Guía para desarrolladores

Amazon Elastic Container Service

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Elastic Container Service: Guía para desarrolladores

Copyright © 2025 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, sino que son propiedad de sus respectivos propietarios, que pueden o no estar afiliados, conectados o patrocinados por Amazon.

Table of Contents

¿Qué es Amazon ECS? ... 1
Terminología y componentes ... 1
Características ... 3
Aprovisionando ... 3
Precios ... 4
Servicios relacionados ... 4
Introducción ... 6
Configuración ... 6
AWS Management Console ... 7
Cómo crear una Cuenta de AWS ... 7
Creación de un usuario con acceso administrativo ... 8
Creación de una nube virtual privada ... 9
Creación de un grupo de seguridad ... 10
Crear las credenciales para conectarse a la instancia de EC2 ... 13
Instalar la AWS CLI ... 14
Pasos siguientes para utilizar Amazon ECS ... 15
Creación de una imagen de contenedor ... 16
Requisitos previos ... 16
Creación de una imagen de Docker ... 18
Envío de la imagen a Amazon Elastic Container Registry ... 20
Limpieza ... 22
Pasos a seguir a continuación ... 22
Obtenga información sobre cómo crear una tarea de Linux para el tipo de lanzamiento de Fargate. ... 22
Requisitos previos ... 23
Paso 1: Crear el clúster ... 24
Paso 2: Crear una definición de tarea ... 24
Paso 3: Crear el servicio ... 26
Paso 4: Ver el servicio ... 26
Paso 5: Eliminar ... 26
Obtenga información sobre cómo crear una tarea de Windows para el tipo de lanzamiento de Fargate. ... 27
Requisitos previos ... 27
Paso 1: creación de un clúster ... 28

Paso 2: Registrar una definición de tareas de Windows ... 29
Paso 3: Crear un servicio con la definición de tarea ... 30
Paso 4: Ver el servicio ... 31
Paso 5: Eliminación ... 31
Obtenga información sobre cómo crear una tarea de Windows para el tipo de lanzamiento de EC2. ... 32
Requisitos previos ... 32
Paso 1: creación de un clúster ... 33
Paso 2: Registrar una definición de tareas ... 35
Paso 3: Crear un servicio ... 36
Paso 4: Ver el servicio ... 37
Paso 5: Eliminación ... 38
Uso de AWS CDK ... 38
Paso 1: Configuración del proyecto AWS CDK ... 39
Paso 2: usar el AWS CDK para definir un servidor web en contenedores en Fargate ... 42
Paso 3: probar el servidor web ... 49
Paso 4: Limpiar ... 50
Pasos a seguir a continuación ... 50
Creación de recursos mediante la CLI de AWS Copilot ... 51
Instalación de la CLI de AWS Copilot ... 52
Implementación de una aplicación de Amazon ECS de ejemplo mediante la CLI de AWS Copilot ... 60
Amazon ECS con AWS CloudFormation ... 63
Plantillas de AWS CloudFormation ejemplo ... 64
Definiciones de tareas ... 64
Clústeres ... 76
Servicios ... 89
Roles de IAM para Amazon ECS ... 146
Creación de recursos de Amazon ECS con la consola de AWS CloudFormation ... 149
Requisitos previos ... 149
Paso 1: creación de una plantilla de pila ... 150
Paso 2: creación de una pila para recursos de Amazon ECS ... 155
Paso 3: verificación ... 156
Paso 4: limpie los recursos ... 157
Creación de recursos de Amazon ECS con comandos de la AWS CLI para AWS CloudFormation ... 157

Requisitos previos ... 158
Paso 1: creación de una pila ... 158
Paso 2: verificación de la creación de recursos ... 164
Paso 3: Limpieza ... 165
Prácticas recomendadas ... 166
AWS Fargate ... 171
Tutoriales ... 171
Proveedores de capacidad ... 172
Definiciones de tareas ... 172
Versiones de la plataforma ... 172
Equilibrio de carga de los servicios ... 173
Métricas de uso ... 173
Consideraciones de seguridad sobre cuándo utilizar el tipo de lanzamiento de Fargate ... 174
Prácticas recomendadas sobre seguridad de Fargate ... 174
Uso de AWS KMS para cifrar el almacenamiento efímero de Fargate ... 174
Capacidad SYS_PTRACE para el seguimiento de llamadas al sistema del kernel con Fargate ... 175
Uso de la supervisión en tiempo de ejecución de Amazon GuardDuty con Fargate ... 175
Consideraciones de seguridad de Fargate ... 176
Versiones de la plataforma Fargate ... 177
... 178
Migración a la versión 1.4.0 de la plataforma Linux ... 179
Registro de cambios de la versión de la plataforma de Linux ... 180
Baja de versiones de la plataforma Linux ... 183
Registro de cambios de la versión de la plataforma de Windows ... 185
Consideraciones sobre los contenedores de Windows en Fargate para Amazon ECS ... 186
Almacenamiento efímero para las tareas de Fargate ... 187
Versiones de plataforma de contenedores Fargate Linux ... 187
Versiones de plataforma de contenedores Windows Fargate ... 189
Claves administradas por el cliente para el almacenamiento efímero de AWS Fargate ... 189
Retirada y mantenimiento de tareas ... 202
Descripción general del aviso de retirada de tareas ... 204
¿Puedo no participar en el retiro de tareas? ... 207
¿Puedo recibir notificaciones de retiro de tareas a través de otros servicios de AWS? ... 208
¿Puedo cambiar el retiro de una tarea una vez programada? ... 208
¿Cómo administra Amazon ECS las tareas que forman parte de un servicio? ... 208

¿Amazon ECS puede administrar automáticamente las tareas independientes? 209
Prepárese para la retirada de tareas de AWS Fargate en Amazon ECS 209
Regiones de AWS Fargate 211
Contenedores de Linux en AWS Fargate 211
Contenedores de Windows en AWS Fargate 214
Arquitectura de la solución para Amazon ECS 216
Capacidad 217
Puntos de conexión de servicio 218
Red 218
Acceso a la característica 219
Roles de IAM 220
Registro 220
Tipos de lanzamiento 221
Tipo de lanzamiento de Fargate 222
Tipo de lanzamiento de EC2 226
Tipo de lanzamiento externo 228
Usar puntos de enlace de doble pila 229
Usar puntos de enlace de doble pila (AWS CLI) 230
Uso de los puntos de enlace de doble pila (SDK de AWS) 231
Usar los puntos de enlace de doble pila desde la API REST 232
Aplicaciones en subredes compartidas, zonas locales y zonas de Wavelength 232
Subredes compartidas 233
Zonas locales 234
Zonas de Wavelength 234
Amazon Elastic Container Service en AWS Outposts 235
Consideraciones 235
Requisitos previos 236
Descripción general de la creación de clústeres en AWS Outposts 236
Prácticas recomendadas para redes 239
Conexión de las aplicaciones a Internet 240
Prácticas recomendadas para recibir conexiones entrantes a Amazon ECS 243
Prácticas recomendadas para conectarse a los servicios de AWS 248
Prácticas recomendadas para conectar servicios 251
Prácticas recomendadas para conectar en red los servicios entre cuentas de AWS y las VPC 256
Servicios de AWS para la solución de problemas de redes 257

Optimización del tiempo de lanzamiento de tareas ... 258
Uso a escala ... 260
Cuotas de servicio de Amazon ECS y limitaciones de las API ... 260
Gestión de los problemas de limitación ... 266
Prácticas recomendadas de escalado automático y administración de la capacidad ... 269
Determinación del tamaño de las tareas ... 269
Optimización del escalado automático de servicios ... 271
Capacidad y disponibilidad ... 277
Capacidad de clústeres ... 282
Elección de los tamaños de las tareas de Fargate ... 283
Aceleración del escalado automático de clústeres ... 283
Acceso a las características con la configuración de la cuenta ... 285
Nombres de recursos de Amazon (ARN) e ID ... 286
Escala de tiempo del formato de ARN e ID de recurso ... 289
Información de contenedores ... 289
Conformidad de AWS Fargate con el Estándar Federal de Procesamiento de la Información(FIPS-140) ... 291
Autorización de etiquetado ... 292
Plazos de la autorización de etiquetado ... 293
Tiempo de espera para el retiro de tareas de AWS Fargate ... 294
Aumento de las interfaces de red de instancias de contenedor de Linux ... 296
Supervisión en tiempo de ejecución (integración con Amazon GuardDuty) ... 296
VPC IPv6 de doble pila ... 297
Modo de controlador de registro predeterminado ... 298
Para ver la configuración de cuenta mediante la consola ... 298
Modificación de la configuración de cuenta ... 299
Reversión a la configuración de cuenta predeterminada ... 300
Administración de la configuración de la cuenta mediante la AWS CLI ... 301
Roles de IAM para Amazon ECS ... 303
Definiciones de tareas ... 307
Estados de definiciones de tareas ... 308
Recursos de Amazon ECS que pueden bloquear una eliminación ... 309
Diseño de la arquitectura de su aplicación ... 310
Prácticas recomendadas para las imágenes de contenedores ... 312
Prácticas recomendadas para el tamaño de las tareas ... 314
Redes de tareas para el tipo de lanzamiento de EC2 ... 315

Redes de tareas para el tipo de lanzamiento de Fargate ... 328
Opciones de almacenamiento para las tareas ... 332
Administración del espacio de memoria de intercambio de contenedores ... 425
Diferencias en la definición de tareas para el tipo de lanzamiento de Fargate ... 426
Diferencias en la definición de tareas para instancias de EC2 que ejecutan Windows ... 436
Creación de una definición de tareas con la consola ... 437
Validación de JSON ... 438
Pilas de AWS CloudFormation ... 438
Procedimiento ... 438
Actualización de una definición de tareas con la consola ... 470
Validación de JSON ... 471
Procedimiento ... 471
Anulación del registro de la revisión de una definición de tareas con la consola ... 472
Pilas de AWS CloudFormation ... 438
Procedimiento ... 473
Eliminación de una revisión de definición de tareas con la consola ... 473
Recursos de Amazon ECS que pueden bloquear una eliminación ... 309
Procedimiento ... 475
Casos de uso de las definiciones de tareas ... 475
Definiciones de tareas para cargas de trabajo de GPU ... 476
Definiciones de tareas para cargas de trabajo de transcodificación de video ... 487
Definiciones de tareas para cargas de trabajo de machine learning de AWS Neuron ... 501
Definiciones de tareas para instancias de aprendizaje profundo ... 510
Definiciones de tareas para cargas de trabajo de ARM de 64 bits ... 513
Envío de registros a CloudWatch ... 515
Envío de registros a un servicio de AWS o AWS Partner ... 519
Uso de imágenes de contenedor que no sean de AWS ... 533
Reinicio de contenedores individuales en las tareas ... 536
Transferencia de datos confidenciales a un contenedor ... 539
Parámetros en la definición de tareas para el tipo de lanzamiento de Fargate ... 564
Familia ... 564
Tipos de lanzamiento ... 564
Rol de la tarea ... 565
Rol de ejecución de tareas ... 565
Modo de red ... 566
Plataforma de tiempo de ejecución ... 566

Tamaño de tarea ... 567
Definiciones de contenedores ... 571
Nombre del acelerador de Elastic Inference ... 618
Configuración del proxy ... 618
Volúmenes ... 621
Etiquetas ... 628
Otros parámetros de definición de tarea ... 629
Parámetros en la definición de tareas para el tipo de lanzamiento de EC2 ... 631
Familia ... 632
Tipos de lanzamiento ... 632
Rol de la tarea ... 633
Rol de ejecución de tareas ... 633
Modo de red ... 634
Plataforma de tiempo de ejecución ... 635
Tamaño de tarea ... 636
Definiciones de contenedores ... 637
Nombre del acelerador de Elastic Inference ... 689
Restricciones para ubicación de tareas ... 689
Configuración del proxy ... 690
Volúmenes ... 692
Etiquetas ... 699
Otros parámetros de definición de tarea ... 700
Plantilla de definición de tareas ... 703
Ejemplos de definiciones de tarea ... 714
Servidor web ... 714
Controlador de registros de `splunk` ... 717
Controlador de registros de `fluentd` ... 717
Controlador de registros de `gelf` ... 718
Cargas de trabajo en instancias externas ... 719
Rol de IAM de definición de tarea e imagen de Amazon ECR ... 720
Punto de entrada con comando ... 721
Dependencia de contenedor ... 721
Volúmenes en definiciones de tareas ... 723
Definiciones de tareas de muestra de Windows ... 724
Clústeres ... 726
Proveedores de capacidad ... 727

Clústeres para Fargate ... 728
Avisos de terminación de Fargate Spot ... 730
Creación de un clúster para el tipo de lanzamiento de Fargate ... 732
Proveedores de capacidad para el tipo de lanzamiento de EC2 ... 735
Seguridad de instancia de contenedor de EC2 ... 738
Creación de un clúster para el tipo de lanzamiento de Amazon EC2 ... 738
Escalado automático de clústeres ... 745
Instancias de contenedor de Amazon EC2 ... 782
Clústeres para el tipo de lanzamiento externo ... 931
Sistemas operativos y arquitecturas de sistemas compatibles ... 931
Consideraciones ... 932
Creación de un clúster para el tipo de lanzamiento externo ... 936
Registro de una instancia externa en un clúster de Amazon ECS ... 939
Anulación del registro de una instancia de externa ... 945
Actualización del agente de AWS Systems Manager y del agente de contenedor de Amazon ECS ... 951
Actualización de un clúster ... 956
Eliminación de un clúster ... 958
Anulación del registro de una instancia de contenedor ... 959
Procedimiento ... 960
Vaciado de instancias de contenedor ... 960
Comportamiento de drenaje de los servicios ... 961
Comportamiento de drenaje para tareas independientes ... 962
Procedimiento ... 962
Instancias de contenedor de Amazon EC2 ... 963
Ciclo de vida ... 964
Compatibilidad con Docker ... 965
AMI optimizada para Amazon ECS ... 965
Información adicional ... 966
Configuración del agente de contenedor ... 966
Instalación del agente de contenedor de Amazon ECS ... 970
Parámetros de configuración del registro del agente de contenedor ... 976
Configuración de instancias de contenedor para imágenes de Docker privadas ... 977
Limpieza de tareas e imágenes ... 982
Programación de los contenedores ... 985
Opciones de computación ... 987

Ciclo de vida de las tareas ... 988
Estados del ciclo de vida ... 989
Cómo coloca Amazon ECS las tareas en las instancias de contenedor ... 991
Tipo de lanzamiento de EC2 ... 991
Tipo de lanzamiento de Fargate ... 992
Uso de estrategias para definir la ubicación de las tareas ... 993
Agrupación de tareas relacionadas ... 1000
Definir de las instancias de contenedor que se utilizan para las tareas ... 1000
Tareas independientes ... 1012
Flujo de trabajo de tareas ... 1012
Ejecución de una aplicación como tarea ... 1013
Uso de Programador de Amazon EventBridge para programar tareas ... 1025
Detención de una tarea ... 1033
Servicios ... 1034
Opción de computación de infraestructura ... 1036
Escalado automático de servicios ... 1036
Equilibrio de carga de los servicios ... 1037
Interconexión de servicios ... 1038
Controladores y estrategias de implementación de servicios ... 1039
Implementaciones de servicios ... 1199
Revisiones de servicios ... 1210
Reequilibrio de la zona de disponibilidad ... 1220
Uso del equilibrador de carga para distribuir el tráfico de servicio ... 1230
Escalado automático de servicios ... 1245
Interconexión de los servicios ... 1287
Protección de reducción horizontal de tareas ... 1349
Inyección de errores con Amazon ECS y Fargate ... 1358
Actualización de los parámetros de servicio de Amazon ECS ... 1368
Eliminación de un servicio ... 1400
Migración de un ARN de servicio corto ... 1401
Lógica de limitación controlada de servicios ... 1407
Parámetros de definición de servicio ... 1409
Plantilla de definición de servicio ... 1438
Etiquetado de recursos ... 1446
Cómo se etiquetan los recursos ... 1447
Etiquetado de recursos durante la creación ... 1450

Restricciones ... 1451
Etiquetas administradas por Amazon ECS ... 1451
Uso de etiquetas para facturación ... 1452
Adición de etiquetas a recursos ... 1453
Adición de etiquetas a una instancia de contenedor ... 1456
Instancias de contenedor externas ... 1457
Informes de uso de ... 1458
Costo y uso de nivel de tarea ... 1459
Monitorización ... 1461
Prácticas recomendadas de supervisión de Amazon ECS ... 1462
Herramientas de supervisión ... 1462
Herramientas automatizadas ... 1462
Herramientas manuales ... 1464
Supervisión de Amazon ECS con CloudWatch ... 1466
Consideraciones ... 1466
Métricas recomendadas ... 1467
Visualización de métricas de Amazon ECS ... 1467
Métricas de Amazon ECS CloudWatch ... 1469
Métricas de uso de AWS Fargate ... 1498
Métricas de reserva del clúster de Amazon ECS ... 1500
Métricas de uso del clúster de Amazon ECS ... 1502
Métricas de uso de los servicios de Amazon ECS ... 1504
Automaticización de las respuestas a los errores de Amazon ECS mediante EventBridge ... 1508
Eventos de Amazon ECS ... 1509
Gestión de eventos ... 1534
Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada ... 1538
Determine el estado de las tareas mediante comprobaciones de estado de los contenedores 1539
Determinación del estado de la tarea ... 1541
Comprobaciones de estado y desconexiones de agentes ... 1542
Ver el estado de los contenedores ... 1542
Supervisión del estado de la instancia de contenedor de Amazon ECS ... 1543
Problemas de estado de las instancias de contenedor ... 1544
Identifique las oportunidades de optimización de Amazon ECS mediante los datos de seguimiento de la aplicación ... 1544

Permisos de IAM necesarios para la integración AWS Distro for OpenTelemetry con AWS X-Ray ... 1545
Especificación del sidecar de AWS Distro for OpenTelemetry para la integración AWS X-Ray de la definición de tarea ... 1546
Correlacionar el rendimiento de las aplicaciones de Amazon ECS mediante métricas de aplicaciones ... 1548
Exportación de métricas de aplicaciones a Amazon CloudWatch ... 1548
Exportación de métricas de aplicaciones a Amazon Managed Service for Prometheus ... 1553
Registro de llamadas a la API de Amazon ECS mediante AWS CloudTrail ... 1557
Eventos de administración de Amazon ECS en CloudTrail ... 1559
Ejemplos de eventos de Amazon ECS ... 1559
Visualización de Live Tail de Registros de CloudWatch para los servicios y tareas de Amazon ECS ... 1561
Permisos necesarios ... 1561
Procedimiento ... 1562
Referencias de AWS CLI ... 1563
Supervisión de las cargas de trabajo mediante metadatos ... 1563
Variables de entorno ... 1564
Archivo de metadatos de contenedores ... 1565
Metadatos de tareas disponibles para tareas de Amazon ECS en EC2 ... 1572
Metadatos de tareas disponibles para tareas de Fargate ... 1616
Introspección de contenedor ... 1640
Identificación de comportamientos no autorizados mediante la supervisión en tiempo de ejecución ... 1643
Cómo funciona la supervisión en tiempo de ejecución con Amazon ECS ... 1644
Consideraciones ... 1645
Uso de los recursos ... 1645
Supervisión en tiempo de ejecución de las cargas de trabajo de Fargate ... 1646
Supervisión en tiempo de ejecución de las cargas de trabajo de EC2 ... 1650
Solucionar problemas de la supervisión en tiempo de ejecución ... 1655
Supervisión de los contenedores de Amazon ECS con ECS Exec ... 1659
Consideraciones ... 1659
Requisitos previos ... 1662
Arquitectura ... 1662
Uso de ECS Exec ... 1663
Registro mediante ECS Exec ... 1666

Utilización de políticas de IAM para limitar el acceso a ECS Exec 1670
Recomendaciones de Compute Optimizer 1674
Recomendaciones de tamaño de tareas para Fargate 1674
Solución de problemas 1675
Solución de los errores de las tareas detenidas 1678
Actualizaciones de los mensajes de error de las tareas detenidas 1679
Visualización de los errores de las tareas detenidas 1684
Mensajes de error de las tareas detenidas 1686
Comprobación de la conectividad de las tareas 1708
Visualización de solicitudes de roles de IAM 1713
Visualización de los mensajes de eventos del servicio 1714
Mensajes de eventos del servicio de Amazon ECS 1715
Mensajes de eventos que no están en buen estado del servicio de Amazon ECS 1727
Mensajes de eventos del servicio de reequilibrio del servicio de zonas de disponibilidad de Amazon ECS 1728
Solución de problemas de los equilibradores de carga de servicio en Amazon ECS 1731
Solución de problemas de escalado automático de servicios en Amazon ECS 1733
Solución de errores de CPU o memoria no válidos en la definición de tareas 1733
Visualización de los registros del agente de contenedor 1736
Recopilación de registros de contenedor con el recopilador de registros de Amazon ECS 1737
Introspección del agente 1740
Diagnósticos de Docker en Amazon ECS 1742
Enumeración de los contenedores de Docker en Amazon ECS 1743
Visualización de los registros de Docker en Amazon ECS 1744
Inspección de los contenedores de Docker en Amazon ECS 1745
Configuración de la salida detallada desde el daemon de Docker en Amazon ECS 1746
Solución del problema de Docker `API error (500): devmapper` en Amazon ECS 1747
Solución de problemas de ECS Exec 1749
Verificación mediante Exec Checker 1749
Error al ejecutar `execute-command` 1749
Solución de problemas de Amazon ECS Anywhere 1750
Problemas de registro de instancias externas 1750
Problemas de red de instancias externas 1751
Problemas al ejecutar tareas 1751
Cuotas de limitación de AWS Fargate 1752
Limitación de la API `RunTask` en Fargate 1753

Ajuste de cuotas de tarifas en Fargate ... 1753
Motivos de los errores de la API ... 1753
Seguridad ... 1766
Identity and Access Management ... 1767
Público ... 1767
Autenticación con identidades ... 1768
Administración de acceso mediante políticas ... 1772
Cómo funciona Amazon Elastic Container Service con IAM ... 1775
Ejemplos de políticas basadas en identidades ... 1787
Políticas administradas por AWS para Amazon ECS ... 1799
Cómo utilizar roles vinculados a servicios ... 1814
Roles de IAM para Amazon ECS ... 1818
Permisos necesarios para la consola de Amazon ECS ... 1884
Permisos necesarios para las funciones de Lambda en las implementaciones azul/verde de Amazon ECS ... 1893
Permisos de IAM necesarios para el escalado automático del servicio de Amazon ECS ... 1895
Recursos de etiquetas durante la creación ... 1896
Solución de problemas ... 1901
Prácticas recomendadas de IAM ... 1903
Registro y supervisión ... 1906
Validación de conformidad ... 1908
Prácticas recomendadas sobre seguridad y conformidad ... 1909
AWS Fargate Conformidad con la norma FIPS-140 ... 1911
Consideraciones sobre FIPS-140 de AWS Fargate ... 1912
Utilice la norma FIPS en Fargate. ... 1912
Uso de CloudTrail para la auditoría de FIPS-140 de Fargate ... 1913
Seguridad de infraestructuras ... 1914
Puntos de conexión de VPC de tipo interfaz (AWS PrivateLink) ... 1915
Modelo de responsabilidad compartida ... 1922
Tipo de lanzamiento de Fargate ... 1923
Tipo de lanzamiento de EC2 ... 1924
Prácticas recomendadas de seguridad de red ... 1925
Cifrado en tránsito ... 1925
Integración en red de las tareas ... 1926
AWS PrivateLink y Amazon ECS ... 1927
Configuración del agente de contenedor ... 1928

Recomendaciones de seguridad de red ... 1929
Prácticas recomendadas de seguridad para las tareas y contenedores ... 1930
Cree imágenes minimalistas o utilice imágenes sin distribución. ... 1931
Escanear las imágenes para detectar vulnerabilidades ... 1932
Eliminar los permisos especiales de sus imágenes ... 1933
Crear un conjunto de imágenes mantenidas ... 1934
Escanear los paquetes y las bibliotecas de aplicaciones para detectar vulnerabilidades ... 1932
Realizar un análisis de código estático ... 1934
Ejecutar contenedores como usuario no raíz ... 1935
Utilizar un sistema de archivos raíz de solo lectura ... 1935
Configurar tareas con límites de CPU y memoria (Amazon EC2) ... 1936
Utilizar etiquetas inmutables con Amazon ECR ... 1936
Evitar ejecutar contenedores con privilegios (Amazon EC2) ... 1936
Eliminar del contenedor las capacidades innecesarias de Linux ... 1937
Utilizar una clave administrada por el cliente (CMK) para cifrar imágenes enviadas a Amazon ECR ... 1937
Tutoriales ... 1938
Creación de una tarea de Linux para el tipo de lanzamiento de Fargate con la AWS CLI ... 1941
Requisitos previos ... 1942
Paso 1: Crear un clúster ... 1943
Paso 2: Registrar una definición de tarea Linux ... 1944
Paso 3: Mostrar la lista de definiciones de tareas ... 1945
Paso 4: Crear un servicio ... 1945
Paso 5: Mostrar la lista de los servicios ... 1946
Paso 6: Describir el servicio en ejecución ... 1947
Paso 7: Prueba ... 1949
Paso 8: eliminación ... 1953
Creación de una tarea de Windows para el tipo de lanzamiento de Fargate con la AWS CLI ... 1953
Requisitos previos ... 1954
Paso 1: Crear un clúster ... 1955
Paso 2: Registrar una definición de tareas de Windows ... 1955
Paso 3: Mostrar la lista de definiciones de tareas ... 1957
Paso 4: Crear un servicio ... 1957
Paso 5: Mostrar la lista de los servicios ... 1958
Paso 6: Describir el servicio en ejecución ... 1958
Paso 7: Eliminación ... 1960

Creación de una tarea para el tipo de lanzamiento de EC2 con la AWS CLI 1961
Requisitos previos 1962
Creación de un clúster 1962
Lanzamiento de una instancia de contenedor de AMI para Amazon ECS 1963
Enumeración de instancias de contenedor 1966
Descripción de la instancia de contenedor 1966
Registro de una definición de tareas 1969
Enumeración de definiciones de tareas 1971
Crear un servicio 1971
Enumeración de servicios 1972
Descripción de los servicios 1972
Descripción de la tarea en ejecución 1973
Prueba del servidor web 1974
Eliminar recursos 1975
Configuración de Amazon ECS para escuchar los eventos de Eventos de CloudWatch 1977
Requisito previo: configurar un clúster de prueba 1977
Paso 1: Crear la función de Lambda 1977
Paso 2: Registrar una regla de eventos 1978
Paso 3: Cree una definición de tarea 1979
Paso 4: Probar la regla 1981
Envío de alertas de Amazon Simple Notification Service para eventos de tareas detenidas ... 1981
Requisito previo: configurar un clúster de prueba 1981
Requisito previo: configurar los permisos para Amazon SNS 1982
Paso 1: Crear y suscribirse a un tema de Amazon SNS 1982
Paso 2: Registrar una regla de eventos 1983
Paso 3: Pruebe la regla 1984
Concatenación de mensajes de registro de seguimiento de pila o de varias líneas 1985
Permisos de IAM necesarios 1986
Determine cuándo utilizar la configuración de registro multilínea 1987
Opciones de análisis y concatenación 1989
Implementación de Fluent Bit en contenedores para Windows 2008
Requisitos previos 2011
Paso 1: Crear roles de acceso de IAM 2012
Paso 2: Crear una instancia de contenedor de Amazon ECS para Windows 2013
Paso 3: Configurar Fluent Bit 2014

Paso 4: Registrar una definición de tarea de Fluent Bit para Windows que dirija los registros a CloudWatch ... 2016
Paso 5: Ejecutar la definición de tarea `ecs-windows-fluent-bit` como un servicio de Amazon ECS mediante la estrategia de programación de daemon ... 2018
Paso 6: Registrar una definición de tarea de Windows que genere los registros ... 2019
Paso 7: Ejecutar la definición de tarea `windows-app-task` ... 2021
Paso 8: Verificar los registros en CloudWatch ... 2021
Paso 9: limpiar ... 2023
Uso de gMSA para contenedores de EC2 Linux ... 2023
Consideraciones ... 2024
Requisitos previos ... 2025
Configuración ... 2026
CredSpec file ... 2033
Uso de gMSA para contenedores de Linux en Fargate ... 2034
Consideraciones ... 2035
Requisitos previos ... 2035
Configuración ... 2036
CredSpec file ... 2039
Uso de contenedores para Windows con gMSA sin dominio mediante la AWS CLI ... 2040
Requisitos previos ... 2041
Paso 1: Crear y configurar la cuenta de gMSA en los servicios de dominio de Active Directory (AD DS) ... 2043
Paso 2: Cargar credenciales a Secrets Manager ... 2045
Paso 3: Modifique el JSON CredSpec para incluir información de gMSA sin dominio ... 2046
Paso 4: Cargar CredSpec a Amazon S3 ... 2048
Paso 5: (Opcional) crear un clúster de Amazon ECS ... 2048
Paso 6: crear un rol de IAM para instancias de contenedor ... 2049
Paso 7: Crear un rol de ejecución de tarea personalizado ... 2049
Paso 8: crear un rol de tarea para Amazon ECS Exec ... 2050
Paso 9: Registrar una definición de tareas ... 2052
Paso 10: Registrar una instancia de contenedor de Windows ... 2054
Paso 11: Verificar la instancia de contenedor ... 2054
Paso 12: Ejecutar una tarea de Windows ... 2056
Paso 13: Comprobar que el contenedor tenga credenciales de gMSA ... 2056
Paso 14: Limpiar ... 2057
Debugging ... 2058

Obtenga información sobre cómo utilizar gMSA para contenedores de EC2 para Windows 2059
Consideraciones 2060
Requisitos previos 2061
Configuración 2062
Uso del Generador de imágenes para crear AMI personalizadas optimizadas para Amazon ECS 2068
Uso del ARN de imagen con la infraestructura como código (IaC) 2070
Uso del ARN de la imagen con AWS CloudFormation 2072
Uso del ARN de la imagen con Terraform 2074
Uso de contenedores de aprendizaje profundo de AWS 2074
Service Quotas 2075
Administración de las cuotas de servicio en la AWS Management Console 2075
Gestión de las cuotas de servicio y los límites de las API 2077
Elastic Load Balancing 262
Interfaces de red elásticas 263
AWS Cloud Map 265
Referencia de la API de Amazon ECS 2083
Historial de documentos 2084

¿Qué es Amazon Elastic Container Service?

Amazon Elastic Container Service (Amazon ECS) es un servicio de orquestación de contenedores completamente administrado que facilita la implementación, la administración y el escalado de aplicaciones en contenedores. Como servicio totalmente administrado, Amazon ECS incluye configuración y prácticas recomendadas operativas integradas de AWS. Se integra tanto con herramientas de AWS, como Amazon Elastic Container Registry, como con herramientas de terceros, como Docker. Esta integración facilita a los equipos centrarse en crear las aplicaciones, no en el entorno. Puede ejecutar y escalar las cargas de trabajo de contenedores en todas las Regiones de AWS en la nube y en las instalaciones, sin la complejidad de administrar un plano de control.

Terminología y componentes

Amazon ECS consta de tres capas:

- Capacidad: la infraestructura en la que se ejecutan sus contenedores
- Controlador: implementan y administran las aplicaciones que se ejecutan en los contenedores
- Aprovisionamiento: las herramientas que puede utilizar para interactuar con el programador a fin de implementar y administrar las aplicaciones y los contenedores

En el siguiente diagrama se muestran las capas de Amazon ECS.

La capacidad es la infraestructura en la que se ejecutan sus contenedores. A continuación, se muestran una descripción general de las opciones de capacidad:

- Instancias de Amazon EC2 en la nube de AWS

  Usted elige el tipo de instancia y la cantidad de instancias y administra la capacidad.
- Sin servidor (AWS Fargate) en la nube de AWS

  Fargate es un motor de cálculos de pago por uso, sin servidor. Con Fargate, no necesita administrar servidores, gestionar la planificación de la capacidad ni aislar las cargas de trabajo de contenedores por seguridad.
- Máquinas virtuales (VM) o servidores locales en las instalaciones

  Amazon ECS Anywhere admite el registro de una instancia externa, por ejemplo, un servidor ubicado en las instalaciones o una máquina virtual (VM), en el clúster de Amazon ECS.

El programador de Amazon ECS es el software que administra sus aplicaciones.

Características

Amazon ECS ofrece las siguientes características de nivel alto:

Definición de tarea

El esquema de la aplicación.

Clúster

La infraestructura en la que se ejecuta la aplicación.

Tarea

Una aplicación, como un trabajo por lotes, que realiza un trabajo y, a continuación, se detiene.

Servicio

Una aplicación sin estado de larga duración.

Configuración de cuenta

Permite el acceso a las características.

Escalado automático de clústeres

Amazon ECS puede administrar el escalado de las instancias de Amazon EC2 registradas en el clúster.

Escalado automático de servicios

Amazon ECS aumenta o disminuye la cantidad de tareas deseada en el servicio de manera automática.

Aprovisionando

Existen varias opciones para aprovisionar Amazon ECS:

- AWS Management Console: proporciona una interfaz web que se puede utilizar para obtener acceso a los recursos de Amazon ECS.

- AWS Command Line Interface (AWS CLI): proporciona comandos para un amplio conjunto de servicios AWS, incluido Amazon ECS. Es compatible con Windows, Mac y Linux. Para obtener más información, consulte AWS Command Line Interface.
- SDK de AWS: proporciona API específicas de cada lenguaje y se encargan de muchos de los detalles de la conexión. Incluyen cálculos de firmas, control de reintentos de solicitud y control de errores. Para obtener más información, consulte SDK de AWS.
- AWS CDK: proporciona un marco de desarrollo de software de código abierto que puede utilizar para modelar y aprovisionar los recursos de sus aplicaciones en la nube mediante lenguajes de programación conocidos. La AWS CDK aprovisiona sus recursos de forma segura y repetible a través de AWS CloudFormation.

Precios

Los precios de Amazon ECS dependen de la opción de capacidad que elija para sus contenedores.

- Precios de Amazon ECS: información sobre los precios de Amazon ECS.
- Precios de AWS Fargate: información sobre los precios de Fargate.

Servicios relacionados

Servicios para utilizar con Amazon ECS

Puede utilizar otros Servicios de AWS como ayuda para implementar las tareas y los servicios en Amazon ECS.

Amazon EC2 Auto Scaling

Le ayuda a garantizar que cuenta con la cantidad correcta de instancias de Amazon EC2 disponibles para controlar la carga de su aplicación.

Amazon CloudWatch

Supervise sus servicios y tareas.

Amazon Elastic Container Registry

Almacene y administre imágenes de contenedor.

Elastic Load Balancing

Distribuya de manera automática el tráfico de servicio entrante.

Amazon GuardDuty

Detecte el uso potencialmente no autorizado o malintencionado de las instancias de contenedor y las cargas de trabajo.

Obtenga información sobre cómo crear y utilizar los recursos de Amazon ECS.

Las siguientes guías sirven como introducción a las herramientas disponibles para acceder a Amazon ECS y los procedimientos básicos para ejecutar contenedores. Los conceptos básicos de Docker explican los pasos básicos para crear una imagen de contenedor de Docker y cargarla a un repositorio privado de Amazon ECR. Las guías de introducción lo guiarán a través de la interfaz de línea de comandos de AWS Copilot y la AWS Management Console a fin de completar las tareas comunes para ejecutar los contenedores en Amazon ECS y AWS Fargate.

Contenido

- Configuración para utilizar Amazon ECS
- Creación de una imagen de contenedor para utilizarla en Amazon ECS
- Obtenga información sobre cómo crear una tarea de Linux de Amazon ECS para el tipo de lanzamiento de Fargate.
- Obtenga información sobre cómo crear una tarea de Windows de Amazon ECS para el tipo de lanzamiento de Fargate.
- Obtenga información sobre cómo crear una tarea de Windows de Amazon ECS para el tipo de lanzamiento de EC2.
- Creación de recursos de Amazon ECS con AWS CDK
- Creación de recursos de Amazon ECS mediante la interfaz de la línea de comandos de AWS Copilot

Configuración para utilizar Amazon ECS

Si ya se ha registrado en Amazon Web Services (AWS) y ha empezado a utilizar Amazon Elastic Compute Cloud (Amazon EC2), está muy de cerca de poder usar Amazon ECS. El proceso de configuración para los dos servicios es similar. La siguiente guía lo prepara para lanzar su primer clúster de Amazon ECS.

Lleve a cabo las siguientes tareas para la configuración de Amazon ECS.

AWS Management Console

La AWS Management Console es una interfaz basada en navegador que permite administrar los recursos de Amazon ECS. La consola proporciona información general visual del servicio, lo que facilita la exploración de las características y funciones de Amazon ECS sin necesidad de utilizar herramientas adicionales. Hay disponibles muchas explicaciones y tutoriales relacionados que pueden guiarlo en cuanto a la utilización de la consola.

Para ver un tutorial que le oriente a través de la consola, consulte Obtenga información sobre cómo crear y utilizar los recursos de Amazon ECS..

Al comenzar, muchos clientes prefieren usar la consola porque les proporciona una indicación visual inmediata de si las acciones que realizan correctamente o no. Los clientes de AWS que están familiarizados con la AWS Management Console, puede administrar fácilmente los recursos relacionados, como balanceadores de carga e instancias de Amazon EC2.

Comience con la AWS Management Console.

Cómo crear una Cuenta de AWS

Si no dispone de una Cuenta de AWS, siga estos pasos para crear una.

Procedimiento para registrarse en Cuenta de AWS

1. Abra https://portal.aws.amazon.com/billing/signup.
2. Siga las instrucciones que se le indiquen.

   Parte del procedimiento de registro consiste en recibir una llamada telefónica o mensaje de texto e indicar un código de verificación en el teclado del teléfono.

   Al registrarse en una Cuenta de AWS, se crea un Usuario raíz de la cuenta de AWS. El usuario raíz tendrá acceso a todos los Servicios de AWS y recursos de esa cuenta. Como práctica recomendada de seguridad, asigne acceso administrativo a un usuario y utilice únicamente el usuario raíz para realizar tareas que requieren acceso de usuario raíz.

AWS le enviará un correo electrónico de confirmación cuando complete el proceso de registro. Puede ver la actividad de la cuenta y administrarla en cualquier momento entrando en https://aws.amazon.com/ y seleccionando Mi cuenta.

Creación de un usuario con acceso administrativo

Después de registrarse para obtener una Cuenta de AWS, proteja su Usuario raíz de la cuenta de AWS, habilite AWS IAM Identity Center y cree un usuario administrativo para no usar el usuario raíz en las tareas cotidianas.

Protección de Usuario raíz de la cuenta de AWS

1. Inicie sesión en AWS Management Console como propietario de la cuenta; para ello, elija Usuario raíz e introduzca el correo electrónico de su Cuenta de AWS. En la siguiente página, escriba su contraseña.

   Para obtener ayuda para iniciar sesión con el usuario raíz, consulte Iniciar sesión como usuario raíz en la Guía del usuario de AWS Sign-In.

2. Active la autenticación multifactor (MFA) para el usuario raíz.

   Para obtener instrucciones, consulte Habilitación de un dispositivo MFA virtual para su usuario raíz de la Cuenta de AWS (consola) en la Guía del usuario de IAM.

Creación de un usuario con acceso administrativo

1. Activar IAM Identity Center.

   Consulte las instrucciones en Activar AWS IAM Identity Center en la Guía del usuario de AWS IAM Identity Center.

2. En IAM Identity Center, conceda acceso administrativo a un usuario.

   Para ver un tutorial sobre cómo usar Directorio de IAM Identity Center como origen de identidad, consulte Configuración del acceso de los usuarios con el Directorio de IAM Identity Center predeterminado en la Guía del usuario de AWS IAM Identity Center.

Inicio de sesión como usuario con acceso de administrador

- Para iniciar sesión con el usuario de IAM Identity Center, use la URL de inicio de sesión que se envió a la dirección de correo electrónico cuando creó el usuario de IAM Identity Center.

  Para obtener ayuda para iniciar sesión con un usuario de IAM Identity Center, consulte Inicio de sesión en el portal de acceso de AWS en la Guía del usuario de AWS Sign-In.

Concesión de acceso a usuarios adicionales

1. En IAM Identity Center, cree un conjunto de permisos que siga la práctica recomendada de aplicar permisos de privilegios mínimos.

   Para conocer las instrucciones, consulte Create a permission set en la Guía del usuario de AWS IAM Identity Center.
2. Asigne usuarios a un grupo y, a continuación, asigne el acceso de inicio de sesión único al grupo.

   Para conocer las instrucciones, consulte Add groups en la Guía del usuario de AWS IAM Identity Center.

Creación de una nube virtual privada

Puede usar Amazon Virtual Private Cloud (Amazon VPC) para lanzar recursos de AWS en una red virtual que haya definido. Sugerimos lanzar las instancias de contenedor en una VPC.

Si dispone de una VPC predeterminada, puede omitir esta sección y pasar a la siguiente tarea, Creación de un grupo de seguridad. Para determinar si tiene una VPC predeterminada, consulte Utilización de su VPC y subredes predeterminadas en la Guía del usuario de Amazon VPC. Si no tiene, puede crear una no predeterminada en su cuenta. Para ello, siga los pasos que se indican a continuación.

Para obtener información acerca de cómo crear una VPC, consulte Creación de una VPC en la Guía del usuario de Amazon VPC y utilice la siguiente tabla para determinar qué opciones seleccionar.

Opción	Valor
Recursos para crear	VPC solo
Nombre	De manera opcional, indique un nombre para su VPC.
IPv4 CIDR block	Entrada manual de IPv4 CIDR El bloque de CIDR debe ser de un tamaño de entre /16 y /28.

Opción	Valor
IPv6 CIDR block	No hay bloque de CIDR IPv6
Propiedad	Predeterminado/a

Para obtener más información acerca de Amazon VPC, consulte ¿Qué es Amazon VPC? en la Guía del usuario de Amazon VPC.

Creación de un grupo de seguridad

Los grupos de seguridad actúan como un firewall para las instancias de contenedor asociadas, al controlar el tráfico entrante y saliente en el nivel de instancia del contenedor. Es posible añadir reglas a un grupo de seguridad que le permita conectarse a la instancia de contenedor desde su dirección IP mediante SSH. También se pueden añadir reglas que permitan HTTP de entrada y salida y acceso HTTPS desde cualquier lugar. Añada reglas para abrir los puertos requeridos por las tareas. Las instancias de contenedor necesitan acceso de red externo para comunicarse con el punto de enlace de servicio de Amazon ECS.

Si pretende lanzar instancias de contenedor en varias regiones deberá crear un grupo de seguridad por región. Para obtener más información, consulte Regiones y zonas de disponibilidad en la Guía del usuario de Amazon EC2.

> **Tip**
>
> Necesitará la dirección IP pública de su equipo local, que puede obtener mediante un servicio. Por ejemplo, proporcionamos el siguiente servicio: http://checkip.amazonaws.com/ o https://checkip.amazonaws.com/. Para buscar otro servicio que le brinde su dirección IP, utilice la frase de búsqueda "what is my IP address" (cuál es mi dirección IP). Si se conecta a través de un proveedor de Internet (ISP) o protegido por un firewall sin una dirección IP estática, debe identificar el rango de direcciones IP utilizadas por los equipos cliente.

Para obtener información acerca de cómo crear un grupo de seguridad, consulte Creación de un grupo de seguridad para una instancia de Amazon EC2 en la Guía del usuario de Amazon EC2 y utilice la siguiente tabla para determinar qué opciones seleccionar.

Opción	Valor
Región	La misma región en la que creó el par de claves.
Nombre	Un nombre que sea fácil de recordar, como ecs-instances-default-cluster.
VPC	La VPC predeterminada (marcada con “(default)”). **Note** Si su cuenta admite Amazon EC2 Classic, seleccione la VPC que creó en la tarea anterior.

Para obtener información sobre las reglas de salida que puede agregar para sus casos de uso, consulte Reglas de grupo de seguridad para diferentes casos de uso en la Guía del usuario de Amazon EC2.

Las instancias de contenedor de Amazon ECS no requieren la apertura de ningún puerto de entrada. Sin embargo, probablemente desee añadir una regla SSH para iniciar sesión en la instancia del contenedor y examinar las tareas con comandos de Docker. También puede añadir reglas para HTTP y HTTPS si desea que su instancia de contenedor pueda alojar una tarea que se ejecuta un servidor web. Las instancias de contenedor necesitan acceso de red externo para comunicarse con el punto de enlace de servicio de Amazon ECS. Siga los pasos a continuación para añadir estas reglas de grupo de seguridad opcionales.

Agregue las siguientes tres reglas de entrada al grupo de seguridad. Para obtener información acerca de cómo crear un grupo de seguridad, consulte Configuración de reglas para un grupo de seguridad en la Guía del usuario de Amazon EC2.

Opción	Valor	
Regla HTTP	Tipo: HTTP Source (Origen): Anywhere (Cualquiera) (`0.0.0.0/0`) Esta opción agrega automátic amente el bloque IPv4 0.0.0.0/0 de CIDR como fuente. Esto es aceptable para un periodo corto en un entorno de prueba, pero no es seguro en entornos de producción. En entornos de producción, debe autorizar el acceso a su instancia únicamente a una dirección IP o a un rango de direcciones IP específico.	
Regla HTTPS	Tipo: HTTPS Source (Origen): Anywhere (Cualquiera) (`0.0.0.0/0`) Esto es aceptable para un periodo corto en un entorno de prueba, pero no es seguro en entornos de producción. En entornos de producción, debe autorizar el acceso a su instancia únicamente a una dirección IP o a un rango de direcciones IP específico.	
Regla SSH	Tipo: SSH	

Opción	Valor	
	Source (Origen): Custom (Personalizada), especifiq ue la dirección IP pública de su equipo o red en notación CIDR. Para especificar una dirección IP individual en notación CIDR, añada el prefijo de enrutamiento `/32`. Por ejemplo, si su dirección IP es `203.0.113.25` , especifiq ue `203.0.113.25/32` . Si su empresa asigna direccion es de un rango, especifíq uelo; por ejemplo, `203.0.113.0/24` . **Important** Por motivos de seguridad, recomenda mos que no permita acceso SSH desde todas las direcciones IP (`0.0.0.0/0`) a la instancia, excepto con fines de prueba y solamente durante un breve periodo.	

Crear las credenciales para conectarse a la instancia de EC2

Para Amazon ECS, solo se necesita un par de claves si va a utilizar el tipo de lanzamiento de EC2.

AWS utiliza criptografía de clave pública para proteger la información de inicio de sesión de la instancia. Una instancia de Linux como, por ejemplo, una instancia de contenedor de Amazon ECS, no tiene contraseña para el acceso a través de SSH. Utilice un par de claves para iniciar sesión de forma segura en la instancia. Usted especifica el nombre del par de claves cuando lanza la instancia de contenedor, luego proporciona la clave privada cuando inicia sesión con SSH.

Si aún no ha creado un par de claves, puede crear uno con la consola de Amazon EC2. Si pretende lanzar instancias en varias regiones deberá crear un par de claves en cada una de ellas. Para obtener más información acerca de las regiones, consulte Regiones y zonas de disponibilidad en la Guía del usuario de Amazon EC2.

Creación de un par de claves

- Utilice la consola de Amazon EC2 para crear un par de claves. Para obtener más información sobre la creación de un par de claves, consulte Crear un par de claves en la Guía del usuario de Amazon EC2.

Para obtener más información acerca de cómo conectarse a su instancia, consulte Conexión con la instancia de Linux en la Guía del usuario de Amazon EC2.

Instalar la AWS CLI

La AWS Management Console se puede utilizar para administrar todas las operaciones manualmente mediante Amazon ECS. Sin embargo, puede instalar la AWS CLI en el escritorio local o en un equipo de desarrollo para crear scripts que puedan automatizar tareas de administración comunes en Amazon ECS.

Para utilizar la AWS CLI con Amazon ECS, instale la última versión de la AWS CLI. Para obtener información acerca de cómo instalar la AWS CLI o actualizarla a la versión más reciente, consulte Instalación o actualización de la versión más reciente de la AWS CLI en la Guía del usuario de AWS Command Line Interface.

La AWS Command Line Interface (AWS CLI) es una herramienta unificada que se puede utilizar para administrar los servicios de AWS. Con esta única herramienta, puede controlar múltiples servicios de AWS y automatizarlos a través de scripts. Los comandos de Amazon ECS de la AWS CLI son idénticos a los de la API de Amazon ECS.

La AWS CLI es la opción perfecta para los clientes que prefieren y están acostumbrados al scripting y a interactuar con herramientas de línea de comandos y saben exactamente qué acciones quieren

realizar en sus recursos de Amazon ECS. La AWS CLI también es útil para quienes desean familiarizarse con las API de Amazon ECS. Los clientes pueden usar la AWS CLI para realizar diversas operaciones en recursos de Amazon ECS, incluidas las operaciones de crear, leer, actualizar y eliminar, directamente desde la interfaz de la línea de comandos.

Utilice la AWS CLI si está familiarizado o desea familiarizarse con las API de Amazon ECS y los comandos correspondientes de la CLI y desea escribir scripts automatizados y realizar acciones específicas en los recursos de Amazon ECS.

AWS también proporciona las herramientas de línea de comandos AWS Tools for Windows PowerShell. Para obtener más información, consulte la Guía del usuario de AWS Tools for Windows PowerShell.

Pasos siguientes para utilizar Amazon ECS

Tras la instalación de la AWS CLI, hay muchas herramientas diferentes que puede emplear para seguir utilizando Amazon ECS. En los siguientes enlaces se explican algunas de esas herramientas y se ofrecen ejemplos de cómo utilizarlas con Amazon ECS.

- Cree su primera imagen de contenedor con Docker y envíela a Amazon ECR para utilizarla en las definiciones de tareas de Amazon ECS.
- Obtenga información sobre cómo crear una tarea de Linux de Amazon ECS para el tipo de lanzamiento de Fargate..
- Obtenga información sobre cómo crear una tarea de Windows de Amazon ECS para el tipo de lanzamiento de Fargate..
- Obtenga información sobre cómo crear una tarea de Windows de Amazon ECS para el tipo de lanzamiento de EC2..
- Utilice su lenguaje de programación preferido para definir la infraestructura o arquitectura como código con el Creación de recursos de Amazon ECS con AWS CDK.
- Defina y administre todos los recursos de AWS en su entorno con implementación automatizada mediante el Uso de Amazon ECS con AWS CloudFormation.
- Utilice el flujo de trabajo de desarrollador de extremo a extremo de Creación de recursos de Amazon ECS mediante la interfaz de la línea de comandos de AWS Copilot para crear, lanzar y operar aplicaciones de contenedor que cumplan con las prácticas recomendadas para infraestructura de AWS.

Creación de una imagen de contenedor para utilizarla en Amazon ECS

Amazon ECS utiliza imágenes de Docker en las definiciones de tareas para lanzar contenedores. Docker es una tecnología que brinda herramientas para crear, ejecutar, probar e implementar aplicaciones distribuidas en contenedores.

Amazon ECS programa las aplicaciones en contenedores en instancias de contenedores o en AWS Fargate. Las aplicaciones en contenedores se empaquetan como imágenes de contenedores. En este ejemplo, se crea una imagen de contenedor para un servidor web.

Puede crear su primera imagen de Docker y enviarla a Amazon ECR, que es un registro de contenedores, para utilizarla en las definiciones de tareas de Amazon ECS. Este tutorial presupone que el lector posee conocimientos básicos sobre Docker y su funcionamiento. Para obtener más información sobre Docker, consulte ¿Qué es Docker? y la documentación de Docker.

Requisitos previos

Antes de comenzar, asegúrese de que se cumplen los siguientes requisitos previos:

- Asegúrese de que ha completado los pasos de configuración de Amazon ECR. Para obtener más información, consulte Moving an image through its lifecycle in Amazon ECR en la Guía del usuario de Amazon Elastic Container Registry.
- El usuario tiene los permisos de IAM requeridos para tener acceso y usar el servicio de Amazon ECR. Para obtener más información, consulte Políticas administradas de Amazon ECR.
- Tiene Docker instalado. Para los pasos de instalación de Docker para Amazon Linux 2023, consulte Instalación de Docker en AL2023. Para los demás sistemas operativos, consulte la documentación de Docker en Descripción general de Docker de Escritorio.
- Tener instalada y configurada la AWS CLI. Para obtener más información, consulte Instalación o actualización de la versión más reciente de la AWS CLI en la Guía del usuario de AWS Command Line Interface.

Si no necesita o tiene un entorno de desarrollo local y prefiere utilizar una instancia de Amazon EC2 para utilizar Docker, siga estos pasos para lanzar una instancia de Amazon EC2 usando Amazon Linux 2023 e instalar Docker Engine y la CLI de Docker.

Instalación de Docker en AL2023

Docker está disponible en muchos sistemas operativos diferentes, incluidas las distribuciones de Linux más modernas, como Ubuntu, e incluso en macOS y Windows. Para obtener más información sobre cómo instalar Docker en su sistema operativo concreto, consulte la guía de instalación de Docker.

No necesita un sistema de desarrollo local para usar Docker. Si ya utiliza Amazon EC2, puede lanzar una instancia de Amazon Linux 2023 e instalar Docker para empezar.

Si ya tiene Docker instalado, pase a Creación de una imagen de Docker.

Instalación de Docker en una instancia de Amazon EC2 mediante una AMI de Amazon Linux 2023

1. Lance una instancia con la AMI de Amazon Linux 2023. Para obtener más información, consulte Inicialización de una instancia de EC2 mediante el asistente de inicialización de instancias de la consola en la Guía del usuario de Amazon EC2.
2. Conecte con la instancia . Para obtener más información, consulte Conexión con instancias de EC2 en la Guía del usuario de Amazon EC2.
3. Actualice la caché de paquetes y los paquetes instalados en la instancia.

   ```
   sudo yum update -y
   ```

4. Instale el paquete de Community Edition de Docker más reciente.

   ```
   sudo yum install docker
   ```

5. Abra el servicio de Docker.

   ```
   sudo service docker start
   ```

6. Agregue el `ec2-user` al grupo `docker` para que pueda ejecutar comandos de Docker sin usar `sudo`.

   ```
   sudo usermod -a -G docker ec2-user
   ```

7. Cierre sesión y vuelva a iniciarla para actualizar los nuevos permisos de grupo de `docker`. Para ello, cierre la ventana de su terminal de SSH actual y vuelva a conectarse a la instancia en una ventana nueva. De esta forma, la nueva sesión de SSH tendrá los permisos de grupo de `docker` adecuados.

8. Compruebe que el `ec2-user` puede ejecutar comandos de Docker sin `sudo`.

```
docker info
```

> **Note**
>
> En algunos casos, es posible que tenga que reiniciar su instancia para que el `ec2-user` tenga los permisos necesarios para acceder al daemon de Docker. Intente reiniciar su instancia si ve el siguiente error:
>
> ```
> Cannot connect to the Docker daemon. Is the docker daemon running on this
> host?
> ```

Creación de una imagen de Docker

Las definiciones de tareas de Amazon ECS utilizan imágenes de contenedores para lanzar contenedores en las instancias de contenedor de los clústeres. En esta sección, va a crear una imagen de Docker de una aplicación web simple y la va a probar en su sistema local o en la instancia de Amazon EC2. Luego, enviará la imagen a un registro de contenedores de Amazon ECR para poder utilizarla en una definición de tarea de Amazon ECS.

Creación de una imagen Docker de una aplicación web simple

1. Cree un archivo denominado `Dockerfile`. Un Dockerfile es un manifiesto que describe la imagen base para su imagen Docker y qué desea instalar y que se ejecute en ella. Para obtener más información acerca de los archivos Docker, consulte Docker Reference.

```
touch Dockerfile
```

2. Edite el `Dockerfile` que acaba de crear y agregue el siguiente contenido.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest

# Update installed packages and install Apache
RUN yum update -y && \
 yum install -y httpd

# Write hello world message
```

```
RUN echo 'Hello World!' > /var/www/html/index.html

# Configure Apache
RUN echo 'mkdir -p /var/run/httpd' >> /root/run_apache.sh && \
 echo 'mkdir -p /var/lock/httpd' >> /root/run_apache.sh && \
 echo '/usr/sbin/httpd -D FOREGROUND' >> /root/run_apache.sh && \
 chmod 755 /root/run_apache.sh

EXPOSE 80

CMD /root/run_apache.sh
```

Este Dockerfile utiliza la imagen pública de Amazon Linux 2023 alojada en Amazon ECR Public. Las instrucciones `RUN` actualizan la caché del paquete, instalan algunos paquetes de software para el servidor web y, a continuación, escriben el contenido “Hello World!”. en la raíz de documentos del servidor web. La instrucción `EXPOSE` significa que el puerto 80 del contenedor es el que está escuchando y la instrucción `CMD` inicia el servidor web.

3. Cree la imagen Docker desde el Dockerfile.

> **Note**
>
> Algunas versiones de Docker pueden requerir la ruta completa a su Dockerfile en el siguiente comando en lugar de la ruta relativa que se muestra a continuación.
> Si ejecuta el comando en un sistema basado en ARM, como Apple Silicon, utilice la opción de plataforma “--platform linux/amd64”.

```
docker build -t hello-world .
```

4. Incluya la imagen de su contenedor.

```
docker images --filter reference=hello-world
```

Salida:

```
REPOSITORY          TAG                 IMAGE ID            CREATED
 SIZE
hello-world         latest              e9ffedc8c286        4 minutes ago
 194MB
```

5. Ejecute la nueva imagen. La opción `-p 80:80` asigna el puerto 80 expuesto en el contenedor al puerto 80 del sistema de host.

```
docker run -t -i -p 80:80 hello-world
```

> **Note**
>
> La salida desde el servidor web Apache se muestra en la ventana de la terminal. Puede hacer caso omiso del mensaje "`Could not reliably determine the fully qualified domain name`"

6. Abra un navegador y encuentre el servidor que está ejecutando Docker y alojando su contenedor.

   - Si utiliza una instancia de EC2, este es el valor DNS público para el servidor, que es la misma dirección que utiliza para conectarse a la instancia con SSH. Asegúrese de que el grupo de seguridad para la instancia permite el tráfico entrante en el puerto 80.
   - Si ejecuta Docker de forma local, dirija el navegador a http://localhost/.

   Debería ver una página web que diga "Hello, World!" statement.

7. Detenga el contenedor de Docker escribiendo Ctrl + c.

Envío de la imagen a Amazon Elastic Container Registry

Amazon ECR es un servicio de registro de imágenes administrado por AWS. Puede utilizar la CLI de Docker para enviar, extraer y administrar imágenes de sus repositorios de Amazon ECR. Para obtener información detallada sobre los productos de Amazon ECR, casos prácticos destacados de clientes y preguntas frecuentes, consulte las páginas de detalles de productos de Amazon Elastic Container Registry.

Para etiquetar la imagen y enviarla a Amazon ECR

1. Cree un repositorio de Amazon ECR para almacenar la imagen `hello-world`. En los resultados, anote el `repositoryUri`.

   Sustituya `region`, con la Región de AWS, por ejemplo, `us-east-1`.

```
aws ecr create-repository --repository-name hello-repository --region region
```

Salida:

```
{
    "repository": {
        "registryId": "aws_account_id",
        "repositoryName": "hello-repository",
        "repositoryArn": "arn:aws:ecr:region:aws_account_id:repository/hello-
repository",
        "createdAt": 1505337806.0,
        "repositoryUri": "aws_account_id.dkr.ecr.region.amazonaws.com/hello-
repository"
    }
}
```

2. Etiquete la imagen de `hello-world` con el valor de `repositoryUri` del paso anterior.

```
docker tag hello-world aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository
```

3. Ejecute el comando aws ecr get-login-password. Especifique el URI del registro en el que desea autenticar. Para obtener más información, consulte Autenticación de registros en la Guía del usuario de Amazon Elastic Container Registry.

```
aws ecr get-login-password --region region | docker login --username AWS --
password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
```

Salida:

```
Login Succeeded
```

> **Important**
>
> Si recibe un error, instale o actualice a la versión más reciente de la AWS CLI. Para obtener más información, consulte Instalación o actualización de la versión más reciente de la AWS CLI en la Guía del usuario de AWS Command Line Interface.

4. Envíe la imagen a Amazon ECR con el valor `repositoryUri` del paso anterior.

```
docker push aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository
```

Limpieza

Para seguir creando una definición de tarea de Amazon ECS e iniciando una tarea con la imagen de contenedor, vaya al Pasos a seguir a continuación. Cuando haya terminado de experimentar con su imagen de Amazon ECR, puede eliminar el repositorio para que no se le cobre por el almacenamiento de imágenes.

```
aws ecr delete-repository --repository-name hello-repository --region region --force
```

Pasos a seguir a continuación

Las definiciones de tareas requieren un rol de ejecución de tarea. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

Después de crear e insertar la imagen de contenedor en Amazon ECR, puede utilizar esa imagen en una definición de tarea. Para obtener más información, consulte una de las siguientes:

- the section called “Obtenga información sobre cómo crear una tarea de Linux para el tipo de lanzamiento de Fargate.”
- the section called “Obtenga información sobre cómo crear una tarea de Windows para el tipo de lanzamiento de Fargate.”
- Creación de una tarea de Linux de Amazon ECS para el tipo de lanzamiento de Fargate con la AWS CLI

Obtenga información sobre cómo crear una tarea de Linux de Amazon ECS para el tipo de lanzamiento de Fargate.

Amazon Elastic Container Service (Amazon ECS) es un servicio de administración de contenedores altamente escalable y rápido que facilita la ejecución, detención y administración de los contenedores. Para alojar los contenedores en una infraestructura sin servidor administrada por Amazon ECS, puede lanzar sus servicios o tareas en AWS Fargate. Para obtener más información, consulte AWS Fargate para Amazon ECS.

Introducción a Amazon ECS en AWS Fargate con el tipo de lanzamiento de Fargate para las tareas en las regiones donde Amazon ECS admite a AWS Fargate.

Para comenzar a utilizar Amazon ECS en AWS Fargate, siga estos pasos.

Requisitos previos

Antes de comenzar, complete los pasos en Configuración para utilizar Amazon ECS y corrobore que su usuario de IAM disponga de los permisos que se especifican en la política de IAM `AdministratorAccess` de ejemplo.

La consola intenta crear automáticamente el rol de IAM de ejecución de tareas, que se requiere para las tareas de Fargate. Para asegurarse de que la consola pueda crear este rol de IAM, una de las siguientes condiciones debe ser verdadera:

- El usuario tiene acceso de administrador. Para obtener más información, consulte Configuración para utilizar Amazon ECS.
- El usuario tiene los permisos de IAM para crear un rol de servicio. Para obtener más información, consulte Creación de un rol para delegar permisos a un servicio de AWS.
- Un usuario con acceso de administrador ha creado manualmente el rol de ejecución de tareas para que se encuentre disponible en la cuenta que se va a utilizar. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

> **Important**
>
> El grupo de seguridad que seleccione al crear un servicio con su definición de tareas debe tener el puerto 80 abierto para el tráfico entrante. Agregue las siguientes reglas de entrada al grupo de seguridad. Para obtener información acerca de cómo crear un grupo de seguridad, consulte Cree un grupo de seguridad para su instancia de Amazon EC2 en la Guía del usuario de Amazon EC2.
>
> - Tipo: HTTP
> - Protocolo: TCP
> - Intervalo de puertos: 80
> - Source (Origen): Anywhere (Cualquiera) (`0.0.0.0/0`)

Paso 1: Crear el clúster

Cree un clúster que use la VPC predeterminada.

Antes de empezar, asigne el permiso de IAM correspondiente. Para obtener más información, consulte the section called “Ejemplos de clústeres de Amazon ECS”.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, seleccione la región a utilizar.
3. En el panel de navegación, seleccione Clusters (Clústeres).
4. En la página Clusters (Clústeres), elija Create Cluster (Crear clúster).
5. En CLuster configuration (Configuración de clúster), para Cluster name (Nombre del clúster), introduzca un nombre único.

   El nombre puede contener hasta 255 letras (minúsculas y mayúsculas), números y guiones.
6. (Opcional) Para activar Container Insights, expanda Monitoring (Supervisión) y, a continuación, active Use Container Insights (Uso de Container Insights).
7. (Opcional) Para ayudar a identificar el clúster, expanda Tags (Etiquetas) y, a continuación, configure sus etiquetas.

   [Añadir una etiqueta] Elija Add tag (Añadir etiqueta) y haga lo siguiente:

   - En Key (Clave), escriba el nombre de la clave.
   - En Valor, escriba el valor de la clave.

   [Eliminar una etiqueta] Elija Eliminar a la derecha de la clave y el valor de la etiqueta.
8. Seleccione Create (Crear).

Paso 2: Crear una definición de tarea

Una definición de tarea es una especie de plano de la aplicación. Cada vez que lance una tarea en Amazon ECS, debe especificar una definición de tarea. Esto permite que el servicio sepa qué imagen de Docker debe usar para los contenedores, cuántos contenedores debe utilizar en la tarea, así como la asignación de recursos para cada contenedor.

1. En el panel de navegación, elija Task Definitions.

2. Elija Create new Task Definition (Crear nueva definición de tarea) y Create new revision with JSON (Crear nueva revisión con JSON).
3. Copie y pegue la siguiente definición de tarea de ejemplo en el cuadro y, a continuación, elija Save (Guardar).

```
{
    "family": "sample-fargate",
    "networkMode": "awsvpc",
    "containerDefinitions": [
        {
            "name": "fargate-app",
            "image": "public.ecr.aws/docker/library/httpd:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": [
                "sh",
  "-c"
            ],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
            ]
        }
    ],
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "cpu": "256",
    "memory": "512"
}
```

4. Seleccione Crear.

Paso 3: Crear el servicio

Cree un servicio mediante la definición de tarea.

1. En el panel de navegación, elija Clusters (Clústeres) y, a continuación, seleccione el clúster que creó en Paso 1: Crear el clúster.
2. En la pestaña Services (Servicios), elija Create (Crear).
3. En Deployment configuration (Configuración de implementación), especifique cómo se implementa su aplicación.
   a. En Task definition (Definición de tarea), elija la definición de tarea que creó en Paso 2: Crear una definición de tarea.
   b. En Service name (Nombre del servicio), ingrese un nombre para el servicio.
   c. En Desired tasks (Tareas deseadas), ingrese 1.
4. En Redes, puede crear un nuevo grupo de seguridad o elegir uno existente para la tarea. Asegúrese de que el grupo de seguridad que utilice tenga la regla de entrada que se indica en Requisitos previos.
5. Seleccione Crear.

Paso 4: Ver el servicio

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. Elija el clúster en el que ejecutó el servicio.
4. En la pestaña Servicios, en Nombre del servicio, elija el servicio que creó en Paso 3: Crear el servicio.
5. Seleccione la pestaña Tareas y, a continuación, elija la tarea de su servicio.
6. En la página de tareas, en la sección Configuración, en IP pública, elija Dirección abierta.

Paso 5: Eliminar

Cuando termine de utilizar un clúster de Amazon ECS, debe limpiar los recursos asociados para evitar que se generen cargos por recursos que no está utilizando.

Algunos recursos de Amazon ECS, tales como tareas, servicios, clústeres e instancias de contenedor, se limpian a través de la consola de Amazon ECS. Otros recursos como, por ejemplo, las instancias de Amazon EC2, los balanceadores de carga de Elastic Load Balancing y los grupos de Auto Scaling, se deben limpiar manualmente en la consola de Amazon EC2 o eliminando la pila de AWS CloudFormation que los creó.

1. En el panel de navegación, seleccione Clusters (Clústeres).
2. En la página Clústeres, seleccione el clúster que creó para este tutorial.
3. Seleccione la pestaña Servicios.
4. Seleccione el servicio y, a continuación, elija Eliminar.
5. En la pregunta de confirmación, escriba delete (eliminar) y, a continuación, elija Delete (Eliminar). Como alternativa, puede utilizar la opción `Force delete` para que Amazon ECS reduzca verticalmente el servicio en su nombre antes de eliminarlo.

   Espere hasta que se elimine el servicio.
6. Elija Delete cluster. En la pregunta de confirmación, ingrese delete ***`cluster-name`*** (eliminar nombre de clúster) y, a continuación, elija Delete (Eliminar). Al eliminar el clúster, se limpian los recursos asociados que se crearon con él, incluidos los grupos de Auto Scaling, las VPC o los balanceadores de carga.

Obtenga información sobre cómo crear una tarea de Windows de Amazon ECS para el tipo de lanzamiento de Fargate.

Introducción a Amazon ECS en AWS Fargate con el tipo de lanzamiento de Fargate para las tareas en las regiones donde Amazon ECS admite a AWS Fargate.

Para comenzar a utilizar Amazon ECS en AWS Fargate, siga estos pasos.

Requisitos previos

Antes de comenzar, complete los pasos en Configuración para utilizar Amazon ECS y corrobore que su usuario de IAM disponga de los permisos que se especifican en la política de IAM `AdministratorAccess` de ejemplo.

La consola intenta crear automáticamente el rol de IAM de ejecución de tareas, que se requiere para las tareas de Fargate. Para asegurarse de que la consola pueda crear este rol de IAM, una de las siguientes condiciones debe ser verdadera:

- El usuario tiene acceso de administrador. Para obtener más información, consulte Configuración para utilizar Amazon ECS.
- El usuario tiene los permisos de IAM para crear un rol de servicio. Para obtener más información, consulte Creación de un rol para delegar permisos a un servicio de AWS.
- Un usuario con acceso de administrador ha creado manualmente el rol de ejecución de tareas para que se encuentre disponible en la cuenta que se va a utilizar. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

> **Important**
>
> El grupo de seguridad que seleccione al crear un servicio con su definición de tareas debe tener el puerto 80 abierto para el tráfico entrante. Agregue las siguientes reglas de entrada al grupo de seguridad. Para obtener información acerca de cómo crear un grupo de seguridad, consulte Cree un grupo de seguridad para su instancia de Amazon EC2 en la Guía del usuario de Amazon EC2.
>
> - Tipo: HTTP
> - Protocolo: TCP
> - Intervalo de puertos: 80
> - Source (Origen): Anywhere (Cualquiera) (`0.0.0.0/0`)

Paso 1: creación de un clúster

Puede crear un nuevo clúster denominado Windows que utilice la VPC predeterminada.

Para crear un clúster con la AWS Management Console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, seleccione la región a utilizar.
3. En el panel de navegación, seleccione Clusters (Clústeres).
4. En la página Clusters (Clústeres), elija Create Cluster (Crear clúster).
5. En Cluster configuration (Configuración de clúster), para Cluster name (Nombre del clúster), ingrese windows.

6. (Opcional) Para activar Container Insights, expanda Monitoring (Supervisión) y, a continuación, active Use Container Insights (Uso de Container Insights).
7. (Opcional) Para ayudar a identificar el clúster, expanda Tags (Etiquetas) y, a continuación, configure sus etiquetas.

   [Añadir una etiqueta] Elija Add tag (Añadir etiqueta) y haga lo siguiente:

   - En Key (Clave), escriba el nombre de la clave.
   - En Valor, escriba el valor de la clave.

   [Eliminar una etiqueta] Elija Eliminar a la derecha de la clave y el valor de la etiqueta.
8. Seleccione Create (Crear).

Paso 2: Registrar una definición de tareas de Windows

Antes de poder ejecutar los contenedores de Windows en el clúster de Amazon ECS, debe registrar una definición de tarea. El siguiente ejemplo de definición de tareas muestra una página web sencilla en el puerto 8080 de una instancia de contenedor con la imagen de contenedor `mcr.microsoft.com/windows/servercore/iis`.

Para registrar la definición de tarea de muestra con la AWS Management Console

1. En el panel de navegación, elija Task Definitions (Definiciones de tareas).
2. Elija Create new task definition (Crear nueva definición de tarea) y Create new task definition with JSON (Crear nueva definición de tarea con JSON).
3. Copie y pegue la siguiente definición de tarea de ejemplo en el cuadro y, a continuación, elija Save (Guardar).

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html
 -Type file -Value '<html> <head> <title>Amazon ECS Sample App</title>
 <style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
 <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p>'; C:\\ServiceMonitor.exe w3svc"],
            "entryPoint": [
```

```
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

4. Verifique su información y seleccione Create (Crear).

Paso 3: Crear un servicio con la definición de tarea

Después de haber registrado la definición de tarea, puede colocar tareas en el clúster con ella. El procedimiento siguiente crea un servicio con su definición de tarea y coloca una tarea en el clúster.

Para crear un servicio a partir de la definición de tarea con la consola

1. En el panel de navegación, elija Clusters (Clústeres) y, a continuación, seleccione el clúster que creó en Paso 1: creación de un clúster.
2. En la pestaña Services (Servicios), elija Create (Crear).
3. En Deployment configuration (Configuración de implementación), especifique cómo se implementa su aplicación.

   a. En Task definition (Definición de tarea), elija la definición de tarea que creó en Paso 2: Registrar una definición de tareas de Windows.
   b. En Service name (Nombre del servicio), ingrese un nombre para el servicio.
   c. En Desired tasks (Tareas deseadas), ingrese 1.
4. En Redes, puede crear un grupo de seguridad o elegir uno existente. Asegúrese de que el grupo de seguridad que utilice tenga la regla de entrada que se indica en Requisitos previos.
5. Seleccione Crear.

Paso 4: Ver el servicio

Después de que el servicio haya lanzado una tarea en el clúster, puede ver el servicio y abrir la página de prueba de IIS en un navegador para verificar que el contenedor se está ejecutando.

> **Note**
>
> La instancia de contenedor puede tardar hasta 15 minutos en descargarse y extraer las capas de base de contenedor de Windows.

Para ver el servicio

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. Elija el clúster en el que ejecutó el servicio.
4. En la pestaña Servicios, en Nombre del servicio, elija el servicio que creó en Paso 3: Crear un servicio con la definición de tarea.
5. Seleccione la pestaña Tareas y, a continuación, elija la tarea de su servicio.
6. En la página de tareas, en la sección Configuración, en IP pública, elija Dirección abierta.

Paso 5: Eliminación

Cuando termine de utilizar un clúster de Amazon ECS, debe limpiar los recursos asociados para evitar que se generen cargos por recursos que no está utilizando.

Algunos recursos de Amazon ECS, tales como tareas, servicios, clústeres e instancias de contenedor, se limpian a través de la consola de Amazon ECS. Otros recursos como, por ejemplo, las instancias de Amazon EC2, los balanceadores de carga de Elastic Load Balancing y los grupos de Auto Scaling, se deben limpiar manualmente en la consola de Amazon EC2 o eliminando la pila de AWS CloudFormation que los creó.

1. En el panel de navegación, seleccione Clusters (Clústeres).
2. En la página Clústeres, seleccione el clúster que creó para este tutorial.
3. Seleccione la pestaña Servicios.
4. Seleccione el servicio y, a continuación, elija Eliminar.
5. En la pregunta de confirmación, escriba delete (eliminar) y, a continuación, elija Delete (Eliminar).

   Espere hasta que se elimine el servicio.
6. Elija Delete cluster. En la pregunta de confirmación, ingrese delete ***`cluster-name`*** (eliminar nombre de clúster) y, a continuación, elija Delete (Eliminar). Al eliminar el clúster, se limpian los recursos asociados que se crearon con él, incluidos los grupos de Auto Scaling, las VPC o los balanceadores de carga.

Obtenga información sobre cómo crear una tarea de Windows de Amazon ECS para el tipo de lanzamiento de EC2.

Para comenzar a utilizar Amazon ECS mediante el tipo de lanzamiento de EC2, registre una definición de tarea y cree un clúster y un servicio en la consola.

Siga estos pasos para comenzar a utilizar Amazon ECS mediante el tipo de lanzamiento de EC2.

Requisitos previos

Antes de comenzar, complete los pasos en Configuración para utilizar Amazon ECS y corrobore que su usuario de IAM disponga de los permisos que se especifican en la política de IAM `AdministratorAccess` de ejemplo.

La consola intenta crear automáticamente el rol de IAM de ejecución de tareas, que se requiere para las tareas de Fargate. Para asegurarse de que la consola pueda crear este rol de IAM, una de las siguientes condiciones debe ser verdadera:

- El usuario tiene acceso de administrador. Para obtener más información, consulte Configuración para utilizar Amazon ECS.
- El usuario tiene los permisos de IAM para crear un rol de servicio. Para obtener más información, consulte Creación de un rol para delegar permisos a un servicio de AWS.
- Un usuario con acceso de administrador ha creado manualmente el rol de ejecución de tareas para que se encuentre disponible en la cuenta que se va a utilizar. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

> **Important**
>
> El grupo de seguridad que seleccione al crear un servicio con su definición de tareas debe tener el puerto 80 abierto para el tráfico entrante. Agregue las siguientes reglas de entrada al grupo de seguridad. Para obtener información acerca de cómo crear un grupo de seguridad, consulte Cree un grupo de seguridad para su instancia de Amazon EC2 en la Guía del usuario de Amazon EC2.
>
> - Tipo: HTTP
> - Protocolo: TCP
> - Intervalo de puertos: 80
> - Source (Origen): Anywhere (Cualquiera) (`0.0.0.0/0`)

Paso 1: creación de un clúster

Un clúster de Amazon ECS es una agrupación lógica de tareas, servicios e instancias de contenedores.

Los siguientes pasos lo guiarán a través de la creación de un clúster con una instancia de Amazon EC2 registrada en él que nos permitirá ejecutar una tarea en el clúster. Si no se menciona un campo específico, deje el valor predeterminado de la consola.

Para crear un nuevo clúster (consola de Amazon ECS)

Antes de empezar, asigne el permiso de IAM correspondiente. Para obtener más información, consulte the section called “Ejemplos de clústeres de Amazon ECS”.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.

2. En la barra de navegación, seleccione la región a utilizar.
3. En el panel de navegación, seleccione Clusters (Clústeres).
4. En la página Clusters (Clústeres), elija Create Cluster (Crear clúster).
5. En CLuster configuration (Configuración de clúster), para Cluster name (Nombre del clúster), introduzca un nombre único.

   El nombre puede contener hasta 255 letras (minúsculas y mayúsculas), números y guiones.
6. (Opcional) Para cambiar la VPC y las subredes donde se inician sus tareas y servicios, en Networking (Redes), realice cualquiera de las siguientes operaciones:
   - Para eliminar una subred, en Subnets (Subredes), elija Xpara cada subred que desea eliminar.
   - Para cambiar a una VPC distinta de la VPC predeterminada, enVPC, elija una VPC existente y, a continuación, en Subnets (Subredes), seleccione cada subred.
7. Para agregar instancias de Amazon EC2 al clúster, expanda Infraestructura y, a continuación, seleccione Instancias de Amazon EC2. A continuación, configure el grupo de Auto Scaling que actúa como proveedor de capacidad:
   a. Para utilizar un grupo de Auto Scaling existente, desde Auto Scaling group (ASG) (Grupo de Auto Scaling), seleccione el grupo.
   b. Para crear un grupo de Auto Scaling, desde Auto Scaling group (ASG) (Grupo de Auto Scaling), seleccione Create new group (Crear nuevo grupo) y, a continuación, proporcione los siguientes detalles sobre el grupo:
      - Para Operating system/Architecture (Arquitectura y sistema operativo), elija la AMI optimizada para Amazon ECS para las instancias de grupo de Auto Scaling.
      - Para EC2 instance type (Tipo de instancia EC2), elija el tipo de instancia para sus cargas de trabajo. Para obtener más información acerca de los diferentes tipos de instancias y sus casos de uso, consulte Instancias de Amazon EC2.

        El escalado administrado funciona mejor si el grupo de Auto Scaling utiliza los mismos tipos de instancia o similares.
      - Para Par de clave de SSH, elija el par que demuestre su identidad cuando se conecta a la instancia.
      - Para Capacity (Capacidad), introduzca el número mínimo y el número máximo de instancias que va a lanzar en el grupo de Auto Scaling. Las instancias de Amazon EC2

generan costos mientras existan en los recursos de AWS. Para obtener más información, consulte Precios de Amazon EC2.

8. (Opcional) Para activar Container Insights, expanda Monitoring (Supervisión) y, a continuación, active Use Container Insights (Uso de Container Insights).
9. (Opcional) Para administrar las etiquetas de clúster, expanda Tags (Etiquetas) y, a continuación, realice una de las siguientes operaciones:

   [Añadir una etiqueta] Elija Add tag (Añadir etiqueta) y haga lo siguiente:

   - En Key (Clave), escriba el nombre de la clave.
   - En Valor, escriba el valor de la clave.

   [Eliminar una etiqueta] Elija Eliminar a la derecha de la clave y el valor de la etiqueta.
10. Seleccione Create (Crear).

Paso 2: Registrar una definición de tareas

Para registrar la definición de tarea de muestra con la AWS Management Console

1. En el panel de navegación, elija Task Definitions.
2. Elija Create new task definition (Crear nueva definición de tarea) y Create new task definition with JSON (Crear nueva definición de tarea con JSON).
3. Copie y pegue la siguiente definición de tarea de ejemplo en el cuadro y, a continuación, elija Guardar.

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html
 -Type file -Value '<html> <head> <title>Amazon ECS Sample App</title>
 <style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
 <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p>'; C:\\ServiceMonitor.exe w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
```

```
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 443,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["EC2"]
}
```

4. Verifique su información y seleccione Create (Crear).

Paso 3: Crear un servicio

Un servicio Amazon ECS le ayuda a ejecutar y mantener simultáneamente un número determinado de instancias de una definición de tareas en un clúster de Amazon ECS. En caso de que alguna de las tareas falle o se detenga por algún motivo, el programador de servicio de Amazon ECS lanza otra instancia de la definición de tarea para sustituirla y mantener el recuento deseado de tareas en el servicio. Para obtener más información sobre los servicios, consulte Servicios de Amazon ECS.

Para crear un servicio

1. En el panel de navegación, seleccione Clusters (Clústeres).
2. Seleccione el clúster que creó en Paso 1: creación de un clúster.
3. En la pestaña Services (Servicios), elija Create (Crear).
4. En la sección Environment (Entorno), haga lo siguiente:

   a. Para Compute options, (Opciones de procesamiento), elija Launch type (Tipo de lanzamiento).
   b. En Launch type (Tipo de lanzamiento), seleccione EC2
5. En la sección Deployment configuration (Configuración de implementación), haga lo siguiente:
   a. En Family (Familia), elija la definición de tarea que creó en Paso 2: Registrar una definición de tareas.
   b. En Service name (Nombre del servicio), ingrese un nombre para el servicio.
   c. En Desired tasks (Tareas deseadas), ingrese 1.
6. Revise las opciones y elija Crear.
7. Elija View service (Ver servicio) para revisar el servicio.

Paso 4: Ver el servicio

El servicio es una aplicación web para que pueda ver sus contenedores con un navegador web.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. Elija el clúster en el que ejecutó el servicio.
4. En la pestaña Servicios, en Nombre del servicio, elija el servicio que creó en Paso 3: Crear un servicio.
5. Seleccione la pestaña Tareas y, a continuación, elija la tarea de su servicio.
6. En la página de tareas, en la sección Configuración, en IP pública, elija Dirección abierta. En la captura de pantalla siguiente se muestra el resultado esperado.

Paso 5: Eliminación

Cuando termine de utilizar un clúster de Amazon ECS, debe limpiar los recursos asociados para evitar que se generen cargos por recursos que no está utilizando.

Algunos recursos de Amazon ECS, tales como tareas, servicios, clústeres e instancias de contenedor, se limpian a través de la consola de Amazon ECS. Otros recursos como, por ejemplo, las instancias de Amazon EC2, los balanceadores de carga de Elastic Load Balancing y los grupos de Auto Scaling, se deben limpiar manualmente en la consola de Amazon EC2 o eliminando la pila de AWS CloudFormation que los creó.

1. En el panel de navegación, seleccione Clusters (Clústeres).
2. En la página Clústeres, seleccione el clúster de clúster que creó para este tutorial.
3. Seleccione la pestaña Servicios.
4. Seleccione el servicio y, a continuación, elija Eliminar.
5. En la pregunta de confirmación, escriba delete (eliminar) y, a continuación, elija Delete (Eliminar).

   Espere hasta que se elimine el servicio.
6. Elija Delete cluster. En la pregunta de confirmación, ingrese delete ***`cluster-name`*** (eliminar nombre de clúster) y, a continuación, elija Delete (Eliminar). Al eliminar el clúster, se limpian los recursos asociados que se crearon con él, incluidos los grupos de Auto Scaling, las VPC o los balanceadores de carga.

Creación de recursos de Amazon ECS con AWS CDK

AWS Cloud Development Kit (AWS CDK) es un marco de infraestructura como código (IAC) que le permite definir la infraestructura de nube de AWS mediante un lenguaje de programación de su elección. Para definir su propia infraestructura de nube, primero escriba una aplicación (en uno de los lenguajes compatibles con CDK) que contenga una o más pilas. Luego, sintetícela en una plantilla de AWS CloudFormation e implemente sus recursos en suCuenta de AWS. Siga los pasos de este tema para implementar un servidor web en contenedores con Amazon Elastic Container Service (Amazon ECS) y el AWS CDK en Fargate.

La biblioteca de construcción de AWS, incluida con el CDK, ofrece módulos que puede usar para modelar los recursos proporcionados por Servicios de AWS. Para los servicios más populares, la biblioteca proporciona construcciones seleccionadas con valores predeterminados inteligentes y

prácticas recomendadas. Uno de estos módulos, `aws-ecs-patterns` en particular, proporciona abstracciones de alto nivel que le permiten definir su servicio en contenedores y todos los recursos de soporte necesarios en solo unas pocas líneas de código.

Este tema emplea la construcción `ApplicationLoadBalancedFargateService`. Esta construcción implementa un servicio Amazon ECS en Fargate detrás de un equilibrador de carga de aplicación. El módulo `aws-ecs-patterns` también incluye construcciones que usan un equilibrador de carga de red y se ejecutan en Amazon EC2.

Antes de comenzar esta tarea, configure su entorno de desarrollo AWS CDK y ejecute el comando siguiente para instalar el AWS CDK. Para obtener instrucciones sobre cómo configurar su entorno de desarrollo AWS CDK, consulte Getting Started With the AWS CDK - Prerequisites (Primeros pasos con CDK: requisitos previos).

```
npm install -g aws-cdk
```

> **Note**
>
> En estas instrucciones, se presupone que está utilizando AWS CDK v2.

Temas

- Paso 1: Configuración del proyecto AWS CDK
- Paso 2: usar el AWS CDK para definir un servidor web en contenedores en Fargate
- Paso 3: probar el servidor web
- Paso 4: Limpiar
- Pasos a seguir a continuación

Paso 1: Configuración del proyecto AWS CDK

Cree un directorio para la nueva aplicación AWS CDK e inicialice el proyecto.

TypeScript

```
mkdir hello-ecs
cd hello-ecs
```

```
cdk init --language typescript
```

JavaScript

```
mkdir hello-ecs
cd hello-ecs
cdk init --language javascript
```

Python

```
mkdir hello-ecs
cd hello-ecs
cdk init --language python
```

Después de inicializar el proyecto, active el entorno virtual del proyecto e instale las dependencias de referencia de AWS CDK.

```
source .venv/bin/activate
python -m pip install -r requirements.txt
```

Java

```
mkdir hello-ecs
cd hello-ecs
cdk init --language java
```

Importe este proyecto de Maven a su IDE de Java. Por ejemplo, en Eclipse, use Archivo > Importar > Maven > Proyectos de Maven existentes.

C#

```
mkdir hello-ecs
cd hello-ecs
cdk init --language csharp
```

Go

```
mkdir hello-ecs
cd hello-ecs
cdk init --language go
```

> **Note**
>
> La plantilla de la aplicación AWS CDK utiliza el nombre del directorio del proyecto para generar nombres para los archivos y las clases fuente. En este ejemplo, el directorio se llama `hello-ecs`. Si usa otro nombre de directorio de proyecto, la aplicación no coincidirá con estas instrucciones.

AWS CDK v2 incluye construcciones estables para todos los Servicios de AWS en un solo paquete denominado `aws-cdk-lib`. Este paquete se instala como dependencia cuando inicializa el proyecto. Al trabajar con ciertos lenguajes de programación, el paquete se instala al crear el proyecto por primera vez. Este tema abarca cómo usar una construcción Amazon ECS Patterns, la cual proporciona abstracciones de alto nivel para trabajar con Amazon ECS. Este módulo se basa en construcciones de Amazon ECS y en otras para aprovisionar los recursos necesarios para su aplicación Amazon ECS.

Los nombres que usa para importar estas bibliotecas a su aplicación CDK pueden diferir ligeramente según el lenguaje de programación que use. Como referencia, estos son los nombres usados en cada lenguaje de programación CDK compatible.

TypeScript

```
aws-cdk-lib/aws-ecs
aws-cdk-lib/aws-ecs-patterns
```

JavaScript

```
aws-cdk-lib/aws-ecs
aws-cdk-lib/aws-ecs-patterns
```

Python

```
aws_cdk.aws_ecs
aws_cdk.aws_ecs_patterns
```

Java

```
software.amazon.awscdk.services.ecs
```

```
software.amazon.awscdk.services.ecs.patterns
```

C#

```
Amazon.CDK.AWS.ECS
Amazon.CDK.AWS.ECS.Patterns
```

Go

```
github.com/aws/aws-cdk-go/awscdk/v2/awsecs
github.com/aws/aws-cdk-go/awscdk/v2/awsecspatterns
```

Paso 2: usar el AWS CDK para definir un servidor web en contenedores en Fargate

Utilice la imagen del contenedor `amazon-ecs-sample`. Esta imagen contiene una aplicación web PHP que se ejecuta en Ngingx.

En el proyecto AWS CDK que ha creado, edite el archivo que contiene la definición de pila para que se parezca a uno de los siguientes ejemplos.

> **Note**
>
> Una pila es una unidad de implementación. Todos los recursos deben estar en una pila y todos los recursos que están en una pila se implementan al mismo tiempo. Si un recurso no se puede implementar, se restaurarán todos los otros recursos ya implementados. Una aplicación AWS CDK puede contener varias pilas y los recursos de una pila pueden hacer referencia a los recursos de otra.

TypeScript

Actualice `lib/hello-ecs-stack.ts` para que se parezca a lo siguiente.

```
import * as cdk from 'aws-cdk-lib';
import { Construct } from 'constructs';
import * as ecs from 'aws-cdk-lib/aws-ecs';
```

```
import * as ecsp from 'aws-cdk-lib/aws-ecs-patterns';

export class HelloEcsStack extends cdk.Stack {
  constructor(scope: Construct, id: string, props?: cdk.StackProps) {
    super(scope, id, props);

    new ecsp.ApplicationLoadBalancedFargateService(this, 'MyWebServer', {
      taskImageOptions: {
        image: ecs.ContainerImage.fromRegistry('public.ecr.aws/ecs-sample-image/
amazon-ecs-sample:latest'),
      },
      publicLoadBalancer: true
    });
  }
}
```

JavaScript

Actualice `lib/hello-ecs-stack.js` para que se parezca a lo siguiente.

```
const cdk = require('aws-cdk-lib');
const { Construct } = require('constructs');
const ecs = require('aws-cdk-lib/aws-ecs');
const ecsp = require('aws-cdk-lib/aws-ecs-patterns');

class HelloEcsStack extends cdk.Stack {
  constructor(scope = Construct, id = string, props = cdk.StackProps) {
    super(scope, id, props);

    new ecsp.ApplicationLoadBalancedFargateService(this, 'MyWebServer', {
      taskImageOptions: {
        image: ecs.ContainerImage.fromRegistry('amazon/amazon-ecs-sample'),
      },
      publicLoadBalancer: true
    });
  }
}

module.exports = { HelloEcsStack }
```

Python

Actualice `hello-ecs/hello_ecs_stack.py` para que se parezca a lo siguiente.

```
import aws_cdk as cdk
from constructs import Construct

import aws_cdk.aws_ecs as ecs
import aws_cdk.aws_ecs_patterns as ecsp

class HelloEcsStack(cdk.Stack):

    def __init__(self, scope: Construct, construct_id: str, **kwargs) -> None:
        super().__init__(scope, construct_id, **kwargs)

        ecsp.ApplicationLoadBalancedFargateService(self, "MyWebServer",
            task_image_options=ecsp.ApplicationLoadBalancedTaskImageOptions(
                image=ecs.ContainerImage.from_registry("amazon/amazon-ecs-sample")),
            public_load_balancer=True
        )
```

Java

Actualice `src/main/java/com.myorg/HelloEcsStack.java` para que se parezca a lo siguiente.

```
package com.myorg;

import software.constructs.Construct;
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;

import software.amazon.awscdk.services.ecs.ContainerImage;
import
 software.amazon.awscdk.services.ecs.patterns.ApplicationLoadBalancedFargateService;
import
 software.amazon.awscdk.services.ecs.patterns.ApplicationLoadBalancedTaskImageOptions;

public class HelloEcsStack extends Stack {
    public HelloEcsStack(final Construct scope, final String id) {
        this(scope, id, null);
    }

    public HelloEcsStack(final Construct scope, final String id, final StackProps
 props) {
        super(scope, id, props);
```

```
        ApplicationLoadBalancedFargateService.Builder.create(this, "MyWebServer")
         .taskImageOptions(ApplicationLoadBalancedTaskImageOptions.builder()
           .image(ContainerImage.fromRegistry("amazon/amazon-ecs-sample"))
           .build())
         .publicLoadBalancer(true)
         .build();
    }
}
```

C#

Actualice `src/HelloEcs/HelloEcsStack.cs` para que se parezca a lo siguiente.

```
using Amazon.CDK;
using Constructs;
using Amazon.CDK.AWS.ECS;
using Amazon.CDK.AWS.ECS.Patterns;
namespace HelloEcs
{
    public class HelloEcsStack : Stack
    {
        internal HelloEcsStack(Construct scope, string id, IStackProps props =
 null) : base(scope, id, props)
        {
            new ApplicationLoadBalancedFargateService(this, "MyWebServer",
                new ApplicationLoadBalancedFargateServiceProps
                {
                    TaskImageOptions = new ApplicationLoadBalancedTaskImageOptions
                    {
                        Image = ContainerImage.FromRegistry("amazon/amazon-ecs-
sample")
                    },
                    PublicLoadBalancer = true
                });
        }
    }
}
```

Go

Actualice `hello-ecs.go` para que se parezca a lo siguiente.

```
package main
```

```
import (
 "github.com/aws/aws-cdk-go/awscdk/v2"
 // "github.com/aws/aws-cdk-go/awscdk/v2/awssqs"
 "github.com/aws/aws-cdk-go/awscdk/v2/awsecs"
 "github.com/aws/aws-cdk-go/awscdk/v2/awsecspatterns"
 "github.com/aws/constructs-go/constructs/v10"
 "github.com/aws/jsii-runtime-go"
)

type HelloEcsStackProps struct {
 awscdk.StackProps
}

func NewHelloEcsStack(scope constructs.Construct, id string, props
 *HelloEcsStackProps) awscdk.Stack {
 var sprops awscdk.StackProps
 if props != nil {
  sprops = props.StackProps
 }
 stack := awscdk.NewStack(scope, &id, &sprops)

 // The code that defines your stack goes here

 // example resource
 // queue := awssqs.NewQueue(stack, jsii.String("HelloEcsQueue"),
 &awssqs.QueueProps{
 //  VisibilityTimeout: awscdk.Duration_Seconds(jsii.Number(300)),
 // })
 res := awsecspatterns.NewApplicationLoadBalancedFargateService(stack,
 jsii.String("MyWebServer"),
  &awsecspatterns.ApplicationLoadBalancedFargateServiceProps{
   TaskImageOptions: &awsecspatterns.ApplicationLoadBalancedTaskImageOptions{
    Image: awsecs.ContainerImage_FromRegistry(jsii.String("amazon/amazon-ecs-
sample"), &awsecs.RepositoryImageProps{}),
   },
  },
 )
 awscdk.NewCfnOutput(stack, jsii.String("LoadBalancerDNS"),
 &awscdk.CfnOutputProps{Value: res.LoadBalancer().LoadBalancerDnsName()})

 return stack
}
```

```
func main() {
 defer jsii.Close()

 app := awscdk.NewApp(nil)

 NewHelloEcsStack(app, "HelloEcsStack", &HelloEcsStackProps{
  awscdk.StackProps{
   Env: env(),
  },
 })

 app.Synth(nil)
}

// env determines the AWS environment (account+region) in which our stack is to
// be deployed. For more information see: https://docs.aws.amazon.com/cdk/latest/
guide/environments.html
func env() *awscdk.Environment {
 // If unspecified, this stack will be "environment-agnostic".
 // Account/Region-dependent features and context lookups will not work, but a
 // single synthesized template can be deployed anywhere.
 //---------------------------------------------------------------------------
 return nil

 // Uncomment if you know exactly what account and region you want to deploy
 // the stack to. This is the recommendation for production stacks.
 //---------------------------------------------------------------------------
 // return &awscdk.Environment{
 //  Account: jsii.String("123456789012"),
 //  Region:  jsii.String("us-east-1"),
 // }

 // Uncomment to specialize this stack for the AWS Account and Region that are
 // implied by the current CLI configuration. This is recommended for dev
 // stacks.
 //---------------------------------------------------------------------------
 // return &awscdk.Environment{
 //  Account: jsii.String(os.Getenv("CDK_DEFAULT_ACCOUNT")),
 //  Region:  jsii.String(os.Getenv("CDK_DEFAULT_REGION")),
 // }
}
```

El fragmento breve anterior incluye lo siguiente:

- El nombre lógico del servicio: `MyWebServer`.
- La imagen del contenedor que se obtuvo de la galería pública de Amazon ECR: `amazon/amazon-ecs-sample`.
- Otra información relevante, como el hecho de que el equilibrador de carga tiene una dirección pública y se puede acceder a él desde Internet.

La AWS CDK creará todos los recursos necesarios para implementar el servidor web, incluidos los siguientes recursos. Estos recursos se omitieron en este ejemplo.

- Clúster de Amazon ECS
- Instancias de Amazon VPC y Amazon EC2
- Grupo de escalado automático
- Equilibrador de carga de aplicación
- Roles y políticas de IAM

Algunos recursos aprovisionados automáticamente se compartirán con todos los servicios de Amazon ECS definidos en la pila.

Guarde el archivo fuente y a continuación ejecute el comando `cdk synth` en el directorio principal de la aplicación. El AWS CDK ejecuta la aplicación, sintetiza una plantilla de AWS CloudFormation a partir de ella y luego muestra la plantilla. La plantilla es un archivo YAML de aproximadamente 600 líneas. Aquí se muestra el principio del archivo. Su plantilla puede diferir de este ejemplo.

```
Resources:
  MyWebServerLB3B5FD3AB:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      LoadBalancerAttributes:
        - Key: deletion_protection.enabled
          Value: "false"
      Scheme: internet-facing
      SecurityGroups:
        - Fn::GetAtt:
            - MyWebServerLBSecurityGroup01B285AA
            - GroupId
      Subnets:
        - Ref: EcsDefaultClusterMnL3mNNYNVpcPublicSubnet1Subnet3C273B99
```

```
        - Ref: EcsDefaultClusterMnL3mNNYNVpcPublicSubnet2Subnet95FF715A
      Type: application
    DependsOn:
      - EcsDefaultClusterMnL3mNNYNVpcPublicSubnet1DefaultRouteFF4E2178
      - EcsDefaultClusterMnL3mNNYNVpcPublicSubnet2DefaultRouteB1375520
    Metadata:
      aws:cdk:path: HelloEcsStack/MyWebServer/LB/Resource
  MyWebServerLBSecurityGroup01B285AA:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Automatically created Security Group for ELB
 HelloEcsStackMyWebServerLB06757F57
      SecurityGroupIngress:
        - CidrIp: 0.0.0.0/0
          Description: Allow from anyone on port 80
          FromPort: 80
          IpProtocol: tcp
          ToPort: 80
      VpcId:
        Ref: EcsDefaultClusterMnL3mNNYNVpc7788A521
    Metadata:
      aws:cdk:path: HelloEcsStack/MyWebServer/LB/SecurityGroup/Resource
# and so on for another few hundred lines
```

Para implementar el servicio en su Cuenta de AWS, ejecute el comando `cdk deploy` en el directorio principal de la aplicación. Se le pedirá que apruebe las políticas de IAM que ha generado el AWS CDK.

La implementación lleva varios minutos, durante los cuales el AWS CDK crea varios recursos. Las últimas líneas del resultado de la implementación incluyen el nombre de host público del equilibrador de carga y la dirección URL de su nuevo servidor web. Son los siguientes:

```
Outputs:
HelloEcsStack.MyWebServerLoadBalancerDNSXXXXXXX = Hello-MyWeb-ZZZZZZZZZZZZZ-
ZZZZZZZZZZ.us-west-2.elb.amazonaws.com
HelloEcsStack.MyWebServerServiceURLYYYYYYYY = http://Hello-MyWeb-ZZZZZZZZZZZZZ-
ZZZZZZZZZZ.us-west-2.elb.amazonaws.com
```

Paso 3: probar el servidor web

Copie la dirección URL del resultado de la implementación y péguela en el navegador web. Se muestra el siguiente mensaje de bienvenida del servidor web.

Paso 4: Limpiar

Cuando haya terminado con el servidor web, finalice el servicio mediante la CDK. Para ello, ejecute el comando `cdk destroy` en el directorio principal de la aplicación. Esto evita que incurra en cargos imprevistos en el futuro.

Pasos a seguir a continuación

Para obtener más información sobre cómo desarrollar la infraestructura de AWS mediante AWS CDK, consulte la Guía para desarrolladores de AWS CDK.

Para obtener información sobre cómo escribir aplicaciones de AWS CDK en el lenguaje de su elección, consulte:

TypeScript

- Utilización del AWS CDK en TypeScript

JavaScript

- Utilización del AWS CDK en JavaScript

Python

- Utilización del AWS CDK en Python

Java

- Utilización del AWS CDK en Java

C#

Utilización del AWS CDK en C#

Go

Uso del AWS CDK en Go

Para obtener más información sobre los módulos de la biblioteca de construcción de AWS que se usan en este tema, consulte la siguiente información general de la Referencia de la API de AWS CDK.

- aws-ecs
- aws-ecs-patterns

Creación de recursos de Amazon ECS mediante la interfaz de la línea de comandos de AWS Copilot

La interfaz de línea de comandos (CLI) de AWS Copilot simplifican la creación, el lanzamiento y la operación de aplicaciones en contenedores listas para producción en Amazon ECS desde un entorno de desarrollo local. La CLI de AWS Copilot se alinea con los flujos de trabajo de los desarrolladores que admiten prácticas recomendadas de aplicaciones modernas: desde la utilización de infraestructura como código hasta la creación de una canalización CI/CD aprovisionada en nombre de un usuario. Utilice la CLI de AWS Copilot en los ciclos de pruebas y desarrollos cotidianos como alternativa a la AWS Management Console.

En la actualidad, AWS Copilot es compatible con los sistemas Linux, macOS y Windows. Para obtener más información acerca de la versión más reciente de la CLI de AWS Copilot, consulte Versiones.

> **Note**
>
> El código fuente de la CLI de AWS Copilot está disponible en GitHub. Le recomendamos informar problemas y enviar solicitudes de extracción para los cambios que le gustaría que incluyamos. No obstante, en la actualidad, Amazon Web Services no admite la ejecución de copias modificadas del código de AWS Copilot. Para informar los problemas de AWS Copilot, contáctenos en Gitter o GitHub, donde puede plantear problemas, brindar comentarios e informar errores.

Para obtener información sobre el desarrollo de Copilot, consulte What's the future of Copilot?.

Temas

- Instalación de la CLI de AWS Copilot
- Implementación de una aplicación de Amazon ECS de ejemplo mediante la CLI de AWS Copilot

La documentación adicional de la CLI de AWS Copilot está disponible en el sitio web de AWS Copilot

Instalación de la CLI de AWS Copilot

Puede instalar la CLI de AWS Copilot mediante Homebrew o la descarga manual del archivo binario con los siguientes pasos.

Uso de Homebrew

El siguiente comando se utiliza para instalar la CLI de AWS Copilot en el sistema macOS o Linux mediante Homebrew. Antes de la instalación, debe tener instalado Homebrew. Para obtener más información, consulte Homebrew.

```
brew install aws/tap/copilot-cli
```

Descarga del archivo binario

Como alternativa a Homebrew, puede instalar manualmente la CLI de AWS Copilot en el sistema macOS, Windows o Linux. Utilice el comando siguiente para su sistema operativo para descargar el archivo binario. Los ejemplos de macOS y Linux también incluyen comandos que aplican permisos de ejecución al archivo binario y muestran el menú de ayuda para comprobar que la instalación funciona.

macOS

Para macOS:

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-darwin \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Para sistemas ARM de macOS:

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-darwin-arm64 \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Linux

Para sistemas Linux x86 (64 bits):

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-linux \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Para sistemas Linux ARM:

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-linux-arm64 \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Windows

O ejecute el siguiente comando de PowerShell:

```
New-Item -Path 'C:\copilot' -ItemType directory; `
  Invoke-WebRequest -OutFile 'C:\copilot\copilot.exe' https://github.com/aws/
copilot-cli/releases/latest/download/copilot-windows.exe
```

(Opcional) Verifique la CLI de AWS Copilot instalada manualmente mediante firmas PGP

Los ejecutables de la CLI de AWS Copilot están firmados criptográficamente mediante firmas PGP. Las firmas PGP se pueden utilizar para verificar la validez del ejecutable de la CLI de AWS Copilot. Siga estos pasos para verificar las firmas mediante la herramienta GnuPG.

1. Descargue e instale GnuPG. Para obtener más información, consulte el sitio web de GnuPG.

macOS

Recomendamos utilizar Homebrew. Instale Homebrew siguiendo las instrucciones de su sitio web. Para obtener más información, consulte Homebrew. Una vez que haya instalado Homebrew, utilice el siguiente comando desde el terminal de macOS.

```
brew install gnupg
```

Linux

Instale gpg utilizando el administrador de paquetes de su versión de Linux.

Windows

Descargue el instalador sencillo de Windows desde el sitio web de GnuPG e instálelo como administrador. Después de instalar GnuPG, cierre y vuelva a abrir el Administrador PowerShell.

Para obtener más información, consulte GnuPG Download.

2. Verifique que la ruta de GnuPG se haya agregado a la ruta de su entorno.

macOS

```
echo $PATH
```

Si no ve la ruta de GnuPG en la salida, ejecute el siguiente comando para agregarlo a la ruta.

```
PATH=$PATH:<path to GnuPG executable files>
```

Linux

```
echo $PATH
```

Si no ve la ruta de GnuPG en la salida, ejecute el siguiente comando para agregarlo a la ruta.

```
export PATH=$PATH:<path to GnuPG executable files>
```

Windows

```
Write-Output $Env:PATH
```

Si no ve la ruta de GnuPG en la salida, ejecute el siguiente comando para agregarlo a la ruta.

```
$Env:PATH += ";<path to GnuPG executable files>"
```

3. Cree un archivo de texto sin formato local.

macOS

En la terminal, introduzca:

```
touch <public_key_filename.txt>
```

Abra el archivo en con TextEdit.

Linux

Cree un archivo de texto en un editor de texto como gedit. Guardar como `public_key_filename.txt`

Windows

Cree un archivo de texto en un editor de texto como el Bloc de notas. Guardar como `public_key_filename.txt`

4. Agregue el siguiente contenido de la clave pública PGP de Amazon ECS y guarde el archivo.


```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2

mQINBFq1SasBEADliGcT1NVJ1ydfN8DqebYYe9ne3dt6jqKFmKowLmm6LLGJe7HU
jGtqhCWRDkN+qPpHqdArRgDZAtn2pXY5fEipHgar4CP8QgRnRMO2fl74lmavr4Vg
7K/KH8VHlq2uRw32/B94XLEgRbGTMdWFdKuxoPCttBQaMj3LGn6Pe+6xVWRkChQu
BoQAhjBQ+bEm0kNy0LjNgjNlnL3UMAG56t8E3LANIgGgEnpNsB1UwfWluPoGZoTx
N+6pHBJrKIL/1v/ETU4FXpYw2zvhWNahxeNRnoYj3uycHkeliCrw4kj0+skizBgO
2K7oVX8Oc3j5+ZilhL/qDLXmUCb2az5cMM1mOoF8EKX5HaNuq1KfwJxqXE6NNIcO
lFTrT7QwD5fMNld3FanLgv/ZnIrsSaqJOL6zRSq8O4LN1OWBVbndExk2Kr+5kFxn
5lBPgfPgRj5hQ+KTHMa9Y8Z7yUc64BJiN6F9Nl7FJuSsfqbdkvRLsQRbcBG9qxX3
rJAEhieJzVMEUNl+EgeCkxj5xuSkNU7zw2c3hQZqEcrADLV+hvFJktOz9Gm6xzbq
```

```
lTnWWCz4xrIWtuEBA2qE+MlDheVd78a3gIsEaSTfQq0osYXaQbvlnSWOoc1y/5Zb
zizHTJIhLtUyls9WisP2s0emeHZicVMfW61EgPrJAiupgc7kyZvFt4YwfwARAQAB
tCRBbWF6b24gRUNTIDxlY3Mtc2VjdXJpdHlAYW1hem9uLmNvbT6JAhwEEAECAAYF
AlrjL0YACgkQHivRXs0TaQrg1g/+JppwPqHnlVPmv7lessB8I5UqZeD6p6uVpHd7
Bs3pcPp8BV7BdRbs3sPLt5bV1+rkqOlw+0gZ4Q/ue/YbWtOAt4qY0OcEo0HgcnaX
lsB827QIfZIVtGWMhuh94xzm/SJkvngml6KB3YJNnWP61A9qJ37/VbVVLzvcmazA
McWB4HUMNrhd0JgBCo0gIpqCbpJEvUc02Bjn23eEJsS9kC7OUAHyQkVnx4d9UzXF
4OoISF6hmQKIBoLnRrAlj5Qvs3GhvHQ0ThYq0Grk/KMJJX2CSqt7tWJ8gk1n3H3Y
SReRXJRnv7DsDDBwFgT6r5Q2HW1TBUvaoZy5hF6maD09nHcNnvBjqADzeT8Tr/Qu
bBCLzkNSYqqkpgtwv7seoD2P4n1giRvDAOEfMZpVkUr+C252IaH1HZFEz+TvBVQM
Y8OWWxmIJW+J6evjo3N1eO19UHv71jvoF8zljbI4bsL2c+QTJmOv7nRqzDQgCWyp
Id/v2dUVVTk1j9omuLBBwNJzQCB+72LcIzJhYmaP1HC4LcKQG+/f41exuItenatK
lEJQhYtyVXcBlh6Yn/wzNg2NWOwb3vqY/F7m6u9ixAwgtIMgPCDE4aJ86zrrXYFz
N2HqkTSQh77Z8KPKmyGopsmN/reMuilPdINb249nA0dzoN+nj+tTFOYCIaLaFyjs
Z0r1QAOJAjkEEwECACMFAlq1SasCGwMHCwkIBwMCAQYVCAIJCgsEFgIDAQIeAQIX
gAAKCRC86dmkLVF4T9iFEACEnkm1dNXsWUx34R3c0vamHrPxvfkyI1FlEUen8D1h
uX9xy6jCEROHWEp0rjGK4QDPgM93sWJ+s1UAKg214QRVzft0y9/DdR+twApA0fzy
uavIthGd6+03jAAo6udYDE+cZC3P7XBbDiYEWk4XAF9I1JjB8hTZUgvXBL046JhG
eM17+crgUyQeetkiOQemLbsbXQ40Bd9V7zf7XJraFd8VrwNUwNb+9KFtgAsc9rk+
YIT/PEf+YOPysgcxI4sTWghtyCulVnuGoskgDv4v73PALU0ieUrvvQVqWMRvhVx1
0X90J7cC1KOyhlEQQ1aFTgmQjmXexVTwIBm8LvysFK6YXM41KjOrlz3+6xBIm/qe
bFyLUnf4WoiuOplAaJhK9pRY+XEnGNxdtN4D26Kd0F+PLkm3Tr3Hy3b1Ok34FlGr
KVHUq1TZD7cvMnnNKEELTUcKX+1mV3an16nmAg/my1JSUt6BNK2rJpY1s/kkSGSE
XQ4zuF2IGCpvBFhYAlt5Un5zwqkwwQR3/n2kwAoDzonJcehDw/C/cGos5D0aIU7I
K2X2aTD3+pA7Mx3IMe2hqmYqRt9X42yF1PIEVRneBRJ3HDezAgJrNh0GQWRQkhIx
gz6/cTR+ekr5TptVszS9few2GpI5bCgBKBisZIssT89aw7mAKWut0Gcm4qM9/yK6
1bkCDQRatUmrARAAxNPvVwreJ2yAiFcUpdRlVhsuOgnxvs1QgsIw3H7+Pacr9Hpe
8uftYZqdC82KeSKhpHq7c8gMTMucIINtH25x9BCc73E33EjCL9Lqov1TL7+QkgHe
T+JIhZwdD8Mx2K+LVVVu/aWkNrfMuNwyDUciSI4D5QHa8T+F8fgN4OTpwYjirzel
5yoICMr9hVcbzDNv/ozKCxjx+XKgnFc3wrnDfJfntfDAT7ecwbUTL+viQKJ646s+
psiqXRYtVvYInEhLVrJ0aV6zHFoigE/Bils6/g7ru1Q6CEHqEw++APs5CcE8VzJu
WAGSVHZgun5Y9N4quR/M9Vm+IPMhTxrAg7rOvyRN9cAXfeSMf77I+XTifigNna8x
t/MOdjXr1fjF4pThEi5u6WsuRdFwjY2azEv3vevodTi4HoJReH6dFRa6y8c+UDgl
2iHiOKIpQqLbHEfQmHcDd2fix+AaJKMnPGNku9qCFEMbgSRJpXz6BfwnY1QuKE+I
R6jA0frUNt2jhiGG/F8RceXzohaaC/Cx7LUCUFWc0n7z32C9/Dtj7I1PMOacdZzz
bjJzRKO/ZDv+UN/c9dwAkllzAyPMwGBkUaY68EBstnIliW34aWm6IiHhxioVPKSp
VJfyiXPO0EXqujtHLAeChfjcns3I12YshT1dv2PafG53fp33ZdzeUgsBo+EAEQEA
AYkCHwQYAQIACQUCWrVJqwIbDAAKCRC86dmkLVF4T+ZdD/9x/8APzgNJF3o3STrF
jvnV1ycyhWYGAeBJiu7wjsNWwzMFOv15tLjB7AqeVxZn+WKDD/mIOQ45OZvnYZuy
X7DR0JszaH9wrYTxZLVruAu+t6UL0y/XQ4L1GZ9QR6+r+7t1Mvbfy7BlHbvX/gYt
Rwe/uwdibI0CagEzyX+2D3kTOlHO5XThbXaNf8AN8zha91Jt2Q2UR2X5T6JcwtMz
FBvZnl3LSmZyE0EQehS2iUurU4uWOpGppuqVnbi0jbCvCHKgDGrqZ0smKNAQng54
F365W3g8AfY48s8XQwzmcliowYX9bT8PZiEi0J4QmQh0aXkpqZyFefuWeOL2R94S
XKzr+gRh3BAULoqF+qK+IUMxTip9KTPNvYDpiC66yBiT6gFDji5Ca9pGpJXrC3xe
```

```
TXiKQ8DBWDhBPVPrruLIaenTtZEOsPc4I85yt5U9RoPTStcOr34s3w5yEaJagt6S
Gc5r9ysjkfH6+6rbi1ujxMgROSqtqr+RyB+V9A5/OgtNZc8llK6u4UoOCde8jUUW
vqWKvjJB/Kz3u4zaeNu2ZyyHaOqOuH+TETcW+jsY9IhbEzqN5yQYGi4pVmDkY5vu
lXbJnbqPKpRXgM9BecV9AMbPgbDq/5LnHJJXg+G8YQOgp4lR/hC1TEFdIp5wM8AK
CWsENyt2o1rjgMXiZOMF8A5oBLkCDQRatUuSARAAr77kj7j2QR2SZeOSlFBvV7oS
mFeSNnz9xZssqrsm6bTwSHM6YLDwc7Sdf2esDdyzONETwqrVCg+FxgL8hmo9hS4c
rR6tmrP0mOmptr+xLLsKcaP7ogIXsyZnrEAEsvW8PnfayoiPCdc3cMCR/lTnHFGA
7EuR/XLBmi7Qg9tByVYQ5Yj5wB9V4B2yeCt3XtzPqeLKvaxl7PNelaHGJQY/xo+m
V0bndxf9IY+4oFJ4blD32WqvyxESo7vW6WBh7oqv3Zbm0yQrr8a6mDBpqLkvWwNI
3kpJR974tg5o5LfDu1BeeyHWPSGm4U/G4JB+JIG1ADy+RmoWEt4BqTCZ/knnoGvw
D5sTCxbKdmuOmhGyTssoG+3OOcGYHV7pWYPhazKHMPm201xKCjH1RfzRULzGKjD+
yMLT1I3AXFmLmZJXikAOlvE3/wgMqCXscbycbLjLD/bXIuFWo3rzoezeXjgi/DJx
jKBAyBTYO5nMcth1O9oaFd9d0HbsOUDkIMnsgGBE766Piro6MHo0T0rXl07Tp4pI
rwuSOsc6XzCzdImj0Wc6axS/HeUKRXWdXJwno5awTwXKRJMXGfhCvSvbcbc2Wx+L
IKvmB7EB4K3fmjFFE67yolmiw2qRcUBfygtH3eL5XZU28MiCpue8Y8GKJoBAUyvf
KeM1rO8Jm3iRAc5a/D0AEQEAAYkEPgQYAQIACQUCWrVLkgIbAgIpCRC86dmkLVF4
T8FdIAQZAQIABgUCWrVLkgAKCRDePL1hra+LjtHYD/9MucxdFe6bXO1dQR4tKhhQ
P0LRqy6zlBY9ILCLowNdGZdqorogUiUymgn3VhEhVtxTOoHcN7qOuM01PNsRnOeS
EYjf8Xrb1clzkD6xULwmOclTb9bBxnBc/4PFvHAbZW3QzusaZniNgkuxt6BTfloS
Of4inq71kjmGK+TlzQ6mUMQUg228NUQC+a84EPqYyAeY1sgvgB7hJBhYL0QAxhcW
6m20Rd8iEc6HyzJ3yCOCsKip/nRWAbf0OvfHfRBp0+m0ZwnJM8cPRFjOqqzFpKH9
HpDmTrC4wKP1+TL52LyEqNh4yZitXmZNV7giSRIkk0eDSko+bFy6VbMzKUMkUJK3
D3eHFAMkujmbfJmSMTJOPGn5SB1HyjCZNx6bhIIbQyEUB9gKCmUFaqXKwKpF6rj0
iQXAJxLR/shZ5Rk96VxzOphUl7T90m/PnUEEPwq8KsBhnMRgxa0RFidDP+n9fgtv
HLmrOqX9zBCVXh0mdWYLrWvmzQFWzG7AoE55fkf8nAEPsalrCdtaNUBHRXA0OQxG
AHMOdJQQvBsmqMvuAdjkDWpFu5y0My5ddU+hiUzUyQLjL5Hhd5LOUDdewlZgIw1j
xrEAUzDKetnemM8GkHxDgg8koev5frmShJuce7vSjKpCNg3EIJSgqMOPFjJuLWtZ
vjHeDNbJy6uNL65ckJy6WhGjEADS2WAW1D6Tfekkc21SsIXk/LqEpLMR/0g5OUif
wcEN1rS9IJXBwIy8MelN9qr5KcKQLmfdfBNEyyceBhyVl0MDyHOKC+7PofMtkGBq
13QieRHv5GJ8LB3fclqHV8pwTTo3Bc8z2g0TjmUYAN/ixETdReDoKavWJYSE9yoM
aaJu279ioVTrwpECse0XkiRyKToTjwOb73CGkBZZpJyqux/rmCV/fp4ALdSW8zbz
FJVORaivhoWwzjpfQKhwcU9lABXi2UvVm14v0AfeI7oiJPSU1zM4fEny4oiIBXlR
zhFNih1UjIu82X16mTm3BwbIga/s1fnQRGzyhqUIMii+mWra23EwjChaxpvjjcUH
5ilLc5Zq781aCYRygYQw+hu5nFkOH1R+Z50Ubxjd/aqUfnGIAX7kPMD3Lof4KldD
Q8ppQriUvxVo+4nPV6rpTy/PyqCLWDjkguHpJsEFsMkwajrAz0QNSAU5CJ0G2Zu4
yxvYlumHCEl7nbFrm0vIiA75Sa8KnywTDsyZsu3XcOcf3g+g1xWTpjJqy2bYXlqz
9uDOWtArWHOis6bq8l9RE6xr1RBVXS6uqgQIZFBGyq66b0dIq4D2JdsUvgEMaHbc
e7tBfeB1CMBdA64e9Rq7bFR7Tvt8gasCZYlNr3lydh+dFHIEkH53HzQe6l88HEic
+0jVnLkCDQRa55wJARAAyLya2Lx6gyoWoJN1a6740q3o8e9d4KggQOfGMTCflmeq
ivuzgN+3DZHN+9ty2KxXMtn0mhHBerZdbNJyjMNT1gAgrhPNB4HtXBXum2wS57WK
DNmade914L7FWTPAWBG2Wn448OEHTqsClICXXWy9IICgclAEyIq0Yq5mAdTEgRJS
Z8t4GpwtDL9gNQyFXaWQmDmkAsCygQMvhAlmu9xOIzQG5CxSnZFk7zcuL60k14Z3
Cmt49k4T/7ZU8goWi8tt+rU78/IL3J/fF9+1civ1OwuUidgfPCSvOUW1JojsdCQA
L+RZJcoXq7lfOFj/eNjeOSstCTDPfTCL+kThE6E5neDtbQHBYkEX1BRiTedsV4+M
```

```
ucgiTrdQFWKf89G72xdv8ut9AYYQ2BbEYU+JAYhUH8rYYui2dHKJIgjNvJscuUWb
+QEqJIRleJRhrO+/CHgMs4fZAkWF1VFhKBkcKmEjLn1f7EJJUUW84ZhKXjO/AUPX
1CHsNjziRceuJCJYox1cwsoq6jTE50GiNzcIxTn9xUc0UMKFeggNAFys1K+TDTm3
Bzo8H5ucjCUEmUm9lhkGwqTZgOlRX5eqPX+JBoSaObqhgqCa5IPinKRa6MgoFPHK
6sYKqroYwBGgZm6Js5chpNchvJMs/3WXNOEVg0J3z3vP0DMhxqWm+r+n9zlW8qsA
EQEAAYkEPgQYAQgACQUCWuecCQIbAgIpCRC86dmkLVF4T8FdIAQZAQgABgUCWuec
CQAKCRBQ3szEcQ5hr+ykD/4tOLRHFHXuKUcxgGaubUcVtsFrwBKma1cYjqaPms8u
6Sk0wfGRI32G/GhOrp0Ts/MOkbObq6VLTh8N5Yc/53MEl8zQFw9Y5AmRoW4PZXER
ujs5s7p4oR7xHMihMjCCBn1bvrR+34YPfgzTcgLiOEFHYT8UTxwnGmXOvNkMM7md
xD3CV5q6VAte8WKBo/220II3fcQlc9r/oWX4kXXkb0v9hoGwKbDJ1tzqTPrp/xFt
yohqnvImpnlz+Q9zXmbrWYL9/g8VCmW/NN2gju2G3Lu/TlFUWIT4v/5OPK6TdeNb
VKJO4+S8bTayqSG9CML1S57KSgCo5HUhQWeSNHI+fpe5oX6FALPT9JLDce8OZz1i
cZZ0MELP37mOOQun0AlmHm/hVzf0f311PtbzcqWaE51tJvgUR/nZFo6Ta3O5Ezhs
3VlEJNQ1Ijf/6DH87SxvAoRIARCuZd0qxBcDK0avpFzUtbJd24lRA3WJpkEiMqKv
RDVZkE4b6TW61f0o+LaVfK6E8oLpixegS4fiqC16mFrOdyRk+RJJfIUyz0WTDVmt
g0U1CO1ezokMSqkJ7724pyjr2xf/r9/sC6aOJwB/lKgZkJfC6NqL7TlxVA31dUga
LEOvEJTTE4gl+tYtfsCDvALCtqL0jduSkUo+RXcBItmXhA+tShW0pbS2Rtx/ixua
KohVD/0R4QxiSwQmICNtm9mw9ydIl1yjYXX5a9x4wMJracNY/LBybJPFnZnT4dYR
z4XjqysDwvvYZByaWoIe3QxjX84V6MlI2IdAT/xImu8gbaCI8tmyfpIrLnPKiR9D
VFYfGBXuAX7+HgPPSFtrHQONCALxxzlbNpS+zxt9r0MiLgcLyspWxSdmoYGZ6nQP
RO5Nm/ZVS+u2imPCRzNUZEMa+dlE6kHx0rS0dPiuJ4O7NtPeYDKkoQtNagspsDvh
cK7CSqAiKMq06UBTxqlTSRkm62eOCtcs3p3OeHu5GRZF1uzTET0ZxYkaPgdrQknx
ozjP5mC7X+45lcCfmcVt94TFNL5HwEUVJpmOgmzILCI8yoDTWzloo+i+fPFsXX4f
kynhE83mSEcr5VHFYrTY3mQXGmNJ3bCLuc/jq7ysGq69xiKmTlUeXFm+aojcRO5i
zyShIRJZ0GZfuzDYFDbMV9amA/YQGygLw//zP5ju5SW26dNxlf3MdFQE5JJ86rn9
MgZ4gcpazHEVUsbZsgkLizRp9imUiH8ymLqAXnfRGlU/LpNSefnvDFTtEIRcpOHc
bhayG0bk51Bd4mioOXnIsKy4j63nJXA27x5EVVHQ1sYRN8Ny4Fdr2tMAmj2O+X+J
qX2yy/UX5nSPU492e2CdZ1UhoU0SRFY3bxKHKB7SDbVeav+K5g==
=Gi5D
-----END PGP PUBLIC KEY BLOCK-----
```


Los detalles de la clave pública PGP de Amazon ECS como referencia:

```
Key ID: BCE9D9A42D51784F
Type: RSA
Size: 4096/4096
Expires: Never
User ID: Amazon ECS
Key fingerprint: F34C 3DDA E729 26B0 79BE AEC6 BCE9 D9A4 2D51 784F
```

5. Importe el archivo con la clave pública PGP de Amazon ECS a través del siguiente comando en el terminal.

```
gpg --import <public_key_filename.txt>
```

6. Descargue las firmas de la CLI de AWS Copilot. Las firmas son firmas PGP separadas en formato ASCII que se almacenan en archivos con la extensión `.asc`. El archivo de firmas tiene el mismo nombre que su archivo ejecutable correspondiente, al que se le añade `.asc`.

   macOS

   Para sistemas macOS, ejecute el siguiente comando.

   ```
   sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/
   download/copilot-darwin.asc
   ```

   Linux

   Para sistemas Linux x86 (64 bits), ejecute el siguiente comando.

   ```
   sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/
   download/copilot-linux.asc
   ```

   Para sistemas Linux ARM, ejecute el siguiente comando.

   ```
   sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/
   download/copilot-linux-arm64.asc
   ```

   Windows

   O ejecute el siguiente comando de PowerShell.

   ```
   Invoke-WebRequest -OutFile 'C:\copilot\copilot.asc' https://github.com/aws/
   copilot-cli/releases/latest/download/copilot-windows.exe.asc
   ```

7. Verifique la firma mediante el siguiente comando.

   - Para sistemas macOS y Linux:

     ```
     gpg --verify copilot.asc /usr/local/bin/copilot
     ```

   - Para sistemas Windows:

```
gpg --verify 'C:\copilot\copilot.asc' 'C:\copilot\copilot.exe'
```

Resultado previsto:

```
gpg: Signature made Tue Apr  3 13:29:30 2018 PDT
gpg:                using RSA key DE3CBD61ADAF8B8E
gpg: Good signature from "Amazon ECS <ecs-security@amazon.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
     Subkey fingerprint: EB3D F841 E2C9 212A 2BD4  2232 DE3C BD61 ADAF 8B8E
```

> **Important**
>
> La advertencia en la salida se espera y no es problemática. Esto se produce porque no hay una cadena de confianza entre su clave PGP personal (si tiene una) y la clave PGP de Amazon ECS. Para obtener más información, consulta Red de confianza.

8. En las instalaciones de Windows, ejecute el siguiente comando en Powershell para agregar el directorio de AWS Copilot a la ruta.

```
$Env:PATH += ";<path to Copilot executable files>"
```

Implementación de una aplicación de Amazon ECS de ejemplo mediante la CLI de AWS Copilot

Tras instalar la CLI de AWS Copilot, puede seguir estos pasos para implementar una aplicación de ejemplo, comprobar la implementación y eliminar los recursos.

Requisitos previos

Antes de comenzar, asegúrese de que cumple los siguientes requisitos previos:

- Instalar y configurar la AWS CLI. Para obtener más información, consulte AWS Command Line Interface.

- Ejecute `aws configure` para configurar el perfil predeterminado que va a utilizar la CLI de AWS Copilot para administrar la aplicación y los servicios.
- Instale y ejecute Docker. Si necesita más información, consulte Introducción a Docker.

Implementación de una aplicación de Amazon ECS de ejemplo con un solo comando

1. Implemente una aplicación web de ejemplo clonada desde un repositorio de GitHub mediante el comando siguiente. Para obtener información sobre `init` de AWS Copilot y sus marcas, consulte la documentación de AWS Copilot.

```
git clone https://github.com/aws-samples/aws-copilot-sample-service.git demo-app &&
 \
cd demo-app &&                                \
copilot init --app demo                       \
  --name api                                  \
  --type 'Load Balanced Web Service'          \
  --dockerfile './Dockerfile'                 \
  --port 80                                   \
  --tag  latest                               \
  --deploy
```

2. Una vez completada la implementación, la CLI de AWS Copilot mostrará una URL que puede utilizar para comprobar la implementación. También puede utilizar los siguientes comandos para comprobar el estado de la aplicación.

   - Enumere todas las aplicaciones de AWS Copilot.

```
copilot app ls
```

   - Muestre información sobre los entornos y servicios de la aplicación.

```
copilot app show
```

   - Muestre información acerca de los entornos.

```
copilot env ls
```

   - Muestre información acerca del servicio, incluidos los puntos de enlace, la capacidad y los recursos relacionados.

```
copilot svc show
```

- Lista de todos los servicios de una aplicación.

```
copilot svc ls
```

- Muestre registros de un servicio implementado.

```
copilot svc logs
```

- Muestre el estado del servicio.

```
copilot svc status
```

3. Cuando termine esta demostración, ejecute el comando siguiente para eliminar los recursos asociados y evitar incurrir en cargos generados por recursos sin utilizar.

```
copilot app delete
```

Uso de Amazon ECS con AWS CloudFormation

Amazon ECS está integrado con AWS CloudFormation, un servicio que puede utilizar para modelar y configurar recursos de AWS con plantillas que defina. AWS CloudFormation usa plantillas con formato de archivo de texto `YAML` o `JSON`. Las plantillas son como esquemas del recurso de AWS que desea crear. Al crear y enviar una plantilla, AWS CloudFormation crea una pila. Los recursos definidos en su plantilla se administran a través de la pila. Cuando desee crear, actualizar o eliminar un recurso, debe crear, actualizar o eliminar la pila que se ha creado a partir de ese recurso. Cuando se trata de actualizar las pilas, primero debe crear un conjunto de cambios. Los conjuntos de cambios le muestran qué es lo que se ve afectado por el cambio antes de que lo realice. Por ejemplo, esto evita que elimine bases de datos de manera accidental al cambiar el nombre de la base de datos. Para obtener más información sobre plantillas, pilas y conjuntos de cambios, consulte Cómo funciona AWS CloudFormation en la Guía del usuario AWS CloudFormation.

Al utilizar AWS CloudFormation, puede dedicar menos tiempo a crear y administrar sus recursos e infraestructura. Puede crear una plantilla que describa todos los recursos de AWS que desee, como clústeres, definiciones de tareas y servicios de Amazon ECS. A continuación, AWS CloudFormation se encarga de aprovisionar y configurar esos recursos para usted.

AWS CloudFormation también le permite volver a usar la plantilla para configurar los recursos de Amazon ECS de forma coherente y repetida. Solo tiene que describir los recursos una vez y luego aprovisionar de nuevo los mismos recursos a lo largo de varios Cuentas de AWS y Regiones de AWS.

Las plantillas de AWS CloudFormation se pueden utilizar tanto con la AWS Management Console como la AWS Command Line Interface para crear recursos.

Para conocer más información acerca de AWS CloudFormation, consulte los siguientes recursos:

- AWS CloudFormation
- Guía del usuario de AWS CloudFormation
- Guía del usuario de la interfaz de la línea de comandos de AWS CloudFormation

Temas

- Plantillas de AWS CloudFormation de ejemplo para Amazon ECS
- Creación de recursos de Amazon ECS con la consola de AWS CloudFormation
- Creación de recursos de Amazon ECS con comandos de la AWS CLI para AWS CloudFormation

Plantillas de AWS CloudFormation de ejemplo para Amazon ECS

Para aprovisionar y configurar los recursos de Amazon ECS y los servicios relacionados, asegúrese de estar familiarizado con las plantillas de AWS CloudFormation. Para obtener más información, consulte Plantillas de AWS CloudFormation en la Guía del usuario de AWS CloudFormation. Las plantillas de AWS CloudFormation son archivos de texto en formato JSON o YAML que describen los recursos que desea aprovisionar en sus pilas de AWS CloudFormation. Si no está familiarizado con los formatos JSON o YAML, o ambos, puede usar AWS Infrastructure Composer para comenzar a usar las plantillas de AWS CloudFormation. Para obtener más información, consulte Creación visual de plantillas con Infrastructure Composer en la Guía del usuario de AWS CloudFormation.

Puede crear clústeres, definiciones de tareas y servicios de Amazon ECS mediante AWS CloudFormation. Los siguientes temas incluyen plantillas que demuestran cómo crear recursos con diferentes configuraciones. Puede crear estos recursos con estas plantillas mediante la consola de AWS CloudFormation o la AWS CLI. Para obtener más información sobre cómo crear recursos con la consola AWS CloudFormation, consulte la Guía del usuario de AWS CloudFormation.

Los siguientes temas enumeran plantillas de ejemplo para definiciones de tareas, clústeres y servicios de Amazon ECS.

Temas

- Definiciones de tareas
- Clústeres
- Servicios
- Roles de IAM para Amazon ECS

Definiciones de tareas

Una definición de tarea es un esquema para su aplicación que describe los parámetros y uno o más contenedores que forman su aplicación. A continuación se muestran plantillas de AWS CloudFormation de ejemplo para las definiciones de tareas de Amazon ECS. Para obtener más información sobre definiciones de tareas de Amazon ECS, consulte Definiciones de tareas de Amazon ECS.

Definición de tarea de Fargate Linux

Puede usar la siguiente plantilla para crear una tarea de Fargate Linux de ejemplo.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "ECS Task Definition with parameterized values",
  "Parameters": {
    "ContainerImage": {
      "Type": "String",
      "Default": "public.ecr.aws/docker/library/httpd:2.4",
      "Description": "The container image to use for the task"
    },
    "ContainerCpu": {
      "Type": "Number",
      "Default": 256,
      "Description": "The number of CPU units to reserve for the container",
      "AllowedValues": [256, 512, 1024, 2048, 4096]
    },
    "ContainerMemory": {
      "Type": "Number",
      "Default": 512,
      "Description": "The amount of memory (in MiB) to reserve for the container",
      "AllowedValues": [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192]
    },
    "TaskFamily": {
      "Type": "String",
      "Default": "task-definition-cfn",
      "Description": "The name of the task definition family"
    },
    "ContainerName": {
      "Type": "String",
      "Default": "sample-fargate-app",
      "Description": "The name of the container"
    },
    "ContainerPort": {
      "Type": "Number",
      "Default": 80,
      "Description": "The port number on the container"
    },
    "HostPort": {
      "Type": "Number",
      "Default": 80,
      "Description": "The port number on the host"
    },
    "ExecutionRoleArn": {
```

```
      "Type": "String",
      "Default": "arn:aws:iam::aws_account_id:role/ecsTaskExecutionRole",
      "Description": "The ARN of the task execution role"
    },
    "LogGroup": {
      "Type": "String",
      "Default": "/ecs/fargate-task-definition",
      "Description": "The CloudWatch log group for container logs"
    },
    "NetworkMode": {
      "Type": "String",
      "Default": "awsvpc",
      "Description": "The Docker networking mode to use",
      "AllowedValues": ["awsvpc", "bridge", "host", "none"]
    },
    "OperatingSystemFamily": {
      "Type": "String",
      "Default": "LINUX",
      "Description": "The operating system for the task",
      "AllowedValues": ["LINUX", "WINDOWS_SERVER_2019_FULL",
 "WINDOWS_SERVER_2019_CORE", "WINDOWS_SERVER_2022_FULL", "WINDOWS_SERVER_2022_CORE"]
    }
  },
  "Resources": {
    "ECSTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ContainerDefinitions": [
          {
            "Command": [
              "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</
title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && -
foreground\""
            ],
            "EntryPoint": [
              "sh",
              "-c"
            ],
            "Essential": true,
            "Image": {"Ref": "ContainerImage"},
            "LogConfiguration": {
```

```
              "LogDriver": "awslogs",
              "Options": {
                "mode": "non-blocking",
                "max-buffer-size": "25m",
                "awslogs-create-group": "true",
                "awslogs-group": {"Ref": "LogGroup"},
                "awslogs-region": {"Ref": "AWS::Region"},
                "awslogs-stream-prefix": "ecs"
              }
            },
            "Name": {"Ref": "ContainerName"},
            "PortMappings": [
              {
                "ContainerPort": {"Ref": "ContainerPort"},
                "HostPort": {"Ref": "HostPort"},
                "Protocol": "tcp"
              }
            ]
          }
        ],
        "Cpu": {"Ref": "ContainerCpu"},
        "ExecutionRoleArn": {"Ref": "ExecutionRoleArn"},
        "Family": {"Ref": "TaskFamily"},
        "Memory": {"Ref": "ContainerMemory"},
        "NetworkMode": {"Ref": "NetworkMode"},
        "RequiresCompatibilities": [
          "FARGATE"
        ],
        "RuntimePlatform": {
          "OperatingSystemFamily": {"Ref": "OperatingSystemFamily"}
        }
      }
    }
  },
  "Outputs": {
    "TaskDefinitionArn": {
      "Description": "The ARN of the created task definition",
      "Value": {"Ref": "ECSTaskDefinition"}
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: 'ECS Task Definition to deploy a sample app'
Parameters:
  ContainerImage:
    Type: String
    Default: 'public.ecr.aws/docker/library/httpd:2.4'
    Description: The container image to use for the task
  ContainerCpu:
    Type: Number
    Default: 256
    Description: The number of CPU units to reserve for the container
    AllowedValues: [256, 512, 1024, 2048, 4096]
  ContainerMemory:
    Type: Number
    Default: 512
    Description: The amount of memory (in MiB) to reserve for the container
    AllowedValues: [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192]
  TaskFamily:
    Type: String
    Default: 'task-definition-cfn'
    Description: The name of the task definition family
  ContainerName:
    Type: String
    Default: 'sample-fargate-app'
    Description: The name of the container
  ContainerPort:
    Type: Number
    Default: 80
    Description: The port number on the container
  HostPort:
    Type: Number
    Default: 80
    Description: The port number on the host
  ExecutionRoleArn:
    Type: String
    Default: 'arn:aws:iam::111122223333:role/ecsTaskExecutionRole'
    Description: The ARN of the task execution role
  LogGroup:
    Type: String
    Default: '/ecs/fargate-task-definition'
    Description: The CloudWatch log group for container logs
  NetworkMode:
```

```
    Type: String
    Default: 'awsvpc'
    Description: The Docker networking mode to use
    AllowedValues: ['awsvpc', 'bridge', 'host', 'none']
  OperatingSystemFamily:
    Type: String
    Default: 'LINUX'
    Description: The operating system for the task
    AllowedValues: ['LINUX', 'WINDOWS_SERVER_2019_FULL', 'WINDOWS_SERVER_2019_CORE',
 'WINDOWS_SERVER_2022_FULL', 'WINDOWS_SERVER_2022_CORE']
Resources:
  ECSTaskDefinition:
    Type: 'AWS::ECS::TaskDefinition'
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
              awslogs-stream-prefix: ecs
          Name: !Ref ContainerName
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              HostPort: !Ref HostPort
              Protocol: tcp
      Cpu: !Ref ContainerCpu
```

```
      ExecutionRoleArn: !Ref ExecutionRoleArn
      Family: !Ref TaskFamily
      Memory: !Ref ContainerMemory
      NetworkMode: !Ref NetworkMode
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: !Ref OperatingSystemFamily
Outputs:
  TaskDefinitionArn:
    Description: The ARN of the created task definition
    Value: !Ref ECSTaskDefinition
```

Definición de tarea de Amazon EFS

Puede usar la siguiente plantilla para crear una tarea que utilice un sistema de archivos Amazon EFS que haya creado. Para obtener más información sobre el uso de volúmenes de Amazon EBS con Amazon ECS, consulte Uso de volúmenes de Amazon EFS con Amazon ECS.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Create a task definition for a web server with parameterized
 values.",
  "Parameters": {
    "ExecutionRoleArn": {
      "Type": "String",
      "Default": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
      "Description": "The ARN of the task execution role"
    },
    "NetworkMode": {
      "Type": "String",
      "Default": "awsvpc",
      "Description": "The Docker networking mode to use",
      "AllowedValues": ["awsvpc", "bridge", "host", "none"]
    },
    "TaskFamily": {
      "Type": "String",
      "Default": "my-ecs-task",
      "Description": "The name of the task definition family"
    },
```

```
    "ContainerCpu": {
      "Type": "String",
      "Default": "256",
      "Description": "The number of CPU units to reserve for the container",
      "AllowedValues": ["256", "512", "1024", "2048", "4096"]
    },
    "ContainerMemory": {
      "Type": "String",
      "Default": "512",
      "Description": "The amount of memory (in MiB) to reserve for the container",
      "AllowedValues": ["512", "1024", "2048", "3072", "4096", "5120", "6144",
 "7168", "8192"]
    },
    "ContainerName": {
      "Type": "String",
      "Default": "nginx",
      "Description": "The name of the container"
    },
    "ContainerImage": {
      "Type": "String",
      "Default": "public.ecr.aws/nginx/nginx:latest",
      "Description": "The container image to use for the task"
    },
    "ContainerPort": {
      "Type": "Number",
      "Default": 80,
      "Description": "The port number on the container"
    },
    "InitProcessEnabled": {
      "Type": "String",
      "Default": "true",
      "Description": "Whether to enable the init process inside the container",
      "AllowedValues": ["true", "false"]
    },
    "EfsVolumeName": {
      "Type": "String",
      "Default": "efs-volume",
      "Description": "The name of the EFS volume"
    },
    "EfsContainerPath": {
      "Type": "String",
      "Default": "/usr/share/nginx/html",
      "Description": "The path in the container where the EFS volume will be
 mounted"
```

```
    },
    "LogGroup": {
      "Type": "String",
      "Default": "LogGroup",
      "Description": "The CloudWatch log group for container logs"
    },
    "LogStreamPrefix": {
      "Type": "String",
      "Default": "efs-task",
      "Description": "The prefix for the log stream"
    },
    "EfsFilesystemId": {
      "Type": "String",
      "Default": "fs-1234567890abcdef0",
      "Description": "The ID of the EFS filesystem"
    },
    "EfsRootDirectory": {
      "Type": "String",
      "Default": "/",
      "Description": "The root directory in the EFS filesystem"
    },
    "EfsTransitEncryption": {
      "Type": "String",
      "Default": "ENABLED",
      "Description": "Whether to enable transit encryption for EFS",
      "AllowedValues": ["ENABLED", "DISABLED"]
    }
  },
  "Resources": {
    "ECSTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ExecutionRoleArn": {"Ref": "ExecutionRoleArn"},
        "NetworkMode": {"Ref": "NetworkMode"},
        "RequiresCompatibilities": ["FARGATE"],
        "Family": {"Ref": "TaskFamily"},
        "Cpu": {"Ref": "ContainerCpu"},
        "Memory": {"Ref": "ContainerMemory"},
        "ContainerDefinitions": [
          {
            "Name": {"Ref": "ContainerName"},
            "Image": {"Ref": "ContainerImage"},
            "Essential": true,
            "PortMappings": [
```

```
              {
                "ContainerPort": {"Ref": "ContainerPort"},
                "Protocol": "tcp"
              }
            ],
            "LinuxParameters": {
              "InitProcessEnabled": {"Ref": "InitProcessEnabled"}
            },
            "MountPoints": [
              {
                "SourceVolume": {"Ref": "EfsVolumeName"},
                "ContainerPath": {"Ref": "EfsContainerPath"}
              }
            ],
            "LogConfiguration": {
              "LogDriver": "awslogs",
              "Options": {
                "mode": "non-blocking",
                "max-buffer-size": "25m",
                "awslogs-group": {"Ref": "LogGroup"},
                "awslogs-region": {"Ref": "AWS::Region"},
                "awslogs-create-group": "true",
                "awslogs-stream-prefix": {"Ref": "LogStreamPrefix"}
              }
            }
          }
        ],
        "Volumes": [
          {
            "Name": {"Ref": "EfsVolumeName"},
            "EFSVolumeConfiguration": {
              "FilesystemId": {"Ref": "EfsFilesystemId"},
              "RootDirectory": {"Ref": "EfsRootDirectory"},
              "TransitEncryption": {"Ref": "EfsTransitEncryption"}
            }
          }
        ]
      }
    }
  },
  "Outputs": {
    "TaskDefinitionArn": {
      "Description": "The ARN of the created task definition",
      "Value": {"Ref": "ECSTaskDefinition"}
```

```
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: Create a task definition for a web server with parameterized values.
Parameters:
  ExecutionRoleArn:
    Type: String
    Default: arn:aws:iam::123456789012:role/ecsTaskExecutionRole
    Description: The ARN of the task execution role
  NetworkMode:
    Type: String
    Default: awsvpc
    Description: The Docker networking mode to use
    AllowedValues: [awsvpc, bridge, host, none]
  TaskFamily:
    Type: String
    Default: my-ecs-task
    Description: The name of the task definition family
  ContainerCpu:
    Type: String
    Default: "256"
    Description: The number of CPU units to reserve for the container
    AllowedValues: ["256", "512", "1024", "2048", "4096"]
  ContainerMemory:
    Type: String
    Default: "512"
    Description: The amount of memory (in MiB) to reserve for the container
    AllowedValues: ["512", "1024", "2048", "3072", "4096", "5120", "6144", "7168",
 "8192"]
  ContainerName:
    Type: String
    Default: nginx
    Description: The name of the container
  ContainerImage:
    Type: String
    Default: public.ecr.aws/nginx/nginx:latest
    Description: The container image to use for the task
  ContainerPort:
    Type: Number
```

```
    Default: 80
    Description: The port number on the container
  InitProcessEnabled:
    Type: String
    Default: "true"
    Description: Whether to enable the init process inside the container
    AllowedValues: ["true", "false"]
  EfsVolumeName:
    Type: String
    Default: efs-volume
    Description: The name of the EFS volume
  EfsContainerPath:
    Type: String
    Default: /usr/share/nginx/html
    Description: The path in the container where the EFS volume will be mounted
  LogGroup:
    Type: String
    Default: LogGroup
    Description: The CloudWatch log group for container logs
  LogStreamPrefix:
    Type: String
    Default: efs-task
    Description: The prefix for the log stream
  EfsFilesystemId:
    Type: String
    Default: fs-1234567890abcdef0
    Description: The ID of the EFS filesystem
  EfsRootDirectory:
    Type: String
    Default: /
    Description: The root directory in the EFS filesystem
  EfsTransitEncryption:
    Type: String
    Default: ENABLED
    Description: Whether to enable transit encryption for EFS
    AllowedValues: [ENABLED, DISABLED]
Resources:
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ExecutionRoleArn: !Ref ExecutionRoleArn
      NetworkMode: !Ref NetworkMode
      RequiresCompatibilities:
        - FARGATE
```

```
      Family: !Ref TaskFamily
      Cpu: !Ref ContainerCpu
      Memory: !Ref ContainerMemory
      ContainerDefinitions:
        - Name: !Ref ContainerName
          Image: !Ref ContainerImage
          Essential: true
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              Protocol: tcp
          LinuxParameters:
            InitProcessEnabled: !Ref InitProcessEnabled
          MountPoints:
            - SourceVolume: !Ref EfsVolumeName
              ContainerPath: !Ref EfsContainerPath
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
              awslogs-create-group: "true"
              awslogs-stream-prefix: !Ref LogStreamPrefix
      Volumes:
        - Name: !Ref EfsVolumeName
          EFSVolumeConfiguration:
            FilesystemId: !Ref EfsFilesystemId
            RootDirectory: !Ref EfsRootDirectory
            TransitEncryption: !Ref EfsTransitEncryption
Outputs:
  TaskDefinitionArn:
    Description: The ARN of the created task definition
    Value: !Ref ECSTaskDefinition
```

Clústeres

Un clúster de Amazon ECS es una agrupación lógica de tareas o servicios. Puede usar las siguientes plantillas para crear clústeres con diferentes configuraciones. Para obtener más información sobre los clústeres de Amazon ECS, consulte Clústeres de Amazon ECS.

Creación de un clúster vacío con la configuración predeterminada

Puede usar la siguiente plantilla para crear un clúster vacío con la configuración predeterminada.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": "MyEmptyCluster"
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  ECSCluster:
    Type: 'AWS::ECS::Cluster'
    Properties:
      ClusterName: MyEmptyCluster
```

Creación de un clúster vacío con cifrado de almacenamiento administrado e Información de contenedores mejorado

Puede usar la siguiente plantilla para crear un clúster con almacenamiento administrado a nivel de clúster e Información de contenedores mejorado activado. El cifrado a nivel de clúster se aplica a los volúmenes de datos administrados por Amazon ECS, como los volúmenes de Amazon EBS. Para obtener más información sobre el cifrado de Amazon EBS, consulte Datos cifrados almacenados en volúmenes de Amazon EBS adjuntos a tareas de Amazon ECS. Para obtener más información sobre el uso de Información de contenedores con una observabilidad mejorada, consulte Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
        "Cluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": "EncryptedEnhancedCluster",
                "ClusterSettings": [
                    {
                        "Name": "containerInsights",
                        "Value": "enhanced"
                    }
                ],
                "Configuration": {
                    "ManagedStorageConfiguration": {
                        "KmsKeyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
                    }
                }
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  Cluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: EncryptedEnhancedCluster
      ClusterSettings:
        - Name: containerInsights
          Value: enhanced
      Configuration:
        ManagedStorageConfiguration:
          KmsKeyId: a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Cree un clúster con la AMI optimizada para Amazon ECS basada en AL2023

Puede usar la siguiente plantilla para crear un clúster que utilice un proveedor de capacidad que lance instancias de AL2023 en Amazon EC2.

> **Important**
>
> Para los últimos ID de la AMI, consulte la AMI optimizada para Amazon ECS en la Guía para desarrolladores de Amazon Elastic Container Service.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "EC2 ECS cluster that starts out empty, with no EC2 instances
 yet. An ECS capacity provider automatically launches more EC2 instances as required
 on the fly when you request ECS to launch services or standalone tasks.",
    "Parameters": {
        "InstanceType": {
            "Type": "String",
            "Description": "EC2 instance type",
            "Default": "t2.medium",
            "AllowedValues": [
                "t1.micro",
                "t2.2xlarge",
                "t2.large",
                "t2.medium",
                "t2.micro",
                "t2.nano",
                "t2.small",
                "t2.xlarge",
                "t3.2xlarge",
                "t3.large",
                "t3.medium",
                "t3.micro",
                "t3.nano",
                "t3.small",
                "t3.xlarge"
            ]
        },
        "DesiredCapacity": {
```

```
            "Type": "Number",
            "Default": "0",
            "Description": "Number of EC2 instances to launch in your ECS cluster."
        },
        "MaxSize": {
            "Type": "Number",
            "Default": "100",
            "Description": "Maximum number of EC2 instances that can be launched in
 your ECS cluster."
        },
        "ECSAMI": {
            "Description": "The Amazon Machine Image ID used for the cluster",
            "Type": "AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>",
            "Default": "/aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended/image_id"
        },
        "VpcId": {
            "Type": "AWS::EC2::VPC::Id",
            "Description": "VPC ID where the ECS cluster is launched",
            "Default": "vpc-1234567890abcdef0"
        },
        "SubnetIds": {
            "Type": "List<AWS::EC2::Subnet::Id>",
            "Description": "List of subnet IDs where the EC2 instances will be
 launched",
            "Default": "subnet-021345abcdef67890"
        }
    },
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterSettings": [
                    {
                        "Name": "containerInsights",
                        "Value": "enabled"
                    }
                ]
            }
        },
        "ECSAutoScalingGroup": {
            "Type": "AWS::AutoScaling::AutoScalingGroup",
            "DependsOn": [
                "ECSCluster",
```

```
                "EC2Role"
            ],
            "Properties": {
                "VPCZoneIdentifier": {
                    "Ref": "SubnetIds"
                },
                "LaunchTemplate": {
                    "LaunchTemplateId": {
                        "Ref": "ContainerInstances"
                    },
                    "Version": {
                        "Fn::GetAtt": [
                            "ContainerInstances",
                            "LatestVersionNumber"
                        ]
                    }
                },
                "MinSize": 0,
                "MaxSize": {
                    "Ref": "MaxSize"
                },
                "DesiredCapacity": {
                    "Ref": "DesiredCapacity"
                },
                "NewInstancesProtectedFromScaleIn": true
            },
            "UpdatePolicy": {
                "AutoScalingReplacingUpdate": {
                    "WillReplace": "true"
                }
            }
        },
        "ContainerInstances": {
            "Type": "AWS::EC2::LaunchTemplate",
            "Properties": {
                "LaunchTemplateName": "asg-launch-template-2",
                "LaunchTemplateData": {
                    "ImageId": {
                        "Ref": "ECSAMI"
                    },
                    "InstanceType": {
                        "Ref": "InstanceType"
                    },
                    "IamInstanceProfile": {
```

```
                        "Name": {
                            "Ref": "EC2InstanceProfile"
                        }
                    },
                    "SecurityGroupIds": [
                        {
                            "Ref": "ContainerHostSecurityGroup"
                        }
                    ],
                    "UserData": {
                        "Fn::Base64": {
                            "Fn::Sub": "#!/bin/bash -xe\n echo ECS_CLUSTER=
${ECSCluster} >> /etc/ecs/ecs.config\n yum install -y aws-cfn-bootstrap\n /opt/aws/
bin/cfn-init -v --stack ${AWS::StackId} --resource ContainerInstances --configsets
 full_install --region ${AWS::Region} &\n"
                        }
                    },
                    "MetadataOptions": {
                        "HttpEndpoint": "enabled",
                        "HttpTokens": "required"
                    }
                }
            }
        },
        "EC2InstanceProfile": {
            "Type": "AWS::IAM::InstanceProfile",
            "Properties": {
                "Path": "/",
                "Roles": [
                    {
                        "Ref": "EC2Role"
                    }
                ]
            }
        },
        "CapacityProvider": {
            "Type": "AWS::ECS::CapacityProvider",
            "Properties": {
                "AutoScalingGroupProvider": {
                    "AutoScalingGroupArn": {
                        "Ref": "ECSAutoScalingGroup"
                    },
                    "ManagedScaling": {
                        "InstanceWarmupPeriod": 60,
```

```
                        "MinimumScalingStepSize": 1,
                        "MaximumScalingStepSize": 100,
                        "Status": "ENABLED",
                        "TargetCapacity": 100
                    },
                    "ManagedTerminationProtection": "ENABLED"
                }
            }
        },
        "CapacityProviderAssociation": {
            "Type": "AWS::ECS::ClusterCapacityProviderAssociations",
            "Properties": {
                "CapacityProviders": [
                    {
                        "Ref": "CapacityProvider"
                    }
                ],
                "Cluster": {
                    "Ref": "ECSCluster"
                },
                "DefaultCapacityProviderStrategy": [
                    {
                        "Base": 0,
                        "CapacityProvider": {
                            "Ref": "CapacityProvider"
                        },
                        "Weight": 1
                    }
                ]
            }
        },
        "ContainerHostSecurityGroup": {
            "Type": "AWS::EC2::SecurityGroup",
            "Properties": {
                "GroupDescription": "Access to the EC2 hosts that run containers",
                "VpcId": {
                    "Ref": "VpcId"
                }
            }
        },
        "EC2Role": {
            "Type": "AWS::IAM::Role",
            "Properties": {
                "AssumeRolePolicyDocument": {
```

```
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": [
                                    "ec2.amazonaws.com"
                                ]
                            },
                            "Action": [
                                "sts:AssumeRole"
                            ]
                        }
                    ]
                },
                "Path": "/",
                "ManagedPolicyArns": [
                    "arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role",
                    "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore"
                ]
            }
        },
        "ECSTaskExecutionRole": {
            "Type": "AWS::IAM::Role",
            "Properties": {
                "AssumeRolePolicyDocument": {
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": [
                                    "ecs-tasks.amazonaws.com"
                                ]
                            },
                            "Action": [
                                "sts:AssumeRole"
                            ],
                            "Condition": {
                                "ArnLike": {
                                    "aws:SourceArn": {
                                        "Fn::Sub": "arn:${AWS::Partition}:ecs:
${AWS::Region}:${AWS::AccountId}:*"
                                    }
                                },
```

```
                                "StringEquals": {
                                    "aws:SourceAccount": {
                                        "Fn::Sub": "${AWS::AccountId}"
                                    }
                                }
                            }
                        }
                    ]
                },
                "Path": "/",
                "ManagedPolicyArns": [
                    "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy"
                ]
            }
        }
    },
    "Outputs": {
        "ClusterName": {
            "Description": "The ECS cluster into which to launch resources",
            "Value": "ECSCluster"
        },
        "ECSTaskExecutionRole": {
            "Description": "The role used to start up a task",
            "Value": "ECSTaskExecutionRole"
        },
        "CapacityProvider": {
            "Description": "The cluster capacity provider that the service should
 use to request capacity when it wants to start up a task",
            "Value": "CapacityProvider"
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: EC2 ECS cluster that starts out empty, with no EC2 instances yet. An
 ECS capacity provider automatically launches more EC2 instances as required on the
 fly when you request ECS to launch services or standalone tasks.
Parameters:
  InstanceType:
    Type: String
```

```
    Description: EC2 instance type
    Default: t2.medium
    AllowedValues:
      - t1.micro
      - t2.2xlarge
      - t2.large
      - t2.medium
      - t2.micro
      - t2.nano
      - t2.small
      - t2.xlarge
      - t3.2xlarge
      - t3.large
      - t3.medium
      - t3.micro
      - t3.nano
      - t3.small
      - t3.xlarge
  DesiredCapacity:
    Type: Number
    Default: '0'
    Description: Number of EC2 instances to launch in your ECS cluster.
  MaxSize:
    Type: Number
    Default: '100'
    Description: Maximum number of EC2 instances that can be launched in your ECS
 cluster.
  ECSAMI:
    Description: The Amazon Machine Image ID used for the cluster
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ecs/optimized-ami/amazon-linux-2023/recommended/image_id
  VpcId:
    Type: AWS::EC2::VPC::Id
    Description: VPC ID where the ECS cluster is launched
    Default: vpc-1234567890abcdef0
  SubnetIds:
    Type: List<AWS::EC2::Subnet::Id>
    Description: List of subnet IDs where the EC2 instances will be launched
    Default: subnet-021345abcdef67890
Resources:
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterSettings:
```

```
        - Name: containerInsights
          Value: enabled
  ECSAutoScalingGroup:
    Type: AWS::AutoScaling::AutoScalingGroup
    DependsOn:
      - ECSCluster
      - EC2Role
    Properties:
      VPCZoneIdentifier: !Ref SubnetIds
      LaunchTemplate:
        LaunchTemplateId: !Ref ContainerInstances
        Version: !GetAtt ContainerInstances.LatestVersionNumber
      MinSize: 0
      MaxSize: !Ref MaxSize
      DesiredCapacity: !Ref DesiredCapacity
      NewInstancesProtectedFromScaleIn: true
    UpdatePolicy:
      AutoScalingReplacingUpdate:
        WillReplace: 'true'
  ContainerInstances:
    Type: AWS::EC2::LaunchTemplate
    Properties:
      LaunchTemplateName: asg-launch-template-2
      LaunchTemplateData:
        ImageId: !Ref ECSAMI
        InstanceType: !Ref InstanceType
        IamInstanceProfile:
          Name: !Ref EC2InstanceProfile
        SecurityGroupIds:
          - !Ref ContainerHostSecurityGroup
        UserData: !Base64
          Fn::Sub: |
            #!/bin/bash -xe
            echo ECS_CLUSTER=${ECSCluster} >> /etc/ecs/ecs.config
            yum install -y aws-cfn-bootstrap
            /opt/aws/bin/cfn-init -v --stack ${AWS::StackId} --resource
ContainerInstances --configsets full_install --region ${AWS::Region} &
        MetadataOptions:
          HttpEndpoint: enabled
          HttpTokens: required
  EC2InstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Path: /
```

```
      Roles:
        - !Ref EC2Role
  CapacityProvider:
    Type: AWS::ECS::CapacityProvider
    Properties:
      AutoScalingGroupProvider:
        AutoScalingGroupArn: !Ref ECSAutoScalingGroup
        ManagedScaling:
          InstanceWarmupPeriod: 60
          MinimumScalingStepSize: 1
          MaximumScalingStepSize: 100
          Status: ENABLED
          TargetCapacity: 100
        ManagedTerminationProtection: ENABLED
  CapacityProviderAssociation:
    Type: AWS::ECS::ClusterCapacityProviderAssociations
    Properties:
      CapacityProviders:
        - !Ref CapacityProvider
      Cluster: !Ref ECSCluster
      DefaultCapacityProviderStrategy:
        - Base: 0
          CapacityProvider: !Ref CapacityProvider
          Weight: 1
  ContainerHostSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Access to the EC2 hosts that run containers
      VpcId: !Ref VpcId
  EC2Role:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - ec2.amazonaws.com
            Action:
              - sts:AssumeRole
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonEC2ContainerServiceforEC2Role
        - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
```

```
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - ecs-tasks.amazonaws.com
            Action:
              - sts:AssumeRole
            Condition:
              ArnLike:
                aws:SourceArn: !Sub arn:${AWS::Partition}:ecs:${AWS::Region}:
${AWS::AccountId}:*
              StringEquals:
                aws:SourceAccount: !Sub ${AWS::AccountId}
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
Outputs:
  ClusterName:
    Description: The ECS cluster into which to launch resources
    Value: ECSCluster
  ECSTaskExecutionRole:
    Description: The role used to start up a task
    Value: ECSTaskExecutionRole
  CapacityProvider:
    Description: The cluster capacity provider that the service should use to
 request capacity when it wants to start up a task
    Value: CapacityProvider
```

Servicios

Puede utilizar un servicio de Amazon ECS para ejecutar y mantener un número determinado de instancias de una definición de tarea de manera simultánea en un clúster de Amazon ECS. Si una de las tareas falla o se detiene, el programador de servicios de Amazon ECS lanza otra instancia de su definición de tarea para sustituirla. Esto ayuda a mantener el número deseado de tareas en el servicio. Las siguientes plantillas se pueden utilizar para implementar servicios. Para obtener más información sobre los servicios de Amazon ECS, consulte Servicios de Amazon ECS.

Implementación de una aplicación de

La siguiente plantilla también crea un servicio de Amazon ECS denominado `cfn-service` que lanza y mantiene la tarea definida por la definición de tareas. La plantilla crea un clúster denominado `CFNCluster`. El clúster contiene una definición de tarea de Linux Fargate que configura un servidor web. La plantilla también crea los recursos de red y registro necesarios. La plantilla también crea un rol de ejecución de tareas de Amazon ECS. Para obtener más información sobre el rol de ejecución de tareas, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "A template that deploys an application that is built on an
 Apache web server Docker image by creating an Amazon ECS cluster, task definition,
 and service. The template also creates networking and logging resources, and an
 Amazon ECS task execution role.",
    "Parameters": {
        "ClusterName": {
            "Type": "String",
            "Default": "CFNCluster",
            "Description": "Name of the ECS Cluster"
        },
        "TaskFamily": {
            "Type": "String",
            "Default": "task-definition-cfn",
            "Description": "Family name for the Task Definition"
        },
        "ServiceName": {
            "Type": "String",
            "Default": "cfn-service",
            "Description": "Name of the ECS Service"
        },
        "ContainerImage": {
            "Type": "String",
            "Default": "public.ecr.aws/docker/library/httpd:2.4",
            "Description": "Container image to use for the task"
        },
        "TaskCpu": {
            "Type": "Number",
            "Default": 256,
            "AllowedValues": [
                256,
```

```
                512,
                1024,
                2048,
                4096
            ],
            "Description": "CPU units for the task"
        },
        "TaskMemory": {
            "Type": "Number",
            "Default": 512,
            "AllowedValues": [
                512,
                1024,
                2048,
                4096,
                8192,
                16384
            ],
            "Description": "Memory (in MiB) for the task"
        },
        "DesiredCount": {
            "Type": "Number",
            "Default": 1,
            "Description": "Desired number of tasks to run"
        },
        "LogGroupName": {
            "Type": "String",
            "Default": "/ecs/fargate-task-definition",
            "Description": "CloudWatch Log Group name"
        },
        "VpcCidr": {
            "Type": "String",
            "Default": "10.0.0.0/16",
            "Description": "CIDR block for the VPC"
        },
        "PublicSubnet1Cidr": {
            "Type": "String",
            "Default": "10.0.0.0/24",
            "Description": "CIDR block for public subnet 1"
        },
        "PublicSubnet2Cidr": {
            "Type": "String",
            "Default": "10.0.1.0/24",
            "Description": "CIDR block for public subnet 2"
```

```
        }
    },
    "Resources": {
        "VPC": {
            "Type": "AWS::EC2::VPC",
            "Properties": {
                "CidrBlock": {
                    "Ref": "VpcCidr"
                },
                "EnableDnsSupport": true,
                "EnableDnsHostnames": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-VPC"
                        }
                    }
                ]
            }
        },
        "InternetGateway": {
            "Type": "AWS::EC2::InternetGateway",
            "Properties": {
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-IGW"
                        }
                    }
                ]
            }
        },
        "InternetGatewayAttachment": {
            "Type": "AWS::EC2::VPCGatewayAttachment",
            "Properties": {
                "InternetGatewayId": {
                    "Ref": "InternetGateway"
                },
                "VpcId": {
                    "Ref": "VPC"
                }
            }
```

```
        },
        "PublicSubnet1": {
            "Type": "AWS::EC2::Subnet",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "AvailabilityZone": {
                    "Fn::Select": [
                        0,
                        {
                            "Fn::GetAZs": ""
                        }
                    ]
                },
                "CidrBlock": {
                    "Ref": "PublicSubnet1Cidr"
                },
                "MapPublicIpOnLaunch": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-PublicSubnet1"
                        }
                    }
                ]
            }
        },
        "PublicSubnet2": {
            "Type": "AWS::EC2::Subnet",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "AvailabilityZone": {
                    "Fn::Select": [
                        1,
                        {
                            "Fn::GetAZs": ""
                        }
                    ]
                },
                "CidrBlock": {
```

```
                    "Ref": "PublicSubnet2Cidr"
                },
                "MapPublicIpOnLaunch": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-PublicSubnet2"
                        }
                    }
                ]
            }
        },
        "PublicRouteTable": {
            "Type": "AWS::EC2::RouteTable",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-PublicRouteTable"
                        }
                    }
                ]
            }
        },
        "DefaultPublicRoute": {
            "Type": "AWS::EC2::Route",
            "DependsOn": "InternetGatewayAttachment",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "DestinationCidrBlock": "0.0.0.0/0",
                "GatewayId": {
                    "Ref": "InternetGateway"
                }
            }
        },
        "PublicSubnet1RouteTableAssociation": {
            "Type": "AWS::EC2::SubnetRouteTableAssociation",
```

```
            "Properties": {
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "SubnetId": {
                    "Ref": "PublicSubnet1"
                }
            }
        },
        "PublicSubnet2RouteTableAssociation": {
            "Type": "AWS::EC2::SubnetRouteTableAssociation",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "SubnetId": {
                    "Ref": "PublicSubnet2"
                }
            }
        },
        "ECSSecurityGroup": {
            "Type": "AWS::EC2::SecurityGroup",
            "Properties": {
                "GroupDescription": "Security group for ECS tasks",
                "VpcId": {
                    "Ref": "VPC"
                },
                "SecurityGroupIngress": [
                    {
                        "IpProtocol": "tcp",
                        "FromPort": 80,
                        "ToPort": 80,
                        "CidrIp": "0.0.0.0/0"
                    },
                    {
                        "IpProtocol": "tcp",
                        "FromPort": 443,
                        "ToPort": 443,
                        "CidrIp": "0.0.0.0/0"
                    }
                ]
            }
        },
        "ECSTaskExecutionRole": {
```

```
            "Type": "AWS::IAM::Role",
            "Properties": {
                "AssumeRolePolicyDocument": {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": "ecs-tasks.amazonaws.com"
                            },
                            "Action": "sts:AssumeRole"
                        }
                    ]
                },
                "ManagedPolicyArns": [
                    "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy"
                ]
            }
        },
        "TaskLogGroup": {
            "Type": "AWS::Logs::LogGroup",
            "DeletionPolicy": "Retain",
            "UpdateReplacePolicy": "Retain",
            "Properties": {
                "LogGroupName": {
                    "Ref": "LogGroupName"
                },
                "RetentionInDays": 30
            }
        },
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": {
                    "Ref": "ClusterName"
                }
            }
        },
        "ECSTaskDefinition": {
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "ContainerDefinitions": [
                    {
```

```
                        "Command": [
                        "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
                        ],
                        "EntryPoint": [
                            "sh",
                            "-c"
                        ],
                        "Essential": true,
                        "Image": {
                            "Ref": "ContainerImage"
                        },
                        "LogConfiguration": {
                            "LogDriver": "awslogs",
                            "Options": {
                                "mode": "non-blocking",
                                "max-buffer-size": "25m",
                                "awslogs-create-group": "true",
                                "awslogs-group": {
                                    "Ref": "LogGroupName"
                                },
                                "awslogs-region": {
                                    "Ref": "AWS::Region"
                                },
                                "awslogs-stream-prefix": "ecs"
                            }
                        },
                        "Name": "sample-fargate-app",
                        "PortMappings": [
                            {
                                "ContainerPort": 80,
                                "HostPort": 80,
                                "Protocol": "tcp"
                            }
                        ]
                    }
                ],
                "Cpu": {
                    "Ref": "TaskCpu"
                },
```

```
                "ExecutionRoleArn": {
                    "Fn::GetAtt": [
                        "ECSTaskExecutionRole",
                        "Arn"
                    ]
                },
                "Family": {
                    "Ref": "TaskFamily"
                },
                "Memory": {
                    "Ref": "TaskMemory"
                },
                "NetworkMode": "awsvpc",
                "RequiresCompatibilities": [
                    "FARGATE"
                ],
                "RuntimePlatform": {
                    "OperatingSystemFamily": "LINUX"
                }
            }
        },
        "ECSService": {
            "Type": "AWS::ECS::Service",
            "DependsOn": [
                "PublicSubnet1RouteTableAssociation",
                "PublicSubnet2RouteTableAssociation"
            ],
            "Properties": {
                "ServiceName": {
                    "Ref": "ServiceName"
                },
                "Cluster": {
                    "Ref": "ECSCluster"
                },
                "DesiredCount": {
                    "Ref": "DesiredCount"
                },
                "LaunchType": "FARGATE",
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "AssignPublicIp": "ENABLED",
                        "SecurityGroups": [
                            {
                                "Ref": "ECSSecurityGroup"
```

```
                            }
                        ],
                        "Subnets": [
                            {
                                "Ref": "PublicSubnet1"
                            },
                            {
                                "Ref": "PublicSubnet2"
                            }
                        ]
                    }
                },
                "TaskDefinition": {
                    "Ref": "ECSTaskDefinition"
                }
            }
        }
    },
    "Outputs": {
        "ClusterName": {
            "Description": "The name of the ECS cluster",
            "Value": {
                "Ref": "ECSCluster"
            }
        },
        "TaskDefinitionArn": {
            "Description": "The ARN of the task definition",
            "Value": {
                "Ref": "ECSTaskDefinition"
            }
        },
        "ServiceName": {
            "Description": "The name of the ECS service",
            "Value": {
                "Ref": "ECSService"
            }
        },
        "VpcId": {
            "Description": "The ID of the VPC",
            "Value": {
                "Ref": "VPC"
            }
        },
        "PublicSubnet1": {
```

```
            "Description": "The ID of public subnet 1",
            "Value": {
                "Ref": "PublicSubnet1"
            }
        },
        "PublicSubnet2": {
            "Description": "The ID of public subnet 2",
            "Value": {
                "Ref": "PublicSubnet2"
            }
        },
        "SecurityGroup": {
            "Description": "The ID of the security group",
            "Value": {
                "Ref": "ECSSecurityGroup"
            }
        },
        "ExecutionRoleArn": {
            "Description": "The ARN of the task execution role",
            "Value": {
                "Fn::GetAtt": [
                    "ECSTaskExecutionRole",
                    "Arn"
                ]
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: A template that deploys an application that is built on an Apache
 web server Docker image by creating an Amazon ECS cluster, task definition, and
 service. The template also creates networking and logging resources, and an Amazon
 ECS task execution role.
Parameters:
  ClusterName:
    Type: String
    Default: CFNCluster
    Description: Name of the ECS Cluster
  TaskFamily:
    Type: String
```

```
    Default: task-definition-cfn
    Description: Family name for the Task Definition
  ServiceName:
    Type: String
    Default: cfn-service
    Description: Name of the ECS Service
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  TaskCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the task
  TaskMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 4096, 8192, 16384]
    Description: Memory (in MiB) for the task
  DesiredCount:
    Type: Number
    Default: 1
    Description: Desired number of tasks to run
  LogGroupName:
    Type: String
    Default: /ecs/fargate-task-definition
    Description: CloudWatch Log Group name
  VpcCidr:
    Type: String
    Default: 10.0.0.0/16
    Description: CIDR block for the VPC
  PublicSubnet1Cidr:
    Type: String
    Default: 10.0.0.0/24
    Description: CIDR block for public subnet 1
  PublicSubnet2Cidr:
    Type: String
    Default: 10.0.1.0/24
    Description: CIDR block for public subnet 2
Resources:
  # VPC and Networking Resources
  VPC:
    Type: AWS::EC2::VPC
```

```
    Properties:
      CidrBlock: !Ref VpcCidr
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-VPC
  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-IGW
  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC
  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PublicSubnet1Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet1
  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PublicSubnet2Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet2
  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
```

```
        Value: !Sub ${AWS::StackName}-PublicRouteTable
  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1
  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2
  # Security Group
  ECSSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Security group for ECS tasks
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0

  # IAM Roles
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
```

```
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
  # CloudWatch Logs
  TaskLogGroup:
    Type: AWS::Logs::LogGroup
    DeletionPolicy: Retain
    UpdateReplacePolicy: Retain
    Properties:
      LogGroupName: !Ref LogGroupName
      RetentionInDays: 30
  # ECS Resources
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ClusterName
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"s
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroupName
              awslogs-region: !Ref 'AWS::Region'
              awslogs-stream-prefix: ecs
          Name: sample-fargate-app
```

```
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
              Protocol: tcp
      Cpu: !Ref TaskCpu
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      Family: !Ref TaskFamily
      Memory: !Ref TaskMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX
  ECSService:
    Type: AWS::ECS::Service
    DependsOn:
      - PublicSubnet1RouteTableAssociation
      - PublicSubnet2RouteTableAssociation
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ECSCluster
      DesiredCount: !Ref DesiredCount
      LaunchType: FARGATE
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups:
            - !Ref ECSSecurityGroup
          Subnets:
            - !Ref PublicSubnet1
            - !Ref PublicSubnet2
      TaskDefinition: !Ref ECSTaskDefinition
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref ECSCluster
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref ECSTaskDefinition
  ServiceName:
    Description: The name of the ECS service
    Value: !Ref ECSService
  VpcId:
    Description: The ID of the VPC
```

```
    Value: !Ref VPC
  PublicSubnet1:
    Description: The ID of public subnet 1
    Value: !Ref PublicSubnet1
  PublicSubnet2:
    Description: The ID of public subnet 2
    Value: !Ref PublicSubnet2
  SecurityGroup:
    Description: The ID of the security group
    Value: !Ref ECSSecurityGroup
  ExecutionRoleArn:
    Description: The ARN of the task execution role
    Value: !GetAtt ECSTaskExecutionRole.Arn
```

Implementación de un servicio con ECS Exec activado

Puede usar la siguiente plantilla para implementar un servicio con ECS Exec activado. El servicio se ejecuta en un clúster con una clave de KMS para cifrar las sesiones de ECS Exec y una configuración de registro para redirigir los registros de las sesiones de comandos de ejecución a un bucket de Amazon S3. Para obtener más información, consulte Supervisión de los contenedores de Amazon ECS con ECS Exec.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "ECS Cluster with Fargate Service and Task Definition and ECS
 Exec enabled.",
    "Parameters": {
        "ClusterName": {
            "Type": "String",
            "Default": "CFNCluster",
            "Description": "Name of the ECS Cluster"
        },
        "S3BucketName": {
            "Type": "String",
            "Default": "amzn-s3-demo-bucket",
            "Description": "S3 bucket for ECS execute command logs"
        },
        "KmsKeyId": {
            "Type": "String",
            "Default": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
```

```
            "Description": "KMS Key ID for ECS execute command encryption"
        },
        "ContainerImage": {
            "Type": "String",
            "Default": "public.ecr.aws/docker/library/httpd:2.4",
            "Description": "Container image to use for the task"
        },
        "ContainerCpu": {
            "Type": "Number",
            "Default": 256,
            "AllowedValues": [256, 512, 1024, 2048, 4096],
            "Description": "CPU units for the container (256 = 0.25 vCPU)"
        },
        "ContainerMemory": {
            "Type": "Number",
            "Default": 512,
            "AllowedValues": [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192],
            "Description": "Memory for the container (in MiB)"
        },
        "DesiredCount": {
            "Type": "Number",
            "Default": 1,
            "Description": "Desired count of tasks in the service"
        },
       "SecurityGroups": {
  "Type": "List<AWS::EC2::SecurityGroup::Id>",
  "Description": "Security Group IDs for the ECS Service"
 },
 "Subnets": {
  "Type": "List<AWS::EC2::Subnet::Id>",
  "Description": "Subnet IDs for the ECS Service"
 },
        "ServiceName": {
            "Type": "String",
            "Default": "cfn-service",
            "Description": "Name of the ECS service"
        },
        "TaskFamily": {
            "Type": "String",
            "Default": "task-definition-cfn",
            "Description": "Family name for the task definition"
        },
        "TaskExecutionRoleArn": {
            "Type": "String",
```

```
            "Description": "ARN of an existing IAM role for ECS task execution",
            "Default": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole"
        },
        "TaskRoleArn": {
            "Type": "String",
            "Description": "ARN of an existing IAM role for ECS tasks",
            "Default": "arn:aws:iam::111122223333:role/execTaskRole"
        }
    },
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": {"Ref": "ClusterName"},
                "Configuration": {
                    "ExecuteCommandConfiguration": {
                        "Logging": "OVERRIDE",
                        "LogConfiguration": {
                            "S3BucketName": {"Ref": "S3BucketName"}
                        },
                        "KmsKeyId": {"Ref": "KmsKeyId"}
                    }
                },
                "Tags": [
                    {
                        "Key": "Environment",
                        "Value": {"Ref": "AWS::StackName"}
                    }
                ]
            }
        },
        "ECSTaskDefinition": {
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "ContainerDefinitions": [
                    {
                        "Command": [
                            "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS
 Sample App</title> <style>body {margin-top: 40px; background-color: #333;} </style>
 </head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
                        ],
```

```
                        "EntryPoint": [
                            "sh",
                            "-c"
                        ],
                        "Essential": true,
                        "Image": {"Ref": "ContainerImage"},
                        "LogConfiguration": {
                            "LogDriver": "awslogs",
                            "Options": {
                                "mode": "non-blocking",
                                "max-buffer-size": "25m",
                                "awslogs-create-group": "true",
                                "awslogs-group": {"Fn::Sub": "/ecs/
${AWS::StackName}"},
                                "awslogs-region": {"Ref": "AWS::Region"},
                                "awslogs-stream-prefix": "ecs"
                            }
                        },
                        "Name": "sample-fargate-app",
                        "PortMappings": [
                            {
                                "ContainerPort": 80,
                                "HostPort": 80,
                                "Protocol": "tcp"
                            }
                        ]
                    }
                ],
                "Cpu": {"Ref": "ContainerCpu"},
                "ExecutionRoleArn": {"Ref": "TaskExecutionRoleArn"},
                "TaskRoleArn": {"Ref": "TaskRoleArn"},
                "Family": {"Ref": "TaskFamily"},
                "Memory": {"Ref": "ContainerMemory"},
                "NetworkMode": "awsvpc",
                "RequiresCompatibilities": [
                    "FARGATE"
                ],
                "RuntimePlatform": {
                    "OperatingSystemFamily": "LINUX"
                },
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {"Fn::Sub": "${AWS::StackName}-TaskDefinition"}
```

```
                    }
                ]
            }
        },
        "ECSService": {
            "Type": "AWS::ECS::Service",
            "Properties": {
                "ServiceName": {"Ref": "ServiceName"},
                "Cluster": {"Ref": "ECSCluster"},
                "DesiredCount": {"Ref": "DesiredCount"},
                "EnableExecuteCommand": true,
                "LaunchType": "FARGATE",
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "AssignPublicIp": "ENABLED",
                        "SecurityGroups": {"Ref": "SecurityGroups"},
                        "Subnets": {"Ref": "Subnets"}
                    }
                },
                "TaskDefinition": {"Ref": "ECSTaskDefinition"},
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {"Fn::Sub": "${AWS::StackName}-Service"}
                    }
                ]
            }
        }
    },
    "Outputs": {
        "ClusterName": {
            "Description": "The name of the ECS cluster",
            "Value": {"Ref": "ECSCluster"}
        },
        "ServiceName": {
            "Description": "The name of the ECS service",
            "Value": {"Ref": "ServiceName"}
        },
        "TaskDefinitionArn": {
            "Description": "The ARN of the task definition",
            "Value": {"Ref": "ECSTaskDefinition"}
        },
        "ClusterArn": {
            "Description": "The ARN of the ECS cluster",
```

```
            "Value": {"Fn::GetAtt": ["ECSCluster", "Arn"]}
        },
        "StackName": {
            "Description": "The name of this stack",
            "Value": {"Ref": "AWS::StackName"}
        },
        "StackId": {
            "Description": "The unique identifier for this stack",
            "Value": {"Ref": "AWS::StackId"}
        },
        "Region": {
            "Description": "The AWS Region where the stack is deployed",
            "Value": {"Ref": "AWS::Region"}
        },
        "AccountId": {
            "Description": "The AWS Account ID",
            "Value": {"Ref": "AWS::AccountId"}
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: ECS Cluster with Fargate Service and Task Definition and ECS Exec
 enabled.
Parameters:
  ClusterName:
    Type: String
    Default: CFNCluster
    Description: Name of the ECS Cluster
  S3BucketName:
    Type: String
    Default: amzn-s3-demo-bucket
    Description: S3 bucket for ECS execute command logs
  KmsKeyId:
    Type: String
    Default: a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
    Description: KMS Key ID for ECS execute command encryption
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
```

```
  ContainerCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the container (256 = 0.25 vCPU)
  ContainerMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192]
    Description: Memory for the container (in MiB)
  DesiredCount:
    Type: Number
    Default: 1
    Description: Desired count of tasks in the service
 SecurityGroups:
    Type: List<AWS::EC2::SecurityGroup::Id>
    Description: Security Group IDs for the ECS Service
  Subnets:
    Type: List<AWS::EC2::Subnet::Id>
    Description: Subnet IDs for the ECS Service
  ServiceName:
    Type: String
    Default: cfn-service
    Description: Name of the ECS service
  TaskFamily:
    Type: String
    Default: task-definition-cfn
    Description: Family name for the task definition
  TaskExecutionRoleArn:
    Type: String
    Description: ARN of an existing IAM role for ECS task execution
    Default: 'arn:aws:iam::111122223333:role/ecsTaskExecutionRole'
  TaskRoleArn:
    Type: String
    Description: ARN of an existing IAM role for ECS tasks
    Default: 'arn:aws:iam::111122223333:role/execTaskRole'
Resources:
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ClusterName
      Configuration:
        ExecuteCommandConfiguration:
          Logging: OVERRIDE
```

```
          LogConfiguration:
            S3BucketName: !Ref S3BucketName
          KmsKeyId: !Ref KmsKeyId
      Tags:
        - Key: Environment
          Value: !Ref AWS::StackName
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
             /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Sub /ecs/${AWS::StackName}
              awslogs-region: !Ref AWS::Region
              awslogs-stream-prefix: ecs
          Name: sample-fargate-app
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
              Protocol: tcp
      Cpu: !Ref ContainerCpu
      ExecutionRoleArn: !Ref TaskExecutionRoleArn
      TaskRoleArn: !Ref TaskRoleArn
      Family: !Ref TaskFamily
      Memory: !Ref ContainerMemory
      NetworkMode: awsvpc
```

```
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-TaskDefinition
  ECSService:
    Type: AWS::ECS::Service
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ECSCluster
      DesiredCount: !Ref DesiredCount
      EnableExecuteCommand: true
      LaunchType: FARGATE
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups: !Ref SecurityGroups
          Subnets: !Ref Subnets
      TaskDefinition: !Ref ECSTaskDefinition
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-Service
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref ECSCluster
  ServiceName:
    Description: The name of the ECS service
    Value: !Ref ServiceName
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref ECSTaskDefinition
  ClusterArn:
    Description: The ARN of the ECS cluster
    Value: !GetAtt ECSCluster.Arn
  StackName:
    Description: The name of this stack
    Value: !Ref AWS::StackName
  StackId:
    Description: The unique identifier for this stack
    Value: !Ref AWS::StackId
  Region:
```

```
    Description: The AWS Region where the stack is deployed
    Value: !Ref AWS::Region
  AccountId:
    Description: The AWS Account ID
    Value: !Ref AWS::AccountId
```

Implementación de un servicio que utilice Amazon VPC Lattice

Puede usar la siguiente plantilla para empezar a crear un servicio de Amazon ECS con VPC Lattice. Es posible que tenga que completar los siguientes pasos adicionales para configurar VPC Lattice:

- Actualice las reglas de entrada de su grupo de seguridad para VPC Lattice a fin de permitir el prefijo de la regla de entrada `vpc-lattice` y permitir el tráfico en el puerto 80.
- Asocie la VPC del servicio a una red de servicios de VPC Lattice.
- Configure una zona alojada pública o privada con Amazon Route 53.
- Configure los oyentes y sus reglas en un servicio de VPC Lattice.
- Verifique las configuraciones de comprobación de estado del grupo de destino.

Para obtener más información acerca del uso de VPC Lattice con Amazon VPC, consulte Uso de Amazon VPC Lattice para conectar, observar y proteger los servicios de Amazon ECS.

JSON

```
{
 "AWSTemplateFormatVersion": "2010-09-09",
 "Description": "The template used to create an ECS Service with VPC Lattice.",
 "Parameters": {
  "ECSClusterName": {
   "Type": "String",
   "Default": "vpc-lattice-cluster"
  },
  "ECSServiceName": {
   "Type": "String",
   "Default": "vpc-lattice-service"
  },
  "SecurityGroupIDs": {
   "Type": "List<AWS::EC2::SecurityGroup::Id>",
   "Description": "Security Group IDs for the ECS Service"
  },
```

```
  "SubnetIDs": {
   "Type": "List<AWS::EC2::Subnet::Id>",
   "Description": "Subnet IDs for the ECS Service"
  },
  "VpcID": {
   "Type": "AWS::EC2::VPC::Id",
   "Description": "VPC ID for the resources"
  },
  "ContainerImage": {
   "Type": "String",
   "Default": "public.ecr.aws/docker/library/httpd:2.4",
   "Description": "Container image to use for the task"
  },
  "TaskCpu": {
   "Type": "Number",
   "Default": 256,
   "AllowedValues": [256, 512, 1024, 2048, 4096],
   "Description": "CPU units for the task"
  },
  "TaskMemory": {
   "Type": "Number",
   "Default": 512,
   "AllowedValues": [512, 1024, 2048, 4096, 8192, 16384],
   "Description": "Memory (in MiB) for the task"
  },
  "LogGroupName": {
   "Type": "String",
   "Default": "/ecs/vpc-lattice-task",
   "Description": "CloudWatch Log Group name"
  },
  "EnableContainerInsights": {
   "Type": "String",
   "Default": "enabled",
   "AllowedValues": ["enabled", "disabled"],
   "Description": "Enable or disable CloudWatch Container Insights for the cluster"
  }
 },
 "Resources": {
  "ECSCluster": {
   "Type": "AWS::ECS::Cluster",
   "Properties": {
    "ClusterName": {"Ref": "ECSClusterName"},
    "ClusterSettings": [
     {
```

```
      "Name": "containerInsights",
      "Value": {"Ref": "EnableContainerInsights"}
     }
    ],
    "Tags": [
     {
      "Key": "Name",
      "Value": {"Ref": "ECSClusterName"}
     }
    ]
   }
  },
  "ECSTaskExecutionRole": {
   "Type": "AWS::IAM::Role",
   "Properties": {
    "AssumeRolePolicyDocument": {
     "Version": "2012-10-17",
     "Statement": [
      {
       "Effect": "Allow",
       "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
       },
       "Action": "sts:AssumeRole"
      }
     ]
    },
    "ManagedPolicyArns": [
     "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
    ]
   }
  },
  "TaskLogGroup": {
   "Type": "AWS::Logs::LogGroup",
   "DeletionPolicy": "Retain",
   "UpdateReplacePolicy": "Retain",
   "Properties": {
    "LogGroupName": {"Ref": "LogGroupName"},
    "RetentionInDays": 30
   }
  },
  "VpcLatticeTaskDefinition": {
   "Type": "AWS::ECS::TaskDefinition",
   "Properties": {
```

```
    "ContainerDefinitions": [
     {
      "Command": [
       "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</title>
<style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
<div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
<h2>Congratulations!</h2> <p>Your application is now running on a container in
Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
httpd-foreground\""
      ],
      "EntryPoint": [
       "sh",
       "-c"
      ],
      "Essential": true,
      "Image": {"Ref": "ContainerImage"},
      "LogConfiguration": {
       "LogDriver": "awslogs",
       "Options": {
        "mode": "non-blocking",
        "max-buffer-size": "25m",
        "awslogs-create-group": "true",
        "awslogs-group": {"Ref": "LogGroupName"},
        "awslogs-region": {"Ref": "AWS::Region"},
        "awslogs-stream-prefix": "ecs"
       }
      },
      "Name": "vpc-lattice-container",
      "PortMappings": [
       {
        "ContainerPort": 80,
        "HostPort": 80,
        "Protocol": "tcp",
        "Name": "vpc-lattice-port"
       }
      ]
     }
    ],
    "Cpu": {"Ref": "TaskCpu"},
    "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
    "Family": "vpc-lattice-task-definition",
    "Memory": {"Ref": "TaskMemory"},
    "NetworkMode": "awsvpc",
    "RequiresCompatibilities": [
```

```
      "FARGATE"
     ],
     "RuntimePlatform": {
      "OperatingSystemFamily": "LINUX"
     }
    }
   },
   "ECSService": {
    "Type": "AWS::ECS::Service",
    "Properties": {
     "Cluster": {"Ref": "ECSCluster"},
     "TaskDefinition": {"Ref": "VpcLatticeTaskDefinition"},
     "LaunchType": "FARGATE",
     "ServiceName": {"Ref": "ECSServiceName"},
     "SchedulingStrategy": "REPLICA",
     "DesiredCount": 2,
     "AvailabilityZoneRebalancing": "ENABLED",
     "NetworkConfiguration": {
      "AwsvpcConfiguration": {
       "AssignPublicIp": "ENABLED",
       "SecurityGroups": {
        "Ref": "SecurityGroupIDs"
       },
       "Subnets": {
        "Ref": "SubnetIDs"
       }
      }
     },
     "PlatformVersion": "LATEST",
     "VpcLatticeConfigurations": [
      {
       "RoleArn": "arn:aws:iam::111122223333:role/ecsInfrastructureRole",
       "PortName": "vpc-lattice-port",
       "TargetGroupArn": {
        "Ref": "TargetGroup1"
       }
      }
     ],
     "DeploymentConfiguration": {
      "DeploymentCircuitBreaker": {
       "Enable": true,
       "Rollback": true
      },
      "MaximumPercent": 200,
```

```
      "MinimumHealthyPercent": 100
     },
     "DeploymentController": {
      "Type": "ECS"
     },
     "ServiceConnectConfiguration": {
      "Enabled": false
     },
     "Tags": [],
     "EnableECSManagedTags": true
    }
   },
   "TargetGroup1": {
    "Type": "AWS::VpcLattice::TargetGroup",
    "Properties": {
     "Type": "IP",
     "Name": "first-target-group",
     "Config": {
      "Port": 80,
      "Protocol": "HTTP",
      "VpcIdentifier": {"Ref": "VpcID"},
      "HealthCheck": {
       "Enabled": true,
       "Path": "/"
      }
     },
     "Tags": [
      {
       "Key": "ecs-application-networking/ServiceName",
       "Value": {"Ref": "ECSServiceName"}
      },
      {
       "Key": "ecs-application-networking/ClusterName",
       "Value": {"Ref": "ECSClusterName"}
      },
      {
       "Key": "ecs-application-networking/TaskDefinition",
       "Value": {"Ref": "VpcLatticeTaskDefinition"}
      },
      {
       "Key": "ecs-application-networking/VpcId",
       "Value": {"Ref": "VpcID"}
      }
     ]
```

```
    }
   }
  },
  "Outputs": {
   "ClusterName": {
    "Description": "The cluster used to create the service.",
    "Value": {
     "Ref": "ECSCluster"
    }
   },
   "ClusterArn": {
    "Description": "The ARN of the ECS cluster",
    "Value": {
     "Fn::GetAtt": ["ECSCluster", "Arn"]
    }
   },
   "ECSService": {
    "Description": "The created service.",
    "Value": {
     "Ref": "ECSService"
    }
   },
   "TaskDefinitionArn": {
    "Description": "The ARN of the task definition",
    "Value": {
     "Ref": "VpcLatticeTaskDefinition"
    }
   }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: The template used to create an ECS Service with VPC Lattice.

Parameters:
  ECSClusterName:
    Type: String
    Default: vpc-lattice-cluster
  ECSServiceName:
    Type: String
    Default: vpc-lattice-service
```

```
  SecurityGroupIDs:
    Type: List<AWS::EC2::SecurityGroup::Id>
    Description: Security Group IDs for the ECS Service
  SubnetIDs:
    Type: List<AWS::EC2::Subnet::Id>
    Description: Subnet IDs for the ECS Service
  VpcID:
    Type: AWS::EC2::VPC::Id
    Description: VPC ID for the resources
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  TaskCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the task
  TaskMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 4096, 8192, 16384]
    Description: Memory (in MiB) for the task
  LogGroupName:
    Type: String
    Default: /ecs/vpc-lattice-task
    Description: CloudWatch Log Group name
  EnableContainerInsights:
    Type: String
    Default: 'enhanced'
    AllowedValues: ['enabled', 'disabled', 'enhanced']
    Description: Enable or disable CloudWatch Container Insights for the cluster

Resources:
  # ECS Cluster
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ECSClusterName
      ClusterSettings:
        - Name: containerInsights
          Value: !Ref EnableContainerInsights
      Tags:
        - Key: Name
```

```
          Value: !Ref ECSClusterName

  # IAM Roles
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy

  # CloudWatch Logs
  TaskLogGroup:
    Type: AWS::Logs::LogGroup
    DeletionPolicy: Retain
    UpdateReplacePolicy: Retain
    Properties:
      LogGroupName: !Ref LogGroupName
      RetentionInDays: 30

  # Task Definition
  VpcLatticeTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
```

```
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroupName
              awslogs-region: !Ref 'AWS::Region'
              awslogs-stream-prefix: ecs
          Name: vpc-lattice-container
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
              Protocol: tcp
              Name: vpc-lattice-port
      Cpu: !Ref TaskCpu
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      Family: vpc-lattice-task-definition
      Memory: !Ref TaskMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX

  ECSService:
    Type: AWS::ECS::Service
    Properties:
      Cluster: !Ref ECSCluster
      TaskDefinition: !Ref VpcLatticeTaskDefinition
      LaunchType: FARGATE
      ServiceName: !Ref ECSServiceName
      SchedulingStrategy: REPLICA
      DesiredCount: 2
      AvailabilityZoneRebalancing: ENABLED
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups: !Ref SecurityGroupIDs
          Subnets: !Ref SubnetIDs
      PlatformVersion: LATEST
      VpcLatticeConfigurations:
        - RoleArn: arn:aws:iam::111122223333:role/ecsInfrastructureRole
          PortName: vpc-lattice-port
```

```
          TargetGroupArn: !Ref TargetGroup1
      DeploymentConfiguration:
        DeploymentCircuitBreaker:
          Enable: true
          Rollback: true
        MaximumPercent: 200
        MinimumHealthyPercent: 100
      DeploymentController:
        Type: ECS
      ServiceConnectConfiguration:
        Enabled: false
      Tags: []
      EnableECSManagedTags: true

  TargetGroup1:
    Type: AWS::VpcLattice::TargetGroup
    Properties:
      Type: IP
      Name: first-target-group
      Config:
        Port: 80
        Protocol: HTTP
        VpcIdentifier: !Ref VpcID
        HealthCheck:
          Enabled: true
          Path: /
      Tags:
        - Key: ecs-application-networking/ServiceName
          Value: !Ref ECSServiceName
        - Key: ecs-application-networking/ClusterName
          Value: !Ref ECSClusterName
        - Key: ecs-application-networking/TaskDefinition
          Value: !Ref VpcLatticeTaskDefinition
        - Key: ecs-application-networking/VpcId
          Value: !Ref VpcID

Outputs:
  ClusterName:
    Description: The cluster used to create the service.
    Value: !Ref ECSCluster
  ClusterArn:
    Description: The ARN of the ECS cluster
    Value: !GetAtt ECSCluster.Arn
  ECSService:
```

```
    Description: The created service.
    Value: !Ref ECSService
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref VpcLatticeTaskDefinition
```

Implementación del servicio con una configuración de volumen

La siguiente plantilla incluye una configuración de volumen en la definición del servicio. Amazon ECS admite la configuración de los siguientes volúmenes de datos mediante una configuración de volumen en el momento del lanzamiento: volúmenes de Amazon EBS. Para obtener más información sobre los volúmenes Amazon EBS, consulte Uso de volúmenes de Amazon EBS con Amazon ECS.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "The template used to create an ECS Service that includes a
 volume configuration. The configuration is used to create Amazon EBS volumes for
 attachment to the tasks. One volume is attached per task.",
  "Parameters": {
    "ECSClusterName": {
      "Type": "String",
      "Default": "volume-config-cluster",
      "Description": "Name of the ECS cluster"
    },
    "SecurityGroupIDs": {
      "Type": "List<AWS::EC2::SecurityGroup::Id>",
      "Description": "Security Group IDs for the ECS Service"
    },
    "SubnetIDs": {
      "Type": "List<AWS::EC2::Subnet::Id>",
      "Description": "Subnet IDs for the ECS Service"
    },
    "InfrastructureRoleArn": {
      "Type": "String",
      "Description": "ARN of the IAM role that ECS will use to manage EBS volumes"
    },
    "ContainerImage": {
      "Type": "String",
      "Default": "public.ecr.aws/nginx/nginx:latest",
      "Description": "Container image to use for the task"
```

```
    },
    "TaskCpu": {
      "Type": "String",
      "Default": "2048",
      "Description": "CPU units for the task"
    },
    "TaskMemory": {
      "Type": "String",
      "Default": "4096",
      "Description": "Memory (in MiB) for the task"
    },
    "VolumeSize": {
      "Type": "String",
      "Default": "10",
      "Description": "Size of the EBS volume in GiB"
    },
    "VolumeType": {
      "Type": "String",
      "Default": "gp3",
      "AllowedValues": ["gp2", "gp3", "io1", "io2", "st1", "sc1", "standard"],
      "Description": "EBS volume type"
    },
    "VolumeIops": {
      "Type": "String",
      "Default": "3000",
      "Description": "IOPS for the EBS volume (required for io1, io2, and gp3)"
    },
    "VolumeThroughput": {
      "Type": "String",
      "Default": "125",
      "Description": "Throughput for the EBS volume (only for gp3)"
    },
    "FilesystemType": {
      "Type": "String",
      "Default": "xfs",
      "AllowedValues": ["xfs", "ext4"],
      "Description": "Filesystem type for the EBS volume"
    },
    "EnableContainerInsights": {
      "Type": "String",
      "Default": "enhanced",
      "AllowedValues": ["enabled", "disabled", "enhanced"],
      "Description": "Enable or disable CloudWatch Container Insights for the
cluster"
```

```
    }
  },
  "Resources": {
    "ECSCluster": {
      "Type": "AWS::ECS::Cluster",
      "Properties": {
        "ClusterName": {"Ref": "ECSClusterName"},
        "ClusterSettings": [
          {
            "Name": "containerInsights",
            "Value": {"Ref": "EnableContainerInsights"}
          }
        ],
        "Tags": [
          {
            "Key": "Name",
            "Value": {"Ref": "ECSClusterName"}
          }
        ]
      }
    },
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
      }
    },
    "EBSTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
```

```
        "Family": "ebs-task-attach-task-def",
        "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
        "NetworkMode": "awsvpc",
        "RequiresCompatibilities": [
          "EC2",
          "FARGATE"
        ],
        "Cpu": {"Ref": "TaskCpu"},
        "Memory": {"Ref": "TaskMemory"},
        "ContainerDefinitions": [
          {
            "Name": "nginx",
            "Image": {"Ref": "ContainerImage"},
            "Essential": true,
            "PortMappings": [
              {
                "Name": "nginx-80-tcp",
                "ContainerPort": 80,
                "HostPort": 80,
                "Protocol": "tcp",
                "AppProtocol": "http"
              }
            ],
            "MountPoints": [
              {
                "SourceVolume": "ebs-vol",
                "ContainerPath": "/foo-container-path",
                "ReadOnly": false
              }
            ]
          }
        ],
        "Volumes": [
          {
            "Name": "ebs-vol",
            "ConfiguredAtLaunch": true
          }
        ]
      }
    },
    "ECSService": {
      "Type": "AWS::ECS::Service",
      "Properties": {
        "Cluster": {"Ref": "ECSCluster"},
```

```
        "TaskDefinition": {"Ref": "EBSTaskDefinition"},
        "LaunchType": "FARGATE",
        "ServiceName": "ebs",
        "SchedulingStrategy": "REPLICA",
        "DesiredCount": 1,
        "NetworkConfiguration": {
          "AwsvpcConfiguration": {
            "AssignPublicIp": "ENABLED",
            "SecurityGroups": {"Ref": "SecurityGroupIDs"},
            "Subnets": {"Ref": "SubnetIDs"}
          }
        },
        "PlatformVersion": "LATEST",
        "DeploymentConfiguration": {
          "MaximumPercent": 200,
          "MinimumHealthyPercent": 100,
          "DeploymentCircuitBreaker": {
            "Enable": true,
            "Rollback": true
          }
        },
        "DeploymentController": {
          "Type": "ECS"
        },
        "Tags": [],
        "EnableECSManagedTags": true,
        "VolumeConfigurations": [
          {
            "Name": "ebs-vol",
            "ManagedEBSVolume": {
              "RoleArn": {"Ref": "InfrastructureRoleArn"},
              "VolumeType": {"Ref": "VolumeType"},
              "Iops": {"Ref": "VolumeIops"},
              "Throughput": {"Ref": "VolumeThroughput"},
              "SizeInGiB": {"Ref": "VolumeSize"},
              "FilesystemType": {"Ref": "FilesystemType"},
              "TagSpecifications": [
                {
                  "ResourceType": "volume",
                  "PropagateTags": "TASK_DEFINITION"
                }
              ]
            }
          }
```

```
        ]
      }
    }
  },
  "Outputs": {
    "ClusterName": {
      "Description": "The cluster used to create the service.",
      "Value": {"Ref": "ECSCluster"}
    },
    "ClusterArn": {
      "Description": "The ARN of the ECS cluster",
      "Value": {"Fn::GetAtt": ["ECSCluster", "Arn"]}
    },
    "ECSService": {
      "Description": "The created service.",
      "Value": {"Ref": "ECSService"}
    },
    "TaskDefinitionArn": {
      "Description": "The ARN of the task definition",
      "Value": {"Ref": "EBSTaskDefinition"}
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: The template used to create an ECS Service that includes a volume
 configuration. The configuration is used to create Amazon EBS volumes for
 attachment to the tasks. One volume is attached per task.
Parameters:
  ECSClusterName:
    Type: String
    Default: volume-config-cluster
    Description: Name of the ECS cluster

  SecurityGroupIDs:
    Type: List<AWS::EC2::SecurityGroup::Id>
    Description: Security Group IDs for the ECS Service

  SubnetIDs:
    Type: List<AWS::EC2::Subnet::Id>
    Description: Subnet IDs for the ECS Service
```

```
  InfrastructureRoleArn:
    Type: String
    Description: ARN of the IAM role that ECS will use to manage EBS volumes

  ContainerImage:
    Type: String
    Default: public.ecr.aws/nginx/nginx:latest
    Description: Container image to use for the task

  TaskCpu:
    Type: String
    Default: "2048"
    Description: CPU units for the task

  TaskMemory:
    Type: String
    Default: "4096"
    Description: Memory (in MiB) for the task

  VolumeSize:
    Type: String
    Default: "10"
    Description: Size of the EBS volume in GiB

  VolumeType:
    Type: String
    Default: gp3
    AllowedValues: [gp2, gp3, io1, io2, st1, sc1, standard]
    Description: EBS volume type

  VolumeIops:
    Type: String
    Default: "3000"
    Description: IOPS for the EBS volume (required for io1, io2, and gp3)

  VolumeThroughput:
    Type: String
    Default: "125"
    Description: Throughput for the EBS volume (only for gp3)

  FilesystemType:
    Type: String
    Default: xfs
```

```
    AllowedValues: [xfs, ext4]
    Description: Filesystem type for the EBS volume

  EnableContainerInsights:
    Type: String
    Default: 'enhanced'
    AllowedValues: ['enabled', 'disabled', 'enhanced']
    Description: Enable or disable CloudWatch Container Insights for the cluster

Resources:
  # ECS Cluster
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ECSClusterName
      ClusterSettings:
        - Name: containerInsights
          Value: !Ref EnableContainerInsights
      Tags:
        - Key: Name
          Value: !Ref ECSClusterName

  # IAM Role for Task Execution
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy

  # Task Definition
  EBSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      Family: ebs-task-attach-task-def
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      NetworkMode: awsvpc
      RequiresCompatibilities:
```

```
        - EC2
        - FARGATE
      Cpu: !Ref TaskCpu
      Memory: !Ref TaskMemory
      ContainerDefinitions:
        - Name: nginx
          Image: !Ref ContainerImage
          Essential: true
          PortMappings:
            - Name: nginx-80-tcp
              ContainerPort: 80
              HostPort: 80
              Protocol: tcp
              AppProtocol: http
          MountPoints:
            - SourceVolume: ebs-vol
              ContainerPath: /foo-container-path
              ReadOnly: false
      Volumes:
        - Name: ebs-vol
          ConfiguredAtLaunch: true

  ECSService:
    Type: AWS::ECS::Service
    Properties:
      Cluster: !Ref ECSCluster
      TaskDefinition: !Ref EBSTaskDefinition
      LaunchType: FARGATE
      ServiceName: ebs
      SchedulingStrategy: REPLICA
      DesiredCount: 1
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups: !Ref SecurityGroupIDs
          Subnets: !Ref SubnetIDs
      PlatformVersion: LATEST
      DeploymentConfiguration:
        MaximumPercent: 200
        MinimumHealthyPercent: 100
        DeploymentCircuitBreaker:
          Enable: true
          Rollback: true
      DeploymentController:
```

```
        Type: ECS
      Tags: []
      EnableECSManagedTags: true
      VolumeConfigurations:
        - Name: ebs-vol
          ManagedEBSVolume:
            RoleArn: !Ref InfrastructureRoleArn
            VolumeType: !Ref VolumeType
            Iops: !Ref VolumeIops
            Throughput: !Ref VolumeThroughput
            SizeInGiB: !Ref VolumeSize
            FilesystemType: !Ref FilesystemType
            TagSpecifications:
              - ResourceType: volume
                PropagateTags: TASK_DEFINITION

Outputs:
  ClusterName:
    Description: The cluster used to create the service.
    Value: !Ref ECSCluster
  ClusterArn:
    Description: The ARN of the ECS cluster
    Value: !GetAtt ECSCluster.Arn
  ECSService:
    Description: The created service.
    Value: !Ref ECSService
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref EBSTaskDefinition
```

Implementación del servicio con proveedores de capacidad

La siguiente plantilla define un servicio que utiliza el proveedor de capacidad para solicitar la capacidad de ejecución de AL2023. Los contenedores se incorporarán a las instancias de AL2023 a medida que se encuentren en línea.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "An example service that deploys in AWS VPC networking mode on
 EC2 capacity. Service uses a capacity provider to request EC2 instances to run
```

```
on. Service runs with networking in private subnets, but still accessible to the
internet via a load balancer hosted in public subnets.",
 "Parameters": {
     "VpcId": {
         "Type": "String",
         "Description": "The VPC that the service is running inside of"
     },
     "PublicSubnetIds": {
         "Type": "List<AWS::EC2::Subnet::Id>",
         "Description": "List of public subnet ID's to put the load balancer in"
     },
     "PrivateSubnetIds": {
         "Type": "List<AWS::EC2::Subnet::Id>",
         "Description": "List of private subnet ID's that the AWS VPC tasks are in"
     },
     "ClusterName": {
         "Type": "String",
         "Description": "The name of the ECS cluster into which to launch
capacity."
     },
     "ECSTaskExecutionRole": {
         "Type": "String",
         "Description": "The role used to start up an ECS task"
     },
     "CapacityProvider": {
         "Type": "String",
         "Description": "The cluster capacity provider that the service should use
to request capacity when it wants to start up a task"
     },
     "ServiceName": {
         "Type": "String",
         "Default": "web",
         "Description": "A name for the service"
     },
     "ImageUrl": {
         "Type": "String",
         "Default": "public.ecr.aws/docker/library/nginx:latest",
         "Description": "The url of a docker image that contains the application
process that will handle the traffic for this service"
     },
     "ContainerCpu": {
         "Type": "Number",
         "Default": 256,
         "Description": "How much CPU to give the container. 1024 is 1 CPU"
```

```
        },
        "ContainerMemory": {
            "Type": "Number",
            "Default": 512,
            "Description": "How much memory in megabytes to give the container"
        },
        "ContainerPort": {
            "Type": "Number",
            "Default": 80,
            "Description": "What port that the application expects traffic on"
        },
        "DesiredCount": {
            "Type": "Number",
            "Default": 2,
            "Description": "How many copies of the service task to run"
        }
    },
    "Resources": {
        "TaskDefinition": {
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "Family": {
                    "Ref": "ServiceName"
                },
                "Cpu": {
                    "Ref": "ContainerCpu"
                },
                "Memory": {
                    "Ref": "ContainerMemory"
                },
                "NetworkMode": "awsvpc",
                "RequiresCompatibilities": [
                    "EC2"
                ],
                "ExecutionRoleArn": {
                    "Ref": "ECSTaskExecutionRole"
                },
                "ContainerDefinitions": [
                    {
                        "Name": {
                            "Ref": "ServiceName"
                        },
                        "Cpu": {
                            "Ref": "ContainerCpu"
```

```
                    },
                    "Memory": {
                        "Ref": "ContainerMemory"
                    },
                    "Image": {
                        "Ref": "ImageUrl"
                    },
                    "PortMappings": [
                        {
                            "ContainerPort": {
                                "Ref": "ContainerPort"
                            },
                            "HostPort": {
                                "Ref": "ContainerPort"
                            }
                        }
                    ],
                    "LogConfiguration": {
                        "LogDriver": "awslogs",
                        "Options": {
                            "mode": "non-blocking",
                            "max-buffer-size": "25m",
                            "awslogs-group": {
                                "Ref": "LogGroup"
                            },
                            "awslogs-region": {
                                "Ref": "AWS::Region"
                            },
                            "awslogs-stream-prefix": {
                                "Ref": "ServiceName"
                            }
                        }
                    }
                }
            ]
        }
    },
    "Service": {
        "Type": "AWS::ECS::Service",
        "DependsOn": "PublicLoadBalancerListener",
        "Properties": {
            "ServiceName": {
                "Ref": "ServiceName"
            },
```

```
                "Cluster": {
                    "Ref": "ClusterName"
                },
                "PlacementStrategies": [
                    {
                        "Field": "attribute:ecs.availability-zone",
                        "Type": "spread"
                    },
                    {
                        "Field": "cpu",
                        "Type": "binpack"
                    }
                ],
                "CapacityProviderStrategy": [
                    {
                        "Base": 0,
                        "CapacityProvider": {
                            "Ref": "CapacityProvider"
                        },
                        "Weight": 1
                    }
                ],
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "SecurityGroups": [
                            {
                                "Ref": "ServiceSecurityGroup"
                            }
                        ],
                        "Subnets": {
                            "Ref": "PrivateSubnetIds"
                        }
                    }
                },
                "DeploymentConfiguration": {
                    "MaximumPercent": 200,
                    "MinimumHealthyPercent": 75
                },
                "DesiredCount": {
                    "Ref": "DesiredCount"
                },
                "TaskDefinition": {
                    "Ref": "TaskDefinition"
                },
```

```
            "LoadBalancers": [
                {
                    "ContainerName": {
                        "Ref": "ServiceName"
                    },
                    "ContainerPort": {
                        "Ref": "ContainerPort"
                    },
                    "TargetGroupArn": {
                        "Ref": "ServiceTargetGroup"
                    }
                }
            ]
        }
    },
    "ServiceSecurityGroup": {
        "Type": "AWS::EC2::SecurityGroup",
        "Properties": {
            "GroupDescription": "Security group for service",
            "VpcId": {
                "Ref": "VpcId"
            }
        }
    },
    "ServiceTargetGroup": {
        "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
        "Properties": {
            "HealthCheckIntervalSeconds": 6,
            "HealthCheckPath": "/",
            "HealthCheckProtocol": "HTTP",
            "HealthCheckTimeoutSeconds": 5,
            "HealthyThresholdCount": 2,
            "TargetType": "ip",
            "Port": {
                "Ref": "ContainerPort"
            },
            "Protocol": "HTTP",
            "UnhealthyThresholdCount": 10,
            "VpcId": {
                "Ref": "VpcId"
            },
            "TargetGroupAttributes": [
                {
                    "Key": "deregistration_delay.timeout_seconds",
```

```
                    "Value": 0
                }
            ]
        }
    },
    "PublicLoadBalancerSG": {
        "Type": "AWS::EC2::SecurityGroup",
        "Properties": {
            "GroupDescription": "Access to the public facing load balancer",
            "VpcId": {
                "Ref": "VpcId"
            },
            "SecurityGroupIngress": [
                {
                    "CidrIp": "0.0.0.0/0",
                    "IpProtocol": -1
                }
            ]
        }
    },
    "PublicLoadBalancer": {
        "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
        "Properties": {
            "Scheme": "internet-facing",
            "LoadBalancerAttributes": [
                {
                    "Key": "idle_timeout.timeout_seconds",
                    "Value": "30"
                }
            ],
            "Subnets": {
                "Ref": "PublicSubnetIds"
            },
            "SecurityGroups": [
                {
                    "Ref": "PublicLoadBalancerSG"
                }
            ]
        }
    },
    "PublicLoadBalancerListener": {
        "Type": "AWS::ElasticLoadBalancingV2::Listener",
        "Properties": {
            "DefaultActions": [
```

```
                    {
                        "Type": "forward",
                        "ForwardConfig": {
                            "TargetGroups": [
                                {
                                    "TargetGroupArn": {
                                        "Ref": "ServiceTargetGroup"
                                    },
                                    "Weight": 100
                                }
                            ]
                        }
                    }
                ],
                "LoadBalancerArn": {
                    "Ref": "PublicLoadBalancer"
                },
                "Port": 80,
                "Protocol": "HTTP"
            }
        },
        "ServiceIngressfromLoadBalancer": {
            "Type": "AWS::EC2::SecurityGroupIngress",
            "Properties": {
                "Description": "Ingress from the public ALB",
                "GroupId": {
                    "Ref": "ServiceSecurityGroup"
                },
                "IpProtocol": -1,
                "SourceSecurityGroupId": {
                    "Ref": "PublicLoadBalancerSG"
                }
            }
        },
        "LogGroup": {
            "Type": "AWS::Logs::LogGroup"
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
```

```
Description: >-
  An example service that deploys in AWS VPC networking mode on EC2 capacity.
  Service uses a capacity provider to request EC2 instances to run on. Service
  runs with networking in private subnets, but still accessible to the internet
  via a load balancer hosted in public subnets.
Parameters:
  VpcId:
    Type: String
    Description: The VPC that the service is running inside of
  PublicSubnetIds:
    Type: 'List<AWS::EC2::Subnet::Id>'
    Description: List of public subnet ID's to put the load balancer in
  PrivateSubnetIds:
    Type: 'List<AWS::EC2::Subnet::Id>'
    Description: List of private subnet ID's that the AWS VPC tasks are in
  ClusterName:
    Type: String
    Description: The name of the ECS cluster into which to launch capacity.
  ECSTaskExecutionRole:
    Type: String
    Description: The role used to start up an ECS task
  CapacityProvider:
    Type: String
    Description: >-
      The cluster capacity provider that the service should use to request
      capacity when it wants to start up a task
  ServiceName:
    Type: String
    Default: web
    Description: A name for the service
  ImageUrl:
    Type: String
    Default: 'public.ecr.aws/docker/library/nginx:latest'
    Description: >-
      The url of a docker image that contains the application process that will
      handle the traffic for this service
  ContainerCpu:
    Type: Number
    Default: 256
    Description: How much CPU to give the container. 1024 is 1 CPU
  ContainerMemory:
    Type: Number
    Default: 512
    Description: How much memory in megabytes to give the container
```

```
  ContainerPort:
    Type: Number
    Default: 80
    Description: What port that the application expects traffic on
  DesiredCount:
    Type: Number
    Default: 2
    Description: How many copies of the service task to run
Resources:
  TaskDefinition:
    Type: 'AWS::ECS::TaskDefinition'
    Properties:
      Family: !Ref ServiceName
      Cpu: !Ref ContainerCpu
      Memory: !Ref ContainerMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - EC2
      ExecutionRoleArn: !Ref ECSTaskExecutionRole
      ContainerDefinitions:
        - Name: !Ref ServiceName
          Cpu: !Ref ContainerCpu
          Memory: !Ref ContainerMemory
          Image: !Ref ImageUrl
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              HostPort: !Ref ContainerPort
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
              awslogs-stream-prefix: !Ref ServiceName
  Service:
    Type: AWS::ECS::Service
    DependsOn: PublicLoadBalancerListener
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ClusterName
      PlacementStrategies:
        - Field: 'attribute:ecs.availability-zone'
          Type: spread
```

```
        - Field: cpu
          Type: binpack
      CapacityProviderStrategy:
        - Base: 0
          CapacityProvider: !Ref CapacityProvider
          Weight: 1
      NetworkConfiguration:
        AwsvpcConfiguration:
          SecurityGroups:
            - !Ref ServiceSecurityGroup
          Subnets: !Ref PrivateSubnetIds
      DeploymentConfiguration:
        MaximumPercent: 200
        MinimumHealthyPercent: 75
      DesiredCount: !Ref DesiredCount
      TaskDefinition: !Ref TaskDefinition
      LoadBalancers:
        - ContainerName: !Ref ServiceName
          ContainerPort: !Ref ContainerPort
          TargetGroupArn: !Ref ServiceTargetGroup
  ServiceSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Security group for service
      VpcId: !Ref VpcId
  ServiceTargetGroup:
    Type: 'AWS::ElasticLoadBalancingV2::TargetGroup'
    Properties:
      HealthCheckIntervalSeconds: 6
      HealthCheckPath: /
      HealthCheckProtocol: HTTP
      HealthCheckTimeoutSeconds: 5
      HealthyThresholdCount: 2
      TargetType: ip
      Port: !Ref ContainerPort
      Protocol: HTTP
      UnhealthyThresholdCount: 10
      VpcId: !Ref VpcId
      TargetGroupAttributes:
        - Key: deregistration_delay.timeout_seconds
          Value: 0
  PublicLoadBalancerSG:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
```

```
      GroupDescription: Access to the public facing load balancer
      VpcId: !Ref VpcId
      SecurityGroupIngress:
        - CidrIp: 0.0.0.0/0
          IpProtocol: -1
  PublicLoadBalancer:
    Type: 'AWS::ElasticLoadBalancingV2::LoadBalancer'
    Properties:
      Scheme: internet-facing
      LoadBalancerAttributes:
        - Key: idle_timeout.timeout_seconds
          Value: '30'
      Subnets: !Ref PublicSubnetIds
      SecurityGroups:
        - !Ref PublicLoadBalancerSG
  PublicLoadBalancerListener:
    Type: 'AWS::ElasticLoadBalancingV2::Listener'
    Properties:
      DefaultActions:
        - Type: forward
          ForwardConfig:
            TargetGroups:
              - TargetGroupArn: !Ref ServiceTargetGroup
                Weight: 100
      LoadBalancerArn: !Ref PublicLoadBalancer
      Port: 80
      Protocol: HTTP
  ServiceIngressfromLoadBalancer:
    Type: 'AWS::EC2::SecurityGroupIngress'
    Properties:
      Description: Ingress from the public ALB
      GroupId: !Ref ServiceSecurityGroup
      IpProtocol: -1
      SourceSecurityGroupId: !Ref PublicLoadBalancerSG
  LogGroup:
    Type: 'AWS::Logs::LogGroup'
```

Roles de IAM para Amazon ECS

Puede utilizar plantillas de AWS CloudFormation para crear roles de IAM y utilizarlos con Amazon ECS. Para obtener más información sobre los roles de IAM para Amazon ECS, consulte Roles de IAM para Amazon ECS.

Rol de ejecución de tareas de Amazon ECS

El rol de ejecución de tareas concede al agente de contenedor de Amazon ECS y al agente de Fargate permiso para realizar llamadas a la API de AWS en su nombre. El rol es necesario en función de los requisitos de la tarea. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

La siguiente plantilla se puede utilizar para crear un rol de ejecución de tareas sencillo que utilice la política administrada `AmazonECSTaskExecutionRolePolicy`.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "CloudFormation template for ECS Task Execution Role",
  "Resources": {
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": ["ecs-tasks.amazonaws.com"]
              },
              "Action": ["sts:AssumeRole"],
              "Condition": {
                "ArnLike": {
                  "aws:SourceArn": {
                    "Fn::Sub": "arn:aws:ecs:${AWS::Region}:${AWS::AccountId}:*"
                  }
                },
                "StringEquals": {
                  "aws:SourceAccount": {
                    "Ref": "AWS::AccountId"
                  }
                }
              }
            }
          ]
        },
        "Path": "/",
```

```
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
      }
    }
  },
  "Outputs": {
    "ECSTaskExecutionRoleARN": {
      "Description": "ARN of the ECS Task Execution Role",
      "Value": {
        "Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]
      },
      "Export": {
        "Name": {
          "Fn::Sub": "${AWS::StackName}-ECSTaskExecutionRoleARN"
        }
      }
    },
    "ECSTaskExecutionRoleName": {
      "Description": "Name of the ECS Task Execution Role",
      "Value": {
        "Ref": "ECSTaskExecutionRole"
      },
      "Export": {
        "Name": {
          "Fn::Sub": "${AWS::StackName}-ECSTaskExecutionRoleName"
        }
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: 'CloudFormation template for ECS Task Execution Role'
Resources:
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
```

```
            Principal:
              Service: [ecs-tasks.amazonaws.com]
            Action: ['sts:AssumeRole']
            Condition:
              ArnLike:
                aws:SourceArn: !Sub arn:aws:ecs:${AWS::Region}:${AWS::AccountId}:*
              StringEquals:
                aws:SourceAccount: !Ref AWS::AccountId
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
Outputs:
  ECSTaskExecutionRoleARN:
    Description: ARN of the ECS Task Execution Role
    Value: !GetAtt ECSTaskExecutionRole.Arn
    Export:
      Name: !Sub "${AWS::StackName}-ECSTaskExecutionRoleARN"
  ECSTaskExecutionRoleName:
    Description: Name of the ECS Task Execution Role
    Value: !Ref ECSTaskExecutionRole
    Export:
      Name: !Sub "${AWS::StackName}-ECSTaskExecutionRoleName"
```

Creación de recursos de Amazon ECS con la consola de AWS CloudFormation

Una forma de utilizar Amazon ECS con AWS CloudFormation es a través de la AWS Management Console. Aquí puede crear sus pilas de AWS CloudFormation para los componentes de Amazon ECS, como definiciones de tareas, clústeres y servicios, e implementarlos directamente desde la consola. El siguiente tutorial muestra cómo usar la consola de AWS CloudFormation para crear un servicio, una definición de tareas y un clúster de Amazon ECS.

Requisitos previos

En este tutorial se supone que los siguientes requisitos previos se han completado.

- Se han completado los pasos que se indican en Configuración para utilizar Amazon ECS.
- Su usuario de IAM dispone de los permisos requeridos que se especifican en la política de IAM AmazonECS_FullAccess de ejemplo.

Paso 1: creación de una plantilla de pila

Siga estos pasos para crear una plantilla de pila de AWS CloudFormation para un servicio de Amazon ECS y otros recursos relacionados.

1. Con el editor de texto que desee, cree un archivo llamado `ecs-tutorial-template.yaml`.
2. En el archivo `ecs-tutorial-template.yaml`, pegue la siguiente plantilla y guarde los cambios.

```
AWSTemplateFormatVersion: 2010-09-09
Description: A template that deploys an application that is built on an Apache
 web server Docker image by creating an Amazon ECS cluster, task definition, and
 service. The template also creates networking and logging resources, and an Amazon
 ECS task execution role.
Parameters:
  ClusterName:
    Type: String
    Default: CFNCluster
    Description: Name of the ECS Cluster
  TaskFamily:
    Type: String
    Default: task-definition-cfn
    Description: Family name for the Task Definition
  ServiceName:
    Type: String
    Default: cfn-service
    Description: Name of the ECS Service
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  TaskCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the task
  TaskMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 4096, 8192, 16384]
    Description: Memory (in MiB) for the task
  DesiredCount:
```

```
    Type: Number
    Default: 1
    Description: Desired number of tasks to run
  LogGroupName:
    Type: String
    Default: /ecs/fargate-task-definition
    Description: CloudWatch Log Group name
  VpcCidr:
    Type: String
    Default: 10.0.0.0/16
    Description: CIDR block for the VPC
  PublicSubnet1Cidr:
    Type: String
    Default: 10.0.0.0/24
    Description: CIDR block for public subnet 1
  PublicSubnet2Cidr:
    Type: String
    Default: 10.0.1.0/24
    Description: CIDR block for public subnet 2
Resources:
  # VPC and Networking Resources
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCidr
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-VPC
  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-IGW
  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC
  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
```

```
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PublicSubnet1Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet1
  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PublicSubnet2Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet2
  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicRouteTable
  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1
  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2
  # Security Group
  ECSSecurityGroup:
    Type: AWS::EC2::SecurityGroup
```

```
    Properties:
      GroupDescription: Security group for ECS tasks
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0
  # IAM Roles
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
  # CloudWatch Logs
  TaskLogGroup:
    Type: AWS::Logs::LogGroup
    DeletionPolicy: Retain
    UpdateReplacePolicy: Retain
    Properties:
      LogGroupName: !Ref LogGroupName
      RetentionInDays: 30
  # ECS Resources
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ClusterName
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
```

```
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"s
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroupName
              awslogs-region: !Ref 'AWS::Region'
              awslogs-stream-prefix: ecs
          Name: sample-fargate-app
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
              Protocol: tcp
      Cpu: !Ref TaskCpu
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      Family: !Ref TaskFamily
      Memory: !Ref TaskMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX
  ECSService:
    Type: AWS::ECS::Service
    DependsOn:
      - PublicSubnet1RouteTableAssociation
      - PublicSubnet2RouteTableAssociation
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ECSCluster
      DesiredCount: !Ref DesiredCount
```

```
      LaunchType: FARGATE
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups:
            - !Ref ECSSecurityGroup
          Subnets:
            - !Ref PublicSubnet1
            - !Ref PublicSubnet2
      TaskDefinition: !Ref ECSTaskDefinition
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref ECSCluster
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref ECSTaskDefinition
  ServiceName:
    Description: The name of the ECS service
    Value: !Ref ECSService
  VpcId:
    Description: The ID of the VPC
    Value: !Ref VPC
  PublicSubnet1:
    Description: The ID of public subnet 1
    Value: !Ref PublicSubnet1
  PublicSubnet2:
    Description: The ID of public subnet 2
    Value: !Ref PublicSubnet2
  SecurityGroup:
    Description: The ID of the security group
    Value: !Ref ECSSecurityGroup
  ExecutionRoleArn:
    Description: The ARN of the task execution role
    Value: !GetAtt ECSTaskExecutionRole.Arn
```

Paso 2: creación de una pila para recursos de Amazon ECS

Después de crear un archivo para la plantilla, puede seguir estos pasos para crear una pila con la plantilla mediante la consola de AWS CloudFormation.

1. Inicie sesión en la AWS Management Console y abra la consola de AWS CloudFormation en https://console.aws.amazon.com/cloudformation.
2. En la página Pilas, en la parte superior derecha, elija Crear pila y, a continuación, elija con recursos nuevos (estándar).
3. Elija Elegir una plantilla existente.
4. Elija Cargar un archivo de plantilla y luego elija Elegir archivo para seleccionar el archivo `ecs-tutorial-template`.

   Después de cargar el archivo en un bucket de Amazon S3, puede elegir Ver en Infrastructure Composer para visualizar la plantilla en Infrastructure Composer. Para obtener más información sobre las plantillas de AWS CloudFormation y sobre Infrastructure Composer, consulte Creación visual de plantillas con Infrastructure Composer en la Guía del usuario de AWS CloudFormation.
5. Elija Siguiente.
6. En la página Especificar los detalles de la pila, en Nombre de la pila, proporcione el siguiente nombre para la pila: `ecs-tutorial-stack`. Deje todos los valores de los parámetros en Parámetros como valores predeterminados y, a continuación, seleccione Siguiente.
7. En la página Configurar opciones de pila, en Capacidades, seleccione la casilla de verificación para confirmar la creación de recursos de IAM por parte AWS CloudFormation. Esta confirmación es necesaria para la creación del rol de ejecución de tareas de Amazon ECS tal como se define en la plantilla. Mantenga todas las demás configuraciones en sus valores predeterminados y seleccione Siguiente.
8. Revise los detalles de la pila en la página Revisar y crear y, a continuación, seleccione Enviar para iniciar la creación de la pila.

Paso 3: verificación

Siga los pasos que se indican a continuación para verificar la creación de los recursos de Amazon ECS mediante la plantilla proporcionada.

1. Inicie sesión en la AWS Management Console y abra la consola de AWS CloudFormation en https://console.aws.amazon.com/cloudformation.
2. En la página Pilas, elija ecs-tutorial-stack.
3. Seleccione la pestaña Eventos. Si los estados del evento dicen `CREATE_IN_PROGRESS`, espera hasta que se complete la creación y los estados cambien a `CREATE_COMPLETE`.

4. Cuando los estados del evento cambien a `CREATE_COMPLETE`, selecciona la pestaña Recursos. Verá los recursos con el ID lógico `ECSCluster`, `ECSTaskDefinition` y `ECSService` respectivamente.
5. Para verificar la creación de un clúster de Amazon ECS, elija el ID físico asociado a `ECSCluster`. Se le redirigirá a la consola de Amazon ECS, donde podrá ver el clúster creado denominado `CFNCluster`.
6. Para verificar la creación de un servicio de Amazon ECS, elija el ID físico asociado a `ECSService`. Se le redirigirá a la consola de Amazon ECS, donde podrá ver el servicio creado denominado `cfn-service` que se ha creado en el clúster `cfnCluster`.
7. Para verificar la creación de una definición de tarea de Amazon ECS, elija el ID físico asociado a `ECSTaskDefinition`. Se le redirigirá a la consola de Amazon ECS, donde podrá ver la revisión de la definición de tareas con el nombre `task-definition-cfn`.

Paso 4: limpie los recursos

Para limpiar los recursos y evitar incurrir en costos adicionales, siga estos pasos.

1. Inicie sesión en la AWS Management Console y abra la consola de AWS CloudFormation en https://console.aws.amazon.com/cloudformation.
2. En la página Pilas, elija ecs-tutorial-stack.
3. Elija Eliminar.
4. Cuando se le pida confirmación, elija Eliminar de nuevo.
5. Seleccione la pestaña Eventos. El Estado de los cambios de `ecs-tutorial-stack` cambia a DELETE_IN_PROGRESS y, después, a DELETE_COMPLETE una vez que los recursos se eliminen o cancelen el registro. La eliminación tarda un par de minutos.
6. Elija la pestaña Recursos. Ahora verá una lista de ID lógicos con el Estado actualizado a DELETE_COMPLETE.

Creación de recursos de Amazon ECS con comandos de la AWS CLI para AWS CloudFormation

El siguiente tutorial muestra cómo puede utilizar la AWS CLI para crear recursos de Amazon ECS con una plantilla de AWS CloudFormation.

Requisitos previos

- Se han completado los pasos que se indican en Configuración para utilizar Amazon ECS.
- Su usuario de IAM dispone de los permisos requeridos que se especifican en la política de IAM AmazonECS_FullAccess de ejemplo.

Paso 1: creación de una pila

Para crear una nueva pila mediante la AWS CLI, siga estos pasos.

1. Con el editor de texto que desee, cree un archivo llamado `ecs-tutorial-template.yaml`.
2. En el archivo `ecs-tutorial-template.yaml`, pegue la siguiente plantilla y guarde los cambios.

```
AWSTemplateFormatVersion: 2010-09-09
Description: A template that deploys an application that is built on an Apache
 web server Docker image by creating an Amazon ECS cluster, task definition, and
 service. The template also creates networking and logging resources, and an Amazon
 ECS task execution role.
Parameters:
  ClusterName:
    Type: String
    Default: CFNCluster
    Description: Name of the ECS Cluster
  TaskFamily:
    Type: String
    Default: task-definition-cfn
    Description: Family name for the Task Definition
  ServiceName:
    Type: String
    Default: cfn-service
    Description: Name of the ECS Service
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  TaskCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the task
```

```
  TaskMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 4096, 8192, 16384]
    Description: Memory (in MiB) for the task
  DesiredCount:
    Type: Number
    Default: 1
    Description: Desired number of tasks to run
  LogGroupName:
    Type: String
    Default: /ecs/fargate-task-definition
    Description: CloudWatch Log Group name
  VpcCidr:
    Type: String
    Default: 10.0.0.0/16
    Description: CIDR block for the VPC
  PublicSubnet1Cidr:
    Type: String
    Default: 10.0.0.0/24
    Description: CIDR block for public subnet 1
  PublicSubnet2Cidr:
    Type: String
    Default: 10.0.1.0/24
    Description: CIDR block for public subnet 2
Resources:
  # VPC and Networking Resources
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCidr
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-VPC
  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-IGW
  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
```

```
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC
  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PublicSubnet1Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet1
  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PublicSubnet2Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet2
  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicRouteTable
  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1
  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
```

```
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2
  # Security Group
  ECSSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Security group for ECS tasks
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0
  # IAM Roles
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
  # CloudWatch Logs
  TaskLogGroup:
    Type: AWS::Logs::LogGroup
    DeletionPolicy: Retain
    UpdateReplacePolicy: Retain
    Properties:
      LogGroupName: !Ref LogGroupName
      RetentionInDays: 30
  # ECS Resources
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ClusterName
```

```
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"s
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroupName
              awslogs-region: !Ref 'AWS::Region'
              awslogs-stream-prefix: ecs
          Name: sample-fargate-app
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
              Protocol: tcp
      Cpu: !Ref TaskCpu
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      Family: !Ref TaskFamily
      Memory: !Ref TaskMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX
  ECSService:
    Type: AWS::ECS::Service
    DependsOn:
```

```
      - PublicSubnet1RouteTableAssociation
      - PublicSubnet2RouteTableAssociation
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ECSCluster
      DesiredCount: !Ref DesiredCount
      LaunchType: FARGATE
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups:
            - !Ref ECSSecurityGroup
          Subnets:
            - !Ref PublicSubnet1
            - !Ref PublicSubnet2
      TaskDefinition: !Ref ECSTaskDefinition
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref ECSCluster
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref ECSTaskDefinition
  ServiceName:
    Description: The name of the ECS service
    Value: !Ref ECSService
  VpcId:
    Description: The ID of the VPC
    Value: !Ref VPC
  PublicSubnet1:
    Description: The ID of public subnet 1
    Value: !Ref PublicSubnet1
  PublicSubnet2:
    Description: The ID of public subnet 2
    Value: !Ref PublicSubnet2
  SecurityGroup:
    Description: The ID of the security group
    Value: !Ref ECSSecurityGroup
  ExecutionRoleArn:
    Description: The ARN of the task execution role
    Value: !GetAtt ECSTaskExecutionRole.Arn
```

3. Tras crear un archivo de plantilla, utilice el siguiente comando para crear una pila. El indicador `--capabilities` es obligatorio para crear un rol de ejecución de tareas de Amazon ECS tal y

como se especifica en la plantilla. También puede especificar el indicador `--parameters` para personalizar los parámetros de la plantilla.

```
aws cloudformation create-stack \
      --stack-name ecs-stack \
      --template-body file://ecs-tutorial-template.yaml \
      --capabilities CAPABILITY_IAM
```

Paso 2: verificación de la creación de recursos

Para asegurarse de que los recursos se creen correctamente, siga estos pasos. También puede usar la consola de Amazon ECS.

1. Ejecute el siguiente comando para que se muestren todas las definiciones de tarea en una Región de AWS.

   ```
   aws ecs list-task-definitions
   ```

   El comando devuelve una lista de nombres de recurso de Amazon (ARN) de definiciones de tareas. El ARN de la definición de tarea que creó con la plantilla se mostrará en el siguiente formato.

   ```
   {
       "taskDefinitionArns": [
        .....
           "arn:aws:ecs:aws-region:111122223333:task-definition/task-definition-
   cfn:1",
        .....
       ]
   }
   ```

2. Ejecute el siguiente comando para que se enumeren todos los clústeres en una Región de AWS.

   ```
   aws ecs list-clusters
   ```

   El comando devuelve una lista de ARN de clústeres. El ARN del clúster que creó con la plantilla se mostrará en el siguiente formato.

   ```
   {
   ```

```
    "clusterArns": [
        .....
        "arn:aws:ecs:aws-region:111122223333:cluster/CFNCluster",
        .....
    ]
}
```

3. Ejecute el siguiente comando para enumerar todos los servicios que se ejecutan en el clúster `CFNCluster`.

```
aws ecs list-services \
      --cluster CFNCluster
```

El comando devuelve una lista de ARN de servicios. El ARN del servicio que creó con la plantilla se mostrará en el siguiente formato.

```
{
    "serviceArns": [
        "arn:aws:ecs:aws-region:111122223333:service/CFNCluster/cfn-service"
    ]
}
```

Paso 3: Limpieza

Para limpiar los recursos que ha creado, ejecute el siguiente comando.

```
aws cloudformation delete-stack \
      --stack-name ecs-stack
```

El comando `delete-stack` inicia la eliminación de la pila AWS CloudFormation que se creó en este tutorial y, por tanto, elimina todos los recursos de la pila. Para verificar la eliminación, puede repetir el procedimiento en Paso 2: verificación de la creación de recursos. La lista de ARN de los resultados ya no incluirá una definición de tarea denominada `task-definition-cfn` ni un clúster denominado `CFNCLuster`. La llamada `list-services` fallará.

Prácticas recomendadas para Amazon ECS

Puede utilizar cualquiera de las siguientes páginas para conocer las prácticas recomendadas operativas más importantes para las redes de Amazon ECS.

Vista general de las prácticas recomendadas	Más información
Conexión de las aplicaciones a Internet	Conexión de las aplicaciones de Amazon ECS a Internet
Reciba conexiones entrantes a Amazon ECS de Internet.	Prácticas recomendadas para recibir conexiones entrantes a Amazon ECS desde Internet
Conecte Amazon ECS con otros servicios de AWS dentro de una VPC	Prácticas recomendadas para conectar Amazon ECS a los servicios de AWS desde su VPC
Servicios de red entre Cuentas de AWS y VPC	Prácticas recomendadas para conectar en red los servicios de Amazon ECS entre Cuentas de AWS y las VPC
Solución de problemas de red	Servicios de AWS para la solución de problemas de redes de Amazon ECS

Puede utilizar cualquiera de las siguientes páginas para conocer las prácticas recomendadas operativas más importantes para Fargate en Amazon ECS.

Vista general de las prácticas recomendadas	Más información	
Seguridad en Fargate	Prácticas recomendadas sobre seguridad de Fargate en Amazon ECS	
Consideraciones de seguridad de Fargate	Consideraciones de seguridad de Fargate para Amazon ECS	
Comportamiento de extracción de imágenes de contenedores de Linux en un contenedor de Fargate	Comportamiento de extracción de imágenes de contenedores de Linux en Fargate para Amazon ECS	
Comportamiento de extracción de imágenes de contenedores Windows en un contenedor de Fargate	Comportamiento de extracción de imágenes de contenedor de Windows en Fargate para Amazon ECS	
Retirada de tareas de Fargate	Retirada y mantenimiento de tareas para AWS Fargate en Amazon ECS	

Puede utilizar cualquiera de las siguientes páginas para conocer las prácticas recomendadas operativas más importantes para las definiciones de tareas.

Vista general de las prácticas recomendadas	Más información	
Imágenes de contenedor	Prácticas recomendadas para las imágenes de contenedores de Amazon ECS	
Tamaño de tarea	Prácticas recomendadas para los tamaños de las tareas de Amazon ECS	

Vista general de las prácticas recomendadas	Más información	
Prácticas recomendadas de volumen	Opciones de almacenamiento para las tareas de Amazon ECS	

Puede utilizar cualquiera de las siguientes páginas para conocer las prácticas recomendadas operativas más importantes para clústeres y capacidad.

Vista general de las prácticas recomendadas	Más información	
Seguridad en Fargate	Prácticas recomendadas sobre seguridad de Fargate en Amazon ECS	
Consideraciones de seguridad en instancias de contenedor de EC2	Consideraciones de seguridad en instancias de contenedor de Amazon EC2 para Amazon ECS	
Escalado automático de clústeres	Optimización del escalado automático de clústeres de Amazon ECS	
Uso a escala	Uso de Amazon ECS a escala	
Escalado automático y administración de la capacidad	Prácticas recomendadas de escalado automátic o y administración de la capacidad de Amazon ECS	

Puede utilizar cualquiera de las siguientes páginas para conocer las prácticas recomendadas operativas más importantes para tareas y servicios.

Vista general de las prácticas recomendadas	Más información	
Optimización del tiempo de lanzamiento de tareas	Optimización del tiempo de lanzamiento de tareas de Amazon ECS	
Parámetros de servicio	Prácticas recomendadas para los parámetros de servicio de Amazon ECS	
Optimización de los parámetros de comprobación de estado del equilibrador de carga	Optimización de los parámetros de comprobación de estado del equilibrador de carga para Amazon ECS	
Optimización de los parámetros de drenaje de conexiones del equilibrador de carga	Optimización de los parámetros de drenaje de conexiones del equilibrador de carga para Amazon ECS	
Optimización del escalado automático de servicio	Optimización del escalado automático de servicios de Amazon ECS	

Puede utilizar cualquiera de las siguientes páginas para conocer las prácticas recomendadas operativas más importantes para seguridad.

Vista general de las prácticas recomendadas	Más información	
Seguridad de la red	Prácticas recomendadas de seguridad de red para Amazon ECS	
Seguridad de tareas y contenedores	Prácticas recomendadas de seguridad para las tareas y	

Vista general de las prácticas recomendadas	Más información	
	contenedores de Amazon ECS	

AWS Fargate para Amazon ECS

La tecnología AWS Fargate se puede utilizar en Amazon ECS para ejecutar contenedores sin tener que administrar servidores ni clústeres de instancias de Amazon EC2. Con AWS Fargate ya no tendrá que aprovisionar, configurar ni escalar clústeres de máquinas virtuales para ejecutar los contenedores. De esta manera, se elimina la necesidad de elegir tipos de servidores, decidir cuándo escalar los clústeres u optimizar conjuntos de clústeres.

Al ejecutar las tareas y servicios con el tipo de lanzamiento de Fargate, la aplicación se empaqueta en contenedores, se especifican los requisitos de CPU y de memoria, se definen las políticas de IAM y de redes y se lanza la aplicación. Cada tarea de Fargate tiene su propio límite de aislamiento y no comparte el kernel subyacente, los recursos de CPU, los recursos de memoria ni la interfaz de red elástica con otra tarea. Usted configura las definiciones de tareas para Fargate estableciendo el parámetro de definición de tareas `requiresCompatibilities` en `FARGATE`. Para obtener más información, consulte Tipos de lanzamiento.

Fargate ofrece versiones de la plataforma para Amazon Linux 2 (versión de la plataforma 1.3.0), el sistema operativo Bottlerocket (versión de la plataforma 1.4.0) y las ediciones Full y Core de Microsoft Windows Server 2019. A menos que se especifique lo contrario, la información de esta página se aplica a todas las plataformas Fargate.

En este tema, se describen los diferentes componentes de las tareas y los servicios de Fargate, y se mencionan consideraciones especiales para el uso de Fargate con Amazon ECS.

Para obtener información acerca de las regiones que admiten contenedores Linux, consulte the section called “Contenedores de Linux en AWS Fargate”.

Para obtener información acerca de las regiones que admiten contenedores de Windows en Fargate, consulte the section called “Contenedores de Windows en AWS Fargate”.

Tutoriales

Para obtener información sobre cómo empezar a utilizar la consola, consulte:

- Obtenga información sobre cómo crear una tarea de Linux de Amazon ECS para el tipo de lanzamiento de Fargate.
- Obtenga información sobre cómo crear una tarea de Windows de Amazon ECS para el tipo de lanzamiento de Fargate.

Para obtener información sobre cómo empezar a utilizar la AWS CLI, consulte:

- Creación de una tarea de Linux de Amazon ECS para el tipo de lanzamiento de Fargate con la AWS CLI
- Creación de una tarea de Amazon ECS de Windows para el tipo de lanzamiento de Fargate con la AWS CLI

Proveedores de capacidad

Los siguientes proveedores de capacidad están disponibles:

- Fargate
- Fargate Spot: ejecute tareas de Amazon ECS tolerantes a interrupciones con un descuento respecto al precio de AWS Fargate. Fargate Spot ejecuta las tareas en la capacidad de cómputo adicional. Cuando AWS necesita recuperar esa capacidad, las tareas se interrumpen previa advertencia con dos minutos de antelación. Para obtener más información, consulte Clústeres de Amazon ECS para Fargate.

Definiciones de tareas

Las tareas que utilizan el tipo de lanzamiento Fargate no admiten todos los parámetros de definición de tareas de Amazon ECS que están disponibles. Algunos parámetros directamente no son compatibles, y otros se comportan de forma distinta para tareas de Fargate. Para obtener más información, consulte Memoria y CPU de tarea.

Versiones de la plataforma

Las versiones de la plataforma AWS Fargate se utilizan para hacer referencia a un entorno en tiempo de ejecución específico para la infraestructura de tareas de Fargate. Se trata de una combinación de la versión del kernel y la versión del tiempo de ejecución del contenedor. Selecciona una versión de la plataforma cuando ejecuta una tarea o cuando crea un servicio para mantener varias tareas idénticas.

A medida que evoluciona el entorno de tiempo de ejecución, se lanzan nuevas revisiones de las versiones de la plataforma, por ejemplo, si hay actualizaciones del kernel o del sistema operativo, características nuevas, correcciones de errores o actualizaciones de seguridad. Una versión de

la plataforma de Fargate se actualiza mediante una nueva revisión de la versión de la plataforma. Durante su ciclo de vida, cada tarea se ejecuta en una revisión de la versión de la plataforma. Si desea utilizar la última revisión de la versión de la plataforma, debe iniciar una nueva tarea. Una tarea nueva que se ejecuta en Fargate siempre se ejecuta con la última revisión de la versión de la plataforma, lo que garantiza que las tareas se inicien siempre en una infraestructura segura y con parches.

Si se detecta un problema de seguridad que afecta a una versión de la plataforma existente, AWS crea una nueva revisión con parches de la versión de la plataforma y retira las tareas que se están ejecutando en la revisión vulnerable. En algunos casos, es posible que reciba una notificación de que se ha programado el retiro de sus tareas de Fargate. Para obtener más información, consulte Retirada y mantenimiento de tareas para AWS Fargate en Amazon ECS .

Para obtener más información, consulte Versiones de la plataforma Fargate para Amazon ECS.

Equilibrio de carga de los servicios

El servicio Amazon ECS en AWS Fargate se puede configurar opcionalmente para que utilice Elastic Load Balancing a fin de distribuir el tráfico de manera uniforme entre las tareas del servicio.

Los servicios de Amazon ECS alojados en AWS Fargate admiten los tipos de equilibrador de carga Equilibrador de carga de aplicación y Equilibrador de carga de red. Los Application Load Balancers se utilizan para dirigir el tráfico HTTP/HTTPS (o de capa 7). Los Network Load Balancers se utilizan para dirigir el tráfico TCP o UDP (o de capa 4). Para obtener más información, consulte Uso del equilibrador de carga para distribuir el tráfico de servicio de Amazon ECS.

Al crear un grupo de destino para estos servicios, se debe elegir `ip` como tipo de destino, no `instance`. Esto se debe a que las tareas que utilizan el modo de red `awsvpc` están asociadas a una interfaz de red elástica, no a una instancia de Amazon EC2. Para obtener más información, consulte Uso del equilibrador de carga para distribuir el tráfico de servicio de Amazon ECS.

El uso de un equilibrador de carga de red para direccionar el tráfico UDP a las tareas de Amazon ECS en AWS Fargate solo es compatible cuando se utiliza la versión 1.4 o posterior de la plataforma.

Métricas de uso

Puede utilizar las métricas de uso de CloudWatch para proporcionar visibilidad sobre el uso de los recursos de su cuenta. Utilice estas métricas para visualizar el uso actual del servicio en paneles y gráficos de CloudWatch.

Las métricas de uso de AWS Fargate se corresponden con las cuotas de servicio de AWS. Puede configurar alarmas que le avisen cuando su uso se acerque a una Service Quota. Para obtener más información acerca de las Service Quotas de AWS Fargate, consulte Puntos de conexión y cuotas de Amazon ECS en la Referencia general de Amazon Web Services.

Para obtener más información sobre las métricas de uso de AWS Fargate, consulte Métricas de uso de AWS Fargate.

Consideraciones de seguridad de Amazon ECS sobre cuándo utilizar el tipo de lanzamiento de Fargate

Recomendamos que los clientes que buscan un aislamiento sólido para sus tareas utilicen Fargate. Fargate ejecuta cada tarea en un entorno de virtualización de hardware. Esto garantiza que estas cargas de trabajo en contenedores no compartan interfaces de red, almacenamiento efímero de Fargate, CPU o memoria con otras tareas. Para obtener más información, consulte Security Overview of AWS Fargate.

Prácticas recomendadas sobre seguridad de Fargate en Amazon ECS

Al utilizar AWS Fargate, recomendamos que tenga en cuenta las siguientes prácticas recomendadas. Para obtener más orientación, consulte Descripción general de la seguridad de AWS Fargate.

Uso de AWS KMS para cifrar el almacenamiento efímero de Fargate

Debe cifrar su almacenamiento efímero mediante AWS KMS o mediante sua propiaa clavea administrada por el cliente. En el caso de las tareas alojadas en Fargate que utilizan la versión de la plataforma `1.4.0` o una posterior, cada una de ellas recibe 20 GiB de almacenamiento efímero. Para obtener más información, consulte clave administrada por el cliente (CMK). Puede aumentar la cantidad total de almacenamiento efímero, hasta un máximo de 200 GiB, especificando el parámetro `ephemeralStorage` en la definición de tareas. Para las tareas que se lanzaron el 28 de mayo de 2020 o después, el almacenamiento efímero se cifra con un algoritmo de cifrado AES-256 que utiliza una clave de cifrado administrada por Fargate.

Para obtener más información, consulte Opciones de almacenamiento para las tareas de Amazon ECS.

Ejemplo: lanzar una tarea de Fargate en la versión de la plataforma 1.4.0 con cifrado de almacenamiento efímero

El siguiente comando lanzará una tarea de Fargate en la versión de la plataforma 1.4. Como esta tarea se lanza como parte del clúster, utiliza los 20 GiB de almacenamiento efímero que se cifra automáticamente.

```
aws ecs run-task --cluster clustername \
  --task-definition taskdefinition:version \
  --count 1
  --launch-type "FARGATE" \
  --platform-version 1.4.0 \
  --network-configuration
 "awsvpcConfiguration={subnets=[subnetid],securityGroups=[securitygroupid]}" \
  --region region
```

Capacidad SYS_PTRACE para el seguimiento de llamadas al sistema del kernel con Fargate

Docker proporciona la configuración predeterminada de las capacidades de Linux que se agregan al contenedor o se eliminan de él.

Las tareas lanzadas en Fargate solo admiten la adición de la capacidad del kernel `SYS_PTRACE`.

En el siguiente video se muestra cómo utilizar esta característica mediante el proyecto Falco de Sysdig.

#ContainersFromTheCouch: solución de problemas de una tarea de Fargate con la capacidad SYS_PTRACE

El código mencionado en el video anterior se puede encontrar en GitHub aquí.

Uso de la supervisión en tiempo de ejecución de Amazon GuardDuty con Fargate

Amazon GuardDuty es un servicio de detección de amenazas que ayuda a proteger las cuentas, los contenedores, las cargas de trabajo y los datos de su entorno de AWS. Mediante modelos de machine learning (ML) y capacidades de detección de anomalías y amenazas, GuardDuty supervisa continuamente los diferentes orígenes de registro y la actividad en tiempo de ejecución para identificar y priorizar los posibles riesgos de seguridad y actividades maliciosas en su entorno.

La supervisión en tiempo de ejecución en GuardDuty protege las cargas de trabajo que se ejecutan en Fargate mediante la supervisión continua de la actividad de registro y red de AWS para identificar comportamientos malintencionados o no autorizados. La supervisión en tiempo de ejecución utiliza un agente de seguridad de GuardDuty ligero y totalmente administrado que analiza el comportamiento en el host (como el acceso a los archivos, la ejecución de procesos y las conexiones de red). Esto abarca problemas como el escalado de privilegios, el uso de credenciales expuestas, la comunicación con direcciones IP malintencionadas, los dominios y la presencia de malware en las cargas de trabajo de contenedores e instancias de Amazon EC2. Para obtener más información, consulte GuardDuty Runtime Monitoring en la Guía del usuario de GuardDuty.

Consideraciones de seguridad de Fargate para Amazon ECS

Cada tarea tiene una capacidad de infraestructura dedicada porque Fargate ejecuta cada carga de trabajo en un entorno virtual aislado. Las cargas de trabajo que se ejecutan en Fargate no comparten interfaces de red, almacenamiento efímero, CPU ni memoria con otras tareas. Puede ejecutar varios contenedores dentro de una tarea, incluidos los contenedores de aplicaciones y los contenedores sidecar, o simplemente los sidecar. Un sidecar es un contenedor que se ejecuta junto a un contenedor de aplicaciones en una tarea de Amazon ECS. Si bien el contenedor de aplicaciones ejecuta el código principal de la aplicación, los procesos que se ejecutan en los sidecar pueden mejorar la aplicación. Los sidecar le ayudan a dividir las funciones de la aplicación en contenedores dedicados, lo que facilita actualizar partes de la aplicación.

Los contenedores que forman parte de la misma tarea comparten recursos para el tipo de lanzamiento de Fargate, ya que estos contenedores siempre se ejecutan en el mismo host y comparten recursos informáticos. Estos contenedores también comparten el almacenamiento efímero proporcionado por Fargate. Los contenedores de Linux de una tarea comparten espacios de nombres de red, incluida la dirección IP y los puertos de red. Dentro de una tarea, los contenedores que pertenecen a la tarea pueden intercomunicarse por localhost.

El entorno de tiempo de ejecución de Fargate le impide utilizar determinadas características del controlador que son compatibles con las instancias de EC2. Tenga en cuenta lo siguiente cuando diseñe cargas de trabajo que se ejecutan en Fargate:

- Sin contenedores ni acceso privilegiados: características como los contenedores o el acceso privilegiados no están disponibles actualmente en Fargate. Esto afectará a los casos de uso, como la ejecución de Docker en Docker.
- Acceso limitado a las capacidades de Linux: el entorno en el que se ejecutan los contenedores en Fargate está bloqueado. Las capacidades adicionales de Linux, como CAP_SYS_ADMIN

y CAP_NET_ADMIN, están restringidas para evitar un escalado de privilegios. Fargate admite agregar la capacidad de Linux CAP_SYS_PTRACE a las tareas para permitir que las herramientas de observabilidad y seguridad implementadas dentro de la tarea supervisen la aplicación en contenedores.

- Sin acceso al host subyacente: ni los clientes ni los operadores de AWS pueden conectarse a un host que ejecute las cargas de trabajo de los clientes. Puede utilizar ECS exec para ejecutar comandos en un contenedor u obtener un intérprete de comandos en un contenedor que se ejecute en Fargate. Puede utilizar ECS exec como ayuda para recopilar información de diagnóstico para la depuración. Fargate también impide que los contenedores accedan a los recursos del host subyacente, como el sistema de archivos, los dispositivos, las redes y el tiempo de ejecución del contenedor.
- Redes: puede utilizar grupos de seguridad y ACL de red para controlar el tráfico entrante y saliente. Las tareas de Fargate reciben una dirección IP de la subred configurada en la VPC.

Versiones de la plataforma Fargate para Amazon ECS

Las versiones de la plataforma AWS Fargate se utilizan para hacer referencia a un entorno en tiempo de ejecución específico para la infraestructura de tareas de Fargate. Se trata de una combinación de la versión del kernel y la versión del tiempo de ejecución del contenedor. Selecciona una versión de la plataforma cuando ejecuta una tarea o cuando crea un servicio para mantener varias tareas idénticas.

A medida que evoluciona el entorno de tiempo de ejecución, se lanzan nuevas revisiones de las versiones de la plataforma, por ejemplo, si hay actualizaciones del kernel o del sistema operativo, características nuevas, correcciones de errores o actualizaciones de seguridad. Una versión de la plataforma de Fargate se actualiza mediante una nueva revisión de la versión de la plataforma. Durante su ciclo de vida, cada tarea se ejecuta en una revisión de la versión de la plataforma. Si desea utilizar la última revisión de la versión de la plataforma, debe iniciar una nueva tarea. Una tarea nueva que se ejecuta en Fargate siempre se ejecuta con la última revisión de la versión de la plataforma, lo que garantiza que las tareas se inicien siempre en una infraestructura segura y con parches.

Si se detecta un problema de seguridad que afecta a una versión de la plataforma existente, AWS crea una nueva revisión con parches de la versión de la plataforma y retira las tareas que se están ejecutando en la revisión vulnerable. En algunos casos, es posible que reciba una notificación de que se ha programado el retiro de sus tareas de Fargate. Para obtener más información, consulte Retirada y mantenimiento de tareas para AWS Fargate en Amazon ECS .

Se especifica la versión de la plataforma al ejecutar una tarea o implementar un servicio.

Tenga en cuenta lo siguiente al especificar una versión de plataforma:

- Puede especificar un número de versión específico, por ejemplo `1.4.0` o `LATEST`.

  La ÚLTIMA versión de la plataforma Linux es `1.4.0`.

  La ÚLTIMA versión de la plataforma Windows es `1.0.0`.
- Si desea actualizar la versión de la plataforma de un servicio, cree una implementación. Por ejemplo, supongamos que tiene un servicio que ejecuta tareas en la versión de la plataforma Linux `1.3.0`. Para cambiar el servicio de modo que ejecute tareas en la versión `1.4.0` de la plataforma Linux, puede actualizar el servicio y especificar una nueva versión de la plataforma. Sus tareas se vuelven a implementar con la versión de la plataforma más reciente y la revisión de ella más reciente. Para obtener más información sobre las implementaciones, consulte Servicios de Amazon ECS.
- Si su servicio se amplía sin actualizar la versión de la plataforma, esas tareas recibirán la versión de la plataforma especificada en la implementación actual del servicio. Por ejemplo, supongamos que tiene un servicio que ejecuta tareas en la versión de la plataforma Linux `1.3.0`. Si aumenta el recuento deseado del servicio, el programador de servicios inicia las nuevas tareas con la revisión más reciente de la versión de la plataforma `1.3.0`.
- Las tareas nuevas siempre se ejecutan en la última versión de la plataforma. Esto garantiza que las tareas se realicen siempre en una infraestructura segura y actualizada.
- Los números de versión de la plataforma para los contenedores de Linux y Windows en Fargate son independientes. Por ejemplo, el comportamiento, las características y el software utilizados en la versión de la plataforma `1.0.0` para los contenedores de Windows en Fargate no son comparables a los de la versión de plataforma `1.0.0` para los contenedores de Linux en Fargate.
- La siguiente información se aplica a las versiones de la plataforma Fargate Windows.

  Las imágenes de contenedores de Microsoft Windows Server se deben crear a partir de una versión específica de Windows Server. Debe seleccionar la misma versión de Windows Server en la `platformFamily` cuando ejecute una tarea o cree un servicio que coincida con la imagen del contenedor de Windows Server. Además, puede proporcionar un `operatingSystemFamily` que coincida en la definición de la tarea para evitar que las tareas se ejecuten en una versión de Windows incorrecta. Para obtener más información, consulte Coincidencia de la versión del host de contenedor con las versiones de las imágenes de contenedor en el sitio web de Microsoft Learn.

Migración a la versión 1.4.0 de la plataforma Linux en Amazon ECS

Considere los siguientes aspectos al migrar las tareas de Amazon ECS alojadas en Fargate de la versión `1.0.0`, `1.1.0`, `1.2.0` o `1.3.0` de la plataforma a la versión `1.4.0`. Como práctica recomendada, debe confirmar que la tarea funcione de la manera correcta en la versión `1.4.0` de la plataforma antes de migrar las tareas.

- Se ha actualizado el comportamiento del tráfico de red de entrada y salida de las tareas. A partir de la versión 1.4.0 de la plataforma, todas las tareas de Amazon ECS alojadas en Fargate reciben una única interfaz de red elástica (que se conoce como ENI de la tarea) y todo el tráfico de red fluye a través de esa ENI dentro de la VPC. Puede ver el tráfico a través de los registros de flujo de la VPC. Para obtener más información, consulte Opciones de redes de tareas de Amazon ECS para el tipo de lanzamiento de Fargate.
- Si utiliza puntos de conexión de VPC de interfaz, se deben tener en cuenta los siguientes aspectos.
  - Para las imágenes de contenedor alojadas en Amazon ECR, necesita los siguientes puntos de conexión. Para obtener más información, consulte Puntos de conexión de VCP de la interfaz de Amazon ECR (AWS PrivateLink) en la Guía del usuario de Amazon Elastic Container Registry.
    - com.amazonaws.***región***.ecr.dkr punto de conexión de VPC de Amazon ECR
    - com.amazonaws.***región***.ecr.api punto de conexión de VPC de Amazon ECR
    - Punto de conexión de puerta de enlace de Amazon S3
  - Cuando utilice una definición de tarea que haga referencia a secretos de Secrets Manager para recuperar información confidencial de los contenedores, debe crear los puntos de conexión de VPC de interfaz para Secrets Manager. Para obtener más información, consulte Utilización de Secrets Manager con puntos de enlace de la VPC en la Guía del usuario de AWS Secrets Manager.
  - Cuando utilice una definición de tarea que haga referencia a parámetros del almacén de parámetros de Systems Manager para recuperar información confidencial de los contenedores, debe crear los puntos de conexión de VPC de interfaz para Systems Manager. Para obtener más información, consulte Mejora de la seguridad de las instancias de EC2 mediante puntos de conexión de VPC para Systems Manager en la Guía del usuario de AWS Systems Manager.
  - El grupo de seguridad de la interfaz de red elástica (ENI) asociado a la tarea necesita a las reglas de grupo de seguridad para permitir el tráfico entre la tarea y los puntos de conexión de VPC.

Registro de cambios de la versión de la plataforma de Fargate Linux

Estas son las versiones disponibles de la plataforma de Linux. Para obtener información acerca de las versiones obsoletas de la plataforma, consulte Obsolecencia de la versión de la plataforma AWS Fargate Linux.

1.4.0

A continuación, se incluye el registro de cambios de la versión `1.4.0` de la plataforma.

- A partir del 5 de noviembre de 2020, las nuevas tarea de Amazon ECS lanzadas en Fargate mediante la versión `1.4.0` de la plataforma podrán utilizar las siguientes características:
  - Al usar Secrets Manager para almacenar información confidencial, puede introducir una clave JSON específica o una versión específica de un secreto como variable de entorno o en una configuración de registro. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.
  - Especifique variables de entorno de forma masiva mediante la el parámetro de definición de contenedor `environmentFiles`. Para obtener más información, consulte Transferencia de una variable de entorno individual a un contenedor de Amazon ECS.
  - A las tareas que se ejecuten en una VPC y una subred habilitada para IPv6 se les asignará una dirección IPv4 privada y una dirección IPv6. Para obtener más información, consulte Opciones de redes de tareas de Amazon ECS para el tipo de inicialización de Fargate.
  - El punto de enlace de metadatos de tareas versión 4 proporciona metadatos adicionales acerca de la tarea y el contenedor, incluido el tipo de lanzamiento de la tarea, el nombre de recurso de Amazon (ARN) del contenedor y el controlador de registro utilizado con sus opciones. Al consultar el punto de enlace `/stats`, también recibirá estadísticas de velocidad de red de sus contenedores. Para más información, consulte Task metadata endpoint version 4.
- A partir del 30 de julio de 2020, las nuevas tarea de Amazon ECS lanzadas en Fargate mediante la versión `1.4.0` de la plataforma podrán dirigir el tráfico UDP a través de un Network Load Balancer a las tareas de Amazon ECS alojadas en Fargate. Para obtener más información, consulte Uso del equilibrador de carga para distribuir el tráfico de servicio de Amazon ECS.
- A partir del 28 de mayo de 2020, las nuevas tareas de Amazon ECS que se lancen en Fargate mediante la versión `1.4.0` de la plataforma dispondrán de almacenamiento efímero cifrado con un algoritmo de cifrado AES-256 que utiliza una clave de cifrado de propiedad de AWS. Para obtener más información, consulte Almacenamiento efímero de tareas de Fargate para Amazon ECS y Opciones de almacenamiento para las tareas de Amazon ECS.

- Se ha agregado compatibilidad con el uso de volúmenes del sistema de archivos de Amazon EFS para el almacenamiento de tareas persistentes. Para obtener más información, consulte Uso de volúmenes de Amazon EFS con Amazon ECS.
- El almacenamiento de tareas efímeras se ha incrementado en un mínimo de 20 GB para cada tarea. Para obtener más información, consulte Almacenamiento efímero de tareas de Fargate para Amazon ECS.
- Se ha actualizado el comportamiento del tráfico de red de entrada y salida de las tareas. A partir de la versión 1.4.0 de la plataforma, todas las tareas de Fargate reciben una única interfaz de red elástica (que se conoce como ENI de la tarea) y todo el tráfico de red fluye a través de esa ENI dentro de la VPC y podrá verse a través de los registros de flujo de la VPC. Para obtener más información acerca de las redes para el tipo de lanzamiento de Amazon EC2, consulte Opciones de redes de tareas de Amazon ECS para el tipo de lanzamiento de EC2. Para obtener más información acerca de las redes para el tipo de lanzamiento de Fargate, consulte Opciones de redes de tareas de Amazon ECS para el tipo de lanzamiento de Fargate.
- Las ENI de tarea permiten utilizar tramas gigantes. Las interfaces de red se configuran con una unidad de transmisión máxima (MTU), que es el tamaño de la carga útil más grande que cabe en una sola trama. Cuanto mayor sea la MTU, mayor será la carga de la aplicación que cabe en una sola trama, lo que reduce la sobrecarga por trama y aumenta la eficiencia. Esta compatibilidad con las tramas gigantes reducirá la sobrecarga cuando la ruta de red entre la tarea y el destino admita el uso de estas tramas, como el tráfico que permanece dentro de la VPC.
- CloudWatch Container Insights incluirá métricas de rendimiento de red para las tareas de Fargate. Para obtener más información, consulte Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada.
- Se ha agregado compatibilidad con la versión 4 del punto de enlace de metadatos de tareas, que proporciona información adicional sobre las tareas de Fargate, incluidas estadísticas de red de la tarea y la zona de disponibilidad en la que esta se ejecuta. Para obtener más información, consulte Versión 4 del punto de conexión de metadatos de tareas de Amazon ECS y Versión 4 del punto de conexión de los metadatos de tareas de Amazon ECS para tareas en Fargate.
- Se ha agregado compatibilidad con el parámetro `SYS_PTRACE` de Linux en las definiciones de contenedores. Para obtener más información, consulte Parámetros de Linux.
- El agente de contenedor de Fargate sustituye al de Amazon ECS para todas las tareas de Fargate. Por lo general, este cambio no tiene ningún efecto en el modo en el que se ejecutan las tareas.
- El entorno de ejecución del contenedor ahora utiliza Containerd en lugar de Docker. Lo más habitual es que este cambio no tenga ningún efecto en el modo en el que se ejecutan las tareas. Como podrá ver, algunos mensajes de error que se originan con el entorno de ejecución del

contenedor cambian y ya no mencionan a Docker sino que devuelven errores más generales. Para obtener más información, consulte Mensajes de error de tareas detenidas de Amazon ECS.

- Se basa en Amazon Linux 2.

1.3.0

A continuación, se incluye el registro de cambios de la versión `1.3.0` de la plataforma.

- A partir del 30 de septiembre de 2019, las nuevas tareas de Fargate que se lancen admiten el controlador de registros `awsfirelens`. Configure FireLens para Amazon ECS a fin de usar parámetros de definición de tareas para dirigir registros a un servicio de AWS o al destino de red de socios de AWS (APN) para el almacenamiento y el análisis de registros. Para obtener más información, consulte Envío de registros de Amazon ECS a un servicio de AWS o AWS Partner.
- Se ha agregado el reciclaje de tareas para las tareas de Fargate, un proceso que consiste en actualizar las tareas que forman parte de un servicio de Amazon ECS. Para obtener más información, Retirada y mantenimiento de tareas para AWS Fargate en Amazon ECS.
- A partir del 27 de marzo de 2019, las nuevas tareas de Fargate que se lancen pueden utilizar parámetros de definición de tareas adicionales que le permiten definir una configuración del proxy, dependencias para inicio y apagado de contenedores y un valor de tiempo de espera de inicio y detención por contenedor. Para obtener más información, consulte Configuración del proxy, Dependencia de contenedor y Tiempos de espera de contenedor.
- A partir del 2 de abril de 2019, cualquier nueva tarea de Fargate que se lance admite introducir información confidencial en sus contenedores mediante su almacenamiento en secretos de AWS Secrets Manager o en parámetros del Parameter Store de AWS Systems Manager y la posterior referencia a ellos en la definición de contenedor. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.
- A partir del 1 de mayo de 2019, cualquier nueva tarea de Fargate que se lanza admite hacer referencia a información confidencial en la configuración de registros de un contenedor mediante el parámetro de definición de contenedor `secretOptions`. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.
- A partir del 1 de mayo de 2019, cualquier nueva tarea de Fargate que se lance es compatible con el controlador de registros `splunk` además del controlador de registros `awslogs`. Para obtener más información, consulte Almacenamiento y registro.
- A partir del 9 de julio de 2019, toda nueva tarea de Fargate que se lance admite CloudWatch Container Insights. Para obtener más información, consulte Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada.

- Desde el 3 de diciembre de 2019, se admite el proveedor de capacidad Fargate Spot. Para obtener más información, consulte Clústeres de Amazon ECS para Fargate.
- Se basa en Amazon Linux 2.

Obsolecencia de la versión de la plataforma AWS Fargate Linux

En esta página, se enumeran las versiones de la plataforma que AWS Fargate ha vuelto obsoletas o tiene programado dar de baja. Estas versiones de la plataforma permanecen disponibles hasta la fecha de baja publicada.

Se proporciona una fecha de actualización forzada para cada versión de la plataforma programada para darla de baja. En la fecha de actualización forzada, todo servicio que utilice la versión de la plataforma `LATEST` que apunte a una versión de la plataforma programada para darla de baja se actualizará mediante la opción Force new deployment (Forzar nueva implementación). Cuando el servicio se actualiza mediante esta opción, se detienen todas las tareas que se ejecutan en una versión de la plataforma programada para darla de baja y las nuevas tareas se lanzan a través de la versión de la plataforma a la que apunta la etiqueta `LATEST` en ese momento. Las tareas o servicios independientes que tienen establecido un conjunto explícito de versiones de plataforma no se ven afectados por la fecha de actualización forzada.

Le recomendamos que actualice las tareas independientes de los servicios para que utilicen la versión más reciente de la plataforma. Para obtener más información acerca de cómo migrar a la versión más reciente de la plataforma, consulte Migración a la versión 1.4.0 de la plataforma Linux en Amazon ECS.

Una vez que una versión de la plataforma llega a la fecha de baja, dejará de estar disponible para nuevas tareas o servicios. Toda tarea o servicio independiente que utilice explícitamente una versión obsoleta de la plataforma continuará utilizando esa versión hasta que se detengan las tareas. Después de la fecha de baja, una versión obsoleta de la plataforma ya no recibirá actualizaciones de seguridad ni correcciones de errores.

Versión de la plataforma	Fecha de actualización forzada	Fecha de baja
1.0.0	26 de octubre de 2020	14 de diciembre de 2020
1.1.0	26 de octubre de 2020	14 de diciembre de 2020

Versión de la plataforma	Fecha de actualización forzada	Fecha de baja
1.2.0	26 de octubre de 2020	14 de diciembre de 2020

Para obtener información acerca de las versiones de la plataforma actuales, consulte Versiones de la plataforma Fargate para Amazon ECS.

Registro de cambios de las versiones obsoletas de Fargate Linux

1.2.0

A continuación, se incluye el registro de cambios de la versión `1.2.0` de la plataforma.

> **Note**
>
> La versión de la plataforma `1.2.0` ya no está disponible. Para obtener información acerca de las versiones obsoletas de la plataforma, consulte Obsolecencia de la versión de la plataforma AWS Fargate Linux.

- Se ha agregado compatibilidad con la autenticación de registros privados utilizando AWS Secrets Manager Para obtener más información, consulte Uso de imágenes de contenedor que no sean de AWS en Amazon ECS.

1.1.0

A continuación, se incluye el registro de cambios de la versión `1.1.0` de la plataforma.

> **Note**
>
> La versión de la plataforma `1.1.0` ya no está disponible. Para obtener información acerca de las versiones obsoletas de la plataforma, consulte Obsolecencia de la versión de la plataforma AWS Fargate Linux.

- Se ha agregado compatibilidad con el punto de enlace de metadatos de tareas de Amazon ECS. Para obtener más información, consulte Metadatos de tareas de Amazon ECS disponibles para tareas en Fargate.
- Se ha agregado compatibilidad para las comprobaciones de estado de Docker en las definiciones de contenedor. Para obtener más información, consulte Comprobación de estado.
- Se ha agregado compatibilidad con la detección de servicios de Amazon ECS. Para obtener más información, consulte Uso de la detección de servicios para conectar los servicios de Amazon ECS con nombres de DNS.

1.0.0

A continuación, se incluye el registro de cambios de la versión `1.0.0` de la plataforma.

> **Note**
>
> La versión de la plataforma `1.0.0` ya no está disponible. Para obtener información acerca de las versiones obsoletas de la plataforma, consulte Obsolecencia de la versión de la plataforma AWS Fargate Linux.

- Se basa en Amazon Linux 2017.09.
- Versión inicial.

Registro de cambios de la versión de la plataforma de Fargate Windows

Estas son las versiones de plataforma disponibles de para los contenedores de Windows.

1.0.0

A continuación, se incluye el registro de cambios de la versión `1.0.0` de la plataforma.

- Versión inicial para soporte en los siguientes sistemas operativos Microsoft Windows Server:
  - Windows Server 2019 Full
  - Windows Server 2019 Core
  - Windows Server 2022 Full
  - Windows Server 2022 Core

Consideraciones sobre los contenedores de Windows en Fargate para Amazon ECS

A continuación se muestran las diferencias y consideraciones que debe tener en cuenta al ejecutar contenedores de Windows en AWS Fargate.

Si necesita ejecutar tareas en contenedores de Linux y Windows, debe crear definiciones de tareas independientes para cada sistema operativo.

AWS gestiona la administración de licencias del sistema operativo, por lo que no necesita ninguna licencia de Microsoft Windows Server adicional.

Los contenedores de Windows en Fargate de AWS son compatibles con los siguientes sistemas operativos:

- Windows Server 2019 Full
- Windows Server 2019 Core
- Windows Server 2022 Full
- Windows Server 2022 Core

Los contenedores de Windows en Fargate de AWS son compatibles con el controlador awslogs. Para obtener más información, consulte the section called “Envío de registros a CloudWatch ”.

Las siguientes características no son compatibles en los contenedores de Windows en Fargate:

- Amazon FSx
- Enlace troncal de ENI
- gMSA para contenedores de Windows
- Integración de proxy y servicio App Mesh para tareas
- Integración de enrutador de registro Firelens para tareas
- Volúmenes de EFS
- Volúmenes de EBS
- Los siguientes parámetros de definición de tareas:
  - `maxSwap`

  - `swappiness`
  - `environmentFiles`
- El proveedor de capacidad de Fargate Spot
- Volúmenes de imágenes

  La opción Dockerfile `volume` se omite. En cambio, use un montaje de enlace en la definición de tarea. Para obtener más información, consulte Uso de montajes de unión con Amazon ECS.
- Los parámetros de CPU y memoria de nivel de tarea se omiten para los contenedores de Windows. Le recomendamos que especifique recursos de nivel de contenedor para los contenedores de Windows.
- memoria para tarea
- mermoryReservation en contenedores
- Políticas de reinicio en los contenedores

Almacenamiento efímero de tareas de Fargate para Amazon ECS

Cuando se aprovisiona, cada tarea de Amazon ECS alojada en contenedores Linux en AWS Fargate recibe el siguiente almacenamiento efímero para montajes de enlace. Esto se puede montar y compartir entre contenedores que utilizan los parámetros `volumes`, `mountPoints` y `volumesFrom` de la definición de tareas. Este parámetro no es compatible con contenedores de Windows en AWS Fargate.

Versiones de plataforma de contenedores Fargate Linux

Versión 1.4.0 o posteriores

De forma predeterminada, las tareas de Amazon ECS alojadas en Fargate que utilizan la versión `1.4.0` de la plataforma o una posterior reciben un mínimo de 20 GiB de almacenamiento efímero. La cantidad total de almacenamiento efímero se puede aumentar hasta un máximo de 200 GiB. Para hacerlo, especifique el parámetro `ephemeralStorage` en la definición de tareas.

La imagen del contenedor comprimida y sin comprimir extraída para la tarea se almacena en el almacenamiento efímero. Para determinar la cantidad total de almacenamiento efímero que debe utilizar la tarea, debe restar la cantidad de almacenamiento que utiliza la imagen de contenedor de la cantidad total de almacenamiento efímero que se asigna a su tarea.

Para las tareas que utilizan la versión `1.4.0` de la plataforma o una posterior que se lanzaron a partir del 28 de mayo de 2020, el almacenamiento efímero se cifra con un algoritmo de cifrado AES-256. Este algoritmo utiliza una clave de cifrado de AWS, o usted puede crear su propia clave administrada por el cliente. Para obtener más información, consulte Claves administradas por el cliente para el almacenamiento efímero de AWS Fargate.

Para las tareas que utilizan la versión de la plataforma `1.4.0` o una posterior y que se lanzaron a partir del 18 de noviembre de 2022, el uso efímero del almacenamiento se informa a través del punto de conexión de metadatos de tareas. Las aplicaciones de sus tareas pueden consultar el punto de conexión de metadatos de las tareas, versión 4, para obtener su tamaño reservado de almacenamiento efímero y la cantidad utilizada.

Además, el tamaño reservado para el almacenamiento efímero y la cantidad utilizada se envían a Información de contenedores de Amazon CloudWatch si activa Container Insights.

> **Note**
>
> Fargate reserva espacio en el disco. Solo lo usa Fargate. No se cobra por esto. No se muestra en estas métricas. Sin embargo, puede ver este almacenamiento adicional en otras herramientas, como `df`.

Versión 1.3.0 o anteriores

Para las tareas de Amazon ECS alojadas en Fargate que utilizan la versión `1.3.0` de la plataforma o una anterior, cada tarea recibe el siguiente almacenamiento efímero.

- 10 GB de almacenamiento de capa de Docker

  > **Note**
  >
  > Esta cantidad incluye artefactos de imagen de contenedor comprimida y no comprimida.

- 4 GB adicionales para montaje de volúmenes. Esto se puede montar y compartir entre contenedores que utilizan los parámetros `volumes`, `mountPoints` y `volumesFrom` de la definición de tareas.

Versiones de plataforma de contenedores Windows Fargate

Versión 1.0.0 o posteriores

De forma predeterminada, las tareas de Amazon ECS alojadas en Fargate que utilizan la versión `1.0.0` de la plataforma o una posterior reciben un mínimo de 20 GiB de almacenamiento efímero. La cantidad total de almacenamiento efímero se puede aumentar hasta un máximo de 200 GiB. Para hacerlo, especifique el parámetro `ephemeralStorage` en la definición de tareas.

La imagen del contenedor comprimida y sin comprimir extraída para la tarea se almacena en el almacenamiento efímero. Para determinar la cantidad total de almacenamiento efímero que debe utilizar la tarea, debe restar la cantidad de almacenamiento que utiliza la imagen de contenedor de la cantidad total de almacenamiento efímero que se asigna a su tarea.

Para obtener más información, consulte Uso de montajes de unión con Amazon ECS.

Claves administradas por el cliente para el almacenamiento efímero de AWS Fargate para Amazon ECS

AWS Fargate admite claves administradas por el cliente para cifrar los datos de las tareas de Amazon ECS almacenadas en un almacenamiento efímero para ayudar a los clientes que gestionan datos confidenciales a cumplir sus políticas de seguridad internas. Los clientes siguen disfrutando de las ventajas de Fargate sin servidor y, al mismo tiempo, ofrecen a los auditores de conformidad una mayor visibilidad del cifrado de almacenamiento autoadministrado. Si bien Fargate tiene cifrado de almacenamiento efímero administrado por Fargate de forma predeterminada, los clientes también pueden usar sus propias claves autoadministradas al cifrar datos confidenciales, como información financiera o relacionada con la salud.

Puede importar sus propias claves a AWS KMS o crearlas en AWS KMS. Estas claves autoadministradas se almacenan en AWS KMS y llevan a cabo acciones estándar del ciclo de vida de AWS KMS, como rotar, deshabilitar y eliminar. Puede auditar el acceso a las claves y su uso en los registros de CloudTrail.

De forma predeterminada, la clave KMS admite 50 000 concesiones por clave. Fargate utiliza una única concesión de AWS KMS por cada tarea de clave administrada por el cliente, por lo que admite hasta 50 000 tareas simultáneas para una clave. Si desea aumentar este número, puede solicitar un aumento del límite, que se aprueba caso por caso.

Fargate no cobra nada adicional por usar claves administradas por el cliente. Solo se le cobra el precio estándar por usar las claves de AWS KMS para el almacenamiento y las solicitudes de API.

Temas

- Creación de una clave de cifrado para el almacenamiento efímero de Fargate para Amazon ECS
- Administración de claves de AWS KMS para el almacenamiento efímero de Fargate para Amazon ECS

Creación de una clave de cifrado para el almacenamiento efímero de Fargate para Amazon ECS

Cree claves administradas por el cliente para cifrar los datos almacenados en el almacenamiento efímero de Fargate.

> **Note**
>
> El cifrado de almacenamiento efímero de Fargate con claves administradas por el cliente no está disponible para los clústeres de tareas de Windows.
> El cifrado de almacenamiento efímero de Fargate con claves administradas por el cliente no estaba disponible en las `platformVersions` anteriores a la `1.4.0`.
> Fargate reserva espacio en un almacenamiento efímero que solo utiliza Fargate y no se cobra por el espacio. La asignación puede diferir de las tareas clave no administradas por el cliente, pero el espacio total sigue siendo el mismo. Puede ver este cambio en herramientas como `df`.
> Las claves multiregión no son compatibles con el almacenamiento efímero de Fargate
> Los alias de la clave KMS no son compatibles con el almacenamiento efímero de Fargate

Para crear una clave administrada por el cliente (CMK) a fin de cifrar el almacenamiento efímero para Fargate en AWS KMS, siga estos pasos.

1. Vaya a https://console.aws.amazon.com/kms.
2. Siga las instrucciones de Creating Keys en la Guía para desarrolladores de AWS Key Management Service.
3. Al crear su clave de AWS KMS, asegúrese de proporcionar los permisos de operaciones de AWS KMS pertinentes al servicio Fargate en las políticas de claves. Las siguientes operaciones

de API deben estar permitidas en la política para utilizar su clave administrada por el cliente con los recursos de su clúster de Amazon ECS.

- `kms:GenerateDataKeyWithoutPlainText`: llama a `GenerateDataKeyWithoutPlainText` para generar una clave de datos cifrada a partir de la clave de AWS KMS proporcionada.
- `kms:CreateGrant`: agrega una concesión a una clave administrada por el cliente. Le concede acceso de control a una clave de AWS KMS especificada, lo que le permite acceder a las operaciones de concesión que necesita Fargate para Amazon ECS. Para obtener más información sobre el uso de concesiones, consulte la Guía para desarrolladores de AWS Key Management Service. Esto permite a Fargate para Amazon ECS realizar las siguientes tareas:
  - Llamar a `Decrypt` para AWS KMS a fin de obtener la clave de cifrado para descifrar los datos de almacenamiento efímero.
  - Configurar una entidad principal que se retire para permitir que el servicio `RetireGrant`.
- `kms:DescribeKey`: proporciona detalles de la clave administrada por el cliente para permitir que Amazon ECS valide la clave si es simétrica y está habilitada.

En el siguiente ejemplo, se muestra una política de claves de AWS KMS que se aplicaría a la clave de destino para el cifrado. Para utilizar las instrucciones de la política de ejemplo, sustituya los *`marcadores de posición del usuario`* con su propia información. Como siempre, configure solo los permisos que necesite; sin embargo, deberá proporcionar a AWS KMS los permisos para un usuario, como mínimo, a efectos de evitar errores.

```
{
      "Sid": "Allow generate data key access for Fargate tasks.",
      "Effect": "Allow",
      "Principal": { "Service":"fargate.amazonaws.com" },
      "Action": [
        "kms:GenerateDataKeyWithoutPlaintext"
      ],
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:ecs:clusterAccount": [
            "customerAccountId"
          ],
          "kms:EncryptionContext:aws:ecs:clusterName": [
             "clusterName"
          ]
```

```
        }
      },
      "Resource": "*"
    },
    {
      "Sid": "Allow grant creation permission for Fargate tasks.",
      "Effect": "Allow",
      "Principal": { "Service":"fargate.amazonaws.com" },
      "Action": [
        "kms:CreateGrant"
      ],
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:ecs:clusterAccount": [
            "customerAccountId"
          ],
          "kms:EncryptionContext:aws:ecs:clusterName": [
             "clusterName"
          ]
        },
       "ForAllValues:StringEquals": {
          "kms:GrantOperations": [
             "Decrypt"
          ]
       }
      },
      "Resource": "*"
    },
    {
      "Sid": "Allow describe key permission for cluster operator - CreateCluster
 and UpdateCluster.",
      "Effect": "Allow",
      "Principal": { "AWS":"arn:aws:iam::customerAccountId:role/customer-chosen-
role" },
      "Action": [
        "kms:DescribeKey"
      ],
      "Resource": "*"
    }
```

Las tareas de Fargate utilizan las claves de contexto de cifrado `aws:ecs:clusterAccount` y `aws:ecs:clusterName` para las operaciones criptográficas con la clave. Los clientes deben

agregar estos permisos para restringir el acceso a una cuenta o un clúster específicos. Utilice el nombre del clúster y no el ARN cuando especifique el clúster.

Para obtener más información, consulte Contexto de cifrado en la Guía para desarrolladores de AWS KMS.

Al crear o actualizar un clúster, tiene la opción de utilizar la clave de condición `fargateEphemeralStorageKmsKeyId`. Esta clave de condición permite a los clientes tener un control más detallado de las políticas de IAM. Las actualizaciones de la configuración de `fargateEphemeralStorageKmsKeyId` solo se aplican a las nuevas implementaciones de servicios.

A continuación, se muestra un ejemplo de cómo permitir a los clientes conceder permisos únicamente a un conjunto específico de claves de AWS KMS aprobadas.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:CreateCluster",
        "ecs:UpdateCluster"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "ecs:fargate-ephemeral-storage-kms-key": "arn:aws:kms:us-
west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        }
      }
    }
  ]
}
```

A continuación, se muestra un ejemplo de cómo denegar los intentos de eliminación de las claves de AWS KMS que ya están asociadas a un clúster.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": [
        "ecs:CreateCluster",
        "ecs:UpdateCluster"
    ],
    "Resource": "*",
    "Condition": {
      "Null": {
        "ecs:fargate-ephemeral-storage-kms-key": "true"
      }
    }
  }
}
```

Los clientes pueden comprobar si sus tareas no administradas o de servicio están cifradas con la clave mediante los comandos AWS CLI, `describe-tasks`, `describe-cluster` o `describe-services`.

Para obtener más información, consulte Condition keys for AWS KMS en la Guía para desarrolladores de AWS KMS.

AWS Management Console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. Seleccione Clústeres en el menú de navegación de la izquierda y Crear clúster en la parte superior derecha, o bien elija un clúster existente. Para un clúster existente, seleccione Actualizar clúster en la parte superior derecha.
3. En la sección Cifrado del flujo de trabajo, tendrá la opción de seleccionar su clave de AWS KMS en Almacenamiento administrado y Almacenamiento efímero de Fargate. También puede elegir Crear una clave de AWS KMS desde aquí.
4. Elija Crear una vez que haya terminado de crear el nuevo clúster o Actualizar si estaba actualizando uno existente.

AWS CLI

A continuación, se muestra un ejemplo de cómo crear un clúster y configurar el almacenamiento efímero de Fargate mediante la AWS CLI (sustituya los valores en *color rojo* por los suyos):

```
aws ecs create-cluster --cluster clusterName \
--configuration '{"managedStorageConfiguration":
{"fargateEphemeralStorageKmsKeyId":"arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"}}'
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:012345678901:cluster/clusterName",
        "clusterName": "clusterName",
        "configuration": {
            "managedStorageConfiguration": {
                "fargateEphemeralStorageKmsKeyId": "arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
            }
        },
        "status": "ACTIVE",
        "registeredContainerInstancesCount": 0,
        "runningTasksCount": 0,
        "pendingTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [],
        "settings": [],
        "capacityProviders": [],
        "defaultCapacityProviderStrategy": []
    },
    "clusterCount": 5
}
```

AWS CloudFormation

A continuación, se muestra un ejemplo de plantilla para crear un clúster y configurar el almacenamiento efímero de Fargate mediante AWS CloudFormation (sustituya los valores en *color rojo* por los suyos):

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  MyCluster:
    Type: AWS::ECS::Cluster
```

```
    Properties:
      ClusterName: "clusterName"
      Configuration:
        ManagedStorageConfiguration:
          FargateEphemeralStorageKmsKeyId: "arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Administración de claves de AWS KMS para el almacenamiento efímero de Fargate para Amazon ECS

Tras crear o importar la clave de AWS KMS para cifrar el almacenamiento efímero de Fargate, la administrará de la misma manera en que lo haría con cualquier otra clave de AWS KMS.

Rotación automática de claves de AWS KMS

Puede habilitar la rotación automática de las claves o rotarlas manualmente. La rotación automática de claves hace rotar la clave todos los años mediante la generación de nuevo material criptográfico para la clave. AWS KMS también guarda todas las versiones anteriores del material criptográfico, por lo que podrá descifrar cualquier dato que utilicen las versiones anteriores de la clave. AWS KMS no eliminará el material rotado hasta que usted elimine la clave.

La rotación automática de claves se puede habilitar y deshabilitar en cualquier momento.

Deshabilitación o revocación de claves de AWS KMS

Si deshabilita una clave administrada por el cliente en AWS KMS, esto no repercute en la ejecución de las tareas, y las claves seguirán funcionando durante todo su ciclo de vida. Si una nueva tarea usa la clave deshabilitada o revocada, la tarea falla porque no puede acceder a la clave. Debe configurar una alarma de CloudWatch o similar para asegurarse de que nunca se necesite una clave deshabilitada para descifrar los datos ya cifrados.

Eliminación de claves de AWS KMS

Eliminar las claves siempre debe ser el último recurso y solo debe hacerse si está seguro de que la clave eliminada no volverá a necesitarse. Las tareas nuevas que intenten usar la clave eliminada fallarán porque no podrán acceder a ella. AWS KMS recomienda deshabilitar una clave en lugar de eliminarla. Si cree que es necesario eliminar una clave, le sugerimos que la deshabilite primero y configure una alarma de CloudWatch para asegurarse de que no es necesaria. Si elimina una clave, AWS KMS espera al menos siete días para que cambie de opinión.

Auditoría del acceso a las claves de AWS KMS

Puede usar los registros de CloudTrail para auditar el acceso a su clave de AWS KMS. Puede comprobar las operaciones de AWS KMS `CreateGrant`, `GenerateDataKeyWithoutPlaintext` y `Decrypt`. Estas operaciones también muestran la `aws:ecs:clusterAccount` y el `aws:ecs:clusterName` como parte del `EncryptionContext` registrado en CloudTrail.

A continuación, se muestran ejemplos de eventos de CloudTrail para `GenerateDataKeyWithoutPlaintext`, `GenerateDataKeyWithoutPlaintext (DryRun)`, `CreateGrant`, `CreateGrant (DryRun)` y `RetireGrant` (sustituya los valores en *`color rojo`* por los suyos).

GenerateDataKeyWithoutPlaintext

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ec2-frontend-api.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:13Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ec2-frontend-api.amazonaws.com",
    "userAgent": "ec2-frontend-api.amazonaws.com",
    "requestParameters": {
        "numberOfBytes": 64,
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "encryptionContext": {
            "aws:ecs:clusterAccount": "account-id",
            "aws:ebs:id": "vol-xxxxxxx",
            "aws:ecs:clusterName": "cluster-name"
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
```

```
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

GenerateDataKeyWithoutPlaintext (DryRun)

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "fargate.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "fargate.amazonaws.com",
    "userAgent": "fargate.amazonaws.com",
    "errorCode": "DryRunOperationException",
    "errorMessage": "The request would have succeeded, but the DryRun option is
 set.",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "dryRun": true,
        "numberOfBytes": 64,
        "encryptionContext": {
            "aws:ecs:clusterAccount": "account-id",
            "aws:ecs:clusterName": "cluster-name"
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": true,
    "resources": [
```

```
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

CreateGrant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ec2-frontend-api.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:13Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ec2-frontend-api.amazonaws.com",
    "userAgent": "ec2-frontend-api.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "granteePrincipal": "fargate.us-west-2.amazonaws.com",
        "operations": [
            "Decrypt"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:ecs:clusterAccount": "account-id",
                "aws:ebs:id": "vol-xxxx",
                "aws:ecs:clusterName": "cluster-name"
            }
        },
        "retiringPrincipal": "ec2.us-west-2.amazonaws.com"
```

```
    },
    "responseElements": {
        "grantId":
 "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855",
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
    },
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

CreateGrant (DryRun)

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "fargate.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "fargate.amazonaws.com",
    "userAgent": "fargate.amazonaws.com",
    "errorCode": "DryRunOperationException",
    "errorMessage": "The request would have succeeded, but the DryRun option is
 set.",
    "requestParameters": {
```

```
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "granteePrincipal": "fargate.us-west-2.amazonaws.com",
        "dryRun": true,
        "operations": [
            "Decrypt"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:ecs:clusterAccount": "account-id",
                "aws:ecs:clusterName": "cluster-name"
            }
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

RetireGrant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2024-04-20T18:37:38Z",
    "eventSource": "kms.amazonaws.com",
```

```
    "eventName": "RetireGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
    },
    "additionalEventData": {
        "grantId":
 "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855"
    },
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

Retirada y mantenimiento de tareas para AWS Fargate en Amazon ECS

AWS es responsable del mantenimiento de la infraestructura subyacente de AWS Fargate. AWS determina cuándo se debe reemplazar una revisión de versión de la plataforma por una nueva revisión de la infraestructura. Esto se conoce como retiro de tareas. AWS envía una notificación de retiro de tareas cuando se retira una revisión de la versión de la plataforma. Actualizamos periódicamente las versiones de nuestras plataformas admitidas para introducir una nueva revisión que contenga actualizaciones del software de tiempo de ejecución de Fargate y de las dependencias

subyacentes, como el sistema operativo y el tiempo de ejecución del contenedor. Una vez que una revisión más reciente está disponible, retiramos la anterior para garantizar que todas las cargas de trabajo de los clientes se ejecuten con la versión más actualizada de la plataforma Fargate. Cuando se retira una revisión, se detienen todas las tareas que se estén ejecutando en esa revisión.

Las tareas de Amazon ECS se pueden clasificar como tareas de servicio o tareas independientes. Las tareas de servicio se implementan como parte de un servicio y son controladas por la programación de Amazon ECS. Para obtener más información, consulte Servicios de Amazon ECS. Las tareas independientes son tareas que se inician mediante la API `RunTask` de Amazon ECS, ya sea directamente o mediante un programador externo, como las tareas programadas (que inicia Amazon EventBridge), AWS Batch o AWS Step Functions. No necesita realizar ninguna acción en respuesta a la retirada de tareas de sus tareas de servicio, ya que el programador de Amazon ECS reemplaza automáticamente las tareas.

En el caso de las tareas independientes, es posible que deba realizar una gestión adicional en respuesta a la retirada de la tarea. Para obtener más información, consulte ¿Amazon ECS puede administrar automáticamente las tareas independientes?.

En el caso de las tareas de servicio, no es necesario que realice ninguna acción de retirada de tareas, a menos que desee sustituirlas antes de que AWS lo haga. Cuando el programador de Amazon ECS detiene las tareas, utiliza el `maximumPercent` e inicia una nueva tarea en un intento por mantener el recuento deseado para el servicio. Siga las prácticas recomendadas para minimizar el impacto de la retirada de tareas. El valor predeterminado `maximumPercent` para un servicio que utiliza el programador de servicio REPLICA es 200 %. Por lo tanto, cuando AWS Fargate comienza a retirar tareas, Amazon ECS primero programa una nueva tarea y, a continuación, espera a que se ejecute antes de retirar una tarea anterior. Al establecer el valor `maximumPercent` en 100 %, Amazon ECS detiene primero la tarea y, a continuación, la reemplaza.

Para la retirada de una tarea independiente, AWS detiene la tarea en la fecha de retiro de la tarea o después de esa fecha. Amazon ECS no lanza una tarea de reemplazo cuando se detiene una tarea. Si necesita que estas tareas sigan ejecutándose, debe detenerlas e iniciar una tarea de sustitución antes de la hora indicada en la notificación. Por lo tanto, recomendamos que los clientes supervisen el estado de las tareas independientes y, si es necesario, implementen la lógica para reemplazar las tareas detenidas.

Cuando una tarea se detiene en cualquiera de los escenarios anteriores, puede ejecutar `describe-tasks`. El código `stoppedReason` que aparece en la respuesta es `ECS is performing maintenance on the underlying infrastructure hosting the task`.

El mantenimiento de las tareas se aplica cuando se debe sustituir una nueva revisión de versión de la plataforma por una nueva revisión. Si hay un problema con un host de Fargate subyacente, Amazon ECS sustituye el host sin un aviso de retiro de la tarea.

Descripción general del aviso de retirada de tareas

Cuando AWS marca la necesidad de retirar una revisión de la versión de la plataforma, identificamos todas las tareas que se están ejecutando en esa revisión de la versión de la plataforma en todas las regiones. A continuación, enviamos una notificación por cuenta y región, en la que destacamos las tareas o los servicios afectados y la fecha en que comenzarán a producirse las retiradas.

La siguiente ilustración muestra el ciclo de vida de una revisión de la versión de la plataforma Fargate desde el lanzamiento de una nueva revisión hasta la retirada de la revisión de la plataforma.

La siguiente información proporciona detalles.

- Tras lanzar una nueva revisión de la versión de la plataforma, todas las tareas nuevas se programan en esta revisión.
- Las tareas existentes que se programaron y se están ejecutando permanecen en la revisión en la que se colocaron originalmente mientras dure la tarea, y no se migran a la nueva revisión.
- Las nuevas tareas, por ejemplo, como parte de una actualización de un servicio o de la retirada de tareas de Fargate, se incluyen en la última versión de la plataforma disponible en el momento del lanzamiento.

Las notificaciones de retirada de tareas se envían a través del panel de control de AWS Health y por correo electrónico a la dirección de correo electrónico registrada. Dichas notificaciones incluyen la siguiente información:

- La fecha de retirada de la tarea: la tarea se detiene en esta fecha o después de ella.

- En el caso de las tareas independientes, los ID de las tareas.
- En el caso de las tareas de servicio, el ID del clúster en el que se ejecuta el servicio y los ID del servicio.
- Los siguientes pasos que debe seguir.

Por lo general, se envía una notificación para cada una de las tareas de servicio y tareas independientes de cada Región de AWS. Sin embargo, en algunos casos, es posible que reciba más de un evento por cada tipo de tarea; por ejemplo, cuando hay tantas tareas que deben ser retiradas que sobrepasan los límites de nuestros mecanismos de notificación.

Puede identificar tareas programadas para retirarse de las siguientes formas:

- la AWS Health Dashboard,

  Las notificaciones de AWS Health se pueden enviar por Amazon EventBridge para almacenarse en archivos, como Amazon Simple Storage Service, llevar a cabo acciones automatizadas, como ejecutar una función AWS Lambda, o hacia otros sistemas de notificaciones, como Amazon Simple Notification Service. Para más información, consulte Supervisión de eventos de AWS Health con Amazon EventBridge). Para configurar muestras para enviar notificaciones a Amazon Chime, Slack o Microsoft Teams, consulte el repositorio de AWS Health Aware en GitHub.

  A continuación, se muestra un evento de EventBridge como ejemplo.

```
{
    "version": "0",
    "id": "3c268027-f43c-0171-7425-1d799EXAMPLE",
    "detail-type": "AWS Health Event",
    "source": "aws.health",
    "account": "123456789012",
    "time": "2023-08-16T23:18:51Z",
    "region": "us-east-1",
    "resources": [
        "cluster|service",
        "cluster|service"
    ],
    "detail": {
        "eventArn": "arn:aws:health:us-east-1::event/ECS/
AWS_ECS_TASK_PATCHING_RETIREMENT/AWS_ECS_TASK_PATCHING_RETIREMENT_test1",
        "service": "ECS",
        "eventScopeCode": "ACCOUNT_SPECIFIC",
```

```
        "communicationId":
 "7988399e2e6fb0b905ddc88e0e2de1fd17e4c9fa60349577446d95a18EXAMPLE",
        "lastUpdatedTime": "Wed, 16 Aug 2023 23:18:52 GMT",
        "eventRegion": "us-east-1",
        "eventTypeCode": "AWS_ECS_TASK_PATCHING_RETIREMENT",
        "eventTypeCategory": "scheduledChange",
        "startTime": "Wed, 16 Aug 2023 23:18:51 GMT",
        "endTime": "Fri, 18 Aug 2023 23:18:51 GMT",
        "eventDescription": [
            {
                "language": "en_US",
                "latestDescription": "\\nA software update has been deployed to
 Fargate which includes CVE patches or other critical patches. No action is required
 on your part. All new tasks launched automatically uses the latest software
 version. For existing tasks, your tasks need to be restarted in order for these
 updates to apply. Your tasks running as part of the following ECS Services will
 be automatically updated beginning Wed, 16 Aug 2023 23:18:51 GMT.\\n\\nAfter Wed,
 16 Aug 2023 23:18:51 GMT, the ECS scheduler will gradually replace these tasks,
 respecting the deployment settings for your service. Typically, services should
 see little to no interruption during the update and no action is required. When AWS
 stops tasks, AWS uses the minimum healthy percent (1) and launches a new task in
 an attempt to maintain the desired count for the service. By default, the minimum
 healthy percent of a service is 100 percent, so a new task is started first before
 a task is stopped. Service tasks are routinely replaced in the same way when
 you scale the service or deploy configuration changes or deploy task definition
 revisions. If you would like to control the timing of this restart you can update
 the service before Wed, 16 Aug 2023 23:18:51 GMT, by running the update-service
 command from the ECS command-line interface specifying force-new-deployment for
 services using Rolling update deployment type. For example:\\n\\n$ aws ecs update-
service -service service_name \\\n--cluster cluster_name -force-new-deployment\
\n\\nFor services using Blue/Green deployment type with AWS CodeDeploy:\\nPlease
 refer to create-deployment document (2) and create new deployment using same task
 definition revision.\\n\\nFor further details on ECS deployment types, please
 refer to ECS Deployment Developer Guide (1).\\nFor further details on Fargate's
 update process, please refer to the AWS Fargate User Guide (3).\\nIf you have
 any questions or concerns, please contact AWS Support (4).\\n\\n(1) https://
docs.aws.amazon.com/AmazonECS/latest/developerguide/deployment-types.html\\n(2)
 https://docs.aws.amazon.com/cli/latest/reference/deploy/create-deployment.html\\n(3)
 https://docs.aws.amazon.com/AmazonECS/latest/userguide/task-maintenance.html\\n(4)
 https://aws.amazon.com/support\\n\\nA list of your affected resources(s) can be
 found in the 'Affected resources' tab in the 'Cluster/ Service' format in the AWS
 Health Dashboard. \\n\\n"
            }
        ],
```

```
      "affectedEntities": [
                {
                    "entityValue": "arn:aws:ecs:eu-west-1:111222333444:task/
examplecluster/00805ce1d81940b5a37398e5a2c23333"
                },
                {
                    "entityValue": "arn:aws:ecs:eu-west-1:111222333444:task/
examplecluster/00805ce1d81940b5a37398e5a2c25555"
                }
            }
        ]
    }
}
```

- Correo electrónico

  Se envía un correo electrónico al correo electrónico registrado para obtener el ID de la Cuenta de AWS.

Para obtener información sobre cómo preparar una retirada de tareas, consulte Prepárese para la retirada de tareas de AWS Fargate en Amazon ECS .

¿Puedo no participar en el retiro de tareas?

No. Como parte del modelo de responsabilidad compartida deAWS, AWS es responsable de administrar y mantener la infraestructura subyacente para AWS Fargate. Esto incluye realizar actualizaciones periódicas de la plataforma para garantizar la seguridad y la estabilidad. Estas actualizaciones se aplican automáticamente por AWS, y los clientes no pueden optar por excluirlas. Esta es una ventaja clave del uso de AWS Fargate en comparación con la de ejecutar sus cargas de trabajo en instancias EC2, ya que AWS asume la responsabilidad de mantener la plataforma subyacente. Este modelo le permite centrarse en sus aplicaciones en lugar de en el mantenimiento de la infraestructura. Al aplicar automáticamente estas actualizaciones de la plataforma, AWS puede mantener el entorno de Fargate actualizado y seguro, sin que usted, como cliente, deba realizar ninguna acción. Esto ayuda a proporcionar un entorno contenerizado confiable y seguro para ejecutar sus cargas de trabajo en Fargate.

¿Puedo recibir notificaciones de retiro de tareas a través de otros servicios de AWS?

AWS envía una notificación de retiro de tareas a AWS Health Dashboard y al contacto de correo electrónico principal de la Cuenta de AWS. AWS Health Dashboard ofrece una serie de integraciones en otros servicios de AWS, incluido EventBridge. Puede utilizar EventBridge para automatizar la visibilidad de los avisos (por ejemplo, reenviar el mensaje a una herramienta de ChatOps). Para más información, consulte Solution overview: Capturing task retirement notifications.

¿Puedo cambiar el retiro de una tarea una vez programada?

No. La programación se basa en el tiempo de espera para el retiro de la tarea, que tiene un valor predeterminado de 7 días. Si necesita más tiempo, puede optar por configurar el periodo de espera en 14 días. Para obtener más información, consulte Paso 2: Capture las notificaciones de retirada de tareas para alertar a los equipos y tomar medidas. El cambio en esta configuración se aplica a los retiros que se programarán en el futuro. Los retiros programados actualmente no se ven afectados. Si tiene algún problema, póngase en contacto con Soporte.

¿Cómo administra Amazon ECS las tareas que forman parte de un servicio?

En el caso de las tareas de servicio, no es necesario que realice ninguna acción de retirada de tareas, a menos que desee sustituirlas antes de que AWS lo haga. Cuando el programador de Amazon ECS detiene tareas, utiliza el porcentaje mínimo en buen estado e inicia una nueva tarea en un intento por mantener el recuento deseado para el servicio. Para minimizar el impacto de la retirada de tareas de Fargate, las cargas de trabajo deben implementarse siguiendo las prácticas recomendadas de Amazon ECS. Por ejemplo, al implementar una aplicación sin estado como un servicio de Amazon ECS, como un servidor web o API, los clientes deben implementar múltiples réplicas de tareas y establecer el MinimumHethyPercent en 100 %. Por defecto, el valor del porcentaje mínimo de estado de un servicio es el 100 %. Por lo tanto, cuando Fargate comienza a retirar tareas, Amazon ECS primero programa una nueva tarea y espera a que se ejecute antes de retirar una tarea anterior. Las tareas de servicio se sustituyen de forma rutinaria como parte de la retirada de tareas, del mismo modo que cuando se escala el servicio, se implementan cambios de configuración o se implementan revisiones de la definición de tareas. Para prepararse para el retiro de tareas, consulte Prepárese para la retirada de tareas de AWS Fargate en Amazon ECS .

¿Amazon ECS puede administrar automáticamente las tareas independientes?

No. AWS no puede crear una tarea de sustitución para las tareas independientes que inicie `RunTask`, tareas programadas (por ejemplo, a través del Programador de EventBridge), AWS Batch o AWS Step Functions. Amazon ECS solo administra las tareas que forman parte de un servicio.

Prepárese para la retirada de tareas de AWS Fargate en Amazon ECS

Para prepararse para la retirada de las tareas, lleve a cabo las siguientes operaciones:

1. Establezca el periodo de espera para la retirada de tareas.
2. Capture las notificaciones de retirada de tareas para notificarlas a los miembros del equipo.
3. No puede controlar el momento exacto en el que se retira una tarea, pero puede controlar la sustitución de las tareas actualizando el servicio con la opción de despliegue forzoso.

Paso 1: Configure el tiempo de espera de la tarea

Puede configurar la hora a la que Fargate inicia el retiro de las tareas. Para las cargas de trabajo que requieren la aplicación inmediata de las actualizaciones, elija la configuración inmediata (`0`). Cuando necesite más control, por ejemplo, cuando una tarea solo se pueda detener durante un período determinado, configure la opción de 7 días (`7`) o 14 días (`14`).

Le recomendamos que elija un período de espera más corto para poder seleccionar antes las revisiones de las versiones más recientes de la plataforma.

Para configurar el periodo de espera, ejecute `put-account-setting-default` o `put-account-setting` como usuario raíz o como usuario administrativo. Utilice la opción `fargateTaskRetirementWaitPeriod` para el conjunto de opciones `name` y `value` para uno de los valores siguientes:

- `0` - AWS envía la notificación e inmediatamente comienza a retirar las tareas afectadas.
- `7` - AWS envía la notificación y espera 7 días calendario antes de empezar a retirar las tareas afectadas.
- `14` - AWS envía la notificación y espera 14 días calendario antes de empezar a retirar las tareas afectadas.

El valor predeterminado es 7 días.

Para obtener más información, consulte put-account-setting-default y put-account-setting en la Referencia de la API de Amazon Elastic Container Service.

Paso 2: Capture las notificaciones de retirada de tareas para alertar a los equipos y tomar medidas

Cuando hay una próxima retirada de tareas, AWS envía una notificación de retiro de tareas al panel AWS Health y al contacto de correo electrónico principal de Cuenta de AWS. El panel AWS Health ofrece una serie de integraciones en otros servicios de AWS, incluido Amazon EventBridge. Puede utilizar EventBridge para crear automatizaciones a partir de una notificación de retirada de tarea, como aumentar la visibilidad de la próxima retirada reenviando el mensaje a una herramienta de ChatOps. AWS Health Aware es un recurso que muestra el poder del panel AWS Health y cómo se pueden distribuir las notificaciones en toda la organización. Puede reenviar una notificación de retirada de una tarea a una aplicación de chat, como Slack.

La siguiente ilustración muestra información general de la solución.

La siguiente información proporciona detalles.

- Fargate envía la notificación de retirada de la tarea al panel AWS Health.
- El panel AWS Health envía un correo al contacto principal de correo electrónico de Cuenta de AWS, y lo notifica a EventBridge.
- EventBridge tiene una regla que captura la notificación de retirada.

La regla que busca eventos con el tipo de detalle del evento: `"AWS Health Event" and the Event Detail Type Code: "AWS_ECS_TASK_PATCHING_RETIREMENT"`

- La regla activa una función Lambda que reenvía la información a Slack mediante un webhook entrante de Slack. Para obtener más información, consulte Webhooks entrantes.

Para ver un ejemplo de código, consulte Cómo capturar notificaciones de retirada de tareas de AWS Fargate en Github.

Paso 3: Controle la sustitución de tareas

No puede controlar el momento exacto en que se retira una tarea, pero puede definir un tiempo de espera. Si quiere controlar la sustitución de tareas según su propio cronograma, puede capturar el aviso de retirada de la tarea para conocer primero la fecha de retirada de la tarea. A continuación, puede volver a implementar el servicio para lanzar tareas de sustitución y, del mismo modo, sustituir cualquier tarea independiente. Para los servicios que utilizan la implementación continua, actualice el servicio utilizando `update-service` con la opción `force-deployment` antes de la hora de inicio de la retirada.

En este ejemplo de `update-service` se usa la opción `force-deployment`.

```
aws ecs update-service —-service service_name \
    --cluster cluster_name \
     --force-new-deployment
```

Para los servicios que utilizan la implementación azul/verde, debe crear una nueva implementación en AWS CodeDeploy. Para obtener información sobre cómo crear la implementación, consulte create-deployment en la Referencia de la AWS Command Line Interface.

Regiones compatibles con Amazon ECS en AWS Fargate

Puede utilizar las siguientes tablas para comprobar la compatibilidad regional con los contenedores de Linux en AWS Fargate y los contenedores de Windows en AWS Fargate.

Contenedores de Linux en AWS Fargate

Los contenedores de Linux para Amazon ECS en AWS Fargate se admite en las siguientes Regiones de AWS. Los ID de zona de disponibilidad admitidos se indican cuando corresponda.

Nombre de la región	Región
Este de EE. UU. (Ohio)	us-east-2
Este de EE. UU. (Norte de Virginia)	us-east-1
Oeste de EE. UU. (Norte de California)	us-west-1 (solo `usw1-az1` y `usw1-az3`)
Oeste de EE. UU. (Oregón)	us-west-2
Canadá (centro)	ca-central-1
Oeste de Canadá (Calgary)	ca-west-1
México (central)	mx-central-1
África (Ciudad del Cabo)	af-south-1
Asia-Pacífico (Hong Kong)	ap-east-1
Asia-Pacífico (Bombay)	ap-south-1
Asia-Pacífico (Tokio)	ap-northeast-1 (solo `apne1-az1` , `apne1-az2` y `apne1-az4`)
Asia-Pacífico (Seúl)	ap-northeast-2
Asia-Pacífico (Osaka)	ap-northeast-3
Asia-Pacífico (Hyderabad)	ap-south-2
Asia-Pacífico (Singapur)	ap-southeast-1
Asia-Pacífico (Sídney)	ap-southeast-2
Asia-Pacífico (Tailandia)	ap-southeast-7
Asia-Pacífico (Yakarta)	ap-southeast-3
Asia-Pacífico (Melbourne)	ap-southeast-4
Asia-Pacífico (Malasia)	ap-southeast-5

Nombre de la región	Región
Canadá (centro)	ca-central-1
Oeste de Canadá (Calgary)	ca-west-1
China (Pekín)	cn-north-1 (solo `cnn1-az1` y `cnn1-az2`)
China (Ningxia)	cn-northwest-1
Europe (Frankfurt)	eu-central-1
Europa (Zúrich)	eu-central-2
Europa (Irlanda)	eu-west-1
Europa (Londres)	eu-west-2
Europa (París)	eu-west-3
Europa (Milán)	eu-south-1
Europa (España)	eu-south-2
Europa (Estocolmo)	eu-north-1
América del Sur (São Paulo)	sa-east-1
Israel (Tel Aviv)	il-central-1
Medio Oriente (Baréin)	me-south-1
Medio Oriente (EAU)	me-central-1
AWS GovCloud (Este de EE. UU.)	us-gov-east-1
AWS GovCloud (Oeste de EE.UU.)	us-gov-west-1

Contenedores de Windows en AWS Fargate

Los contenedores de Windows para Amazon ECS en AWS Fargate se admiten en las siguientes Regiones de AWS. Los ID de zona de disponibilidad admitidos se indican cuando corresponda.

Nombre de la región	Región
Este de EE. UU. (Ohio)	us-east-2
Este de EE. UU. (Norte de Virginia)	us-east-1 (`use1-az1`, `use1-az2`, `use1-az4`, `use1-az5` y `use1-az6` únicamente)
Oeste de EE. UU. (Norte de California)	us-west-1 (solo `usw1-az1` y `usw1-az3`)
Oeste de EE. UU. (Oregón)	us-west-2
África (Ciudad del Cabo)	af-south-1
Asia-Pacífico (Hong Kong)	ap-east-1
Asia-Pacífico (Bombay)	ap-south-1
Asia-Pacífico (Hyderabad)	ap-south-2
Asia Pacífico (Osaka)	ap-northeast-3
Asia-Pacífico (Seúl)	ap-northeast-2
Asia-Pacífico (Singapur)	ap-southeast-1
Asia-Pacífico (Sídney)	ap-southeast-2
Asia-Pacífico (Melbourne)	ap-southeast-4
Asia-Pacífico (Malasia)	ap-southeast-5
Asia-Pacífico (Tokio)	ap-northeast-1 (solo `apne1-az1` , `apne1-az2` y `apne1-az4`)
Canadá (centro)	ca-central-1 (solo `cac1-az1` y `cac1-az2`)

Nombre de la región	Región
Oeste de Canadá (Calgary)	ca-west-1
China (Pekín)	cn-north-1 (solo `cnn1-az1` y `cnn1-az2`)
China (Ningxia)	cn-northwest-1
Europe (Frankfurt)	eu-central-1
Europa (Zúrich)	eu-central-2
Europa (Irlanda)	eu-west-1
Europa (Londres)	eu-west-2
Europa (París)	eu-west-3
Europa (Milán)	eu-south-1
Europa (España)	eu-south-2
Europa (Estocolmo)	eu-north-1
América del Sur (São Paulo)	sa-east-1
Israel (Tel Aviv)	il-central-1
Medio Oriente (EAU)	me-central-1
Medio Oriente (Baréin)	me-south-1

Arquitectura de la solución para Amazon ECS

El siguiente diagrama muestra el ciclo de vida de la aplicación y su funcionamiento con los componentes de Amazon ECS.

Debe diseñar sus aplicaciones para que puedan ejecutarse en contenedores. Un contenedor es una unidad estandarizada de desarrollo de software que contiene todo lo que la aplicación de software necesita para ejecutarse. Esto incluye código, tiempo de ejecución, herramientas del sistema y bibliotecas del sistema relevantes. Los contenedores se crean a partir de una plantilla de solo lectura denominada imagen. Las imágenes se crean normalmente a partir de un Dockerfile. Un Dockerfile es un archivo de texto sin formato que contiene las instrucciones para crear un contenedor. Una vez construidas, estas imágenes se almacenan en un registro, como Amazon ECR, desde donde se pueden descargar.

Después de crear y almacenar la imagen, debe crear una definición de tarea de Amazon ECS. Una definición de tarea es un esquema de la aplicación. Se trata de un archivo de texto en formato JSON que describe los parámetros y uno o varios contenedores que forman la aplicación. Por

ejemplo, puede usarla para especificar la imagen y los parámetros para el sistema operativo, qué contenedores utilizar, qué puertos abrir para la aplicación, así como qué volúmenes de datos se van a utilizar con los contenedores en la tarea. Los parámetros específicos disponibles para la definición de tareas dependerán de las necesidades de la aplicación específica.

Después de definir la definición de la tarea, la implementa como un servicio o una tarea en el clúster. Un clúster es una agrupación lógica de tareas o servicios que se ejecuta en la infraestructura de capacidad registrada en un clúster.

Una tarea es la instancia creada de una definición de tarea dentro de un clúster. Puede ejecutar una tarea independiente o ejecutar una tarea como parte de un servicio. Puede utilizar un servicio de Amazon ECS para ejecutar y mantener simultáneamente el número deseado de tareas en un clúster de Amazon ECS. El funcionamiento es que, en caso de que alguna de las tareas falle o se pare por algún motivo, el programador de servicio de Amazon ECS lanza otra instancia en función de la definición de tarea. Lo hace para reemplazarlo y así mantener el número deseado de tareas en el servicio.

El agente de contenedor se ejecuta en cada instancia de contenedor dentro de un clúster de Amazon ECS. El agente envía información a Amazon ECS acerca de la utilización de recursos y tareas actualmente en ejecución en cada recurso. Envía y detiene tareas cada vez que recibe una solicitud proveniente de Amazon ECS.

Tras implementar la tarea o el servicio, puede utilizar cualquiera de las siguientes herramientas para supervisar la implementación y la aplicación:

- CloudWatch
- Supervisión en tiempo de ejecución

Capacidad

La capacidad es la infraestructura en la que se ejecutan sus contenedores. A continuación, se muestran las opciones:

- Instancias de Amazon EC2
- Sin servidor (AWS Fargate)
- Máquinas virtuales (VM) o servidores locales en las instalaciones

Al crear un clúster, especifica la infraestructura. También especifica el tipo de infraestructura al registrar una definición de tarea. La definición de tarea hace referencia a la infraestructura como tipo de lanzamiento. También utiliza el tipo de lanzamiento cuando ejecuta una tarea independiente o implementa un servicio. Para obtener información sobre las opciones de tipo de lanzamiento, consulte Tipos de lanzamiento de Amazon ECS.

Puntos de conexión de servicio

El punto de conexión de servicio es la URL del punto de entrada para Amazon ECS que se utiliza para conectarse al servicio mediante programación a través del Protocolo de Internet versión 4 (IPv4) o el Protocolo de Internet versión 6 (IPv6). De forma predeterminada, las solicitudes que realiza para conectarse a Amazon ECS mediante programación utilizan puntos de conexión de servicio que solo admiten solicitudes de IPv4. Para conectarse mediante programación con Amazon ECS mediante solicitudes IPv4 o IPv6, puede utilizar un punto de conexión de doble pila. Para obtener más información acerca de los puntos de conexión de doble pila, consulte Uso de puntos de conexión de doble pila en Amazon ECS.

Red

Los recursos de AWS se crean en subredes. Cuando utiliza instancias EC2, Amazon ECS lanza las instancias en la subred que se especifica al crear un clúster. Las tareas se ejecutan en la subred de las instancias. En el caso de Fargate o las máquinas virtuales en las instalaciones, la subred se especifica al ejecutar una tarea o al crear un servicio.

Según la aplicación, la subred puede ser pública o privada y puede estar en cualquiera de los siguientes recursos de AWS:

- Zonas de disponibilidad
- Local Zones
- Zonas de Wavelength
- Regiones de AWS
- AWS Outposts

Para obtener más información, consulte Aplicaciones de Amazon ECS en subredes compartidas, zonas locales y zonas de Wavelength o Amazon Elastic Container Service en AWS Outposts.

Puede conectar la aplicación a Internet mediante uno de los métodos siguientes:

- Una subred pública con una puerta de enlace de Internet

  Utilice subredes públicas cuando tenga aplicaciones públicas que requieran grandes cantidades de ancho de banda o una latencia mínima. Entre los escenarios aplicables se encuentran los servicios de streaming de vídeo y juegos.
- Una subred privada con una puerta de enlace NAT

  Utilice subredes privadas cuando quiera proteger los contenedores del acceso externo directo. Entre los escenarios aplicables se encuentran los sistemas de procesamiento de pagos o los contenedores que almacenan datos de usuario y contraseñas.
- AWS PrivateLink

  Utilice AWS PrivateLink para tener conectividad privada entre las VPC, los servicios de AWS y las redes en las instalaciones sin exponer el tráfico a la Internet pública.

Acceso a la característica

Puede usar la configuración de la cuenta de Amazon ECS para acceder a las siguientes características:

- Información de contenedores

  CloudWatch Container Insights recopila, agrega y resume métricas y registros de las aplicaciones y microservicios en contenedores. Las métricas incluyen la utilización de recursos como CPU, memoria, disco y red.
- Enlace troncal de `awsvpc`

  Para algunos tipos de instancias de EC2, puede disponer de interfaces de red (ENI) adicionales en las instancias de contenedor recién lanzadas.
- Autorización de etiquetado

  Los usuarios deben tener permisos para llevar a cabo las acciones que crean recursos, como `ecsCreateCluster`. Si se especifican etiquetas en la acción de creación de recursos, AWS realiza una autorización adicional en la acción de `ecs:TagResource` para verificar que los usuarios o roles tengan permisos para crear etiquetas.
- Conformidad de Fargate con la norma FIPS-140

Fargate presta conformidad con el Federal Information Processing Standard (FIPS, Estándar Federal de Procesamiento de Información) 140, que especifica los requisitos de seguridad para los módulos criptográficos que protegen información confidencial. Es la norma gubernamental actual de los Estados Unidos y Canadá y se aplica a los sistemas que deben cumplir con la Federal Information Security Management Act (FISMA, Ley Federal de Gestión de la Seguridad de la Información) o el Federal Risk and Authorization Management Program (FedRAMP, Programa Federal de Gestión de Riesgos y Autorizaciones).

- Cambios en el tiempo de retirada de tareas de Fargate

  Puede configurar el periodo de espera antes de que se retiren las tareas de Fargate para aplicar parches.

- VPC de doble pila

  Permita que las tareas se comuniquen mediante IPv4, IPv6 o ambos.

- Formato de nombre de recurso de Amazon (ARN)

  Algunas características, como la autorización de etiquetado, requieren un nuevo formato de nombre de recurso de Amazon (ARN).

Para obtener más información, consulte Acceso a las características de Amazon ECS con la configuración de la cuenta.

Roles de IAM

Un rol de IAM es una identidad de IAM que puede crear en su cuenta y que tiene permisos específicos. En Amazon ECS, puede crear roles para conceder permisos a los recursos de Amazon ECS, como contenedores o servicios.

Algunas características de Amazon ECS requieren roles. Para obtener más información, consulte Roles de IAM para Amazon ECS.

Registro

El registro y la supervisión son aspectos importantes del mantenimiento de la fiabilidad, la disponibilidad y el rendimiento de las cargas de trabajo de Amazon ECS. Están disponibles las siguientes opciones:

- Registros de Amazon CloudWatch: dirija los registros a Amazon CloudWatch
- FireLens para Amazon ECS: dirija los registros a un servicio de AWS o a un destino de AWS Partner Network para su almacenamiento y análisis. La AWS Partner Network es una comunidad global de socios que aprovecha los programas, el conocimiento técnico y los recursos para crear, comercializar y vender ofertas para los clientes.

Tipos de lanzamiento de Amazon ECS

El tipo de lanzamiento de la definición de tarea define la capacidad en la que se puede ejecutar la tarea, por ejemplo AWS Fargate.

Tras elegir el tipo de lanzamiento, Amazon verifica que los parámetros de definición de tarea que configure funcionan con el tipo de lanzamiento.

En la tabla siguiente, se describen los tipos de lanzamiento disponibles.

Tipo de lanzamiento		Más información
Fargate	Fargate es un motor de computación de pago por uso sin servidor que le permite centrarse en crear aplicacio nes sin tener que administrar los servidores.	Tipo de lanzamiento de Fargate para Amazon ECS
EC2	Usted elige el tipo de instancia y la cantidad de instancias y administra la capacidad.	Tipo de lanzamiento de EC2 para Amazon ECS
Máquinas virtuales (VM) o servidores locales en las instalaciones	Amazon ECS Anywhere admite el registro de una instancia externa, por ejemplo, un servidor ubicado en las instalaciones o una máquina virtual (VM), en el clúster.	Tipo de lanzamiento externo (Amazon ECS Anywhere) para Amazon ECS

Tipo de lanzamiento de Fargate para Amazon ECS

Fargate es un motor de computación de pago por uso sin servidor que le permite centrarse en crear aplicaciones sin tener que administrar los servidores. Al elegir Fargate, no tiene que administrar la infraestructura de EC2. Lo único que tiene que hacer es crear la imagen del contenedor y definir en qué clúster quiere ejecutar las aplicaciones. Fargate tiene una integración nativa con los servicios de AWS que incluye:

- Amazon VPC
- Auto Scaling
- Elastic Load Balancing
- IAM
- Secrets Manager

Tiene más control con Fargate que con EC2 porque selecciona exactamente la CPU y la memoria que necesita su aplicación. Fargate se encarga de escalar horizontalmente su capacidad, por lo que no tiene que preocuparse por los picos de tráfico. Esto significa que, con Fargate, el esfuerzo operativo es menor.

Fargate cumple con los estándares de los programas de conformidad, entre ellos PCI, FIPS 140-3, FedRAMP e HIPAA. Para obtener más información, consulte Servicios de AWS en el ámbito del programa de conformidad.

Fargate es adecuado para las siguientes cargas de trabajo:

- Cargas de trabajo grandes que requieren una sobrecarga operativa baja
- Pequeñas cargas de trabajo que tienen ráfagas ocasionales
- Cargas de trabajo pequeñas
- Cargas de trabajo en lotes

Para obtener información acerca de las regiones que admiten Fargate, consulte the section called “Regiones de AWS Fargate”.

En el siguiente diagrama, se muestra la arquitectura general.

Para obtener más información acerca de Amazon ECS en Fargate, consulte AWS Fargate para Amazon ECS.

Comportamiento de extracción de imágenes de contenedores de Linux en Fargate para Amazon ECS

Cada tarea de Fargate se ejecuta en su propia instancia de un solo uso y de un solo inquilino. Cuando ejecuta contenedores de Linux en Fargate, las imágenes del contenedor o las capas de imágenes del contenedor no se almacenan en caché en la instancia. Por lo tanto, para cada imagen de contenedor definida en la tarea, es necesario extraer toda la imagen del contenedor del registro de imágenes de contenedor para cada tarea de Fargate. El tiempo que se tarda en extraer las imágenes está directamente relacionado con el tiempo que se tarda en iniciar una tarea de Fargate.

Tenga en cuenta lo siguiente para optimizar el tiempo de extracción de la imagen.

Proximidad de la imagen del contenedor

Para reducir el tiempo que se tarda en descargar las imágenes del contenedor, ubique los datos lo más cerca posible de la computación. Extraer una imagen de un contenedor a través de Internet o en Regiones de AWS puede afectar al tiempo de descarga. Le recomendamos que guarde la imagen del contenedor en la misma región en la que se ejecutará la tarea. Si almacena la imagen del contenedor en Amazon ECR, utilice un punto de conexión de interfaz de VPC para reducir aún más el tiempo de extracción de la imagen. Para obtener más información, consulte Puntos de conexión de VCP de la interfaz de Amazon ECR (AWS PrivateLink) en la Guía del usuario de Amazon ECR.

Reducción del tamaño de la imagen del contenedor

El tamaño de la imagen de un contenedor afecta directamente al tiempo de descarga. Reducir el tamaño de la imagen del contenedor o el número de capas de la imagen del contenedor puede reducir el tiempo que tarda una imagen en descargarse. Las imágenes base ligeras (como la imagen mínima del contenedor de Amazon Linux 2023) pueden ser considerablemente más pequeñas que las basadas en las imágenes base de los sistemas operativos tradicionales. Para obtener más información sobre la imagen mínima, consulte AL2023 Minimal container image en la Guía del usuario de Amazon Linux 2023.

Algoritmos de compresión alternativos

Las capas de imágenes de contenedores suelen comprimirse cuando se insertan en un registro de imágenes de contenedores. Al comprimir la capa de imágenes del contenedor, se reduce la cantidad de datos que deben transferirse a través de la red y almacenarse en el registro de imágenes de contenedores. Una vez que el tiempo de ejecución del contenedor haya descargado una capa de imágenes de contenedor en una instancia, esa capa se descomprime. El algoritmo

de compresión utilizado y la cantidad de vCPU disponibles para el tiempo de ejecución afectan al tiempo que se tarda en descomprimir la imagen del contenedor. En Fargate, puede aumentar el tamaño de la tarea o aprovechar el algoritmo de compresión zstd, de mayor desempeño, para reducir el tiempo necesario para la descompresión. Para obtener más información, consulte ztsd en GitHub. Para obtener información sobre cómo implementar las imágenes para Fargate, consulte Reducing AWS Fargate Startup Times with zstd Compressed Container Images.

Carga diferida de imágenes de contenedores

En el caso de imágenes de contenedores de gran tamaño (> 250 MB), lo mejor sería cargar de forma diferida una imagen de contenedor en lugar de descargar toda la imagen del contenedor. En Fargate, puede utilizar Seekable OCI (SOCI) para cargar de forma diferida una imagen de contenedor desde un registro de imágenes de contenedores. Para obtener más información, consulte soci-snapshotter en GitHub y Carga diferida de imágenes de contenedores mediante Seekable OCI (SOCI).

Comportamiento de extracción de imágenes de contenedor de Windows en Fargate para Amazon ECS

Fargate Windows almacena en caché la imagen base del núcleo del servidor del mes más reciente y del mes anterior proporcionada por Microsoft. Estas imágenes coinciden con los parches de KB/número de compilación que se actualizan cada martes de parches. Por ejemplo, el 9 de abril de 2024, Microsoft lanzó KB5036896 (17763.5696) para Windows Server 2019. El KB del mes anterior, el 12 de marzo de 2024, fue KB5035849 (17763.5576). Por lo tanto, las siguientes imágenes de contenedores se almacenaron en caché para las plataformas `WINDOWS_SERVER_2019_CORE` y `WINDOWS_SERVER_2019_FULL`:

- `mcr.microsoft.com/windows/servercore:ltsc2019`
- `mcr.microsoft.com/windows/servercore:10.0.17763.5696`
- `mcr.microsoft.com/windows/servercore:10.0.17763.5576`

Además, el 9 de abril de 2024, Microsoft lanzó KB5036909 (20348.2402) para Windows Server 2022. El KB del mes anterior, el 12 de marzo de 2024, fue KB5035857 (20348.2340). Por lo tanto, para las plataformas `WINDOWS_SERVER_2022_CORE` y `WINDOWS_SERVER_2022_FULL` los siguientes contenedores, las imágenes se almacenaron en caché:

- `mcr.microsoft.com/windows/servercore:ltsc2022`

- `mcr.microsoft.com/windows/servercore:10.0.20348.2402`
- `mcr.microsoft.com/windows/servercore:10.0.20348.2340`

Tipo de lanzamiento de EC2 para Amazon ECS

El tipo de lanzamiento de EC2 es adecuado para grandes cargas de trabajo cuyo precio se debe optimizar.

A la hora de plantear cómo modelar las definiciones de tareas y servicios mediante el tipo de lanzamiento de EC2, le recomendamos que considere qué procesos se tienen que ejecutar de forma conjunta y cómo se escalaría cada componente.

Por ejemplo, supongamos que una aplicación consta de los siguientes componentes:

- Un servicio frontend que muestre información en una página web
- Un servicio backend que proporciona las API para el servicio frontend
- Un almacén de datos

En este ejemplo, cree definiciones de tareas que agrupen los contenedores que se utilizan para un fin común. Separe los diferentes componentes en definiciones de tareas múltiples e independientes. El clúster de ejemplo siguiente tiene tres instancias de contenedor que se ejecutan en tres contenedores de servicio frontend, dos contenedores de servicio backend y un contenedor de servicios de almacén de datos.

Puede agrupar contenedores relacionados en una definición de tarea, por ejemplo contenedores vinculados que se deben ejecutar conjuntamente. Por ejemplo, agregue un contenedor de flujo de registro a su servicio frontend e inclúyalo en la misma definición de tareas.

Después de tener sus definiciones de tareas, puede crear servicios para mantener la disponibilidad de sus tareas deseadas. Para obtener más información, consulte Creación de una implementación de actualización continua de Amazon ECS. En sus servicios, puede asociar contenedores a los balanceadores de carga de Elastic Load Balancing. Para obtener más información, consulte Uso del equilibrador de carga para distribuir el tráfico de servicio de Amazon ECS. Cuando cambian los requisitos de la aplicación, puede actualizar los servicios para aumentar o reducir el número de tareas deseadas. También puede actualizar los servicios para implementar versiones más nuevas de los contenedores de las tareas. Para obtener más información, consulte Actualización de un servicio de Amazon ECS.

Comportamiento de extracción de imágenes de contenedores para EC2 y tipos de lanzamiento externo para Amazon ECS

El tiempo que tarda un contenedor en iniciarse varía según la imagen del contenedor subyacente. Por ejemplo, una imagen más pesada (versiones completas de Debian, Ubuntu y Amazon1/2) puede tardar más en iniciarse porque hay más servicios que se ejecutan en los contenedores en comparación con sus respectivas versiones compactas (Debian-slim, Ubuntu-slim y Amazon-SLIM) o imágenes base más pequeñas (Alpine).

Cuando el agente de Amazon ECS inicia una tarea, extrae la imagen de Docker de su registro remoto y, a continuación, guarda en caché una copia local. Cuando utiliza una etiqueta de imagen nueva para cada versión de la aplicación, este comportamiento no es necesario.

El parámetro del agente `ECS_IMAGE_PULL_BEHAVIOR` determina el comportamiento de extracción de imágenes. Están disponibles las siguientes opciones:

- `ECS_IMAGE_PULL_BEHAVIOR`: `default`

La imagen se extraerá de forma remota. Si la extracción falla, se usa la imagen en caché de la instancia.

- `ECS_IMAGE_PULL_BEHAVIOR`: `always`

  La imagen se extraerá de forma remota. Si no se puede extraer, se produce un error en la tarea.

Para acelerar la implementación, defina el parámetro del agente Amazon ECS en el siguiente valor:

- `ECS_IMAGE_PULL_BEHAVIOR`: `once`

  La imagen se extrae de forma remota únicamente si no la extrajo una tarea anterior en la misma instancia de contenedor o si la imagen en caché ha sido eliminada por el proceso de limpieza automatizada de imágenes. De lo contrario, se usa la imagen almacenada en la caché de la instancia. De este modo, se garantiza que no se realizan extracciones de imágenes innecesarias.
- `ECS_IMAGE_PULL_BEHAVIOR`: `prefer-cached`

  La imagen se extrae de forma remota si no hay ninguna imagen en caché. De lo contrario, se usa la imagen almacenada en la caché de la instancia. La limpieza automatizada de imágenes se deshabilita para el contenedor con el fin de garantizar que no se elimine la imagen en caché.

Si se establece el parámetro `ECS_IMAGE_PULL_BEHAVIOR` en cualquiera de los valores anteriores, se puede ahorrar tiempo, ya que el agente de Amazon ECS utiliza la imagen descargada existente. En el caso de imágenes de Docker más grandes, el tiempo de descarga puede tardar entre 10 y 20 segundos en llegar a la red.

Tipo de lanzamiento externo (Amazon ECS Anywhere) para Amazon ECS

Amazon ECS Anywhere admite el registro de una instancia externa, por ejemplo, un servidor ubicado en las instalaciones o una máquina virtual (VM), en el clúster de Amazon ECS. Las instancias externas se han optimizado para que puedan ejecutar aplicaciones que generen tráfico o datos del proceso salientes. Si la aplicación requiere tráfico entrante, la falta de compatibilidad con Elastic Load Balancing hace que la ejecución de estas cargas de trabajo sea menos eficiente. Amazon ECS ha agregado un nuevo tipo de lanzamiento `EXTERNAL` que se puede utilizar para crear servicios o ejecutar tareas en las instancias externas.

A continuación, se proporciona información general sobre la arquitectura de sistema de alto nivel de Amazon ECS Anywhere. El servidor en las instalaciones tiene instalados el agente de Amazon ECS y el agente de SSM.

Para obtener más información, consulte Clústeres de Amazon ECS para el tipo de lanzamiento externo.

Uso de puntos de conexión de doble pila en Amazon ECS

Los puntos de conexión de doble pila de Amazon ECS admiten solicitudes a Amazon ECS a través del Protocolo de Internet versión 4 (IPv4) y el Protocolo de Internet versión 6 (IPv6). Para obtener una lista de todos los puntos de conexión de Amazon ECS, consulte Puntos de conexión y cuotas de Amazon ECS en la Referencia general de AWS.

Cuando se utiliza la API de REST, accede directamente a un punto de conexión de Amazon ECS mediante el nombre del punto de conexión (URI). Amazon ECS solo es compatible con nombres de puntos de conexión de doble pila regionales, lo que implica que debe especificar la región como parte del nombre.

Utilice la siguiente convención de nomenclatura para los nombres de puntos de conexión de doble pila: `ecs.`*`region`*`.api.aws`.

Al usar AWS Command Line Interface (AWS CLI) y los SDK de AWS, puede utilizar un parámetro o una marca para cambiar a un punto de conexión de doble pila. También puede especificar el punto de conexión de doble pila directamente como una anulación del punto de conexión de Amazon ECS en el archivo de configuración.

En las secciones siguientes, se describe cómo utilizar los puntos de conexión de doble pila desde la AWS CLI, los SDK de AWS y la API de REST.

Temas

- Usar puntos de enlace de doble pila desde la AWS CLI
- Uso de los puntos de enlace de doble pila de los SDK de AWS
- Usar los puntos de enlace de doble pila desde la API REST

Usar puntos de enlace de doble pila desde la AWS CLI

Esta sección proporciona ejemplos de comandos de la AWS CLI. que se usan para realizar solicitudes a un punto de conexión de doble pila. Para obtener más información acerca de cómo instalar la AWS CLI o actualizarla a la versión más reciente, consulte Instalación o actualización de la versión más reciente de la AWS CLI en la Guía del usuario de la versión 2 de la AWS Command Line Interface.

Para utilizar un punto de conexión de doble pila, puede establecer el valor de configuración `use_dualstack_endpoint` en `true` en su archivo de `config` para la AWS CLI a fin de dirigir todas las solicitudes de Amazon ECS que realicen los comandos `ecs` y AWS CLI de al punto de conexión de doble pila para la región especificada. Puede especificar la región en el archivo `config` o en un comando mediante la opción `--region`. Para obtener más información sobre los archivos de configuración de la AWS CLI, consulte Configuración de archivos de configuración y credenciales en la AWS CLI en la Guía del usuario de la versión 2 de la AWS Command Line Interface.

Si quiere utilizar un punto de conexión de doble pila para comandos de la AWS CLI específicos, puede usar uno de los métodos siguientes:

- Puede utilizar el punto de conexión de doble pila por cada comando si configura el parámetro `--endpoint-url` como `https://ecs.`*`aws-region`*`.api.aws` o `http://ecs.`*`aws-region`*`.api.aws` para cualquier comando de `ecs`.

  El siguiente comando de ejemplo enumera todos los clústeres disponibles y utiliza el punto de conexión de doble pila para la solicitud.

```
$ aws ecs list-clusters --endpoint-url https://ecs.aws-region.api.aws
```

- Puede configurar perfiles separados en su archivo de AWS Config. Por ejemplo, puede crear un perfil que configure `use_dualstack_endpoint` como `true` y un perfil que no configure `use_dualstack_endpoint`. Al ejecutar un comando, especifique qué perfil quiere usar, en función de si quiere usar el punto de conexión de doble pila o no.

Uso de los puntos de enlace de doble pila de los SDK de AWS

En esta sección, se proporcionan ejemplos de cómo obtener acceso a un punto de enlace de doble pila con los SDK de AWS.

AWS SDK for Java 2.x

En los siguientes ejemplos, se muestra cómo especificar un punto de conexión para la región `us-east-1` mediante AWS SDK for Java 2.x.

```
Region region = Region.US_EAST_1
EcsClient client =
 EcsClient.builder().region(region).dualstackEnabled(true).build();
```

AWS SDK para Go

En los siguientes ejemplos, se muestra cómo especificar un punto de conexión para la región `us-east-1` mediante AWS SDK para Go.

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

Para obtener más información, consulte Puntos de conexión de doble pila y FIPS en la Guía de referencia de AWS SDK y las herramientas.

Usar los puntos de enlace de doble pila desde la API REST

Cuando utiliza la API de REST, puede acceder directamente a un punto de conexión de doble pila si lo especifica en su solicitud. El siguiente ejemplo utiliza el punto de conexión de doble pila para enumerar todos los clústeres de Amazon ECS en la región `us-east-1`.

```
POST / HTTP/1.1
Host: ecs.us-east-1.api.aws
Accept-Encoding: identity
Content-Length: 2
X-Amz-Target: AmazonEC2ContainerServiceV20141113.ListClusters
X-Amz-Date: 20150429T170621Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{}
```

Aplicaciones de Amazon ECS en subredes compartidas, zonas locales y zonas de Wavelength

Amazon ECS admite cargas de trabajo que aprovechan las zonas locales, zonas Wavelength y AWS Outposts ante requisitos de baja latencia o de procesamiento de datos locales.

- Las zonas locales son una extensión de una Región de AWS para colocar recursos en varias ubicaciones más cercanas a los usuarios finales.
- Puede usar las zonas Wavelength para crear aplicaciones que ofrecen latencia extremadamente baja para dispositivos 5G y usuarios finales. Wavelength implementa servicios de computación y almacenamiento de AWS estándar al borde de redes 5G de operadores de telecomunicaciones.
- AWS Outposts brinda Servicios de AWS, infraestructura y modelos operativos nativos a prácticamente cualquier centro de datos, espacio de coubicación o instalación en las instalaciones.

> **Important**
>
> En este momento, las cargas de trabajo de Amazon ECS en AWS Fargate no se admiten en zonas locales, zonas Wavelength o AWS Outposts.

Para obtener información acerca de las diferencias entre zonas locales, zonas de Wavelength y AWS Outposts, consulte ¿Cómo se cuándo debo usar AWS Local Zones (zonas locales), AWS Wavelength o AWS Outposts para aplicaciones que requieren latencia baja o procesamiento de datos local? en las Preguntas frecuentes sobre AWS Wavelength.

Subredes compartidas

Puede usar Uso compartido de VPC para compartir subredes con otras cuentas de AWS dentro de la misma AWS Organizations.

Puede usar VPC compartidas para el tipo de lanzamiento de EC2 con las siguientes consideraciones:

- El propietario de la subred de VPC debe compartir una subred con una cuenta participante para que esa cuenta pueda usarla con los recursos de Amazon ECS.
- No puede usar el grupo de seguridad predeterminado para la VPC con las instancias de contenedor porque pertenece al propietario. Además, los participantes no pueden lanzar instancias mediante grupos de seguridad que sean propiedad de otros participantes o del propietario.
- En una subred compartida, el participante y el propietario controlan por separado los grupos de seguridad de cada cuenta respectiva. El propietario de la subred puede ver estos grupos de seguridad creados por los participantes, pero no puede realizar ninguna acción en ellos. Si el propietario de la subred quiere eliminar o modificar estos grupos de seguridad, el participante que ha creado el grupo de seguridad debe realizar la acción.
- El propietario de la VPC compartida no puede ver, actualizar ni eliminar un clúster que un participante cree en la subred compartida. Esto se suma a los recursos de VPC a los que cada cuenta tiene un acceso diferente. Para obtener más información, consulte Responsabilidades y permisos de los propietarios y los participantes en la Guía del usuario de Amazon VPC.

Puede usar VPC compartidas para el tipo de lanzamiento Fargate con las siguientes consideraciones:

- El propietario de la subred de VPC debe compartir una subred con una cuenta participante para que esa cuenta pueda usarla con los recursos de Amazon ECS.
- No puede crear servicios ni ejecutar tareas mediante el grupo de seguridad predeterminado de la VPC porque pertenece al propietario. Además, los participantes no pueden crear servicios ni ejecutar tareas mediante grupos de seguridad que sean propiedad de otros participantes o del propietario.

- En una subred compartida, el participante y el propietario controlan por separado los grupos de seguridad de cada cuenta respectiva. El propietario de la subred puede ver estos grupos de seguridad creados por los participantes, pero no puede realizar ninguna acción en ellos. Si el propietario de la subred quiere eliminar o modificar estos grupos de seguridad, el participante que ha creado el grupo de seguridad debe realizar la acción.
- El propietario de la VPC compartida no puede ver, actualizar ni eliminar un clúster que un participante cree en la subred compartida. Esto se suma a los recursos de VPC a los que cada cuenta tiene un acceso diferente. Para obtener más información, consulte Responsabilidades y permisos de los propietarios y los participantes en la Guía del usuario de Amazon VPC.

Para obtener más información sobre el uso compartido de la subred de VPC, consulte Compartir su VPC con otras cuentas en la Guía del usuario de Amazon VPC.

Zonas locales

Una zona local es una extensión de una Región de AWS que se encuentra geográficamente cerca de los usuarios. Las zonas locales tienen sus propias conexiones a internet y admiten AWS Direct Connect. Los recursos creados en una zona local pueden prestar servicio a los usuarios locales con comunicaciones de baja latencia. Para obtener más información, consulte AWS Local Zones.

Una zona local se representa mediante un código de región seguido de un identificador que indica la ubicación (por ejemplo, `us-west-2-lax-1a`).

Para utilizar una zona local, primero debe registrarse en ella. Una vez registrado, debe crear una VPC de Amazon y una subred en la zona local.

Puede lanzar instancias de Amazon EC2, los servidores de archivos de Amazon FSx y los equilibradores de carga de aplicación para utilizarlos en sus clústeres y tareas de Amazon ECS.

Para obtener más información, consulte la sección ¿Qué son las zonas locales de AWS? en la Guía del usuario de zonas locales de AWS.

Zonas de Wavelength

Puede usar AWS Wavelength para crear aplicaciones que ofrecen una latencia extremadamente baja para dispositivos móviles y usuarios finales. Wavelength implementa servicios de computación y almacenamiento de AWS estándar al borde de redes 5G de operadores de telecomunicaciones. Puede ampliar una Amazon Virtual Private Cloud a una o varias zonas Wavelength. A continuación,

puede utilizar recursos de AWS, como instancias de Amazon EC2, para ejecutar aplicaciones que requieren latencia ultrabaja y una conexión a Servicios de AWS en la región.

Una zona Wavelength es una zona aislada en la ubicación del operador donde se implementa la infraestructura de Wavelength. Las zonas Wavelength están vinculadas a una Región de AWS. Una zona de Wavelength es una extensión lógica de una región y está administrada por el plano de control de la región.

Una zona Wavelength se representa mediante un código de región seguido de un identificador que indica la zona Wavelength, por ejemplo, `us-east-1-wl1-bos-wlz-1`.

Para utilizar una zona Wavelength, primero debe registrarse en la zona. Una vez registrado, debe crear una Amazon VPC y una subred en la zona Wavelength. A continuación, puede lanzar sus instancias de Amazon EC2 en la zona para utilizarlos en sus clústeres y tareas de Amazon ECS.

Para obtener más información, consulte Introducción a AWS Wavelength en la Guía para desarrolladores de AWS Wavelength.

Las zonas Wavelength no están disponibles en todas las Regiones de AWS. Para obtener información sobre las regiones que admiten zonas de Wavelength, consulte Zonas de Wavelength disponibles en la Guía para desarrolladores de AWS Wavelength.

Amazon Elastic Container Service en AWS Outposts

AWS Outposts habilita los servicios, la infraestructura y los modelos operativos nativos de AWS en instalaciones locales. En los entornos de AWS Outposts, puede utilizar las mismas API, herramientas e infraestructura de AWS que utiliza en la Nube de AWS.

Amazon ECS en AWS Outposts es ideal para cargas de trabajo de baja latencia que deben ejecutarse cerca de los datos y las aplicaciones en las instalaciones.

Para obtener más información sobre AWS Outposts, consulte la Guía del usuario de AWS Outposts.

Consideraciones

Estas son las consideraciones para utilizar Amazon ECS en AWS Outposts:

- Amazon Elastic Container Registry, AWS Identity and Access Management y el equilibrador de carga de red se ejecutan en la región de AWS, no en AWS Outposts. Esto aumentará las latencias entre estos servicios y los contenedores.

- AWS Fargate no está disponible en AWS Outposts.

A continuación, se incluyen algunas consideraciones de conectividad de red para AWS Outposts:

- Si se pierde la conectividad de red entre la instancia de AWS Outposts y la región de AWS, los clústeres seguirán ejecutándose. Sin embargo, no podrá crear nuevos clústeres ni realizar nuevas acciones en clústeres existentes hasta que se restablezca la conectividad. En caso de errores en la instancia, la instancia no se reemplazará automáticamente. El agente de CloudWatch Logs no podrá actualizar los registros ni los datos de eventos.
- Le recomendamos que proporcione una conexión fiable, de alta disponibilidad y baja latencia entre el AWS Outposts y la región de AWS.

Requisitos previos

Estos son los requisitos previos para utilizar Amazon ECS en AWS Outposts:

- Debe haber instalado y configurado un Outpost en su centro de datos local.
- Debe contar con una conexión de red fiable entre el Outpost y la región de AWS.

Descripción general de la creación de clústeres en AWS Outposts

A continuación se presenta una descripción general de la configuración:

1. Cree un rol y una política con derechos en el AWS Outposts.
2. Cree un perfil de instancias de IAM con derechos en el AWS Outposts.
3. Cree una VPC o utilice una existente que se encuentre en la misma región que la suya. AWS Outposts
4. Cree una subred o utilice una existente que esté asociada a AWS Outposts.

   Esta es la subred en la que se ejecutan las instancias del contenedor.
5. Cree un grupo de seguridad para las instancias de contenedor en su clúster.
6. Cree un clúster de Amazon ECS.
7. Defina las variables de entorno del agente de contenedor de Amazon ECS para lanzar la instancia en el clúster.
8. Ejecute el contenedor.

Para obtener información detallada sobre cómo integrar Amazon ECS con AWS Outposts, consulte Ampliación de Amazon ECS en dos bastidores AWS Outposts.

En el siguiente ejemplo, se crea un clúster de Amazon ECS en un AWS Outposts.

1. Cree un rol y una política con derechos en el AWS Outposts.

   El archivo `role-policy.json` es el documento de política que contiene el efecto y las acciones de los recursos. Para obtener información sobre el formato de los archivos, consulte PutRolePolicy en la Referencia de la API de IAM

```
aws iam create-role --role-name ecsRole \
    --assume-role-policy-document file://ecs-policy.json
aws iam put-role-policy --role-name ecsRole --policy-name ecsRolePolicy \
    --policy-document file://role-policy.json
```

2. Cree un perfil de instancias de IAM con derechos en el AWS Outposts.

```
aws iam create-instance-profile --instance-profile-name outpost
aws iam add-role-to-instance-profile --instance-profile-name outpost \
    --role-name ecsRole
```

3. Cree una VPC.

```
aws ec2 create-vpc --cidr-block 10.0.0.0/16
```

4. Para crear una subred asociada a su AWS Outposts.

```
aws ec2 create-subnet \
    --cidr-block 10.0.3.0/24 \
    --vpc-id vpc-xxxxxxxx \
    --outpost-arn arn:aws:outposts:us-west-2:123456789012:outpost/op-
xxxxxxxxxxxxxxxx \
    --availability-zone-id usw2-az1
```

5. Cree un grupo de seguridad para las instancias de contenedor y especifique el rango de CIDR adecuado para el AWS Outposts. (Este paso es diferente para AWS Outposts).

```
aws ec2 create-security-group --group-name MyOutpostSG
aws ec2 authorize-security-group-ingress --group-name MyOutpostSG --protocol tcp \
    --port 22 --cidr 10.0.3.0/24
aws ec2 authorize-security-group-ingress --group-name MyOutpostSG --protocol tcp \
```

```
    --port 80 --cidr 10.0.3.0/24
```

6. Cree el clúster.
7. Defina las variables de entorno del agente de contenedor de Amazon ECS para lanzar la instancia en el clúster creado en el paso anterior y defina las etiquetas que desee agregar para ayudar a identificar el clúster (por ejemplo, `Outpost` para indicar que el clúster es para un Outpost).

```
#! /bin/bash
cat << 'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_IMAGE_PULL_BEHAVIOR=prefer-cached
ECS_CONTAINER_INSTANCE_TAGS={"environment": "Outpost"}
EOF
```

> **Note**
>
> Para evitar retrasos causados por la extracción de imágenes de contenedor de Amazon ECR en la región, utilice imágenes almacenadas en caché. Para ello, cada vez que se ejecute una tarea, configure el agente de Amazon ECS de modo que utilice de forma predeterminada la imagen almacenada en caché en la propia instancia estableciendo `ECS_IMAGE_PULL_BEHAVIOR` en `prefer-cached`.

8. Cree la instancia de contenedor y especifique la VPC y la subred para el AWS Outposts donde debe ejecutarse esta instancia, así como un tipo de instancias disponible en el AWS Outposts. (Este paso es diferente para AWS Outposts).

   El archivo `userdata.txt` contiene los datos de usuario que la instancia puede utilizar para llevar a cabo tareas de configuración automatizadas comunes e incluso ejecutar scripts después de que se inicie la instancia. Para obtener información acerca del archivo para las llamadas a la API, consulte Ejecutar comandos en la instancia de Linux durante la inicialización en la Guía del usuario de Amazon EC2.

```
aws ec2 run-instances --count 1 --image-id ami-xxxxxxxx --instance-type c5.large \
    --key-name aws-outpost-key --subnet-id subnet-xxxxxxxxxxxxxxxxx \
    --iam-instance-profile Name outpost --security-group-id sg-xxxxxx \
    --associate-public-ip-address --user-data file://userdata.txt
```

> **Note**
>
> Este comando también se utiliza cuando se añaden instancias adicionales al clúster. Los contenedores implementados en el clúster se colocarán en ese AWS Outposts específico.

Prácticas recomendadas para redes de Amazon ECS

Por lo general, las aplicaciones modernas se crean a partir de varios componentes distribuidos que se comunican entre sí. Por ejemplo, una aplicación móvil o web puede comunicarse con un punto de conexión de la API, y la API puede interactuar con varios microservicios que se comunican a través de Internet.

Para obtener información sobre las prácticas recomendadas para conectar las aplicaciones a Internet, consulte Conexión de las aplicaciones de Amazon ECS a Internet.

Para obtener información sobre las prácticas recomendadas para recibir conexiones entrantes a Amazon ECS desde Internet, consulte Prácticas recomendadas para recibir conexiones entrantes a Amazon ECS desde Internet.

Para obtener información sobre las prácticas recomendadas para conectar Amazon ECS a otros servicios de AWS, consulte Prácticas recomendadas para conectar Amazon ECS a los servicios de AWS desde su VPC.

Para obtener más información sobre las prácticas recomendadas para conectar servicios dentro de la VPC, consulte Prácticas recomendadas para conectar los servicios de Amazon ECS en una VPC.

Para obtener información sobre las prácticas recomendadas para los servicios de red entre las Cuentas de AWS y las VPC, consulte Prácticas recomendadas para conectar en red los servicios de Amazon ECS entre Cuentas de AWS y las VPC.

Para obtener información sobre las prácticas recomendadas para los servicios de solución de problemas de las redes, consulte Servicios de AWS para la solución de problemas de redes de Amazon ECS.

Conexión de las aplicaciones de Amazon ECS a Internet

La mayoría de las aplicaciones en contenedores tienen al menos algunos componentes que necesitan acceso de salida a Internet. Por ejemplo, el backend de una aplicación móvil requiere acceso de salida a las notificaciones push.

Amazon Virtual Private Cloud cuenta con dos métodos principales para facilitar la comunicación entre su VPC e Internet.

Subred pública y puerta de enlace de Internet

Cuando usa una subred pública que tiene una ruta a una puerta de enlace de Internet, la aplicación en contenedores se puede ejecutar en un host dentro de una VPC en una subred pública. Al host que ejecuta el contenedor se le asigna una dirección IP pública. Esta dirección IP pública se puede enrutar desde Internet. Para obtener más información, consulte Puertas de enlace de Internet en la Guía del usuario de Amazon VPC.

Esta arquitectura de red facilita la comunicación directa entre el host que ejecuta la aplicación y otros hosts de Internet. La comunicación es bidireccional. Esto significa que no solo puede establecer una conexión de salida con cualquier otro host de Internet, sino que otros hosts de Internet también podrían intentar conectarse al suyo. Por lo tanto, debe prestar mucha atención a las reglas de firewall y grupos de seguridad. Esto garantiza que otros hosts de Internet no puedan abrir ninguna conexión que no quiera que se abra.

Por ejemplo, si la aplicación se ejecuta en Amazon EC2, asegúrese de que el puerto 22 para el acceso mediante SSH no esté abierto. De lo contrario, su instancia podría recibir intentos constantes de conexión SSH por parte de bots malintencionados de Internet. Estos bots rastrean direcciones IP públicas. Cuando encuentran un puerto SSH abierto, intentan utilizar contraseñas a la fuerza para acceder a la instancia. Por este motivo, muchas organizaciones limitan el uso de las subredes públicas y prefieren tener la mayoría de sus recursos, si no todos, dentro de subredes privadas.

Optar por subredes públicas para las redes es adecuado cuando se trata de aplicaciones públicas que requieran grandes cantidades de ancho de banda o una latencia mínima. Entre los casos de uso aplicables se encuentran los servicios de streaming de vídeo y juegos.

Este enfoque con respecto a las redes se admite tanto cuando se utiliza Amazon ECS en Amazon EC2 y cuando se utiliza en AWS Fargate.

- Amazon EC2: puede lanzar instancias de EC2 en una subred pública. Amazon ECS usa estas instancias de EC2 como capacidad del clúster y cualquier contenedor que se ejecute en las instancias puede usar la dirección IP pública subyacente del host para las redes salientes. Esto se aplica a los modos de red de `host` y `bridge`. Sin embargo, el modo de red `awsvpc` no proporciona las ENI de tarea con direcciones IP públicas. Por lo tanto, no pueden utilizar directamente una puerta de enlace de Internet.
- Fargate: cuando cree su servicio de Amazon ECS, especifique las subredes públicas para la configuración de red de su servicio y utilice la opción Asignar dirección IP pública. Cada tarea de Fargate está conectada a la subred pública y tiene su propia dirección IP pública para la comunicación directa con Internet.

Subred privada y puerta de enlace NAT

Cuando usa una subred privada y una puerta de enlace NAT, puede ejecutar la aplicación en contenedores en un host que se encuentre en una subred privada. Por lo tanto, este host tiene una dirección IP privada que se puede enrutar dentro de la VPC, pero no se puede enrutar desde Internet. Esto significa que otros hosts de la VPC pueden conectarse al host mediante su dirección IP privada, pero los hosts de Internet no pueden establecer ninguna comunicación entrante con el host.

Con una subred privada, puede utilizar una puerta de enlace de traducción de direcciones de red (NAT) para permitir que un host dentro de una subred privada se conecte a Internet. Los hosts de Internet reciben una conexión entrante que parece provenir de la dirección IP pública de la puerta de enlace NAT que se encuentra dentro de una subred pública. La puerta de enlace de NAT es responsable de actuar como puente entre Internet y la subred privada. Esta configuración suele preferirse por motivos de seguridad, ya que significa que su VPC está protegida del acceso directo de los atacantes de Internet. Para obtener información, consulte Gateways NAT en la Guía del usuario de Amazon VPC.

Este enfoque respecto a las redes privadas es adecuado para situaciones en las que desee proteger los contenedores frente al acceso externo directo. Entre los escenarios aplicables se encuentran los sistemas de procesamiento de pagos o los contenedores que almacenan datos de usuario y contraseñas. Se le cobrará por la creación y el uso de una gateway NAT en su cuenta. Además, se aplican tarifas de procesamiento de datos y uso por horas de la puerta de enlace NAT. Para obtener redundancia, debe tener una puerta de enlace NAT en cada zona de disponibilidad. De esta forma, la pérdida de disponibilidad de una única zona de disponibilidad no pone en peligro su conectividad de salida. Por ello, si tiene una carga de trabajo pequeña, podría resultar más rentable utilizar subredes privadas y puertas de enlace NAT.

Este enfoque con respecto a las redes se admite tanto cuando se utiliza Amazon ECS en Amazon EC2 y cuando se utiliza en AWS Fargate.

- Amazon EC2: puede lanzar instancias de EC2 en una subred privada. Los contenedores que se ejecutan en estos hosts de EC2 utilizan las redes de los hosts subyacentes y las solicitudes de salida atraviesan la puerta de enlace NAT.
- Fargate: cuando cree su servicio de Amazon ECS, especifique las subredes privadas para la configuración de red de su servicio y no utilice la opción Asignar dirección IP pública. Cada tarea de Fargate está alojada en una subred privada. Su tráfico de salida se enruta a través de cualquier puerta de enlace NAT que haya asociado a esa subred privada.

Prácticas recomendadas para recibir conexiones entrantes a Amazon ECS desde Internet

Si ejecuta un servicio público, debe aceptar el tráfico entrante de Internet. Por ejemplo, su sitio web público debe aceptar las solicitudes HTTP entrantes de los navegadores. En tal caso, otros hosts de Internet también tienen que iniciar una conexión entrante con el host de su aplicación.

Una forma de solucionar este problema consiste en lanzar los contenedores en los hosts que se encuentren en una subred pública con una dirección IP pública. Sin embargo, no recomendamos esta opción para aplicaciones a gran escala. Para ello, un mejor enfoque consiste en disponer de una capa de entrada escalable que se encuentre entre Internet y la aplicación. Para este enfoque, puede utilizar como entrada cualquiera de los servicios de AWS que se enumeran en esta sección.

Equilibrador de carga de aplicación

En la capa de la aplicación funciona un equilibrador de carga de aplicación. Es la séptima capa del modelo de interconexión de sistemas abiertos (OSI). Esto hace que el equilibrador de carga de

aplicación sea adecuado para los servicios HTTP públicos. Si tiene un sitio web o una API de REST HTTP, entonces el equilibrador de carga de aplicación es el equilibrador de carga adecuado para esta carga de trabajo. Para obtener más información, consulte ¿Qué es un equilibrador de carga de aplicación? en la Guía del usuario de equilibradores de carga de aplicaciones.

Con esta arquitectura, se crea un equilibrador de carga de aplicación en una subred pública para que tenga una dirección IP pública y pueda recibir conexiones entrantes de Internet. Cuando el equilibrador de carga de aplicación recibe una conexión entrante o, más específicamente, una solicitud HTTP, abre una conexión con la aplicación mediante su dirección IP privada. A continuación, reenvía la solicitud a través de la conexión interna.

El equilibrador de carga de aplicación tiene las siguientes ventajas.

- Terminación SSL/TLS: el equilibrador de carga de aplicación puede mantener una comunicación HTTPS segura y los certificados para las comunicaciones con los clientes. Opcionalmente, se puede finalizar la conexión SSL del equilibrador de carga para que no tenga que gestionar los certificados en su propia aplicación.
- Enrutamiento avanzado: un equilibrador de carga de aplicación puede tener varios nombres de host DNS. También cuenta con capacidades de enrutamiento avanzadas para enviar solicitudes HTTP entrantes a diferentes destinos en función de métricas como el nombre de host o la ruta de la solicitud. Esto significa que puede usar un único equilibrador de carga de aplicación como entrada para muchos servicios internos diferentes, o incluso microservicios en diferentes rutas de una API de REST.

- Compatibilidad con gRPC y websockets: un equilibrador de carga de aplicación puede gestionar más que HTTP. También puede equilibrar la carga de servicios basados en gRPC y websocket, con compatibilidad con HTTP/2.
- Seguridad: un equilibrador de carga de aplicación ayuda a proteger su aplicación del tráfico malicioso. Incluye características como las mitigaciones de desincronización de HTTP y está integrado con el firewall de aplicaciones web de AWS (AWS WAF). AWS WAF puede filtrar más aún el tráfico malicioso que podría contener patrones de ataque, como inyección de código SQL o scripting entre sitios.

Equilibrador de carga de red

Un equilibrador de carga de red actúa como la cuarta capa del modelo de interconexión de sistemas abiertos (OSI). Es adecuado para protocolos que no son HTTP o escenarios en los que se necesita cifrado de extremo a extremo, pero no tiene las mismas características específicas de HTTP que un equilibrador de carga de aplicación. Por lo tanto, el equilibrador de carga de red es la opción más adecuada para las aplicaciones que no utilizan HTTP. Para obtener más información, consulte ¿Qué es un equilibrador de carga de red? en la Guía del usuario de equilibradores de carga de red.

Cuando se utiliza un equilibrador de carga de red como entrada, este actúa de manera similar a un equilibrador de carga de aplicación. Esto se debe a que se crea en una subred pública y tiene una dirección IP pública a la que se puede acceder en Internet. A continuación, el equilibrador de carga de red abre una conexión con la dirección IP privada del host que ejecuta el contenedor y envía los paquetes del lado público al lado privado.

Características del equilibrador de carga de red

Como el equilibrador de carga de red funciona en un nivel inferior de la pila de redes, no tiene el mismo conjunto de funciones que el equilibrador de carga de aplicación. Sin embargo, tiene las siguientes características importantes.

- Cifrado de extremo a extremo: dado que el equilibrador de carga de red funciona en la cuarta capa del modelo OSI, no lee el contenido de los paquetes. Esto lo hace adecuado para equilibrar la carga de comunicaciones que requieren cifrado de extremo a extremo.
- Cifrado TLS: además del cifrado de extremo a extremo, el equilibrador de carga de red también puede finalizar las conexiones TLS. De esta forma, las aplicaciones de backend no tienen que implementar su propio TLS.
- Compatibilidad con UDP: dado que el equilibrador de carga de red funciona en la cuarta capa del modelo OSI, es adecuado para cargas de trabajo que no sean HTTP y protocolos distintos de TCP.

Cierre de conexiones

Como el equilibrador de carga de red no observa el protocolo de aplicación en las capas superiores del modelo OSI, no puede enviar mensajes de cierre a los clientes de esos protocolos. A diferencia del equilibrador de carga de aplicación, la aplicación debe cerrar esas conexiones o puede configurar el equilibrador de carga de red para las conexiones de la cuarta capa se cierren cuando se detenga o se reemplace una tarea. Consulte la configuración de finalización de la conexión para los grupos objetivo del equilibrador de carga de red en la documentación del equilibrador de carga de red.

Permitir que el equilibrador de carga de red cierre las conexiones en la cuarta capa puede provocar que los clientes muestren mensajes de error no deseados si el cliente no los gestiona. Consulte la Biblioteca de Constructores para obtener más información sobre la configuración de cliente recomendada aquí .

Los métodos para cerrar las conexiones varían según la aplicación; sin embargo, una forma consiste en garantizar que el retraso de anulación del registro de destino del equilibrador de carga de red sea

superior al tiempo de espera de la conexión del cliente. De este modo, primero el cliente agota el tiempo de espera y se vuelve a conectar correctamente a la siguiente tarea a través del equilibrador de carga de red, mientras que la tarea anterior vacía lentamente todos sus clientes. Para obtener más información sobre el retraso de anulación del registro de destino del equilibrador de carga de red, consulte la documentación del equilibrador de carga de red.

API HTTP de Amazon API Gateway

Amazon API Gateway es adecuado para aplicaciones HTTP con ráfagas repentinas en los volúmenes de solicitudes o volúmenes de solicitudes bajos. Para obtener más información consulte ¿Qué es Amazon API Gateway? en la Guía para desarrolladores de API Gateway.

El modelo de precios tanto del equilibrador de carga de aplicación como del equilibrador de carga de red incluye un precio por hora para mantener los equilibradores de carga disponibles para aceptar conexiones entrantes en todo momento. Por el contrario, API Gateway cobra por cada solicitud por separado. Esto tiene como consecuencia que, si no se recibe ninguna solicitud, no hay cargos. En condiciones de mucho tráfico, un equilibrador de carga de aplicación o equilibrador de carga de red puede gestionar un mayor volumen de solicitudes a un precio por solicitud más económico que API Gateway. Sin embargo, si tiene un número reducido de solicitudes en general o tiene periodos de poco tráfico, el precio acumulado por usar API Gateway debería ser más rentable que pagar una tarifa por hora para mantener un equilibrador de carga que se está infrautilizando. Además, API Gateway puede almacenar en la caché las respuestas de la API, lo que podría dar lugar a tasas de solicitudes de backend más bajas.

Las funciones de API Gateway utilizan un enlace de VPC que permite que el servicio administrado de AWS se conecte a los hosts de la subred privada de la VPC mediante su dirección IP privada. Puede

detectar estas direcciones IP privadas consultando los registros de detección de servicios de AWS Cloud Map administrados por Detección de servicios de Amazon ECS.

API Gateway admite las siguientes características.

- El funcionamiento de API Gateway es similar al de un equilibrador de carga, pero tiene capacidades adicionales exclusivas para la administración de API
- API Gateway ofrece capacidades adicionales en torno a la autorización del cliente, los niveles de uso y la modificación de solicitudes/respuestas. Para obtener más información, consulte Características de Amazon API Gateway.
- API Gateway puede admitir puntos de conexión para puertas de enlace de API privadas, regionales y periféricas. Los puntos de conexión periféricos están disponibles a través de una distribución administrada de CloudFront. Tanto los puntos de conexión regionales como los privados son locales de una región.
- Finalización de SSL/TLS
- Direccionamiento de distintas rutas HTTP a diferentes microservicios de backend

Además de las características anteriores, API Gateway también admite el uso de autorizadores de Lambda personalizados que puede utilizar para proteger su API de usos no autorizados. Para obtener más información, consulte Notas de campo: API basadas en contenedores sin servidor con Amazon ECS y Amazon API Gateway.

Prácticas recomendadas para conectar Amazon ECS a los servicios de AWS desde su VPC

Para que Amazon ECS funcione correctamente, el agente de contenedor de Amazon ECS, que se ejecuta en cada host, debe comunicarse con el plano de control de Amazon ECS. Si va a almacenar las imágenes del contenedor en Amazon ECR, los hosts de Amazon EC2 deben comunicarse con el punto de conexión del servicio de Amazon ECR y con Amazon S3, donde se almacenan las capas de las imágenes. Si utiliza otros servicios de AWS para la aplicación en contenedores, como los datos persistentes almacenados en DynamoDB, compruebe que estos servicios también cuenten con el soporte de red necesario.

Puerta de enlace de NAT

El uso de una puerta de enlace NAT es la forma más sencilla de garantizar que sus tareas de Amazon ECS puedan acceder a otros servicios de AWS. Para obtener más información acerca de este enfoque, consulte Subred privada y puerta de enlace NAT.

A continuación, se muestran las desventajas de utilizar este enfoque:

- No se pueden limitar los destinos con los que puede comunicarse la puerta de enlace NAT. Tampoco se pueden limitar los destinos con los que puede comunicarse su nivel de backend sin interrumpir todas las comunicaciones salientes de su VPC.
- Las puertas de enlace NAT cobran por cada GB de datos que pasan por ellas. Si utiliza la puerta de enlace NAT para alguna de las siguientes operaciones, se le cobrará por cada GB de ancho de banda utilizado:
  - Descarga de archivos grandes desde Amazon S3
  - Realización de consultas extensivas sobre bases de datos en DynamoDB
  - Extracción de imágenes desde Amazon ECR

  Además, las puertas de enlace NAT admiten 5 Gbps de ancho de banda y escalan automáticamente hasta 45 Gbps. Si el enrutamiento se realiza a través de una única puerta de

enlace NAT, las aplicaciones que requieren conexiones de ancho de banda muy elevado pueden experimentar limitaciones de red. Como solución alternativa, puede dividir la carga de trabajo en varias subredes y asignar a cada subred su propia puerta de enlace NAT.

AWS PrivateLink

AWS PrivateLink proporciona conectividad privada entre las VPC, los servicios de AWS y las redes en las instalaciones sin exponer el tráfico a la Internet pública.

Un punto de conexión de VPC habilita conexiones privadas entre la VPC y los servicios de AWS, así como los servicios de punto de conexión de VPC admitidos. El tráfico entre su VPC y el servicio no sale de la red de Amazon. Un punto de conexión de VPC no requiere una puerta de enlace de Internet, una puerta de enlace privada virtual, un dispositivo NAT, una conexión de VPN ni una conexión de AWS Direct Connect. Las instancias de Amazon EC2 en su VPC no necesitan direcciones IP públicas para comunicarse con los recursos del servicio.

El siguiente diagrama muestra cómo funciona la comunicación con los servicios de AWS cuando se utilizan puntos de conexión de VPC en lugar de una puerta de enlace de Internet. AWS PrivateLink proporciona interfaces de red elásticas (ENI) dentro de la subred y las reglas de enrutamiento de VPC se utilizan para enviar cualquier comunicación al nombre de host del servicio a través de la ENI, directamente al servicio de destino de AWS. Este tráfico ya no necesita usar la puerta de enlace NAT ni la puerta de enlace de Internet.

A continuación, se enumeran algunos de los puntos de conexión de VPC comunes que se utilizan con el servicio de Amazon ECS.

- Puntos de conexión de puerta de enlace para Amazon S3
- Punto de conexión de VPC de DynamoDB
- Punto de conexión de VPC de Amazon ECS
- Punto de conexión de VPC de Amazon ECR

Otros servicios de AWS son compatibles con puntos de conexión de VPC. Si hace un uso intensivo de algún servicio de AWS, debe consultar la documentación específica de ese servicio y saber cómo crear un punto de conexión de VPC para ese tráfico.

Prácticas recomendadas para conectar los servicios de Amazon ECS en una VPC

Al utilizar las tareas de Amazon ECS en una VPC, puede dividir las aplicaciones monolíticas en partes independientes que se pueden implementar y escalar de forma independiente en un entorno seguro. Esta arquitectura se conoce como arquitectura orientada a servicios (SOA) o microservicios. Sin embargo, puede resultar difícil garantizar que todas estas partes, tanto dentro como fuera de una VPC, puedan comunicarse entre sí. Existen diversos enfoques para facilitar la comunicación, cada uno con sus respectivas ventajas y desventajas.

Uso de Service Connect

Recomendamos Service Connect, que proporciona la configuración de Amazon ECS para la detección de servicios, la conectividad y la supervisión del tráfico. Con Service Connect, sus aplicaciones pueden utilizar nombres abreviados y puertos estándar para conectarse a los servicios del mismo clúster o a otros clústeres, incluso a través de VPC dentro de la misma región. Para obtener más información, consulte Amazon ECS Service Connect.

Cuando utiliza Service Connect, Amazon ECS administra todas las partes de la detección de servicios: crea los nombres que se pueden descubrir, administra dinámicamente las entradas de cada tarea a medida que se inician y se detienen las tareas y ejecuta un agente en cada tarea que está configurado para descubrir los nombres. La aplicación puede buscar los nombres mediante la funcionalidad estándar para los nombres de DNS y establecer conexiones. Si su aplicación ya lo hace, no tendrá que modificarla para usar Service Connect.

Los cambios solo se realizan durante las implementaciones.

Usted proporciona la configuración completa dentro de cada definición de servicio y tarea. Amazon ECS administra los cambios en esta configuración en cada implementación de servicio para garantizar que todas las tareas de una implementación se comporten de la misma manera. Por ejemplo, un problema común con el DNS en la detección de servicios es el control de una migración. Si cambia un nombre de DNS para que apunte a las nuevas direcciones IP de reemplazo, es posible que pase el tiempo máximo de TTL antes de que todos los clientes comiencen a usar el nuevo servicio. Con Service Connect, la implementación del cliente actualiza la configuración sustituyendo las tareas del cliente. Puede configurar el disyuntor de implementación y otras configuraciones de implementación para que afecten a los cambios de Service Connect de la misma manera que cualquier otra implementación.

Uso de la detección de servicios

Otro enfoque para la comunicación de servicio a servicio es la comunicación directa mediante la detección de servicios. En este enfoque, puede utilizar la integración de detección de servicios AWS

Cloud Map con Amazon ECS. Mediante la detección de servicios, Amazon ECS sincroniza la lista de tareas iniciadas en AWS Cloud Map, que mantiene un nombre de host DNS que se resuelve en las direcciones IP internas de una o más tareas de ese servicio en particular. Otros servicios de Amazon VPC pueden usar este nombre de host DNS para enviar tráfico directamente a otro contenedor mediante su dirección IP interna. Para obtener más información, consulte Detección de servicios.

En el diagrama anterior, hay tres servicios. El `service-a-local` tiene un contenedor y se comunica con el `service-b-local`, que tiene dos contenedores. El `service-b-local` también debe comunicarse con el `service-c-local`, que tiene un contenedor. Cada contenedor de estos tres servicios puede usar los nombres DNS internos desde AWS Cloud Map para buscar las direcciones IP internas de un contenedor del servicio descendente con el que necesita comunicarse.

Este enfoque de comunicación de servicio a servicio proporciona una baja latencia. A primera vista, este enfoque es sencillo porque no hay componentes adicionales entre los contenedores. El tráfico viaja directamente de un contenedor al otro.

Este enfoque es adecuado cuando se utiliza el modo de red `awsvpc`, en el que cada tarea tiene su propia dirección IP única. La mayoría de los programas solo admiten el uso de registros `A` de DNS, que se resuelven directamente en las direcciones IP. Cuando se utiliza el modo de red `awsvpc`, la dirección IP de cada tarea es un registro `A`. Sin embargo, si utiliza el modo de red `bridge`, es

posible que varios contenedores compartan la misma dirección IP. Además, las asignaciones dinámicas de puertos hacen que a los contenedores se les asignen números de puerto de forma aleatoria en esa única dirección IP. En este punto, un registro `A` ya no es suficiente para la detección de servicios. También debe usar un registro `SRV`. Este tipo de registro puede hacer un seguimiento de las direcciones IP y los números de puerto, pero requiere que configure las aplicaciones de forma adecuada. Es posible que algunas aplicaciones prediseñadas que utilice no admitan registros `SRV`.

Otra ventaja del modo de red `awsvpc` es que tiene un grupo de seguridad único para cada servicio. Puede configurar este grupo de seguridad para permitir las conexiones entrantes únicamente desde los servicios ascendentes específicos que necesitan comunicarse con ese servicio.

La principal desventaja de la comunicación directa entre servicios mediante la detección de servicios es que se debe implementar una lógica adicional para llevar a cabo reintentos y solucionar los errores de conexión. Los registros de DNS tienen un período de vida (TTL) que controla el periodo durante el que se almacenan en caché. El registro DNS tarda algún tiempo en actualizarse y la caché en caducar para que las aplicaciones puedan recoger la última versión del registro DNS. Por lo tanto, su aplicación podría terminar resolviendo el registro DNS para que apunte a otro contenedor que ya no está allí. Su aplicación debe gestionar los reintentos y tener una lógica para ignorar los backends defectuosos.

Uso de un equilibrador de carga interno

Otro enfoque para la comunicación entre servicios es utilizar un equilibrador de carga interno. El equilibrador de carga interno reside completamente dentro de la VPC y solo los servicios dentro de la VPC pueden acceder a él.

El equilibrador de carga mantiene una alta disponibilidad mediante la implementación de recursos redundantes en cada subred. Cuando un contenedor de `serviceA` necesita comunicarse con un contenedor de `serviceB`, abre una conexión con el equilibrador de carga. Luego, el equilibrador de carga abre una conexión a un contenedor del `service B`. El equilibrador de carga actúa como un punto centralizado para gestionar todas las conexiones entre los servicios.

Si un contenedor del `serviceB` se detiene, el equilibrador de carga puede retirar ese contenedor del grupo. El equilibrador de carga también realiza comprobaciones de estado de cada uno de los objetivos intermedios de su grupo y puede eliminar automáticamente los objetivos defectuosos del grupo hasta que vuelvan a estar en buen estado. Las aplicaciones ya no necesitan conocer la cantidad de contenedores descendentes. Solo abren sus conexiones a través del equilibrador de carga.

Este enfoque es ventajoso para todos los modos de red. El equilibrador de carga puede realizar un seguimiento de las direcciones IP cuando se usa el modo de red de `awsvpc`, así como de las combinaciones más avanzadas de direcciones IP y puertos cuando se usa el modo de red de

`bridge`. Distribuye el tráfico de manera uniforme entre todas las combinaciones de direcciones IP y puertos, incluso si varios contenedores están alojados en la misma instancia de Amazon EC2, pero en puertos diferentes.

La única desventaja de este enfoque es el costo. Para garantizar una alta disponibilidad, el equilibrador de carga debe tener recursos en cada zona de disponibilidad. Esto implica un costo adicional debido a la sobrecarga de pagar por el equilibrador de carga y por la cantidad de tráfico que maneja.

Sin embargo, puede reducir los costos generales si varios servicios comparten un equilibrador de carga. Esto es particularmente útil para los servicios de REST que utilizan un equilibrador de carga de aplicación. Puede crear reglas de enrutamiento basadas en rutas que dirijan el tráfico a diferentes servicios. Por ejemplo, `/api/user/*` podría dirigir el tráfico a un contenedor que forma parte del servicio de `user`, mientras que `/api/order/*` podría dirigir el tráfico al servicio de `order` asociado. Con este enfoque, solo paga por un equilibrador de carga de aplicación y dispone de una URL coherente para su API. Sin embargo, puede dividir el tráfico en varios microservicios del backend.

Prácticas recomendadas para conectar en red los servicios de Amazon ECS entre Cuentas de AWS y las VPC

Si forma parte de una organización con varios equipos y divisiones, probablemente implemente los servicios de forma independiente en VPC separadas dentro de una Cuenta de AWS compartida o en las VPC asociadas a varias Cuentas de AWS individuales. Independientemente de la forma en que implemente sus servicios, le recomendamos que complemente los componentes de red para ayudar a enrutar el tráfico entre las VPC. Para ello, se pueden utilizar varios servicios de AWS para complementar los componentes de red existentes.

- AWS Transit Gateway: primero debe considerar este servicio de red. Este servicio actúa como un centro para enrutar las conexiones entre las VPC de Amazon, las Cuentas de AWS y las redes en las instalaciones. Para obtener más información, consulte ¿Qué es una puerta de enlace de tránsito? en la Guía de puertas de enlace de tránsito de VPC de Amazon.
- Compatibilidad con Amazon VPC y VPN: puede utilizar este servicio para crear conexiones VPN de sitio a sitio y conectar redes en las instalaciones a su VPC. Para obtener más información, consulte ¿Qué es AWS Site-to-Site VPN? en la Guía del usuario de AWS Site-to-Site VPN.
- Amazon VPC: puede utilizar el emparejamiento de Amazon VPC para conectar varias VPC, ya sea en la misma cuenta o entre cuentas. Para obtener más información, consulte ¿Qué es una conexión de emparejamiento de VPC? en la Amazon VPC Peering Guide.

- VPC compartidas: puede usar una VPC y subredes de VPC en varias cuentas de AWS. Para obtener más información, consulte Uso de VPC compartidas en la Guía del usuario de Amazon VPC.

Servicios de AWS para la solución de problemas de redes de Amazon ECS

Los siguientes servicios y características pueden ayudarlo a obtener información sobre las configuraciones de su red y sus servicios. Puede utilizar esta información para solucionar problemas de red y optimizar sus servicios.

Información de contenedores de CloudWatch

CloudWatch Container Insights recopila, agrega y resume métricas y registros de las aplicaciones y microservicios en contenedores. Las métricas incluyen la utilización de recursos como CPU, memoria, disco y red. Las métricas están disponibles en los paneles automáticos de CloudWatch. Para obtener más información, consulte Configuración de la información de contenedores en Amazon ECS en la Guía del usuario de Amazon CloudWatch.

AWS X-Ray

AWS X-Ray es un servicio de rastreo que puede utilizar para recopilar información sobre las solicitudes de red que realiza su aplicación. Puede usar el SDK para instrumentar su aplicación y capturar los tiempos y los códigos de respuesta del tráfico entre sus servicios, así como entre sus servicios y los puntos de conexión de los servicios de AWS. Para obtener más información, consulte ¿Qué es AWS X-Ray? en la Guía para desarrolladores de AWS X-Ray.

También puede explorar los gráficos de AWS X-Ray que muestran la forma en que sus servicios se conectan entre sí. O utilícelos para explorar estadísticas agregadas sobre el rendimiento de cada enlace de servicio a servicio. Por último, puede profundizar en cualquier transacción específica para ver cómo se asocian los segmentos que representan las llamadas de red a esa transacción en particular.

Puede utilizar estas características para identificar si hay un cuello de botella en la red o si un servicio específico de su red no funciona según lo esperado.

Logs de flujo de VPC

Puede utilizar los registros de flujo de Amazon VPC para analizar el rendimiento de la red y solucionar problemas de conectividad. Con los registros de flujo de la VPC habilitados, puede

capturar un registro de todas las conexiones de la VPC. Entre estas conexiones se incluyen las interfaces de red asociadas a Elastic Load Balancing, Amazon RDS, las puertas de enlace NAT y otros servicios clave de AWS que pueda estar utilizando. Para obtener más información, consulte Logs de flujo de VPC en la Guía del usuario de Amazon VPC.

Consejos de ajuste de red

Existen varias configuraciones que puede ajustar para mejorar el rendimiento de su red.

Nofile ulimit

Si espera que su aplicación tenga mucho tráfico y maneje muchas conexiones simultáneas, debe tener en cuenta la cuota del sistema respecto al número de archivos permitidos. Cuando hay muchos sockets de red abiertos, cada uno debe estar representado por un descriptor de archivo. Si la cuota de descriptores de archivos es demasiado baja, se limitará el número de sockets de red. Esto provocará errores o conexiones fallidas. Puede actualizar la cuota específica del contenedor para el número de archivos en la definición de tareas de Amazon ECS. Si utiliza Amazon EC2 (en lugar de AWS Fargate), es posible que también tenga que ajustar estas cuotas en su instancia de Amazon EC2 subyacente.

Red sysctl

Otra categoría de configuración ajustable es la configuración de red de `sysctl`. Debe consultar la configuración específica de la distribución de Linux que prefiera. Muchas de estas configuraciones ajustan el tamaño de los búferes de lectura y escritura. Esto puede ser útil en situaciones donde se ejecutan instancias de Amazon EC2 a gran escala que contienen muchos contenedores.

Optimización del tiempo de lanzamiento de tareas de Amazon ECS

Para acelerar el lanzamiento de tareas, tenga en cuenta las siguientes recomendaciones.

- Almacenamiento en caché de las imágenes de los contenedores y las instancias de binpack

  Si utiliza el tipo de lanzamiento de EC2, puede configurar el comportamiento de extracción del agente de contenedores de Amazon ECS a `ECS_IMAGE_PULL_BEHAVIOR`: `prefer-cached`. La imagen se extrae de forma remota si no hay ninguna imagen en caché. De lo contrario, se usa la imagen almacenada en la caché de la instancia. La limpieza automatizada de imágenes se deshabilita para el contenedor con el fin de garantizar que no se elimine la imagen en caché. Esto reduce el tiempo de extracción de imágenes para lanzamientos posteriores. El efecto del

almacenamiento en caché es aún mayor cuando hay una alta densidad de tareas en las instancias de contenedor, que se puede configurar mediante la estrategia de ubicación de `binpack`. El almacenamiento en caché de las imágenes de los contenedores es especialmente beneficioso para las cargas de trabajo basadas en Windows, que suelen tener imágenes de contenedores de gran tamaño (decenas de GB). Cuando utilice la estrategia de ubicación de `binpack`, también puede considerar la posibilidad de utilizar el enlace troncal de la interfaz de red elástica (ENI) para colocar más tareas en el modo de red `awsvpc` en cada instancia de contenedor. El enlace troncal de ENI aumenta la cantidad de tareas que puede ejecutar en modo `awsvpc`. Por ejemplo, una instancia c5.large que puede admitir la ejecución simultánea de solo dos tareas, puede ejecutar hasta diez tareas con el enlace troncal de ENI.

- Elección de un modo de red óptimo

  Si bien hay muchos casos en los que el modo de red `awsvpc` es ideal, este modo de red puede aumentar de forma inherente la latencia de inicio de tareas, ya que, para cada tarea en modo `awsvpc`, los flujos de trabajo de Amazon ECS necesitan aprovisionar y adjuntar una ENI mediante la invocación de las API de Amazon EC2, lo que agrega una sobrecarga de varios segundos al lanzamiento de las tareas. Por el contrario, una ventaja clave del uso del modo de red `awsvpc` es que cada tarea tiene un grupo de seguridad para permitir o denegar el tráfico. Esto significa que tiene una mayor flexibilidad para controlar las comunicaciones entre tareas y servicios de forma más detallada. Si la velocidad de implementación es su prioridad, puede considerar usar el modo `bridge` para acelerar el inicio de las tareas. Para obtener más información, consulte the section called “Modo de red AWSVPC”.

- Haga un seguimiento del ciclo de vida de lanzamiento de las tareas para encontrar oportunidades de optimización

  A menudo es difícil saber el tiempo que tarda una aplicación en ponerse en marcha. El lanzamiento de la imagen del contenedor, la ejecución de scripts de inicio y otras configuraciones durante el inicio de la aplicación pueden llevar tiempo. Puede usar el punto de conexión de metadatos de la tarea para publicar métricas y hacer un seguimiento del tiempo de inicio de la aplicación desde `StartedAt` en la respuesta de metadatos del contenedor hasta el momento `StartedAt` de la tarea o el servicio. Con estos datos, puede comprender cómo contribuye su aplicación al tiempo total de lanzamiento y encontrar áreas en las que puede reducir la sobrecarga innecesaria específica de la aplicación y optimizar las imágenes de los contenedores. Para obtener más información, consulte Prácticas recomendadas de escalado automático y administración de la capacidad de Amazon ECS.

- Elección de un tipo de instancia óptimo (para el tipo de lanzamiento de EC2)

La elección del tipo de instancia correcto se basa en la reserva de recursos (por ejemplo, CPU o memoria) que configure en la tarea. Por lo tanto, al dimensionar la instancia, puede calcular cuántas tareas se pueden colocar en una sola instancia. Un ejemplo sencillo de una tarea bien ubicada es alojar cuatro tareas que requieren 0,5 vCPU y 2 GB de reservas de memoria en una instancia m5.large (compatible con 2 vCPU y 8 GB de memoria). Las reservas de esta definición de tarea aprovechan al máximo los recursos de la instancia.

Uso de Amazon ECS a escala

Cuando empiece a utilizar Amazon ECS a escala, tenga en cuenta cómo podrían afectarle las cuotas de servicio y las limitaciones de las API para Amazon ECS y los Servicios de AWS que se integran con Amazon ECS.

Temas

- Cuotas de servicio de Amazon ECS y limitaciones de las API
- Gestión de los problemas de limitación de Amazon ECS

Cuotas de servicio de Amazon ECS y limitaciones de las API

Amazon ECS integra varios Servicios de AWS, como Elastic Load Balancing, AWS Cloud Map y Amazon EC2. Con esta integración estrecha, Amazon ECS incluye varias características, como el equilibrio de carga de servicios, la detección de servicios, la creación de redes de tareas y el escalado automático de clústeres. Amazon ECS y los demás Servicios de AWS con los que se integra mantienen las cuotas de servicio y los límites de velocidad de las API para garantizar un rendimiento y una utilización uniformes. Estas cuotas de servicio también evitan el aprovisionamiento accidental de más recursos de los necesarios y protegen contra acciones malintencionadas que podrían aumentar su factura.

Si se familiariza con las cuotas de servicio y los límites de velocidad de las API de AWS, podrá planificar el escalado de sus cargas de trabajo sin preocuparse por una degradación inesperada del rendimiento. Para más información, consulte Cuotas de servicio de Amazon ECS y Request throttling for the Amazon ECS API.

Al escalar las cargas de trabajo en Amazon ECS, tenga en cuenta la siguiente cuota de servicio. Para obtener las instrucciones sobre cómo solicitar un aumento de las cuotas de servicio, consulte

Administración de Amazon ECS y cuotas de servicio de AWS Fargate en la AWS Management Console.

- AWS Fargate tiene cuotas que limitan el número de tareas que se ejecutan simultáneamente en cada Región de AWS. Existen cuotas para las tareas de spot de Fargate y bajo demanda en Amazon ECS. Cada cuota de servicio también incluye todos los pods de Amazon EKS que ejecute en Fargate. Para más información acerca de las cuotas de Fargate, consulte Cuotas de servicio de AWS Fargate en la Guía del usuario de Amazon Elastic Container Service para AWS Fargate.
- Para las tareas que se ejecutan en instancias de Amazon EC2, el número máximo de instancias de Amazon EC2 que puede registrar para cada clúster es de 5000. Si utiliza el escalado automático de clústeres de Amazon ECS con un proveedor de capacidad del grupo de escalado automático o si administra las instancias de Amazon EC2 para su clúster por su cuenta, esta cuota podría convertirse en un obstáculo para la implementación. Si necesita más capacidad, puede crear más clústeres o solicitar un aumento de la cuota de servicio.
- Si utiliza el escalado automático de clústeres de Amazon ECS con un proveedor de capacidad del grupo de escalado automático, al escalar sus servicios tenga en cuenta la cuota de `Tasks in the PROVISIONING state per cluster`. Esta cuota es la cantidad máxima de tareas en el estado `PROVISIONING` para cada clúster para las que los proveedores de capacidad pueden aumentar la capacidad. Si lanza un gran número de tareas al mismo tiempo, puede cumplir fácilmente con esta cuota. Un ejemplo es si implementa simultáneamente decenas de servicios, cada uno con cientos de tareas. Cuando esto sucede, el proveedor de capacidad debe lanzar nuevas instancias de contenedor para colocar las tareas cuando el clúster no tiene suficiente capacidad. Mientras el proveedor de capacidad lanza instancias de Amazon EC2 adicionales, es probable que el programador de servicios de Amazon ECS continúe lanzando tareas en paralelo. Sin embargo, esta actividad podría verse limitada debido a la capacidad insuficiente del clúster. El programador de servicios de Amazon ECS implementa una estrategia de limitación exponencial y de retroceso para volver a intentar la ubicación de las tareas a medida que se lanzan nuevas instancias de contenedores. Como resultado, es posible que los tiempos de implementación o escalado horizontal sean más lentos. Para evitar esta situación, puede planificar las implementaciones de los servicios de una de las siguientes maneras. Implemente una gran cantidad de tareas que no requieran aumentar la capacidad del clúster o mantenga la capacidad sobrante del clúster para el lanzamiento de nuevas tareas.

Además de tener en cuenta la cuota de servicio de Amazon ECS al escalar sus cargas de trabajo, considere también la cuota de servicio para los demás Servicios de AWS que estén integrados con

Amazon ECS. En la siguiente sección se describen en detalle los límites de tarifas clave de cada servicio y se proporcionan recomendaciones para los posibles problemas de limitación.

Elastic Load Balancing

Puede configurar los servicios de Amazon ECS para que utilicen Elastic Load Balancing a fin de distribuir el tráfico de manera uniforme entre las tareas. Para más información sobre cómo seleccionar un equilibrador de carga, consulte Uso del equilibrio de carga para distribuir el tráfico de servicios de Amazon ECS.

Cuotas de servicio de Elastic Load Balancing

Cuando escale sus cargas de trabajo, tenga en cuenta las siguientes cuotas de servicio de Elastic Load Balancing. La mayoría de las cuotas de servicio de Elastic Load Balancing son ajustables y se puede solicitar un aumento en la consola de Service Quotas.

Equilibrador de carga de aplicación

Cuando utilice un equilibrador de carga de aplicación, en función de su caso de uso, es posible que tenga que solicitar un aumento de cuota para:

- La cuota de `Targets per Application Load Balancer`, que es el número de destinos detrás de su equilibrador de carga de aplicación.
- La cuota de `Targets per Target Group per Region`, que es el número de destinos detrás de sus grupos de destino.

Para obtener más información, consulte Cuotas para sus Equilibradores de carga de aplicación.

Network Load Balancer

Existen limitaciones más estrictas en cuanto a la cantidad de destinos que puede registrar con un equilibrador de carga de red. Cuando utilice un equilibrador de carga de red, a menudo querrá habilitar la compatibilidad entre zonas, lo que conlleva limitaciones de escalado adicionales en `Targets per Availability Zone Per Network Load Balancer` el número máximo de destinos por zona de disponibilidad para cada equilibrador de carga de red. Para más información, consulte Quotas for your Network Load Balancers.

Limitación de la API de Elastic Load Balancing

Al configurar un servicio de Amazon ECS para usar un equilibrador de carga, se deben aprobar las comprobaciones de estado del grupo de destino antes de que se considere que el servicio

está en buen estado. Para llevar a cabo estas comprobaciones de estado, Amazon ECS invoca las operaciones de la API de Elastic Load Balancing en su nombre. Si tiene una gran cantidad de servicios configurados con equilibradores de carga en su cuenta, podría ver una ralentización de las implementaciones de los servicios debido a una posible limitación específica de las operaciones `RegisterTarget`, `DeregisterTarget` y `DescribeTargetHealth` de la API de Elastic Load Balancing. Cuando se produce una limitación, se producen errores de limitación en los mensajes de eventos del servicio de Amazon ECS.

Si experimenta una limitación de la API de AWS Cloud Map, puede contactar con Soporte para obtener orientación sobre cómo aumentar sus límites de limitación de la API de AWS Cloud Map. Para más información acerca de la supervisión y la solución de problemas tales como errores de limitación, consulte Gestión de los problemas de limitación.

Interfaces de red elásticas

Cuando las tareas utilizan el modo de red `awsvpc`, Amazon ECS proporciona una interfaz de red elástica (ENI) única para cada tarea. Cuando sus servicios de Amazon ECS utilizan un equilibrador de carga de Elastic Load Balancing, estas interfaces de red también se registran como destinos para el grupo de destino correspondiente definido en el servicio.

Cuotas de servicio de la interfaz de red elástica

Cuando ejecuta tareas que utilizan el modo de red `awsvpc`, se adjunta una interfaz de red elástica única a cada tarea. Si es necesario acceder a esas tareas a través de Internet, asigne una dirección IP pública a la interfaz de red elástica de esas tareas. Cuando escale sus cargas de trabajo de Amazon ECS, tenga en cuenta estas dos cuotas importantes:

- La cuota de `Network interfaces per Region`, que es el número máximo de interfaces de red en una Región de AWS para su cuenta.
- La cuota de `Elastic IP addresses per Region`, que es el número máximo de direcciones IP elásticas en una Región de AWS.

Estas dos cuotas de servicio son ajustables y puede solicitar un aumento de estas desde la consola de Service Quotas. Para más información, consulte Cuotas de servicio de Amazon VPC.

En cuanto a las cargas de trabajo de Amazon ECS que se alojan en instancias de Amazon EC2, al ejecutar tareas que utilizan el modo de red `awsvpc`, tenga en cuenta la cuota de servicio `Maximum network interfaces` (el número máximo de instancias de red para cada instancia de Amazon EC2). Esta cuota limita la cantidad de tareas que puede colocar en una instancia. No puede ajustar la

cuota y no está disponible en la consola de Service Quotas. Para obtener más información, consulte Direcciones IP por interfaz de red por tipo de instancia en la Guía del usuario de Amazon EC2.

Aunque no se puede cambiar el número de interfaces de red que pueden adjuntarse a una instancia de Amazon EC2, se puede utilizar la característica de enlace troncal de interfaz de red elástica para aumentar el número de interfaces de red disponibles. Por ejemplo, de forma predeterminada, una instancia `c5.large` puede tener hasta tres interfaces de red. La interfaz de red principal de la instancia cuenta como una. Por lo tanto, puede adjuntar dos interfaces de red adicionales a la instancia. Dado que cada tarea que utiliza el modo de red `awsvpc` requiere una interfaz de red, normalmente solo puede ejecutar dos de esas tareas en este tipo de instancia. Esto puede provocar una infrautilización de la capacidad del clúster. Si habilita el enlace troncal de la interfaz de red elástica, puede aumentar la densidad de la interfaz de red para colocar un mayor número de tareas en cada instancia. Con el enlace troncal activado, una instancia `c5.large` puede tener hasta 12 interfaces de red. La instancia tiene la interfaz de red principal y Amazon ECS crea y asocia una interfaz de red “troncal” a la instancia. Como resultado, con esta configuración, puede ejecutar 10 tareas en la instancia en lugar de las dos tareas predeterminadas. Para más información, consulte Enlace troncal de interfaces de red elástica.

Limitación de la API de interfaz de red elástica

Cuando ejecuta tareas que utilizan el modo de red `awsvpc`, Amazon ECS se basa en las siguientes API de Amazon EC2. Cada una de estas API tiene diferentes limitadores de API. Para más información, consulte Request throttling for the Amazon EC2 API.

- CreateNetworkInterface
- AttachNetworkInterface
- DetachNetworkInterface
- DeleteNetworkInterface
- DescribeNetworkInterfaces
- DescribeVpcs
- DescribeSubnets
- DescribeSecurityGroups
- DescribeInstances

Si las llamadas a la API de Amazon EC2 se limitan durante los flujos de trabajo de aprovisionamiento de la interfaz de red elástica, el programador de servicios de Amazon ECS lo volverá a intentar

automáticamente con retrasos exponenciales. En ocasiones, estos reintentos automáticos podrían provocar una demora en el lanzamiento de las tareas, lo que se traduce en velocidades de implementación más lentas. Cuando se produzca una limitación de la API, verá el mensaje `Operations are being throttled. Will try again later.` en los mensajes de eventos del servicio. Si experimenta limitaciones constantes de la API de Amazon EC2, puede ponerse en contacto con Soporte para obtener orientación sobre cómo aumentar sus límites de limitación de la API. Para obtener más información acerca de la supervisión y la solución de problemas tales como errores de limitación, consulte Handling throttling issues.

AWS Cloud Map

La detección de servicios de Amazon ECS utiliza las API de AWS Cloud Map para administrar los espacios de nombres de los servicios de Amazon ECS. Si sus servicios tienen un gran número de tareas, tenga en cuenta las siguientes recomendaciones. Para más información, consulte Consideraciones de la detección de servicios de Amazon ECS.

Service Quotas de AWS Cloud Map

Cuando los servicios de Amazon ECS están configurados para utilizar la detección de servicios, la cuota `Tasks per service`, que es el número máximo de tareas para el servicio, se ve afectada por la cuota de servicio `Instances per service` de AWS Cloud Map, que es el número máximo de instancias para ese servicio. En concreto, la cuota de servicio AWS Cloud Map reduce la cantidad de tareas que puede ejecutar hasta un máximo de 1000 tareas de servicio. No se puede cambiar la cuota AWS Cloud Map. Para más información, consulte Service Quotas de AWS Cloud Map.

Limitación de API de AWS Cloud Map

Amazon ECS llama a las API `ListInstances`, `GetInstancesHealthStatus`, `RegisterInstance` y `DeregisterInstance` de AWS Cloud Map en su nombre. Ayudan en la detección de servicios y en la ejecución de comprobaciones de estado al lanzar una tarea. Cuando se implementan varios servicios que utilizan la detección de servicios con una gran cantidad de tareas al mismo tiempo, esto puede provocar que se superen los límites de limitación de las API de AWS Cloud Map. Cuando esto suceda, es probable que vea el siguiente mensaje: `Operations are being throttled. Will try again later` en los mensajes de eventos del servicio de Amazon ECS y una velocidad de implementación y lanzamiento de tareas más lenta. AWS Cloud Map no documenta los límites de limitación de estas API. Si experimenta limitaciones, puede ponerse en contacto con Soporte para obtener orientación sobre cómo aumentar sus límites de limitación de las API. Para más recomendaciones sobre la supervisión y la solución de problemas tales como los errores de limitación, consulte Gestión de errores de limitación.

Gestión de los problemas de limitación de Amazon ECS

Los errores de limitación se dividen en dos categorías principales: limitación sincrónica y limitación asíncrona.

Limitación sincrónica

Cuando se produce una limitación sincrónica, recibirá inmediatamente una respuesta de error de Amazon ECS. Esta categoría suele producirse cuando llama a las API de Amazon ECS mientras ejecuta tareas o crea servicios. Para obtener más información sobre la limitación en cuestión y los límites de limitación relevantes, consulte Request throttling for the Amazon ECS API.

Cuando su aplicación inicia solicitudes de API, por ejemplo, mediante la AWS CLI o un SDK de AWS, puede corregir la limitación de la API. Para ello, puede diseñar la arquitectura de la aplicación para gestionar los errores o implementar una estrategia de retroceso exponencial y fluctuación con lógica de reintento para las llamadas a la API. Para obtener más información, consulte Tiempos de espera, reintentos y retardo con fluctuación.

Si utiliza un SDK de AWS, la lógica de reintento automática ya está integrada y es configurable.

Limitación asíncrona

La limitación asíncrona se produce debido a flujos de trabajo asíncronos en los que Amazon ECS o AWS CloudFormation podrían estar llamando a las API en su nombre para aprovisionar recursos. Es importante saber qué API de AWS invoca Amazon ECS en su nombre. Por ejemplo, la API `CreateNetworkInterface` se invoca para las tareas que utilizan el modo de red `awsvpc` y se invoca la API `DescribeTargetHealth` cuando se llevan a cabo comprobaciones de estado de las tareas registradas en un equilibrador de carga.

Cuando sus cargas de trabajo alcanzan una escala considerable, es posible que estas operaciones de la API se vean limitadas. Es decir, podrían estar lo suficientemente limitadas como para superar los límites impuestos por Amazon ECS o el Servicio de AWS al que se llama. Por ejemplo, si implementa cientos de servicios, cada uno con cientos de tareas simultáneas que utilizan el modo de red `awsvpc`, Amazon ECS invoca las operaciones de la API de Amazon EC2, como `CreateNetworkInterface` y las operaciones de la API de Elastic Load Balancing, como `RegisterTarget` o `DescribeTargetHealth` para registrar la interfaz de red elástica y el equilibrador de carga, respectivamente. Estas llamadas a la API pueden superar los límites de la API y provocar errores de limitación. El siguiente es un ejemplo de un error de limitación de Elastic Load Balancing que se incluye en el mensaje de evento del servicio.

```
{
   "userIdentity":{
      "arn":"arn:aws:sts::111122223333:assumed-role/AWSServiceRoleForECS/ecs-service-
scheduler",
      "eventTime":"2022-03-21T08:11:24Z",
      "eventSource":"elasticloadbalancing.amazonaws.com",
      "eventName":" DescribeTargetHealth ",
      "awsRegion":"us-east-1",
      "sourceIPAddress":"ecs.amazonaws.com",
      "userAgent":"ecs.amazonaws.com",
      "errorCode":"ThrottlingException",
      "errorMessage":"Rate exceeded",
      "eventID":"0aeb38fc-229b-4912-8b0d-2e8315193e9c"
   }
}
```

Cuando estas llamadas a la API comparten límites con el resto del tráfico de la API de su cuenta, puede resultar difícil supervisarlas aunque se emitan como eventos de servicio.

Supervisión de la limitación

Es importante identificar qué solicitudes de la API están limitadas y quién las emite. Puede utilizar AWS CloudTrail, que supervisa la limitación y se integra con CloudWatch, Amazon Athena y Amazon EventBridge. Puede configurar CloudTrail para enviar eventos a Registros de CloudWatch. Información de registros de CloudWatch analiza los eventos. Esto identifica los detalles de los eventos de limitación, como el usuario o el rol de IAM que hizo la llamada y el número de llamadas a la API que se hicieron. Para obtener más información, consulte Monitoring CloudTrail log files with CloudWatch Logs.

Para obtener más información acerca de Información de registros de CloudWatch e instrucciones sobre cómo consultar los archivos de registro, consulte Analyzing log data with CloudWatch Logs Insights.

Con Amazon Athena, puede crear consultas y analizar datos mediante SQL estándar. Por ejemplo, puede crear una tabla de Athena para analizar eventos de CloudTrail. Para obtener más información, consulte Using the CloudTrail console to create an Athena table for CloudTrail logs.

Tras crear una tabla de Athena, puede utilizar consultas SQL como la siguiente para investigar los errores `ThrottlingException`.

Sustituya las *`entradas del usuario`* por sus valores.

```
select eventname, errorcode,eventsource,awsregion, useragent,COUNT(*) count
FROM cloudtrail_table-name
where errorcode = 'ThrottlingException'
AND eventtime between '2024-09-24T00:00:08Z' and '2024-09-23T23:15:08Z'
group by errorcode, awsregion, eventsource, useragent, eventname
order by count desc;
```

Amazon ECS también emite notificaciones de eventos a Amazon EventBridge. Hay eventos de cambio de estado de los recursos y eventos de acción del servicio. Pueden ser eventos de limitación de API como `ECS_OPERATION_THROTTLED` y `SERVICE_DISCOVERY_OPERATION_THROTTLED`. Para obtener más información, consulte Eventos de acciones de servicio de Amazon ECS.

Un servicio, como, por ejemplo, AWS Lambda, puede consumir estos eventos para llevar a cabo acciones en respuesta. Para obtener más información, consulte Control de eventos de Amazon ECS.

Si ejecuta tareas independientes, algunas operaciones de la API, como `RunTask`, serán asíncronas, y las operaciones de reintento no se harán automáticamente. En esos casos, puede utilizar servicios como AWS Step Functions con la integración de EventBridge para volver a intentar operaciones limitadas o fallidas. Para obtener más información, consulte Manage a container task (Amazon ECS, Amazon SNS).

Uso de CloudWatch para supervisar la limitación

CloudWatch ofrece la supervisión del uso de las API en el espacio de nombres `Usage`, en Por recurso de AWS. Estas métricas se registran con el tipo API y el nombre de métrica CallCount. Puede crear alarmas para que se inicien cada vez que estas métricas alcancen un determinado umbral. Para obtener más información acerca, consulte Visualización de las cuotas de servicio y configuración de alarmas.

CloudWatch también ofrece detección de anomalías. Esta característica utiliza machine learning para analizar y establecer líneas de base en función del comportamiento concreto de la métrica en la que se ha activado. Si hay actividad inusual en la API, puede utilizar esta característica junto con las alarmas de CloudWatch. Para obtener más información, consulte Uso de la detección de anomalías de CloudWatch.

Al supervisar de forma proactiva los errores de limitación, puede ponerse en contacto con Soporte para aumentar los límites de limitación pertinentes y, además, recibir orientación sobre las necesidades específicas de su aplicación.

Prácticas recomendadas de escalado automático y administración de la capacidad de Amazon ECS

Puede ejecutar las cargas de trabajo de las aplicaciones en contenedores de todos los tamaños en Amazon ECS. Esto incluye entornos de pruebas mínimos y entornos de producción de gran tamaño que funcionan a escala global.

Con Amazon ECS, como todos los servicios de AWS, solo se paga por lo que se usa. Si diseña la aplicación de manera adecuada, puede ahorrar costos al consumir solo los recursos que necesita cuando los necesita.

Las recomendaciones siguientes le muestran cómo ejecutar las cargas de trabajo de Amazon ECS para cumplir sus objetivos de servicio y, al mismo tiempo, operar de forma rentable.

Temas

- Determinación del tamaño de las tareas para Amazon ECS
- Optimización del escalado automático de servicios de Amazon ECS
- Capacidad y disponibilidad de Amazon ECS
- Capacidad de clústeres de Amazon ECS
- Elección de los tamaños de las tareas de Fargate para Amazon ECS
- Aceleración del aprovisionamiento de capacidad de clústeres de Amazon ECS con proveedores de capacidad en Amazon EC2

Determinación del tamaño de las tareas para Amazon ECS

Una de las decisiones más importantes al implementar los contenedores en Amazon ECS es el tamaño de los contenedores y las tareas. El tamaño de los contenedores y las tareas es esencial para planificar el escalado y la capacidad.

Amazon ECS utiliza dos métricas de recursos para la capacidad: CPU y memoria. Amazon ECS mide las CPU en unidades de 1/1024 de una vCPU completa (donde 1024 unidades equivalen a 1 vCPU completa). Amazon ECS mide la memoria en megabytes.

En la definición de la tarea, puede declarar las reservas y los límites de los recursos.

Cuando se declara una reserva, se declara la cantidad mínima de recursos que requiere una tarea. La tarea recibe como mínimo la cantidad de recursos que se solicite. Es posible que la aplicación

pueda utilizar más CPU o memoria que la reserva que se declare. Sin embargo, esto está sujeto a los límites que también se hayan declarado.

Utilizar una cantidad superior a la reserva se conoce como ampliación. La expansión significa que la aplicación utiliza más recursos que los que se han reservado, pero se mantiene dentro de los límites declarados. Amazon ECS garantiza las reservas. Por ejemplo, si utiliza las instancias de Amazon EC2 para proporcionar capacidad, Amazon ECS no coloca una tarea en una instancia en la que no pueda gestionar la reserva.

Un límite es la cantidad máxima de unidades de CPU o memoria que pueden utilizar el contenedor o la tarea. Si el contenedor intenta utilizar más CPU que este límite, Amazon ECS lo limita. Si el contenedor intenta utilizar más memoria que este límite, Amazon ECS lo detiene.

Elegir estos valores puede resultar difícil. Los valores que mejor se adaptan a la aplicación dependen en gran medida de los requisitos de recursos de esta.

Las pruebas de carga de la aplicación son la clave para planificar correctamente los requisitos de recursos. Las pruebas de carga ayudan a comprender mejor los requisitos de la aplicación.

Aplicaciones sin estado

En el caso de las aplicaciones sin estado que se escalan horizontalmente, como una aplicación detrás de un equilibrador de carga, recomendamos primero determinar cuánta memoria consume la aplicación cuando atiende las solicitudes.

Para ello, puede utilizar herramientas tradicionales como `ps` o `top`. También puede utilizar soluciones de supervisión como Información de contenedores de CloudWatch.

Al determinar la reserva de CPU, tenga en cuenta cómo quiere escalar la aplicación para que cumpla con los requisitos de su empresa.

Puede utilizar reservas de CPU más pequeñas, como 256 unidades de CPU (o 1/4 de vCPU), para escalar horizontalmente de manera detallada y minimizar los costos. Sin embargo, es posible que no se escalen lo suficientemente rápido como para satisfacer los picos significativos de la demanda.

Puede utilizar las reservas de CPU más grandes para reducir y escalar horizontalmente con mayor rapidez. Esto le es útil para adaptarse a los picos de demanda con mayor rapidez. Sin embargo, las reservas de CPU más grandes cuestan más.

Otras aplicaciones

En el caso de las aplicaciones que no se escalan horizontalmente, como los trabajos singleton o los servidores de bases de datos, las consideraciones más importantes son la capacidad disponible y el costo.

Elija cantidad de memoria y CPU en función de lo que las pruebas de carga muestran que se necesita para atender el tráfico y cumplir su objetivo de servicio. Amazon ECS garantiza que la aplicación se coloque en un host que tenga la capacidad adecuada.

Optimización del escalado automático de servicios de Amazon ECS

Un servicio de Amazon ECS es un conjunto administrado de tareas. Cada servicio tiene una definición de tareas asociada, un recuento de tareas deseado y una estrategia de ubicación opcional.

El escalado automático del servicio de Amazon ECS se implementa mediante el servicio Application Auto Scaling. Application Auto Scaling utiliza las métricas de CloudWatch como origen para las métricas de escalado. También utiliza las alarmas de CloudWatch para establecer umbrales sobre cuándo ampliar o reducir el servicio.

Usted proporciona los umbrales para el escalado. Puede establecer un objetivo métrico, que se denomina escalado de seguimiento de objetivos. También puede especificar umbrales, lo que se denomina escalado por pasos.

Después de que se configura Application Auto Scaling, este calcula continuamente el recuento de tareas deseado adecuado para el servicio. También notifica a Amazon ECS cuando debe cambiar el número de tareas deseado, ya sea escalándolo o reduciéndolo horizontalmente.

Para utilizar el escalado automático del servicio de manera eficaz, debe elegir una métrica de escalado adecuada. En las siguientes secciones se describe cómo seleccionar una métrica.

Caracterización de la aplicación

Para escalar correctamente una aplicación, es necesario conocer las condiciones en las que debe reducirla horizontalmente y cuándo debe escalarla horizontalmente.

Básicamente, debe escalar horizontalmente la aplicación si se prevé que la demanda supere la capacidad. Por el contrario, puede reducirla horizontalmente para ahorrar costos cuando los recursos superan la demanda.

Identificación de una métrica de utilización

Para escalar de forma eficaz, debe identificar una métrica que indique la utilización o la saturación. Esta métrica debe presentar las siguientes propiedades para que sea útil a la hora de escalar.

- La métrica debe estar correlacionada con la demanda. Cuando mantiene los recursos estables, pero la demanda cambia, el valor de la métrica también debe cambiar. La métrica debe aumentar o disminuir cuando la demanda aumenta o disminuye.
- El valor de la métrica debe escalarse en proporción a la capacidad. Cuando la demanda se mantiene constante, la adición de más recursos debe provocar un cambio proporcional en el valor de la métrica. Por lo tanto, si se duplica el número de tareas, la métrica debería disminuir un 50 %.

La mejor forma de identificar una métrica de uso es mediante pruebas de carga en un entorno de preproducción, como un entorno de ensayo. Las soluciones de pruebas de carga comerciales y de código abierto están ampliamente disponibles. Por lo general, estas soluciones pueden generar una carga sintética o simular el tráfico de usuarios reales.

Para iniciar el proceso de pruebas de carga, primero debe crear paneles de control para las métricas de uso de la aplicación. Estas métricas incluyen el uso de la CPU, el uso de la memoria, las operaciones de E/S, la profundidad de las colas de E/S y el rendimiento de la red. Puede recopilar estas métricas con un servicio como Información de contenedores de CloudWatch. También puede recopilarlos mediante Amazon Managed Service para Prometheus junto con Amazon Managed Grafana. Durante este proceso, asegúrese de recopilar y trazar métricas para los tiempos de respuesta de su aplicación o las tasas de finalización del trabajo.

Cuando haga la prueba de carga, comience con una pequeña tasa de solicitudes o de inserción de trabajos. Mantenga esta velocidad constante durante varios minutos para permitir que la aplicación se vaya iniciando. Luego, aumente lentamente la velocidad y manténgala estable durante unos minutos. Repita este proceso y vaya aumentando la velocidad hasta que los tiempos de respuesta o finalización de la aplicación sean demasiado lentos para cumplir sus objetivos de nivel de servicio (SLO).

Cuando haga la prueba de carga, examine cada una de las métricas de utilización. Las métricas que aumentan junto con la carga son las mejores para ser sus mejores métricas de uso.

A continuación, identifique el recurso que alcanza la saturación. Al mismo tiempo, examine también las métricas de utilización para ver cuál se aplana primero en un nivel alto. O bien, examine cuál alcanza su punto máximo y, luego, bloquea primero la aplicación. Por ejemplo, si el uso de la CPU

aumenta del 0 % al 70 u 80 % a medida que se agrega carga y, después, se mantiene en ese nivel después de que agregue aún más carga, se puede decir con seguridad que la CPU está saturada. Según la arquitectura de la CPU, es posible que nunca alcance el 100 %. Por ejemplo, supongamos que el uso de la memoria aumenta a medida que se agrega carga y, después, la aplicación se bloquea repentinamente cuando alcanza el límite de memoria de la tarea o de la instancia de Amazon EC2. En esta situación, es probable que la memoria se haya consumido por completo. Es posible que la aplicación consuma varios recursos. Por lo tanto, elija primero la métrica que represente el recurso que se agota.

Por último, vuelva a hacer las pruebas de carga después de que duplique el número de tareas o instancias de Amazon EC2. Suponga que la métrica clave aumenta o disminuye a la mitad del ritmo anterior. Si este es el caso, la métrica es proporcional a la capacidad. Esta es una buena métrica de uso para el escalado automático.

Consideremos ahora este escenario hipotético. Suponga que lleva a caob una prueba de carga de una aplicación y descubre que el uso de la CPU finalmente alcanza el 80 % con 100 solicitudes por segundo. Al agregar más carga, el uso de la CPU ya no aumenta. Sin embargo, hace que la aplicación responda más lentamente. A continuación, vuelva a ejecutar la prueba de carga y duplique el número de tareas, pero mantenga la velocidad en su valor máximo anterior. Si observa que el uso medio de la CPU se reduce a aproximadamente un 40 %, el uso medio de la CPU es una buena opción para una métrica de escalado. Por otro lado, si el uso de la CPU se mantiene en un 80 % después de aumentar el número de tareas, el uso medio de la CPU no es una buena métrica de escalado. En ese caso, debe investigar más para encontrar una métrica adecuada.

Modelos de aplicación comunes y propiedades de escalado

Puede ejecutar todo tipo de software en AWS. Muchas cargas de trabajo son propias, mientras que otras se basan en el popular software de código abierto. Independientemente de dónde se originen, hemos observado algunos patrones de diseño comunes para los servicios. La forma de escalar de manera efectiva depende en gran medida del patrón.

El servidor eficiente conectado a la CPU

El servidor eficiente conectado a la CPU casi no utiliza recursos distintos del rendimiento de la CPU y la red. La aplicación puede gestionar cada solicitud por sí misma. Las solicitudes no dependen de otros servicios, como las bases de datos. La aplicación puede gestionar cientos de miles de solicitudes simultáneas y, para ello, puede utilizar varias CPU de forma eficiente. Cada solicitud se atiende mediante un subproceso dedicado con poca sobrecarga de memoria, o bien hay un bucle de eventos asíncrono que se ejecuta en cada CPU que atiende las solicitudes. Cada réplica de la

aplicación tiene la misma capacidad de gestionar una solicitud. El único recurso que puede agotarse antes que la CPU es el ancho de banda de la red. En los servicios dependientes de la CPU, el uso de la memoria, incluso con un rendimiento máximo, es una fracción de los recursos disponibles.

Puede utilizar el escalado automático basado en CPU para este tipo de aplicación. La aplicación disfruta de la máxima flexibilidad en términos de escalado. Puede escalarla verticalmente si se le proporcionan instancias de Amazon EC2 más grandes o vCPU de Fargate. Además, también puede escalarla horizontalmente mediante la adición de más réplicas. Agregar más réplicas o duplicar el tamaño de la instancia reduce a la mitad el uso medio de la CPU en relación con la capacidad.

Si utiliza la capacidad de Amazon EC2 para esta aplicación, considere colocarla en instancias optimizadas para la computación, como la familia `c5` o `c6g`.

El servidor eficiente dependiente de la memoria

El servidor eficiente dependiente de la memoria asigna una cantidad significativa de memoria por solicitud. Con la máxima simultaneidad, pero no necesariamente con el rendimiento, la memoria se agota antes de que se agoten los recursos de la CPU. La memoria asociada a una solicitud se libera cuando la solicitud finaliza. Se pueden aceptar solicitudes adicionales siempre que haya memoria disponible.

Puede utilizar el escalado automático basado en memoria para este tipo de aplicación. La aplicación disfruta de la máxima flexibilidad en términos de escalado. Puede escalarla verticalmente si se le proporcionan recursos de memoria más grandes de Amazon EC2 o Fargate. Además, también puede escalarla horizontalmente mediante la adición de más réplicas. Agregar más réplicas o duplicar el tamaño de la instancia puede reducir a la mitad el uso medio de la memoria en relación con la capacidad.

Si utiliza la capacidad de Amazon EC2 para esta aplicación, considere colocarla en instancias optimizadas para memoria, como la familia `r5` o `r6g`.

Algunas aplicaciones vinculadas a la memoria no liberan la memoria asociada a una solicitud cuando finaliza, de modo que una reducción de la simultaneidad no se traduce en una reducción de la memoria utilizada. Para ello, no es recomendable que utilice el escalado basado en memoria.

El servidor basado en trabajadores

El servidor basado en trabajadores procesa una solicitud para cada subproceso de trabajo individual, una tras otra. Los subprocesos de trabajo pueden ser subprocesos ligeros, como los subprocesos

POSIX. También pueden ser hilos más pesados, como los procesos UNIX. No importa de qué subprocesos se trate, siempre hay una simultaneidad máxima que la aplicación puede admitir. Por lo general, el límite de simultaneidad se establece proporcionalmente a los recursos de memoria disponibles. Si se alcanza el límite de simultaneidad, la aplicación coloca más solicitudes en una cola de trabajos pendientes. Si la cola de trabajos pendientes se desborda, la aplicación rechaza de inmediato las solicitudes entrantes adicionales. Entre las aplicaciones habituales que se ajustan a este patrón se incluyen el servidor web Apache y Gunicorn.

La simultaneidad de solicitudes suele ser la mejor métrica para escalar esta aplicación. Como hay un límite de simultaneidad para cada réplica, es importante escalar horizontalmente antes de alcanzar el límite medio.

La mejor forma de obtener métricas de simultaneidad de solicitudes es hacer que la aplicación las informe a CloudWatch. Cada réplica de su aplicación puede publicar el número de solicitudes simultáneas como una métrica personalizada con una frecuencia alta. Recomendamos que la frecuencia se establezca como mínimo una vez cada minuto. Tras recopilar varios informes, puede utilizar la simultaneidad media como métrica de escalado. Para calcular esta métrica, tome la simultaneidad total y divídala entre el número de réplicas. Por ejemplo, si la simultaneidad total es 1000 y el número de réplicas es 10, la simultaneidad media es 100.

Si su aplicación está detrás de un equilibrador de carga de aplicaciones, también puede usar la métrica `ActiveConnectionCount` del equilibrador de carga como un factor en la métrica de escalado. Debe dividir la métrica `ActiveConnectionCount` entre el número de réplicas para obtener un valor promedio. Debe utilizar el valor promedio para escalar, en lugar del valor de recuento sin procesar.

Para que este diseño funcione mejor, la desviación estándar de la latencia de respuesta debe ser pequeña a tasas de solicitud bajas. Recomendamos que, durante los periodos de baja demanda, la mayoría de las solicitudes se respondan en poco tiempo, y no hay muchas solicitudes que tarden mucho más tiempo que la media en responder. El tiempo de respuesta promedio debe estar cerca del percentil 95. De lo contrario, podrían producirse desbordamientos de colas como consecuencia de ello. Esto provoca errores. Le recomendamos que proporcione réplicas adicionales cuando sea necesario para mitigar el riesgo de desbordamiento.

El servidor de espera

El servidor de espera procesa en parte cada solicitud, pero depende en gran medida de uno o más servicios intermedios para funcionar. Las aplicaciones contenedoras suelen hacer un uso intensivo de los servicios secundarios, como las bases de datos y otros servicios de API. Estos servicios

pueden tardar algún tiempo en responder, especialmente en escenarios de alta capacidad o alta simultaneidad. Esto se debe a que estas aplicaciones suelen utilizar pocos recursos de la CPU y utilizan su máxima simultaneidad en términos de memoria disponible.

El servicio de espera es adecuado tanto en el modelo de servidor limitado en memoria como en el modelo de servidor basado en el trabajador, según el diseño de la aplicación. Si la simultaneidad de la aplicación está limitada únicamente por la memoria, se debe utilizar el uso medio de la memoria como métrica de escalado. Si la simultaneidad de la aplicación se basa en un límite de trabajadores, se debe utilizar la simultaneidad media como métrica de escalado.

El servidor basado en Java

Si su servidor basado en Java está vinculado a la CPU y se escala proporcionalmente a los recursos de la CPU, podría ser adecuado para el patrón eficiente de servidor dependiente de la CPU. Si ese es el caso, el uso medio de la CPU podría ser adecuado como métrica de escalado. Sin embargo, muchas aplicaciones Java no dependen de la CPU, lo que dificulta su escalabilidad.

Para obtener el mejor rendimiento, le recomendamos que asigne la mayor cantidad de memoria posible al montón de máquinas virtuales de Java (JVM). Las versiones recientes de la JVM, incluida la actualización 191 o posterior de Java 8, establecen automáticamente el tamaño del montón lo más grande posible para que quepa en el contenedor. Esto significa que, en Java, el uso de la memoria rara vez es proporcional al uso de las aplicaciones. A medida que aumentan la tasa de solicitudes y la simultaneidad, el uso de la memoria permanece constante. Por este motivo, no recomendamos escalar los servidores basados en Java en función del uso de la memoria. En su lugar, solemos recomendar escalar el uso de la CPU.

En algunos casos, los servidores basados en Java sufren el agotamiento del montón antes de agotar la CPU. Si su aplicación es propensa al agotamiento del montón en alta simultaneidad, el promedio de conexiones es la mejor métrica de escalado. Si su aplicación es propensa al agotamiento del montón en alto rendimiento, la tasa media de solicitudes es la mejor métrica de escalado.

Servidores que utilizan otros tiempos de ejecución de recopilación de elementos no utilizados

Muchas aplicaciones de servidor se basan en tiempos de ejecución de recopilación de elementos no utilizados, como .NET y Ruby. Es posible que estas aplicaciones de servidor se ajusten a uno de los patrones descritos anteriormente. Sin embargo, al igual que con Java, no recomendamos escalar estas aplicaciones en función de la memoria, ya que su uso medio de memoria observado no suele estar correlacionado con el rendimiento o la simultaneidad.

En el caso de estas aplicaciones, se recomienda escalar el uso de la CPU si la aplicación está vinculada a esta. De lo contrario, le recomendamos que escale según el rendimiento medio o la simultaneidad media, en función de los resultados de las pruebas de carga.

Procesadores de tareas

Muchas cargas de trabajo implican el procesamiento asíncrono de las tareas. Incluyen aplicaciones que no reciben solicitudes en tiempo real, sino que se suscriben a una cola de tareas para recibir trabajos. Para este tipo de aplicaciones, la métrica de escalado adecuada casi siempre es la profundidad de la cola. El crecimiento de las colas indica que el trabajo pendiente supera la capacidad de procesamiento, mientras que una cola vacía indica que hay más capacidad que trabajo por hacer.

Los servicios de mensajería de AWS, como Amazon SQS y Amazon Kinesis Data Streams, proporcionan métricas de CloudWatch que se pueden utilizar para escalar. Para Amazon SQS, `ApproximateNumberOfMessagesVisible` es la mejor métrica. En Kinesis Data Streams, considere utilizar la métrica `MillisBehindLatest`, publicada por Kinesis Client Library (KCL). Esta métrica debe promediarse entre todos los consumidores antes de utilizarla para escalar.

Capacidad y disponibilidad de Amazon ECS

La disponibilidad de las aplicaciones es fundamental para ofrecer una experiencia sin errores y minimizar la latencia de las aplicaciones. La disponibilidad depende de que los recursos sean accesibles y tengan la capacidad suficiente para satisfacer la demanda. AWS proporciona varios mecanismos para administrar la disponibilidad. En el caso de las aplicaciones que aloja en Amazon ECS, estas incluyen el escalado automático y las zonas de disponibilidad (AZ). El escalado automático administra la cantidad de tareas o instancias en función de las métricas que defina, mientras que las zonas de disponibilidad le permiten alojar la aplicación en ubicaciones aisladas pero cercanas geográficamente.

Al igual que ocurre con el tamaño de las tareas, la capacidad y la disponibilidad presentan ciertas desventajas que debe tener en cuenta. Lo ideal sería que la capacidad estuviera perfectamente alineada con la demanda. Siempre habría suficiente capacidad para atender las solicitudes y procesar los trabajos a fin de cumplir los objetivos de nivel de servicio (SLO), lo que incluye una latencia y una tasa de errores bajas. La capacidad nunca sería demasiado alta, lo que generaría un costo excesivo; tampoco sería demasiado baja, lo que generaría una latencia y una tasa de errores altas.

El escalado automático es un proceso latente. En primer lugar, CloudWatch debe recibir métricas en tiempo real. A continuación, CloudWatch debe agregarlas para su análisis, lo que puede tardar varios minutos en función de la granularidad de la métrica. CloudWatch compara las métricas con los umbrales de alarma para identificar la escasez o el exceso de recursos. Para evitar la inestabilidad, debe configurar las alarmas para que requieran que se supere el umbral establecido durante unos minutos antes de que suene la alarma. También lleva tiempo aprovisionar nuevas tareas y terminar las tareas que ya no son necesarias.

Debido a estos posibles retrasos en el sistema, debe mantener cierto margen mediante el aprovisionamiento excesivo. El aprovisionamiento excesivo puede ayudar a adaptarse a las ampliaciones de demanda a corto plazo. Esto también ayuda a la aplicación a atender solicitudes adicionales sin saturarse. Como práctica recomendada, puede establecer su objetivo de escalado entre el 60 y el 80 % de utilización. Esto ayuda a la aplicación a gestionar mejor las ráfagas de demanda adicional mientras aún se está aprovisionando capacidad adicional.

Otro motivo por el que recomendamos aprovisionar en exceso es para poder responder rápidamente a los errores en las zonas de disponibilidad. AWS recomienda que las cargas de trabajo de producción se atiendan desde varias zonas de disponibilidad. Esto se debe a que, si se produce un error en una zona de disponibilidad, las tareas que se ejecutan en las zonas de disponibilidad restantes pueden seguir satisfaciendo la demanda. Si la aplicación se ejecuta en dos zonas de disponibilidad, debe duplicar el número normal de tareas. Esto es para que pueda proporcionar capacidad inmediata en caso de que se produzca un posible error. Si la aplicación se ejecuta en tres zonas de disponibilidad, le recomendamos que ejecute 1,5 veces el número normal de tareas. Es decir, ejecute tres tareas por cada dos que se necesiten para un servicio normal.

Maximización de la velocidad de escalado

El escalado automático es un proceso reactivo que tarda en surtir efecto. Sin embargo, hay algunas maneras de ayudar a minimizar el tiempo necesario para escalar horizontalmente.

Minimice el tamaño de la imagen. Las imágenes más grandes tardan más en descargarse de un repositorio de imágenes y descomprimirse. Por lo tanto, al reducir el tamaño de las imágenes, se reduce el tiempo necesario para que se inicie un contenedor. Para reducir el tamaño de la imagen, puede seguir estas recomendaciones específicas:

- Si puede crear un binario estático o usar Golang, cree la imagen `FROM` desde cero e incluya solo la aplicación binaria en la imagen resultante.
- Utilice imágenes base minimizadas de proveedores de distribuciones ascendentes, como Amazon Linux o Ubuntu.

- No incluya ningún artefacto de compilación en la imagen final. El uso de compilaciones en varias etapas puede ayudar en este sentido.
- Comprima las etapas de `RUN` siempre que sea posible. Cada etapa de `RUN` crea una nueva capa de imagen, lo que implica un proceso adicional de ida y vuelta para descargar la capa. Una sola etapa de `RUN` que tiene varios comandos unidos por `&&` tiene menos capas que una con varias etapas de `RUN`.
- Si desea incluir datos, como datos de inferencia de ML, en la imagen final, incluya solo los datos necesarios para iniciar y empezar a atender el tráfico. Si obtiene datos bajo demanda de Amazon S3 u otro tipo de almacenamiento sin que ello afecte al servicio, almacene los datos en esos lugares.

Mantenga las imágenes cerca. Cuanto más alta sea la latencia de red, mas tardará en descargarse la imagen. Aloje sus imágenes en un repositorio en la misma región de AWS en la que se encuentra su carga de trabajo. Amazon ECR es un repositorio de imágenes de alto rendimiento que está disponible en todas las regiones en las que está disponible Amazon ECS. Evite utilizar Internet o un enlace de VPN para descargar imágenes de contenedores. El alojamiento de las imágenes en la misma región mejora la fiabilidad general. Mitiga el riesgo de problemas de conectividad de red y de disponibilidad en una región diferente. Como alternativa, también puede implementar la replicación entre regiones de Amazon ECR a modo de ayuda.

Reduzca los umbrales de comprobación de estado del equilibrador de carga. Los equilibradores de carga realizan comprobaciones de estado antes de enviar tráfico a la aplicación. La configuración de comprobación de estado predeterminada para un grupo de destino puede tardar 90 segundos o más. Durante este proceso, el equilibrador de carga comprueba el estado y recibe las solicitudes. Reducir el intervalo de comprobación de estado y el recuento de umbrales puede hacer que la aplicación acepte el tráfico más rápido y reducir la carga de otras tareas.

Considere el rendimiento de arranque en frío. Algunas aplicaciones utilizan tiempos de ejecución como Java para realizar la compilación Just-In-Time (JIT). El proceso de compilación, al menos cuando se inicia, puede mostrar el rendimiento de la aplicación. Una solución alternativa consiste en reescribir las partes de la carga de trabajo que son fundamentales para la latencia en lenguajes que no supongan una penalización del rendimiento al arrancar en frío.

Utilice políticas de escalado por pasos y escalado no centrado en el seguimiento de destino. Dispone de varias opciones de escalado automático de aplicaciones para las tareas de Amazon ECS. El seguimiento de destinos es el modo más fácil de utilizar. Con él, todo lo que necesita hacer es establecer un valor objetivo para una métrica, como la utilización media de la CPU. Luego, el

escalador automático administra de forma automática la cantidad de tareas necesarias para alcanzar ese valor. Con el escalado por pasos, puede reaccionar más rápidamente a los cambios en la demanda, ya que define los umbrales específicos para sus métricas de escalado y cuántas tareas agregar o eliminar cuando se superen los umbrales. Y, lo que es más importante, puede reaccionar muy rápidamente ante los cambios en la demanda al minimizar el tiempo que una alarma supera el umbral. Para obtener más información, consulte Servicio de escalado automático en la Guía para desarrolladores de servicio de contenedor elástico de Amazon.

Si utiliza instancias de Amazon EC2 para proporcionar la capacidad del clúster, considere las recomendaciones siguientes:

Utilice instancias de Amazon EC2 más grandes y volúmenes de Amazon EBS más rápidos. Puede mejorar las velocidades de descarga y preparación de imágenes mediante el uso de una instancia de Amazon EC2 más grande y un volumen de Amazon EBS más rápido. Dentro de una familia de instancias de Amazon EC2 determinada, el rendimiento máximo de la red y de Amazon EBS aumenta a medida que aumenta el tamaño de la instancia (por ejemplo, de `m5.xlarge` a `m5.2xlarge`). Además, también puede personalizar los volúmenes de Amazon EBS para aumentar el rendimiento y las IOPS. Por ejemplo, si utiliza volúmenes `gp2`, utilice volúmenes más grandes que ofrezcan más rendimiento de referencia. Si utiliza volúmenes `gp3`, especifique el rendimiento y las IOPS al crear el volumen.

Utilice el modo de red bridge para las tareas que se ejecutan en las instancias de Amazon EC2. Las tareas que utilizan el modo de red `bridge` en Amazon EC2 se inician más rápido que las tareas que utilizan el modo de red `awsvpc`. Cuando se utiliza el modo de red `awsvpc`, Amazon ECS conecta una interfaz de red elástica (ENI) a la instancia antes de lanzar la tarea. Esto introduce una latencia adicional. Sin embargo, el uso de redes bridge tiene varias desventajas. Estas tareas no tienen su propio grupo de seguridad y el equilibrio de carga tiene algunas implicaciones. Para obtener más información, consulte Load balancer target groups en la Guía del usuario de Elastic Load Balancing.

Gestión de crisis de demanda

Algunas aplicaciones experimentan grandes crisis de demanda repentinas. Esto ocurre por diversas razones: una noticia, una gran venta, un evento mediático o algún otro evento que se vuelva viral y provoque un aumento rápido y significativo del tráfico en muy poco tiempo. Si no se planifica, esto puede provocar que la demanda supere rápidamente los recursos disponibles.

La mejor manera de gestionar las crisis de demanda es anticiparse a ellas y planificar en consecuencia. Como el escalado automático puede llevar tiempo, le recomendamos que escale

horizontalmente la aplicación antes de que comience la crisis de demanda. Para obtener los mejores resultados, recomendamos tener un plan empresarial que implique una estrecha colaboración entre los equipos que utilicen un calendario compartido. El equipo que planifique el evento debe trabajar en estrecha colaboración con el equipo a cargo de la aplicación con antelación. Esto le da al equipo tiempo suficiente para tener un plan de programación claro. Pueden programar el escalado horizontal de la capacidad antes del evento y la reducción horizontal después del evento. Para obtener más información, consulte Escalado programado en la Guía del usuario de Auto Scaling de aplicaciones.

Si tiene el plan Enterprise Support, asegúrese de trabajar también con su administrador técnico de cuentas (TAM). El TAM puede verificar sus cuotas de servicio y asegurarse de que se aumenten las cuotas necesarias antes de que comience el evento. De esta forma, no alcanza accidentalmente ninguna cuota de servicio. También puede ayudarlo con servicios de precalentamiento, como los equilibradores de carga, para garantizar que el evento se desarrolle sin problemas.

Gestionar las crisis no programadas de demanda es un problema más difícil. Las crisis no programadas, si son lo suficientemente grandes en amplitud, pueden provocar rápidamente que la demanda supere a la capacidad. También puede superar la capacidad de reacción del escalado automático. La mejor manera de prepararse para las crisis no programadas es aprovisionar recursos en exceso. Debe disponer de recursos suficientes para gestionar la máxima demanda de tráfico prevista en cualquier momento.

Mantener la máxima capacidad para anticiparse a las crisis no programadas de demanda puede resultar costoso. Para mitigar el impacto en los costos, busque un indicador, métrica o evento principal que prediga la inminencia de una gran crisis de demanda. Si la métrica o el evento avisan de forma fiable y con suficiente antelación, comience el proceso de escalado horizontal inmediatamente cuando se produzca el evento o cuando la métrica supere el umbral específico que haya establecido.

Si su aplicación es propensa a sufrir crisis repentinas y no programadas de demanda, considere la posibilidad de agregar un modo de alto rendimiento a la aplicación que sacrifique la funcionalidad no crítica pero que conserve la funcionalidad crucial para un cliente. Por ejemplo, supongamos que su aplicación puede pasar de generar costosas respuestas personalizadas a ofrecer una página de respuestas estática. En este escenario, puede aumentar el rendimiento de manera significativa sin escalar la aplicación en absoluto.

Por último, puede considerar la posibilidad de separar los servicios monolíticos para hacer frente mejor a las crisis de demanda. Si su aplicación es un servicio monolítico cuya ejecución es costosa y su escalado es lento, es posible que pueda extraer o reescribir las partes fundamentales para el rendimiento y ejecutarlas como servicios independientes. De este modo, estos nuevos servicios

se pueden escalar de forma independiente de los componentes menos críticos. Disponer de la flexibilidad necesaria para escalar horizontalmente las funciones esenciales para el rendimiento de forma independiente de otras partes de la aplicación puede reducir el tiempo que se tarda en agregar capacidad y ayudar a ahorrar costos.

Capacidad de clústeres de Amazon ECS

Puede proporcionar capacidad a un clúster de Amazon ECS de varias maneras. Por ejemplo, puede lanzar instancias de Amazon EC2 y registrarlas en el clúster al inicio mediante el agente del contenedor de Amazon ECS. Sin embargo, este método puede resultar difícil porque debe administrar el escalado por su cuenta. Por lo tanto, se recomienda utilizar proveedores de capacidad de Amazon ECS. Los proveedores de capacidad administran el escalado de los recursos. Hay tres tipos de proveedores de capacidad: Amazon EC2, Fargate y Fargate Spot. Para más información sobre los proveedores de capacidad de Fargate, consulte Clústeres de Amazon ECS para cargas de trabajo del tipo de lanzamiento de Fargate y, para el tipo de lanzamiento de EC2, consulte Clústeres de Amazon ECS para cargas de trabajo del tipo de lanzamiento de EC2.

Los proveedores de capacidad de Fargate y Fargate Spot gestionan el ciclo de vida de las tareas de Fargate. Fargate ofrece capacidad bajo demanda y Fargate Spot proporciona capacidad de Spot. Cuando lanza una tarea, Amazon ECS le proporciona un recurso de Fargate. Este recurso de Fargate incluye las unidades de memoria y CPU que corresponden directamente a los límites de nivel de tarea que declaró en la definición de la tarea. Cada tarea recibe su propio recurso de Fargate, lo que establece una relación 1:1 entre la tarea y los recursos de computación.

Las tareas que se ejecutan en Fargate Spot están sujetas a interrupciones. Las interrupciones se producen tras una advertencia de dos minutos. Se producen durante los periodos de gran demanda. Fargate Spot funciona mejor para las cargas de trabajo tolerantes a las interrupciones, como los trabajos por lotes o los entornos de desarrollo o provisionales. También son adecuados para cualquier otro escenario en el que la alta disponibilidad y la baja latencia no sean un requisito.

Puede ejecutar tareas de Fargate Spot junto con las tareas de Fargate bajo demanda. Al utilizarlas juntas, dispondrá de capacidad de ampliación de aprovisionamiento a un costo menor.

Amazon ECS también puede administrar la capacidad de la instancia de Amazon EC2 para las tareas. Cada proveedor de capacidad de Amazon EC2 está asociado a un grupo de Amazon EC2 Auto Scaling especificado. Cuando utiliza el proveedor de capacidad de Amazon EC2, el escalado automático de clústeres mantiene el tamaño del grupo de Amazon EC2 Auto Scaling para garantizar que se puedan hacer todas las tareas programadas.

Elección de los tamaños de las tareas de Fargate para Amazon ECS

En el caso de las definiciones de tareas de AWS Fargate, debe especificar la CPU y la memoria de las tareas. No es necesario tener en cuenta la sobrecarga. También puede especificar la CPU y la memoria en el nivel de contenedor para tareas de Fargate. Sin embargo, esto no es obligatorio. Los límites de los recursos deben ser superiores o iguales a las reservas que haya declarado. En la mayoría de los casos, puede establecerlos como la suma de las reservas de cada uno de los contenedores declarados en la definición de la tarea. Luego, redondee el número al tamaño de Fargate más cercano. Para más información sobre los tamaños disponibles, consulte CPU y memoria de tareas.

Aceleración del aprovisionamiento de capacidad de clústeres de Amazon ECS con proveedores de capacidad en Amazon EC2

Los clientes que ejecutan Amazon ECS en Amazon EC2 pueden aprovechar el escalado automático de clústeres (CAS) de Amazon ECS para administrar el escalado de los grupos de Amazon EC2 Auto Scaling (ASG). Con CAS, puede configurar Amazon ECS para escalar el ASG automáticamente y centrarse únicamente en ejecutar las tareas. Amazon ECS garantizará que el ASG se reduzca o escale horizontalmente según sea necesario sin necesidad de ninguna otra intervención. Los proveedores de capacidad de Amazon ECS están acostumbrados a administrar la infraestructura del clúster al garantizar que haya instancias de contenedor suficientes para satisfacer las exigencias de la aplicación. Para obtener información sobre cómo funciona el CAS de Amazon ECS, consulte Deep Dive on Amazon ECS Cluster Auto Scaling.

Dado que CAS depende de una integración basada en CloudWatch con ASG para ajustar la capacidad de clústeres, tiene una latencia inherente asociada a la publicación de las métricas de CloudWatch, el tiempo que tarda la métrica `CapacityProviderReservation` en infringir las alarmas de CloudWatch (altas y bajas) y el tiempo que tarda una instancia de Amazon EC2 recién lanzada en prepararse. Puede hacer lo siguiente para que el CAS responda mejor y, de este modo, las implementaciones sean más rápidas:

Tamaños de escalado por pasos del proveedor de capacidad

Con el tiempo, los proveedores de capacidad de Amazon ECS aumentarán o reducirán las instancias de contenedor para satisfacer las demandas de su aplicación. La cantidad mínima de instancias que Amazon ECS lanzará se establece en 1 de forma predeterminada. Esto puede agregar tiempo adicional a las implementaciones si se necesitan varias instancias para realizar las tareas pendientes. Puede aumentar el valor de `minimumScalingStepSize` mediante la API de

Amazon ECS para aumentar el número mínimo de instancias que Amazon ECS reduce o escala horizontalmente cada vez. Un valor de `maximumScalingStepSize` demasiado bajo puede limitar el número de instancias de contenedor que se escalan o reducen horizontalmente cada vez, lo que puede ralentizar las implementaciones.

> **Note**
>
> Actualmente, esta configuración solo está disponible mediante las API `CreateCapacityProvider` o `UpdateCapacityProvider`.

Periodo de preparación de instancias

El periodo de preparación de instancias es el periodo de tiempo después del que una instancia de Amazon EC2 recién lanzada puede contribuir a las métricas de CloudWatch para el grupo de escalado automático. Después de que finaliza el periodo de preparación especificado, la instancia se contabiliza para las métricas agregadas del ASG y el CAS continúa con su siguiente iteración de cálculos para estimar la cantidad de instancias necesarias.

El valor predeterminado de `instanceWarmupPeriod` es de 300 segundos, que puede configurar con un valor inferior mediante las API `CreateCapacityProvider` o `UpdateCapacityProvider` para lograr un escalado con mayor capacidad de respuesta.

Capacidad sobrante

Si su proveedor de capacidad no tiene instancias de contenedor disponibles para colocar tareas, debe aumentar (escalar horizontalmente) la capacidad del clúster mediante el lanzamiento de instancias de Amazon EC2 sobre la marcha y esperar a que se inicien antes de poder lanzar contenedores en ellas. Esto puede reducir considerablemente la frecuencia de lanzamiento de las tareas. Dispone de dos opciones aquí.

En este caso, disponer de capacidad sobrante de Amazon EC2 ya lanzada y lista para ejecutar tareas aumentará la frecuencia efectiva de lanzamiento de tareas. Puede usar la configuración `Target Capacity` para indicar que desea mantener la capacidad sobrante en sus clústeres. Por ejemplo, si se establece el valor de `Target Capacity` en un 80 %, indica que el clúster necesita un 20 % de capacidad sobrante en todo momento. Esta capacidad sobrante puede permitir que cualquier tarea independiente se inicie inmediatamente, lo que garantiza que el lanzamiento de las tareas no se limite. La desventaja de este enfoque es el posible aumento de los costos derivados de mantener la capacidad sobrante de los clústeres.

Un enfoque alternativo que puede considerar es agregar margen de maniobra a su servicio, no al proveedor de capacidad. Esto significa que, en lugar de reducir la configuración `Target Capacity` para lanzar la capacidad sobrante, puede aumentar la cantidad de réplicas de su servicio al modificar la métrica de escalado de seguimiento de destino o los umbrales de escalado por pasos del escalado automático del servicio. Tenga en cuenta que este enfoque solo será útil para cargas de trabajo con picos de actividad, pero no tendrá ningún efecto cuando implemente nuevos servicios y pase de 0 a N tareas por primera vez. Para más información sobre las políticas de escalado relacionadas, consulte Políticas de escalado de seguimiento de destino o Políticas de escalado por pasos

Acceso a las características de Amazon ECS con la configuración de la cuenta

Puede acceder a la configuración de cuenta de Amazon ECS para optar por incluir o no características específicas. En cada Región de AWS, puede optar por incluir o no cada configuración de cuenta en el nivel de cuenta o para un usuario o rol específico.

Puede optar por incluir o no características específicas si alguno de los siguientes aspectos es relevante para usted:

- Un usuario o rol puede optar por incluir o no ajustes de cuenta específicos en su cuenta individual.
- Un usuario o rol puede establecer la configuración predeterminada de características incluidas o no para todos los usuarios de la cuenta.
- El usuario raíz o un usuario con privilegios de administrador puede optar por incluir o no cualquier usuario o rol específico de la cuenta. Si la configuración de cuenta del usuario raíz cambia, se establece el valor predeterminado para todos los usuarios o roles para los que no se hubiera establecido una configuración de cuenta individual.

> **Note**
>
> Los usuarios federados adoptan la configuración de cuenta del usuario raíz y no se puede establecer una configuración de cuenta explícita para ellos de manera individual.

Las configuraciones de cuenta disponibles son las que se indican a continuación. Debe optar por incluir o excluir cada configuración de la cuenta por separado.

Nombre del recurso	Más información
`containerInsights`	Información de contenedores

`serviceLongArnFormat`
 	
`taskLongArnFormat`
 	
`containerInstanceLongArnFormat`
	Nombres de recursos de Amazon (ARN) e ID

`tagResourceAuthorization`	Autorización de etiquetado
`fargateFIPSMode`	Conformidad de AWS Fargate con el Estándar Federal de Procesamiento de la Informaci ón(FIPS-140)
`fargateTaskRetirementWaitPeriod`	Tiempo de espera para el retiro de tareas de AWS Fargate
`guardDutyActivate`	Supervisión en tiempo de ejecución (integrac ión con Amazon GuardDuty)
`dualStackIPv6`	VPC IPv6 de doble pila
`awsvpcTrunking`	Aumento de las interfaces de red de instancias de contenedor de Linux
`defaultLogDriverMode`	Modo de controlador de registro predeterm inado

Nombres de recursos de Amazon (ARN) e ID

Cuando se crean recursos de Amazon ECS, a cada recurso se le asigna un nombre de recurso de Amazon (ARN) y un identificador de recurso (ID) únicos. Si utiliza herramientas de línea de comandos o la API de Amazon ECS para trabajar con Amazon ECS, se requieren ARN o ID de recurso para determinados comandos. Por ejemplo, si utiliza el comando stop-task AWS CLI para detener una tarea, debe especificar el ID o ARN en el comando.

Amazon ECS ha incorporado un nuevo formato para los nombres de recurso de Amazon (ARN) y los ID de recurso para instancias de contenedor, tarea y servicio de Amazon ECS. El estado de opción incluida para cada tipo de recurso determina el formato de Nombre de recurso de Amazon (ARN) que utiliza el recurso. Debe optar por incluir el nuevo formato de ARN para utilizar características como el etiquetado de recursos para ese tipo de recurso.

Puede optar por incluir o no el nuevo formato de nombre de recurso de Amazon (ARN) e ID de recurso a cada región. En la actualidad, las cuentas nuevas que se crean tienen esta opción incluida de forma predeterminada.

Puede optar por inscribirse o no en el nuevo formato de nombre de recurso de Amazon (ARN) e ID de recursos en cualquier momento. Una vez que opte por incluirlo, los nuevos recursos que cree usarán el nuevo formato.

> **Note**
>
> Un ID de recurso no cambia después de haberlo creado. Por lo tanto, la posibilidad de optar por incluir o no el nuevo formato no afecta a los ID de recursos existentes.

En las siguientes secciones se describe cómo están cambiando los formatos de ID de recurso y ARN. Para obtener más información acerca de la transición a los nuevos formatos, consulte Preguntas frecuentes acerca de Amazon Elastic Container Service.

Formato de nombre de recurso de Amazon (ARN)

Algunos de los recursos tienen un nombre fácil de recordar como, por ejemplo, un servicio denominado `production`. En otros casos, debe especificar un recurso utilizando el formato de nombre de recurso de Amazon (ARN). El nuevo formato de ARN para tareas, servicios e instancias de contenedor de Amazon ECS incluye el nombre del clúster. Para obtener más información acerca de cómo optar por incluir el nuevo formato de ARN, consulte Modificación de la configuración de la cuenta de Amazon ECS.

En la siguiente tabla se muestra el formato actual y el nuevo formato para cada tipo de recurso.

Tipo de recurso	ARN
Instancia de contenedor	Actual: `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:container-instance/` *`container-instance-id`*

Tipo de recurso	ARN
	Nuevo: `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:container-instance/` *`cluster-name`* `/`*`container-instance-id`*
Servicio de Amazon ECS	Actual: `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:service/` *`service-name`* Nuevo: `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:service/` *`cluster-name`* `/`*`service-name`*
Tarea de Amazon ECS	Actual: `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:task/`*`task-id`* Nuevo: `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:task/`*`cluster-name`* `/`*`task-id`*

Longitud del ID del recurso

Un ID de recursos adopta la forma de una combinación única de letras y números. Los nuevos formatos de ID de recurso incluyen ID más cortos para tareas e instancias de contenedor de Amazon ECS. El formato de ID de recurso actual tiene 36 caracteres. Los nuevos ID tienen un formato de 32 caracteres y no incluyen guiones. Para obtener más información acerca de cómo optar por incluir el nuevo formato de ID de recurso, consulte Modificación de la configuración de la cuenta de Amazon ECS.

El valor predeterminado es `enabled`.

Solo los recursos que se lancen después de optar por incluirlo recibirán el nuevo formato de ARN e ID de recurso. No se ven afectados todos los recursos existentes. Para hacer la transición de las tareas y los servicios de Amazon ECS a los nuevos formatos de ARN e ID de recurso, se debe volver a crear el servicio o la tarea. Para realizar la transición de una instancia de contenedor al nuevo formato de ARN e ID de recurso, la instancia de contenedor se debe vaciar y se debe lanzar y registrar una nueva en el clúster.

> **Note**
>
> Las tareas lanzadas por un servicio de Amazon ECS solo pueden recibir el nuevo formato de ARN e ID de recurso si el servicio se creó el 16 de noviembre de 2018 o con posterioridad y el usuario que creó el servicio optó por incluir el nuevo formato de las tareas.

Escala de tiempo del formato de ARN e ID de recurso

La escala de tiempo para los periodos en los que se puede optar por incluir o no el nuevo formato de nombre de recurso de Amazon (ARN) y de ID de recursos de Amazon ECS finalizó el 1 de abril de 2021. De forma predeterminada, todas las cuentas nuevas optan por incluir el nuevo formato. Todos los recursos nuevos que se creen recibirán el nuevo formato y ya no puede optar por excluirlo.

Información de contenedores

El 2 de diciembre de 2024, AWS publicó Información de contenedores con observabilidad mejorada para Amazon ECS. Esta versión admite la observabilidad mejorada de los clústeres de Amazon ECS que utilizan los tipos de lanzamiento de Amazon EC2 y Fargate. Después de configurar Información de contenedores con capacidad de observabilidad mejorada en Amazon ECS, Información de contenedores recopilará automáticamente telemetría de infraestructura detallada desde el nivel de clúster hasta el nivel de contenedor en el entorno, y mostrará los datos en paneles con diversas métricas y dimensiones. A continuación, podrá utilizar estos paneles listos para usar en la consola de Información de contenedores y comprender mejor el estado y el rendimiento de los contenedores, así como mitigar los problemas con mayor rapidez mediante la identificación de anomalías.

Recomendamos que utilice Información de contenedores con capacidad de observabilidad mejorada en lugar de Información de contenedores, ya que proporciona una visibilidad detallada del entorno de contenedores y reduce el tiempo medio de resolución. Para obtener más información, consulte Métricas de Información de contenedores con observabilidad mejorada de Amazon ECS en la Guía del usuario de Amazon CloudWatch.

El valor predeterminado del ajuste de cuenta `containerInsights` es `disabled`.

Información de contenedores con observabilidad mejorada

Utilice el siguiente comando para activar Información de contenedores con observabilidad mejorada.

Establezca el ajuste de cuenta `containerInsights` en `enhanced`.

```
aws ecs put-account-setting --name containerInsights --value enhanced
```

Ejemplo de resultado

```
{
    "setting": {
        "name": "containerInsights",
        "value": "enhanced",
        "principalArn": "arn:aws:iam::123456789012:johndoe",
         "type": user
    }
}
```

Tras establecer esta configuración de la cuenta, todos los clústeres nuevos utilizan automáticamente Información de contenedores con observabilidad mejorada. Utilice el comando `update-cluster-settings` para agregar Información de contenedores con capacidad de observabilidad mejorada a un clúster existente, o para actualizar un clúster de Información de contenedores a Información de contenedores con observabilidad mejorada.

```
aws ecs update-cluster-settings --cluster cluster-name --settings
 name=containerInsights,value=enhanced
```

También puede utilizar la consola para configurar Información de contenedores con observabilidad mejorada. Para obtener más información, consulte Modificación de la configuración de la cuenta de Amazon ECS.

Información de contenedores

Al establecer la configuración de la cuenta `containerInsights` en `enabled`, todos los clústeres nuevos tienen habilitado Información de contenedores de forma predeterminada. Puede modificar los clústeres existentes mediante `update-cluster-settings`.

Para usar Información de contenedores, establezca la configuración de la cuenta `containerInsights` en `enabled`. Use el siguiente comando para activar Información de contenedores.

```
aws ecs put-account-setting --name containerInsights --value enabled
```

Ejemplo de resultado

```
{
    "setting": {
        "name": "containerInsights",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:johndoe",
         "type": user
    }
}
```

Al establecer la configuración de la cuenta `containerInsights` en `enabled`, todos los clústeres nuevos tienen habilitado Información de contenedores de forma predeterminada. Usa el comando `update-cluster-settings` para agregar Información de contenedores a un clúster existente.

```
aws ecs update-cluster-settings --cluster cluster-name --settings
 name=containerInsights,value=enabled
```

También puede utilizar la consola para configurar Información de contenedores. Para obtener más información, consulte Modificación de la configuración de la cuenta de Amazon ECS.

Conformidad de AWS Fargate con el Estándar Federal de Procesamiento de la Información(FIPS-140)

Fargate presta conformidad con el Federal Information Processing Standard (FIPS, Estándar Federal de Procesamiento de Información) 140, que especifica los requisitos de seguridad para los módulos criptográficos que protegen información confidencial. Es la norma gubernamental actual de los Estados Unidos y Canadá y se aplica a los sistemas que deben cumplir con la Federal Information Security Management Act (FISMA, Ley Federal de Gestión de la Seguridad de la Información) o el Federal Risk and Authorization Management Program (FedRAMP, Programa Federal de Gestión de Riesgos y Autorizaciones).

El nombre del recurso es `fargateFIPSMode`.

El valor predeterminado es `disabled`.

Debe activar la conformidad de Fargate con el Estándar Federal de Procesamiento de Información (FIPS-140). Para obtener más información, consulte the section called “AWS Fargate Conformidad con la norma FIPS-140”.

> **Important**
>
> La configuración de cuenta `fargateFIPSMode` solo se puede cambiar a través de la API de Amazon ECS o la AWS CLI. Para obtener más información, consulte Modificación de la configuración de la cuenta de Amazon ECS.

Ejecute `put-account-setting-default` con la opción `fargateFIPSMode` establecida en `enabled`. Para obtener más información, consulte put-account-setting-default en la Referencia de la API de Amazon Elastic Container Service.

- Puede utilizar el siguiente comando para activar la conformidad con la norma FIPS-140.

```
aws ecs put-account-setting-default --name fargateFIPSMode --value enabled
```

Ejemplo de resultado

```
{
    "setting": {
        "name": "fargateFIPSMode",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:root",
         "type": user
    }
}
```

Puede ejecutar `list-account-settings` para ver el estado actual de conformidad con la norma FIPS-140. Utilice la opción `effective-settings` para ver la configuración a nivel de cuenta.

```
aws ecs list-account-settings --effective-settings
```

Autorización de etiquetado

Amazon ECS presenta la autorización de etiquetado para la creación de recursos. Los usuarios deben tener permisos de etiquetado para llevar a cabo las acciones que crean el recurso, como `ecsCreateCluster`. Al crear un recurso y especificar etiquetas para ese recurso, AWS realiza una autorización adicional para comprobar que hay permisos para crear etiquetas. Por lo tanto, usted

debe conceder permisos explícitos para utilizar la acción `ecs:TagResource`. Para obtener más información, consulte the section called “Recursos de etiquetas durante la creación”.

Para optar por la autorización de etiquetado, ejecute `put-account-setting-default` con la opción `tagResourceAuthorization` establecida en `enable`. Para obtener más información, consulte put-account-setting-default en la Referencia de la API de Amazon Elastic Container Service. Puede ejecutar `list-account-settings` para ver el estado actual de la autorización de etiquetado.

- Puede utilizar el siguiente comando para activar la autorización de etiquetado.

```
aws ecs put-account-setting-default --name tagResourceAuthorization --value on --
region region
```

Ejemplo de resultado

```
{
    "setting": {
        "name": "tagResourceAuthorization",
        "value": "on",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type": user
    }
}
```

Después de activar la autorización de etiquetado, debe configurar los permisos correspondientes para que los usuarios puedan etiquetar recursos en el momento de su creación. Para obtener más información, consulte the section called “Recursos de etiquetas durante la creación”.

Puede ejecutar `list-account-settings` para ver el estado actual de la autorización de etiquetado. Utilice la opción `effective-settings` para ver la configuración a nivel de cuenta.

```
aws ecs list-account-settings --effective-settings
```

Plazos de la autorización de etiquetado

Puede confirmar si la autorización de etiquetado está activa ejecutando `list-account-settings` para ver el valor `tagResourceAuthorization`. Cuando el valor sea on, quiere decir que la

autorización de etiquetado está en uso. Para obtener más información, consulte list-account-setting-default en la Referencia de la API de Amazon Elastic Container Service.

A continuación, se incluyen las fechas importantes relacionadas con la autorización de etiquetado.

- 18 de abril de 2023: se presenta la autorización de etiquetado. Todas las cuentas nuevas y existentes deben optar por el uso de la característica. Puede optar por comenzar a usar la autorización de etiquetado. Al optar por la opción, debe conceder los permisos correspondientes.
- Del 9 de febrero de 2024 al 6 de marzo de 2024: todas las cuentas nuevas y las cuentas existentes no afectadas tienen activada la autorización de etiquetado de forma predeterminada. Puede habilitar o deshabilitar la configuración de `tagResourceAuthorization` de la cuenta para verificar su política de IAM.

  AWS lo ha notificado a las cuentas afectadas.

  Para desactivar la característica, ejecute `put-account-setting-default` con la opción `tagResourceAuthorization` configurada en `off`.
- 7 de marzo de 2024: si ha activado la autorización de etiquetado, ya no puede desactivar la configuración de la cuenta.

  Le recomendamos que lleve a cabo las pruebas de la política de IAM antes de esta fecha.
- 29 de marzo de 2024: todas las cuentas utilizan la autorización de etiquetado. La configuración de cuenta ya no está disponible en la consola o AWS CLI de Amazon ECS.

Tiempo de espera para el retiro de tareas de AWS Fargate

AWS envía notificaciones cuando tiene tareas de Fargate ejecutándose en una revisión de la versión de la plataforma marcada como retiro. Para obtener más información, consulte Retirada y mantenimiento de tareas para AWS Fargate en Amazon ECS .

AWS es responsable de aplicar parches y mantener la infraestructura subyacente de AWS Fargate. Cuando AWS determina que se necesita una actualización de seguridad o de infraestructura para una tarea de Amazon ECS alojada en Fargate, se deben detener las tareas y lanzar nuevas tareas para sustituirlas. Puede configurar el período de espera antes de que se retiren las tareas para aplicar parches. Tiene la opción de retirar la tarea inmediatamente, esperar 7 días naturales o esperar 14 días naturales.

Esta configuración se hace en el nivel de la cuenta.

Puede configurar la hora a la que Fargate inicia el retiro de las tareas. Para las cargas de trabajo que requieren la aplicación inmediata de las actualizaciones, elija la configuración inmediata (`0`). Cuando necesite más control, por ejemplo, cuando una tarea solo se pueda detener durante un período determinado, configure la opción de 7 días (`7`) o 14 días (`14`).

Le recomendamos que elija un período de espera más corto para poder seleccionar antes las revisiones de las versiones más recientes de la plataforma.

Para configurar el periodo de espera, ejecute `put-account-setting-default` o `put-account-setting` como usuario raíz o como usuario administrativo. Utilice la opción `fargateTaskRetirementWaitPeriod` para el conjunto de opciones `name` y `value` para uno de los valores siguientes:

- `0` - AWS envía la notificación e inmediatamente comienza a retirar las tareas afectadas.
- `7` - AWS envía la notificación y espera 7 días calendario antes de empezar a retirar las tareas afectadas.
- `14` - AWS envía la notificación y espera 14 días calendario antes de empezar a retirar las tareas afectadas.

El valor predeterminado es 7 días.

Para obtener más información, consulte put-account-setting-default y put-account-setting en la Referencia de la API de Amazon Elastic Container Service.

Puede ejecutar el siguiente comando para establecer el periodo de espera en 14 días.

```
aws ecs put-account-setting-default --name fargateTaskRetirementWaitPeriod --value 14
```

Ejemplo de resultado

```
{
    "setting": {
        "name": "fargateTaskRetirementWaitPeriod",
        "value": "14",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type: user"
    }
}
```

Puede ejecutar `list-account-settings` para ver el tiempo de espera actual para el retiro de tareas de Fargate. Use la opción `effective-settings`.

```
aws ecs list-account-settings --effective-settings
```

Aumento de las interfaces de red de instancias de contenedor de Linux

Cada tarea de Amazon ECS que utiliza el modo de red `awsvpc` recibe su propia interfaz de red elástica (ENI), la cual se asocia a la instancia de contenedor que la hospeda. Existe un límite predeterminado en cuanto al número de interfaces de red que pueden asociarse a una instancia de Amazon EC2, y la interfaz de red principal cuenta como una. Por ejemplo, de forma predeterminada una instancia de `c5.large` puede tener asociadas hasta tres ENI. La interfaz de red principal para la instancia cuenta como una, por lo que puede asociar a la instancia dos ENI adicionales. Dado que cada tarea que utiliza el modo de red `awsvpc` requiere una ENI, normalmente solo puede ejecutar dos de esas tareas en este tipo de instancia.

Amazon ECS es compatible con el inicio de instancias de contenedor con mayor densidad de ENI al usar tipos de instancias de Amazon EC2 compatibles. Cuando se utilizan estos tipos de instancias y se activa la configuración de cuenta `awsvpcTrunking`, aparecen ENI adicionales disponibles en las instancias de contenedor recién lanzadas. Esta configuración le permite colocar más tareas en cada instancia de contenedor.

Por ejemplo, una instancia `c5.large` con `awsvpcTrunking` tiene un límite de ENI aumentado de doce. La instancia de contenedor tendrá la interfaz de red principal, y Amazon ECS crea y asocia una interfaz de red “troncal” a la instancia de contenedor. Por lo tanto, esta configuración le permite lanzar diez tareas en la instancia de contenedor, en lugar de las dos tareas actuales.

Supervisión en tiempo de ejecución (integración con Amazon GuardDuty)

Supervisión en tiempo de ejecución es un servicio inteligente de detección de amenazas que protege las cargas de trabajo que se ejecutan en las instancias de contenedor de Fargate y EC2 mediante la supervisión continua de la actividad de registro y red de AWS para identificar comportamientos malintencionados o no autorizados.

El parámetro `guardDutyActivate` es solo de lectura en Amazon ECS e indica si el administrador de seguridad de su cuenta de Amazon ECS ha activado o desactivado la supervisión en tiempo de ejecución. GuardDuty controla esta configuración de la cuenta en su nombre. Para obtener más información, consulte Protecting Amazon ECS workloads with Runtime Monitoring.

Puede ejecutar `list-account-settings` para ver la configuración de integración actual de GuardDuty.

```
aws ecs list-account-settings
```

Ejemplo de resultado

```
{
    "setting": {
        "name": "guardDutyActivate",
        "value": "on",
        "principalArn": "arn:aws:iam::123456789012:doej",
        "type": aws-managed"
    }
}
```

VPC IPv6 de doble pila

Amazon ECS admite proporcionar tareas con una dirección IPv6 además de la dirección IPv4 privada principal.

Para que las tareas reciban una dirección IPv6, la tarea debe utilizar el modo de red `awsvpc`, debe lanzarse en una VPC configurada para el modo de doble pila, y la configuración de cuenta `dualStackIPv6` debe estar habilitada. Para obtener más información sobre otros requisitos, consulte Utilización de una VPC en modo de pila doble para ver el tipo de lanzamiento de EC2 y Utilización de una VPC en modo de pila doble para ver el tipo de lanzamiento de Fargate.

> **Important**
>
> La configuración de cuenta `dualStackIPv6` solo se puede cambiar a través de la API de Amazon ECS o la AWS CLI. Para obtener más información, consulte Modificación de la configuración de la cuenta de Amazon ECS.

Si tenía una tarea en ejecución con el modo de red `awsvpc` en una subred habilitada para IPv6 entre el 1.° de octubre de 2020 y el 2 de noviembre de 2020, la configuración de cuenta `dualStackIPv6` predeterminada en la región en la que se estaba ejecutando la tarea es `disabled`. Si esa condición no se cumple, la configuración `dualStackIPv6` predeterminada en la región es `enabled`.

El valor predeterminado es `disabled`.

Modo de controlador de registro predeterminado

Amazon ECS admite la configuración de un modo de entrega de mensajes de registro predeterminado de un contenedor a un controlador de registro elegido. El modo de entrega afecta la estabilidad de las aplicaciones cuando se interrumpe el flujo de registros del contenedor al controlador de registro.

La configuración de `defaultLogDriverMode` admite dos valores: `blocking` y `non-blocking`. Para obtener más información acerca de estos modos de entrega, consulte LogConfiguration en la Referencia de la API de Amazon Elastic Container Service.

Si no especifica un modo de entrega en `logConfiguration` de la definición de su contenedor, se usará como predeterminado el modo que especifique en la configuración de esta cuenta.

El modo de entrega predeterminado es `non-blocking`.

> **Note**
>
> El 25 de junio de 2025, Amazon ECS cambió el modo de controlador de registro predeterminado de `blocking` a `non-blocking` a fin de priorizar la disponibilidad de las tareas sobre el registro. Para seguir utilizando el modo `blocking` después de este cambio, realice una de las siguientes acciones:
>
> - Establezca la opción `mode` en la `logConfiguration` de su definición de contenedor como `blocking`.
> - Establezca el ajuste de cuenta `defaultLogDriverMode` en `blocking`.

Para configurar un modo de controlador de registro predeterminado en `blocking`, puede ejecutar el siguiente comando:

```
aws ecs put-account-setting-default --name defaultLogDriverMode --value "blocking"
```

Visualización de la configuración de la cuenta de Amazon ECS mediante la consola

Consulte la configuración de la consola.

> **Important**
>
> Las configuraciones de cuenta `dualStackIPv6`, `fargateFIPSMode` y `fargateTaskRetirementWaitPeriod` solo se puede consultar o modificar a través de la AWS CLI.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación de la parte superior, seleccione la región para la que desea consultar la configuración de cuenta.
3. En la página de navegación, elija Account Settings (Configuración de cuenta).

Modificación de la configuración de la cuenta de Amazon ECS

Modifique la configuración de su cuenta para acceder a las funciones de Amazon ECS.

El parámetro `guardDutyActivate` es solo de lectura en Amazon ECS e indica si el administrador de seguridad de su cuenta de Amazon ECS ha activado o desactivado la supervisión en tiempo de ejecución. GuardDuty controla esta configuración de la cuenta en su nombre. Para obtener más información, consulte Protecting Amazon ECS workloads with Runtime Monitoring.

> **Important**
>
> Las configuraciones de cuenta `dualStackIPv6`, `fargateFIPSMode` y `fargateTaskRetirementWaitPeriod` solo se puede consultar o modificar a través de la AWS CLI.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación de la parte superior, seleccione la región para la que desea consultar la configuración de cuenta.
3. En la página de navegación, elija Account Settings (Configuración de cuenta).
4. Elija Actualizar.
5. (Opcional) Para aumentar o disminuir la cantidad de tareas que puede ejecutar en el modo de red awsvpc para cada instancia de EC2, en Enlace troncal AWSVPC, seleccione Enlace troncal AWSVPC.

6. (Opcional) Para utilizar o dejar de utilizar Información de contenedores de CloudWatch de forma predeterminada para los clústeres, en observabilidad de Información de contenedores de CloudWatch, elija una de las siguientes opciones:
   - Para usar Información de contenedores con observabilidad mejorada, elija Información de contenedores con observabilidad mejorada.
   - Para usar Información de contenedores, seleccione Información de contenedores.
   - Para dejar de utilizar Información de contenedores, elija Desactivado.
7. (Opcional) Para activar o desactivar la autorización de etiquetado, en Autorización de etiquetado de recursos, seleccione o borre Autorización de etiquetado de recursos.
8. (Opcional) Para configurar un modo de controlador de registro predeterminado para cuando un modo de entrega de registros no esté definido en un `logConfiguration` de contenedor, en Modo de controlador de registro predeterminado, elija una de las siguientes opciones:
   - Para establecer el modo de controlador de registro predeterminado como `blocking`, seleccione Bloqueo.
   - Para establecer el modo de controlador de registro predeterminado como `non-blocking`, seleccione Sin bloqueo.
9. Seleccione Save changes (Guardar cambios).
10. En la pantalla de confirmación, seleccione Confirm (Confirmar) para guardar la selección.

Pasos a seguir a continuación

Si activó Información de contenedores con observabilidad mejorada o Información de contenedores, puede optar por actualizar los clústeres existentes de modo que utilicen la característica. Para obtener más información, consulte Actualización de un clúster de Amazon ECS.

Revertir a las configuraciones de cuenta predeterminadas de Amazon ECS

Puede utilizar la AWS Management Console para revertir a las configuraciones de cuenta predeterminadas de Amazon ECS.

La opción Revert to account default (Revertir a los valores predeterminados de la cuenta) solo está disponible cuando la configuración de cuenta ya no es la configuración predeterminada.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.

2. En la barra de navegación de la parte superior, seleccione la región para la que desea consultar la configuración de cuenta.
3. En la página de navegación, elija Account Settings (Configuración de cuenta).
4. Elija Actualizar.
5. Elija Revert to account default (Revertir a los valores predeterminados de la cuenta).
6. En la pantalla de confirmación, seleccione Confirm (Confirmar) para guardar la selección.

Administración de la configuración de la cuenta de Amazon ECS mediante la AWS CLI

Para gestionar la configuración de la cuenta, puede utilizar la API de Amazon ECS, la AWS CLI o los SDK. Las configuraciones de cuenta `dualStackIPv6`, `fargateFIPSMode` y `fargateTaskRetirementWaitPeriod` solo se pueden consultar o modificar con esas herramientas.

> **Note**
>
> Puede utilizar puntos de conexión de servicio de doble pila para interactuar con Amazon ECS desde la AWS CLI, los SDK y la API de Amazon ECS a través de IPv4 e IPv6. Para obtener más información, consulte Uso de puntos de conexión de doble pila en Amazon ECS.

Para obtener información sobre las acciones de API disponibles para las definiciones de tareas, consulte Acciones de configuración de cuentas en la Referencia de la API de Amazon Elastic Container Service.

Utilice uno de los siguientes comandos para modificar la configuración de la cuenta predeterminada para todos los usuarios o roles de su cuenta. Estos cambios se aplican a toda la cuenta de AWS, a menos que un usuario o rol anule explícitamente esta configuración para sí mismo.

- put-account-setting-default (AWS CLI)

  ```
  aws ecs put-account-setting-default --name serviceLongArnFormat --value enabled --
  region us-east-2
  ```

  También puede usar este comando para modificar otras configuraciones de cuenta. Para ello, reemplace el parámetro `name` por la configuración de cuenta correspondiente.

- Write-ECSAccountSetting (AWS Tools for Windows PowerShell)

```
Write-ECSAccountSettingDefault -Name serviceLongArnFormat -Value enabled -Region us-
east-1 -Force
```

Para modificar la configuración de cuenta para la cuenta de usuario de IAM (AWS CLI)

Utilice alguno de los comandos siguientes para modificar la configuración de la cuenta para su usuario de . Si utiliza estos comandos como usuario raíz, los cambios se aplican a toda la cuenta de AWS, salvo que un usuario o rol anule explícitamente esta configuración para sí mismo.

- put-account-setting (AWS CLI)

```
aws ecs put-account-setting --name serviceLongArnFormat --value enabled --region us-
east-1
```

  También puede usar este comando para modificar otras configuraciones de cuenta. Para ello, reemplace el parámetro `name` por la configuración de cuenta correspondiente.

- Write-ECSAccountSetting (AWS Tools for Windows PowerShell)

```
Write-ECSAccountSetting -Name serviceLongArnFormat -Value enabled -Force
```

Para modificar la configuración de cuenta de un usuario o rol específico (AWS CLI)

Utilice uno de los comandos siguientes y especifique el ARN de un usuario, un rol o el usuario raíz que figura en la solicitud para modificar la configuración de un usuario o rol específico.

- put-account-setting (AWS CLI)

```
aws ecs put-account-setting --name serviceLongArnFormat --value enabled --principal-
arn arn:aws:iam::aws_account_id:user/principalName --region us-east-1
```

  También puede usar este comando para modificar otras configuraciones de cuenta. Para ello, reemplace el parámetro `name` por la configuración de cuenta correspondiente.

- Write-ECSAccountSetting (AWS Tools for Windows PowerShell)

```
Write-ECSAccountSetting -Name serviceLongArnFormat -Value enabled -PrincipalArn
 arn:aws:iam::aws_account_id:user/principalName -Region us-east-1 -Force
```

Roles de IAM para Amazon ECS

Un rol de IAM es una identidad de IAM que puede crear en su cuenta y que tiene permisos específicos. En Amazon ECS, puede crear roles para conceder permisos a los recursos de Amazon ECS, como contenedores o servicios.

Los roles que Amazon ECS necesita dependen del tipo de lanzamiento de la definición de la tarea y de las características que utilice. Utilice la siguiente tabla para determinar qué roles de IAM necesita para Amazon ECS.

Rol	Definición	Cuando sea necesario	Más información
Rol de ejecución de tareas	Este rol permite que Amazon ECS utilice otros servicios de AWS en su nombre.	La tarea está alojada en AWS Fargate o en instancias externas y: • extrae una imagen de contenedor de un repositorio privado de Amazon ECR. • extrae una imagen de contenedor de un repositorio privado de Amazon ECR en una cuenta diferente de la cuenta que ejecuta la tarea. • envía registros de contenedor a CloudWatch Logs	Rol de IAM de ejecución de tareas de Amazon ECS

Rol	Definición	Cuando sea necesario	Más información
		con el controlador de registros `awslogs`. La tarea está alojada en AWS Fargate o en instancias de Amazon EC2 y: • utiliza la autenticación de registros privados. • utiliza la supervisión en tiempo de ejecución. • la definición de la tarea hace referencia a información confidencial mediante secretos de Secrets Manager o parámetros del Almacén de parámetros de AWS Systems Manager.	

Rol	Definición	Cuando sea necesario	Más información
Rol de la tarea	Este rol permite que el código de la aplicación (en el contenedor) utilice otros servicios de AWS.	La aplicación obtiene acceso a otros servicios de AWS, como Amazon S3.	Rol de IAM de tarea de Amazon ECS
Rol de la instancia de contenedor	Este rol permite que sus instancias de EC2 o instancias externas se registren en el clúster.	La tarea está alojada en instancias de Amazon EC2 o en una instancia externa.	Rol de IAM de instancia de contenedor de Amazon ECS
Rol de Amazon ECS Anywhere	Este rol permite que sus instancias externas accedan a las API de AWS.	La tarea está alojada en instancias externas.	Rol de IAM de Amazon ECS Anywhere
Infraestructura de Amazon ECS para el rol de equilibradores de carga	Este rol permite a Amazon ECS administrar los recursos del equilibrador de carga en los clústeres en su nombre para las implementaciones azul/verde.	Quiere utilizar las implementaciones azul/verde de Amazon ECS.	Rol de IAM de infraestructura de Amazon ECS para los equilibradores de carga
Rol de CodeDeploy de Amazon ECS	Este rol permite a CodeDeploy llevar a cabo actualizaciones en sus servicios.	Utilice el tipo de implementación azul/verde de CodeDeploy para implementar servicios.	Rol de IAM de CodeDeploy de Amazon ECS

Rol	Definición	Cuando sea necesario	Más información
Rol de EventBridge de Amazon ECS	Este rol permite a EventBridge llevar a cabo actualizaciones en sus servicios.	Utilice las reglas y los objetivos de EventBridge para programar las tareas.	Rol de IAM de EventBridge de Amazon ECS
Rol de infraestructura de Amazon ECS	Este rol permite a Amazon ECS administrar los recursos de infraestructura de sus clústeres.	• Adjunte volúmenes de Amazon EBS a sus tareas de Amazon ECS del tipo lanzamiento de Fargate o EC2. El rol de infraestructura permite a Amazon ECS administrar los volúmenes de Amazon EBS para sus tareas. • Utilice la seguridad de la capa de transporte (TLS) para cifrar el tráfico entre sus servicios de Amazon ECS Service Connect. • Quiere crear grupos de destino de VPC Lattice.	Rol de IAM de infraestructura de Amazon ECS

Definiciones de tareas de Amazon ECS

Una definición de tarea es un esquema de la aplicación. Se trata de un archivo de texto en formato JSON que describe los parámetros y uno o varios contenedores que forman la aplicación.

Entre los parámetros que se pueden especificar en una definición de tareas se incluyen los siguientes:

- El tipo de lanzamiento a utilizar, que determina la infraestructura en la que se alojan sus tareas
- La imagen de Docker que se va a utilizar con cada contenedor en su tarea
- La cantidad de CPU y de memoria que se va a utilizar con cada tarea o cada contenedor dentro de una tarea
- Los requisitos de memoria y CPU
- El sistema operativo del contenedor en el que se ejecuta la tarea
- El modo de red de Docker que utilizar para los contenedores en la tarea
- La configuración de registros que se va a utilizar para sus tareas
- Si la tarea se sigue ejecutando si el contenedor finaliza o falla
- El comando que el contenedor ejecuta al iniciarse
- Los volúmenes de datos que se utilizan con los contenedores en la tarea
- El rol de IAM que las tareas utilizan

Para obtener una lista completa de parámetros de definición de tareas, consulte Parámetros en la definición de tareas de Amazon ECS para el tipo de lanzamiento de Fargate.

Tras crear una definición de tarea, puede ejecutarla como una tarea o un servicio.

- Una tarea es la instancia creada de una definición de tarea dentro de un clúster. Después de crear una definición de tareas para la aplicación dentro de Amazon ECS, puede especificar el número de tareas que se ejecutarán en el clúster.
- Un servicio de Amazon ECS ejecuta y mantiene simultáneamente el número deseado de tareas en un clúster de Amazon ECS. El funcionamiento es que, en caso de que alguna de las tareas falle o se pare por algún motivo, el programador de servicio de Amazon ECS lanza otra instancia en función de la definición de tarea. Lo hace para reemplazarlo y así mantener el número deseado de tareas en el servicio.

Temas

- Estados de definiciones de tareas de Amazon ECS
- Diseño de la arquitectura de su aplicación para Amazon ECS
- Creación de una definición de tareas de Amazon ECS mediante la consola
- Actualización de una definición de tareas de Amazon ECS mediante la consola
- Anulación del registro de la revisión de una definición de tareas de Amazon ECS mediante la consola
- Eliminación de una revisión de definición de tareas de Amazon ECS con la consola
- Casos de uso de definiciones de tareas de Amazon ECS
- Parámetros en la definición de tareas de Amazon ECS para el tipo de lanzamiento de Fargate
- Parámetros en la definición de tareas de Amazon ECS para el tipo de lanzamiento de EC2
- Plantilla de definición de tareas de Amazon ECS
- Ejemplo de definiciones de tareas de Amazon ECS

Estados de definiciones de tareas de Amazon ECS

Una definición de tareas cambia de estado al crearla, anular su registro o eliminarla. Puede ver el estado de la definición de tareas en la consola, o mediante `DescribeTaskDefinition`.

A continuación, se muestran los posibles estados de una definición de tareas:

ACTIVE

La definición de tareas está `ACTIVE` después de registrarla en Amazon ECS. Puede utilizar las definiciones de tareas en el estado `ACTIVE` para ejecutar tareas o crear servicios.

INACTIVE

La definición de tareas pasa del estado `ACTIVE` al estado `INACTIVE` cuando se anula su registro. Puede recuperar una definición de tareas en estado `INACTIVE` llamando a `DescribeTaskDefinition`. No puede ejecutar nuevas tareas ni crear nuevos servicios con una definición de tareas en estado `INACTIVE`. No afecta a los servicios ni las tareas existentes.

DELETE_IN_PROGRESS

La definición de tareas pasa del estado `INACTIVE` al estado `DELETE_IN_PROGRESS` después de enviarla para su eliminación. Una vez que la definición de tareas está en el

estado `DELETE_IN_PROGRESS`, Amazon ECS verifica periódicamente que ninguna tarea ni implementación activa haga referencia a la definición de tareas de destino y, a continuación, la elimina de forma permanente. No puede ejecutar nuevas tareas ni crear nuevos servicios con una definición de tareas en estado `DELETE_IN_PROGRESS`. Se puede enviar una definición de tareas para su eliminación en cualquier momento sin que ello repercuta en las tareas ni los servicios existentes.

Las definiciones de tareas que se encuentran en el estado `DELETE_IN_PROGRESS` se pueden consultar en la consola y se puede recuperar la definición de tareas haciendo una llamada a `DescribeTaskDefinition`.

Al eliminar todas las revisiones de la definición de tareas `INACTIVE`, el nombre de la definición de tareas no se muestra en la consola ni se devuelve en la API. Si una revisión de definición de tareas tiene el estado `DELETE_IN_PROGRESS`, el nombre de la definición de tareas se muestra en la consola y se devuelve en la API. Amazon ECS retiene el nombre de la definición de tarea y la revisión se incrementa la próxima vez que cree una definición de tarea con ese nombre.

Si utiliza AWS Config para administrar las definiciones de tareas, AWS Config le cobrará todos los registros de definiciones de tareas. Solo se le cobrará por anular el registro de la última definición de tareas en estado `ACTIVE`. No se aplica ningún cargo por eliminar una definición de tareas. Para obtener más información sobre los precios, consulte Precios de AWS Config.

Recursos de Amazon ECS que pueden bloquear una eliminación

Una solicitud de eliminación de definición de tareas no se completará cuando haya recursos de Amazon ECS que dependan de la revisión de la definición de tareas. Los siguientes recursos pueden impedir que se elimine una definición de tareas:

- Tareas independientes de Amazon ECS: la definición de la tarea es necesaria para que la tarea se mantenga en buen estado.
- Tareas de servicio de Amazon ECS: la definición de la tarea es necesaria para que la tarea se mantenga en buen estado.
- Implementaciones de servicio y conjuntos de tareas de Amazon ECS: la definición de la tarea es necesaria cuando se inicia un evento de escalado para una implementación o conjunto de tareas de Amazon ECS.

Si la definición de la tarea permanece en el estado `DELETE_IN_PROGRESS`, puede utilizar la consola o la AWS CLI para identificar y, a continuación, detener los recursos que bloquean la eliminación de la definición de la tarea.

Eliminación de la definición de tareas después de eliminar el recurso bloqueado

Las siguientes reglas se aplican después de eliminar los recursos que bloquean la eliminación de la definición de tarea:

- Tareas de Amazon ECS: la eliminación de la definición de tareas puede tardar hasta 1 hora en completarse una vez detenida la tarea.
- Implementaciones de servicio y conjuntos de tareas de Amazon ECS: la eliminación de la definición de tareas puede tardar hasta 24 horas en completarse una vez que se haya eliminado la implementación o el conjunto de tareas.

Diseño de la arquitectura de su aplicación para Amazon ECS

La arquitectura de la aplicación se hace mediante la creación de una definición de tareas para la aplicación. La definición de tareas contiene los parámetros que definen la información acerca de la aplicación, entre los que se incluyen los siguientes:

- Tipo de lanzamiento que se debe utilizar que determina la infraestructura en la que se alojan las tareas.

  Cuando se utiliza el tipo de lanzamiento de EC2, también se elige el tipo de instancia. Para algunos tipos de instancias, como la GPU, debe configurar otros parámetros. Para obtener más información, consulte Casos de uso de definiciones de tareas de Amazon ECS.
- Imagen del contenedor que contiene el código de la aplicación y todas las dependencias que el código de la aplicación requiere para ejecutarse.
- El modo de red que se debe utilizar para los contenedores en la tarea.

  El modo de red determina cómo se comunica la tarea a través de la red.

  En el caso de las tareas que se ejecutan en instancias de EC2, hay varias opciones, pero recomendamos utilizar el modo de red de `awsvpc`. El modo de red de `awsvpc` simplifica las redes de contenedores, ya que proporciona mayor control sobre la comunicación de las aplicaciones entre sí y con los demás servicios de las VPC.

En el caso de las tareas que se ejecutan en Fargate, debe utilizar el modo de red de `awsvpc`.

- Configuración de registros que se va a utilizar para las tareas.
- Volúmenes de datos que se utilizan con los contenedores en la tarea.

Para obtener una lista completa de parámetros de definición de tareas, consulte Parámetros en la definición de tareas de Amazon ECS para el tipo de lanzamiento de Fargate.

Utilice las siguientes pautas al crear las definiciones de tareas:

- Utilice cada familia de definiciones de tareas para un solo propósito empresarial.

  Si agrupa varios tipos de contenedores de aplicaciones en la misma definición de tarea, no puede escalar esos contenedores de manera independiente. Por ejemplo, en el caso de un sitio web y una API suelen ser necesarios patrones de escalado distintos. A medida que aumente el tráfico, es posible que sea necesario un número distinto de contenedores web que de contenedores de API. Si estos dos contenedores se implementan en la misma definición de tarea, cada tarea ejecuta la misma cantidad de contenedores web y contenedores de API.
- Haga coincidir cada versión de la aplicación con una revisión de la definición de tareas dentro de una familia de definiciones de tareas.

  En una familia de definiciones de tareas, cada revisión de la definición de tareas representa una instantánea a partir de un momento específico de la configuración de una imagen de contenedor concreta. Es similar a cómo el contenedor es una instantánea de todos los componentes necesarios para ejecutar una versión concreta del código de la aplicación.

  Cree una asignación con correspondencia entre una versión del código de la aplicación, una etiqueta de imagen del contenedor y una revisión de la definición de la tarea. Un proceso de publicación típico implica una confirmación de git que se convierte en una imagen de contenedor etiquetada con el código SHA de la confirmación de git. Luego, esa etiqueta de imagen del contenedor recibe su propia revisión de la definición de tareas de Amazon ECS. Por último, se actualiza el servicio de Amazon ECS para implementar la nueva revisión de la definición de tareas.
- Utilice diferentes roles de IAM para cada familia de definiciones de tareas.

  Defina cada definición de tarea con su propio rol de IAM. Implemente esta práctica y proporcione a cada componente empresarial su propia familia de definiciones de tareas. Al implementar estas dos prácticas recomendadas, puede limitar el acceso de cada servicio a los recursos de la cuenta de AWS. Por ejemplo, puede dar acceso al servicio de autenticación para que se conecte a la base

de datos de contraseñas. Al mismo tiempo, puede asegurarse de que solo el servicio de pedidos tenga acceso a la información de pago con tarjeta de crédito.

Prácticas recomendadas para las imágenes de contenedores de Amazon ECS

Una imagen de contenedor es un conjunto de instrucciones sobre cómo construir el contenedor. Una imagen de contenedor contiene el código de la aplicación y todas las dependencias que el código de la aplicación requiere para ejecutarse. Las dependencias de la aplicación incluyen los paquetes de código fuente en los que se basa el código de la aplicación, el tiempo de ejecución del lenguaje para los lenguajes interpretados y los paquetes binarios en los que se basa el código vinculado dinámicamente.

Utilice las siguientes pautas al diseñar y crear las imágenes de los contenedores:

- Complete las imágenes del contenedor almacenando todas las dependencias de la aplicación como archivos estáticos dentro de la imagen del contenedor.

  Si cambia algo en la imagen del contenedor, cree una nueva con los cambios.
- Ejecute un único proceso de aplicación dentro de un contenedor.

  La vida útil del contenedor dura mientras se ejecute el proceso de la aplicación. Amazon ECS reemplaza los procesos bloqueados y determina dónde iniciar el proceso de reemplazo. Una imagen completa hace que la implementación general sea más resistente.
- Haga que la aplicación gestione `SIGTERM`.

  Cuando Amazon ECS detiene una tarea, primero envía una señal SIGTERM a la tarea para notificarle que la aplicación debe finalizar y cerrarse. A continuación, Amazon ECS envía un mensaje `SIGKILL`. Cuando las aplicaciones ignoran `SIGTERM`, el servicio Amazon ECS debe esperar para enviar la señal `SIGKILL` para terminar el proceso.

  Debe identificar cuánto tarda la aplicación en completar su trabajo y asegurarse de que sus aplicaciones gestionan la señal `SIGTERM`. La gestión de señales de la aplicación debe impedir que la aplicación retome trabajos nuevos y complete los que están en curso, o debe guardar los trabajos pendientes en un almacenamiento externo a la tarea cuando tarden demasiado en completarse.
- Configure las aplicaciones en contenedores para escribir registros en `stdout` y `stderr`.

Desacoplar la gestión de registros del código de la aplicación proporciona flexibilidad para ajustar la gestión de registros de la infraestructura. Un ejemplo de ello es cambiar el sistema de registros. En lugar de modificar los servicios y crear e implementar una nueva imagen de contenedor, puede ajustar la configuración.

- Utilice etiquetas para crear versiones de las imágenes de contenedores.

  Las imágenes de los contenedores se almacenan en un registro de contenedores. Cada imagen de un registro se identifica mediante una etiqueta. Hay una etiqueta llamada `latest`. Esta etiqueta funciona como un puntero de la última versión de la imagen del contenedor de la aplicación, de forma similar a la `HEAD` de un repositorio de git. Le recomendamos que solo utilice la etiqueta `latest` solo para pruebas. Como práctica recomendada, etiquete las imágenes del contenedor con una etiqueta única para cada compilación. Le recomendamos etiquetar las imágenes con el git SHA para la confirmación de git que se utilizó para crear la imagen.

  No es necesario crear una imagen de contenedor para cada confirmación. Sin embargo, le recomendamos que cree una nueva imagen de contenedor cada vez que publique una confirmación de código concreta en el entorno de producción. También le recomendamos etiquetar la imagen con una etiqueta que corresponda a la confirmación de git del código que está dentro de la imagen. Si etiquetó la imagen con la confirmación de git, podrá encontrar más rápidamente qué versión del código está ejecutando la imagen.

  También le recomendamos que active las etiquetas de imagen inmutables en Amazon Elastic Container Registry. Con esta configuración, no puede cambiar la imagen del contenedor a la que apunta una etiqueta. En cambio, Amazon ECR exige que se cargue una imagen nueva en una etiqueta nueva. Para obtener más información, consulte Mutabilidad de las etiquetas de imagen en la Guía del usuario de Amazon ECR.

Al diseñar la arquitectura de la aplicación para que se ejecute en AWS Fargate, debe decidir entre implementar varios contenedores en la misma definición de tareas e implementar contenedores por separado en varias definiciones de tareas. Si se requieren las siguientes condiciones, le recomendamos implementar los contenedores en una sola definición de tareas:

- Los contenedores comparten un ciclo de vida común (es decir, se lanzan y terminan a la vez).
- Los contenedores se deben ejecutar en el mismo host subyacente (es decir, un contenedor hace referencia al otro en un puerto localhost).
- Sus contenedores comparten recursos.

- Sus contenedores comparten volúmenes de datos.

Si no se requieren estas condiciones, le recomendamos implementar los contenedores por separado en varias definiciones de tareas. Esto permite escalar, aprovisionar y desaprovisionar los contenedores por separado.

Prácticas recomendadas para los tamaños de las tareas de Amazon ECS

El tamaño del contenedor y el tamaño de las tareas son esenciales para planificar el escalado y la capacidad. En Amazon ECS, se utilizan dos métricas de recursos para la capacidad: CPU y memoria. La CPU se mide en unidades de 1/1024 de una vCPU completa (donde 1024 unidades equivalen a 1 vCPU completa). La memoria se mide en megabytes. En la definición de la tarea, puede configurar las reservas y los límites de los recursos.

Cuando configura una reserva, se establece la cantidad mínima de recursos que requiere una tarea. La tarea recibe como mínimo la cantidad de recursos solicitada. Es posible que la aplicación pueda utilizar más CPU o memoria que la reserva que se declare. Sin embargo, esto está sujeto a los límites que también se hayan declarado. Utilizar una cantidad superior a la reserva se conoce como ampliación. En Amazon ECS, las reservas están garantizadas. Por ejemplo, si utiliza instancias de Amazon EC2 para proporcionar capacidad, Amazon ECS no coloca una tarea en una instancia en la que no se pueda gestionar la reserva.

Un límite es la cantidad máxima de unidades de CPU o memoria que pueden utilizar el contenedor o la tarea. Cualquier intento de utilizar más CPU por encima de este límite provoca una limitación. Cualquier intento de utilizar más memoria provocará que el contenedor se detenga.

Elegir estos valores puede resultar difícil. Esto se debe a que los valores más adecuados para la aplicación dependen en gran medida de los requisitos de los recursos de la aplicación. Las pruebas de carga de la aplicación son la clave para planificar correctamente los requisitos de recursos y comprender mejor los requisitos de la aplicación.

Aplicaciones sin estado

En el caso de las aplicaciones sin estado que se escalan horizontalmente, como una aplicación detrás de un equilibrador de carga, recomendamos primero determinar la cantidad de memoria que consume la aplicación cuando atiende las solicitudes. Para ello, puede utilizar las herramientas tradicionales, como `ps` o `top`, o las soluciones de supervisión, como Información de contenedores de CloudWatch.

Al determinar la reserva de CPU, tenga en cuenta cómo quiere escalar la aplicación para que cumpla con los requisitos de su empresa. Puede utilizar reservas de CPU más pequeñas, como 256 unidades de CPU (o 1/4 de vCPU), para escalar horizontalmente de manera detallada y minimizar los costos. Sin embargo, es posible que no se escalen lo suficientemente rápido como para satisfacer los picos significativos de la demanda. Puede utilizar reservas de CPU más grandes para reducir horizontalmente y escalar horizontalmente con mayor rapidez y, por lo tanto, adaptarse a los picos de la demanda en menor tiempo. Sin embargo, las reservas de CPU más grandes son más costosas.

Otras aplicaciones

En el caso de las aplicaciones que no se escalan horizontalmente, como los trabajos singleton o los servidores de bases de datos, las consideraciones más importantes son la capacidad disponible y el costo. Debe elegir la cantidad de memoria y CPU en función de lo que las pruebas de carga indiquen que se necesita para atender el tráfico y cumplir su objetivo de nivel de servicio. Amazon ECS garantiza que la aplicación se coloque en un host que tenga la capacidad adecuada.

Opciones de redes de tareas de Amazon ECS para el tipo de lanzamiento de EC2

El comportamiento de las redes de tareas de Amazon ECS alojadas en instancias de Amazon EC2 depende del modo de red que se definió en la definición de tareas. Le recomendamos que utilice el modo de red `awsvpc`, salvo que, por una necesidad específica, deba utilizar otro.

Estos son los modos de red disponibles.

Modo de red	Contenedores de Linux en EC2	Contenedores de Windows en EC2	Descripción
`awsvpc`	Sí	Sí	A la tarea se asigna su propia interfaz de red elástica (ENI) y una dirección IPv4 privada principal. Esto otorga a la tarea las mismas propiedades de redes que las instancias de Amazon EC2.

Modo de red	Contenedores de Linux en EC2	Contenedores de Windows en EC2	Descripción
`bridge`	Sí	No	La tarea usa la red virtual integrada de Docker en Linux que se ejecuta dentro de cada instancia de Amazon EC2 que aloja la tarea. La red virtual integrada en Linux usa el controlador de red `bridge` Docker. Este es el modo de red predeterminado en Linux si no se especifica un modo de red en la definición de la tarea.
`host`	Sí	No	La tarea usa la red del host que omite la red virtual integrada de Docker al asignar los puertos del contenedor directamente a la ENI de la instancia de Amazon EC2 que aloja la tarea. Las asignaciones de puertos dinámicas no se pueden usar en este modo de red. Un contenedor de una definición de tarea que use este modo debe especificar un número de `hostPort` específico. Varias tareas no pueden usar el número de puerto de un host. Por lo tanto, no se pueden ejecutar varias tareas de la misma definición de tarea en una instancia de Amazon EC2 única.
`none`	Sí	No	La tarea no tiene conectividad de red externa.
`default`	No	Sí	La tarea usa la red virtual integrada de Docker en Windows que se ejecuta dentro de cada instancia de Amazon EC2 que aloja la tarea. La red virtual integrada en Windows usa el controlador de red `nat` Docker. Este es el modo de red predeterminado en Windows si no se especifica un modo de red en la definición de la tarea.

Para obtener más información sobre las redes Docker en Linux, consulte Descripción general de las redes en la documentación de Docker.

Para obtener más información sobre las redes Docker en Windows, consulte Redes de contenedores de Windows en la documentación de contenedores en Windows de Microsoft.

Asignación de una interfaz de red para una tarea de Amazon ECS

Las características de integración en red de las tareas que ofrece el modo de red `awsvpc` proporcionan a las tareas de Amazon ECS las mismas propiedades de redes que poseen las instancias de Amazon EC2. Con el modo de red de `awsvpc` se simplifican las redes de contenedores, porque proporciona mayor control sobre la comunicación de las aplicaciones entre sí y con los demás servicios de las VPC. El modo de red `awsvpc` también proporciona mayor seguridad para los contenedores, ya que permite utilizar grupos de seguridad y herramientas de supervisión de red de forma más pormenorizada dentro de las tareas. También puede utilizar otras características de las redes de Amazon EC2, como los registros de flujo de VPC, para supervisar el tráfico entrante y saliente de las tareas. Además, los contenedores que pertenecen a la misma tarea puede comunicarse a través de la interfaz `localhost`.

La interfaz de red elástica (ENI) de la tarea es una característica completamente administrada de Amazon ECS. Amazon ECS crea la ENI y la asocia a la instancia de Amazon EC2 del host con el grupo de seguridad especificado. La tarea envía y recibe el tráfico de red en la ENI, tal y como lo hacen las instancias de Amazon EC2 con sus principales interfaces de red. A cada tarea ENI se le asigna una dirección IPv4 privada de forma predeterminada. Si la VPC está habilitada para el modo de pila doble y utiliza una subred con un bloque de CIDR IPv6, la ENI de la tarea también recibirá una dirección IPv6. Cada tarea puede tener una sola ENI.

Estas ENI se pueden ver en la consola de Amazon EC2 de su cuenta. Su cuenta no puede separar ni modificar las ENI. De este modo, se evita la eliminación accidental de una ENI que esté asociada a una tarea en ejecución. Puede consultar la información de asociación de las ENI de las tareas en la consola de Amazon ECS o con la operación de la API DescribeTasks. Cuando la tarea se detiene o se reduce la escala del servicio, la ENI de tareas se desvincula y se elimina.

Cuando necesite aumentar la densidad de las ENI, utilice la configuración de la cuenta de `awsvpcTrunking`. Amazon ECS también crea y asocia una interfaz de red troncal para la instancia de contenedor. La red troncal está completamente administrada por Amazon ECS. La ENI troncal se elimina al terminar o al anular el registro de la instancia de contenedor del clúster de Amazon ECS. Para más información acerca de la configuración de la cuenta de `awsvpcTrunking`, consulte Requisitos previos.

Especifica `awsvpc` en el parámetro `networkMode` de la definición de la tarea. Para obtener más información, consulte Modo de red.

A continuación, cuando ejecute una tarea o cree un servicio, utilice el parámetro `networkConfiguration` que incluya una o varias subredes en las que deban colocarse las tareas y uno o varios grupos de seguridad que deban vincularse a una ENI. Para obtener más información, consulte Configuración de red. Las tareas se colocan en instancias de Amazon EC2 válidas en las mismas zonas de disponibilidad que esas subredes; por su parte, los grupos de seguridad especificados se asocian con la ENI que se aprovisiona para la tarea.

Consideraciones acerca de Linux

Tenga en cuenta lo siguiente cuando utilice el sistema operativo Linux:

- Si utiliza una instancia p5.48xlarge en modo `awsvpc`, no puede ejecutar más de 1 tarea en la instancia.
- Las tareas y los servicios que utilizan el modo de red `awsvpc` requieren el rol vinculado al servicio de Amazon ECS con el fin de proporcionar a Amazon ECS los permisos necesarios para realizar llamadas a otros servicios de AWS en su nombre. Este rol se crea automáticamente al crear un clúster, o bien al crear o actualizar un servicio en la AWS Management Console. Para obtener más información, consulte Uso de roles vinculados al servicio para Amazon ECS. También puede crear el rol vinculado a servicio con el comando siguiente de la AWS CLI:

  ```
  aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
  ```

- La instancia de Linux de Amazon EC2 requiere la versión `1.15.0` del agente de contenedor o una posterior para ejecutar tareas que utilizan el modo de red `awsvpc`. Si utiliza una AMI optimizada para Amazon ECS, la instancia también necesita al menos la versión `1.15.0-4` del paquete `ecs-init`.
- Amazon ECS rellena el nombre de host de la tarea con un nombre de host DNS (interno) que proporciona Amazon cuando las opciones `enableDnsHostnames` y `enableDnsSupportestán` habilitadas en la VPC. Si estas opciones no están habilitadas, el nombre de host DNS de la tarea se establece en un nombre de host aleatorio. Para obtener más información acerca de la configuración de DNS de una VPC, consulte Utilización de DNS con su VPC en la Guía del usuario de Amazon VPC.
- Cada tarea de Amazon ECS que utiliza el modo de red `awsvpc` recibe su propia interfaz de red elástica (INE), la cual se asocia a la instancia de Amazon EC2 que la aloja. Existe una cuota predeterminada para el número de interfaces de red que se pueden asociar a una instancia

de Linux de Amazon EC2. La interfaz de red principal cuenta como una para esa cuota. Por ejemplo, de forma predeterminada, una instancia de `c5.large` podría tener solo hasta tres ENI asociadas a ella. La interfaz de red principal de la instancia cuenta como una. Puede asociar dos ENI adicionales a la instancia. Dado que cada tarea que utiliza el modo de red `awsvpc` requiere una ENI, normalmente solo puede ejecutar dos de esas tareas en este tipo de instancia. Para obtener más información acerca de los límites de ENI predeterminados para cada tipo de instancia, consulte Direcciones IP por interfaz de red por tipo de instancia en la Guía del usuario de Amazon EC2.

- Amazon ECS admite el lanzamiento de instancias de Linux de Amazon EC2 que utilizan tipos de instancias con mayor densidad de ENI compatibles. Al optar por incluir la configuración de cuenta `awsvpcTrunking` y registrar instancias de Linux de Amazon EC2 con estos tipos de instancias en el clúster, estas instancias tienen cuotas de ENI más altas. Si se utilizan estas instancias con una cuota más alta, es posible colocar más tareas en cada instancia de Linux de Amazon EC2. Para utilizar la mayor densidad de ENI con la característica de enlace troncal, la instancia de Amazon EC2 requiere al menos la versión `1.28.1` del agente de contenedor. Si utiliza una AMI optimizada para Amazon ECS, la instancia también requiere al menos la versión `1.28.1-2` del paquete `ecs-init`. Para obtener más información acerca de la inscripción en el ajuste de cuenta `awsvpcTrunking`, consulte Acceso a las características de Amazon ECS con la configuración de la cuenta. Para obtener más información acerca del enlace troncal de ENI, consulte Aumento de las interfaces de red de instancias de contenedor de Linux de Amazon ECS.
- Cuando se alojan tareas que utilizan el modo de red `awsvpc` en instancias de Linux de Amazon EC2, las ENI de tareas no reciben direcciones IP públicas. Para obtener acceso a Internet, las tareas deben lanzarse en una subred privada configurada para utilizar una puerta de enlace NAT. Para obtener información, consulte Gateways NAT en la Guía del usuario de Amazon VPC. El acceso de red entrante debe tener lugar desde el interior de la VPC mediante la dirección IP privada o dirigirse a través del equilibrador de carga desde dentro de la VPC. Las tareas iniciadas en subredes públicas no tienen acceso a Internet.
- Amazon ECS solo reconoce las ENI que asocia a sus instancias de Linux de Amazon EC2. Si asoció ENI a sus instancias de forma manual, Amazon ECS podría intentar agregar una tarea a una instancia que no tiene suficientes adaptadores de red. Esto puede hacer que se agote el tiempo de espera de la tarea, que la tarea pase a un estado de desaprovisionamiento y, a continuación, a un estado detenido. Recomendamos no asociar ENI manualmente a las instancias.
- Las instancias de Linux de Amazon EC2 deben estar registradas en la función `ecs.capability.task-eni` para que se las tenga en cuenta en la ubicación de tareas con el modo de red `awsvpc`. Las instancias que ejecutan la versión `1.15.0-4` o una posterior de `ecs-init` se registran automáticamente con este atributo.

- La cuenta no puede separar manualmente ni modificar las ENI que se crean y asocian a las instancias de Linux de Amazon EC2. De este modo, se evita la eliminación accidental de una ENI asociada a una tarea en ejecución. Para liberar las ENI de una tarea, detenga la tarea.
- Existe un límite de 16 subredes y 5 grupos de seguridad que se puede especificar en `awsVpcConfiguration` cuando se ejecuta una tarea o se crea un servicio que utilice el modo de red `awsvpc`. Para obtener más información, consulte AwsVpcConfiguration en la Referencia de la API de Amazon Elastic Container Service.
- Cuando se inicia una tarea con el modo de red `awsvpc`, el agente de contenedor de Amazon ECS crea un contenedor `pause` adicional para cada tarea antes de iniciar los contenedores en la definición de tareas. Luego, configura el espacio de nombres de red del contenedor `pause` mediante la ejecución de los complementos de CNI amazon-ecs-cni-plugins. A continuación, el agente inicia el resto de los contenedores en la tarea para que ellos compartan la pila de red del contenedor `pause`. Esto significa que todos los contenedores de una tarea son direccionables mediante las direcciones IP de la ENI y que se pueden comunicar entre sí a través de la interfaz `localhost`.
- Los servicios con tareas que utilizan el modo de red `awsvpc` solo admiten equilibradores de carga de aplicación y equilibradores de carga de red. Cuando crea grupos de destino para estos servicios, debe elegir `ip` como tipo de destino. No utilice `instance`. Esto se debe a que las tareas que utilizan el modo de red `awsvpc` se asocian con una ENI, no con una instancia de Linux de Amazon EC2. Para obtener más información, consulte Uso del equilibrador de carga para distribuir el tráfico de servicio de Amazon ECS.
- Si la VPC se actualiza para cambiar el conjunto de opciones DHCP que utiliza, no es posible aplicar estos cambios a las tareas existentes. Inicie tareas nuevas con estos cambios aplicados, compruebe que funcionan correctamente y, luego, detenga las tareas existentes para cambiar de manera segura estas configuraciones de red.

Consideraciones acerca de Windows

A continuación, se detallan consideraciones que se deben tener en cuenta cuando se utiliza el sistema operativo Windows:

- Las instancias de contenedor que utilizan la AMI de Windows Server 2016 optimizada para Amazon ECS no pueden alojar tareas que utilizan el modo de red `awsvpc`. Si tiene un clúster que contiene AMI de Windows Server 2016 y AMI de Windows optimizadas para Amazon ECS que sí admiten el modo de red `awsvpc`, las tareas que utilizan el modo de red `awsvpc` no se lanzan en

las instancias de Windows Server 2016. En cambio, se lanzarán en las instancias que admiten el modo de red `awsvpc`.

- Su instancia de Windows de Amazon EC2 requiere la versión `1.57.1` del agente de contenedor o una posterior para usar las métricas de CloudWatch para los contenedores de Windows que utilizan el modo de red `awsvpc`.
- Las tareas y los servicios que utilizan el modo de red `awsvpc` requieren el rol vinculado al servicio de Amazon ECS con el fin de proporcionar a Amazon ECS los permisos necesarios para realizar llamadas a otros servicios de AWS en su nombre. Este rol se crea automáticamente al crear un clúster, o bien al crear o actualizar un servicio, en la AWS Management Console. Para obtener más información, consulte Uso de roles vinculados al servicio para Amazon ECS. También puede crear el rol vinculado a un servicio con el siguiente comando de la AWS CLI:

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- Su instancia de Windows de Amazon EC2 requiere la versión `1.54.0` del agente de contenedor o una posterior para ejecutar tareas que utilizan el modo de red `awsvpc`. Al arrancar la instancia, debe configurar las opciones requeridas para el modo de red `awsvpc`. Para obtener más información, consulte the section called “Arranque de instancias de contenedor”.
- Amazon ECS rellena el nombre de host de la tarea con un nombre de host DNS (interno) que proporciona Amazon cuando las opciones `enableDnsHostnames` y `enableDnsSupport` están habilitadas en la VPC. Si estas opciones no están habilitadas, el nombre de host DNS de la tarea es un nombre de host aleatorio. Para obtener más información acerca de la configuración de DNS de una VPC, consulte Utilización de DNS con su VPC en la Guía del usuario de Amazon VPC.
- Cada tarea de Amazon ECS que utiliza el modo de red `awsvpc` recibe su propia interfaz de red elástica (INE), la cual se asocia a la instancia de Windows de Amazon EC2 que la aloja. Existe una cuota predeterminada para el número de interfaces de red que se pueden asociar a una instancia de Windows de Amazon EC2. La interfaz de red principal cuenta como una para esta cuota. Por ejemplo, de forma predeterminada, una instancia de `c5.large` puede tener asociadas hasta tres ENI. La interfaz de red principal de la instancia cuenta como una de ellas. Puede asociar dos ENI adicionales a la instancia. Dado que cada tarea que utiliza el modo de red `awsvpc` requiere una ENI, normalmente solo puede ejecutar dos de esas tareas en este tipo de instancia. Para obtener más información acerca de los límites de ENI predeterminados para cada tipo de instancia, consulte Direcciones IP por interfaz de red por tipo de instancia en la Guía del usuario de Amazon EC2.
- Cuando se alojan tareas que utilizan el modo de red `awsvpc` en instancias de Windows de Amazon EC2, las ENI de tareas no reciben direcciones IP públicas. Para obtener acceso a

Internet, lance las tareas en una subred privada configurada para utilizar una puerta de enlace NAT. Para obtener información, consulte Gateways NAT en la Guía del usuario de Amazon VPC. El acceso de red entrante debe tener lugar desde el interior de la VPC mediante la dirección IP privada o dirigirse a través del equilibrador de carga desde dentro de la VPC. Las tareas iniciadas en subredes públicas no tienen acceso a Internet.

- Amazon ECS solo reconoce las ENI que asoció a sus instancias de Windows de Amazon EC2. Si asoció ENI a sus instancias de forma manual, Amazon ECS podría intentar agregar una tarea a una instancia que no tiene suficientes adaptadores de red. Esto puede hacer que se agote el tiempo de espera de la tarea, que la tarea pase a un estado de desaprovisionamiento y, a continuación, a un estado detenido. Recomendamos no asociar ENI manualmente a las instancias.
- Las instancias de Windows de Amazon EC2 deben estar registradas en la función `ecs.capability.task-eni` para que se las tenga en cuenta en la ubicación de tareas con el modo de red `awsvpc`.
- No es posible separar manualmente ni modificar las ENI que se crean y asocian a las instancias de Windows de Amazon EC2. De este modo, se evita que elimine accidentalmente una ENI que esté asociada a una tarea en ejecución. Para liberar las ENI de una tarea, detenga la tarea.
- Solo es posible especificar hasta 16 subredes y 5 grupos de seguridad en `awsVpcConfiguration` cuando se ejecuta una tarea o se crea un servicio que utiliza el modo de red `awsvpc`. Para obtener más información, consulte AwsVpcConfiguration en la Referencia de la API de Amazon Elastic Container Service.
- Cuando se inicia una tarea con el modo de red `awsvpc`, el agente de contenedor de Amazon ECS crea un contenedor `pause` adicional para cada tarea antes de iniciar los contenedores en la definición de tareas. Luego, configura el espacio de nombres de red del contenedor `pause` mediante la ejecución de los complementos de CNI amazon-ecs-cni-plugins. A continuación, el agente inicia el resto de los contenedores en la tarea para que ellos compartan la pila de red del contenedor `pause`. Esto significa que todos los contenedores de una tarea son direccionables mediante las direcciones IP de la ENI y que se pueden comunicar entre sí a través de la interfaz `localhost`.
- Los servicios con tareas que utilizan el modo de red `awsvpc` solo admiten equilibradores de carga de aplicación y equilibradores de carga de red. Al crear grupos de destino para estos servicios, se debe elegir `ip` como tipo de destino, no `instance`. Esto se debe a que las tareas que utilizan el modo de red `awsvpc` se asocian con una ENI, no con una instancia de Windows de Amazon EC2. Para obtener más información, consulte Uso del equilibrador de carga para distribuir el tráfico de servicio de Amazon ECS.

- Si la VPC se actualiza para cambiar el conjunto de opciones DHCP que utiliza, no es posible aplicar estos cambios a las tareas existentes. Inicie tareas nuevas con estos cambios aplicados, compruebe que funcionan correctamente y, luego, detenga las tareas existentes para cambiar de manera segura estas configuraciones de red.
- Cuando se utiliza el modo de red `awsvpc` en una configuración de EC2 Windows, no se admite lo siguiente:
  - Configuración de pila doble
  - IPv6
  - Enlace troncal de ENI

Utilización de una VPC en modo de pila doble

Cuando se utiliza una VPC en modo de pila doble, las tareas se pueden comunicar mediante IPv4, IPv6 o ambos. Las direcciones IPv4 e IPv6 son independientes entre sí. Por lo tanto, debe configurar el enrutamiento y la seguridad en su VPC por separado para IPv4 e IPv6. Para obtener más información acerca de cómo configurar la VPC para el modo de pila doble, consulte Migración a IPv6 en la Guía del usuario de Amazon VPC.

Si configuró la VPC con una puerta de enlace de Internet o una puerta de enlace de Internet de solo salida, puede utilizar la VPC en modo de pila doble. De este modo, las tareas a las que se les asigna una dirección IPv6 pueden acceder a Internet a través de una puerta de enlace de Internet o una puerta de enlace de Internet de solo salida. Las puertas de enlace NAT son opcionales. Para obtener más información, consulte Gateways de Internet y Gateways de Internet de solo salida en la Guía del usuario de Amazon VPC.

Se asigna una dirección IPv6 a las tareas de Amazon ECS si se cumplen las siguientes condiciones:

- La instancia de Linux de Amazon EC2 que aloja la tarea está utilizando la versión `1.45.0` del agente de contenedor o una posterior. Para obtener información sobre cómo comprobar la versión del agente que está utilizando la instancia y actualizarla si es necesario, consulte Actualización del agente de contenedor de Amazon ECS.
- La configuración de cuenta `dualStackIPv6` está habilitada. Para obtener más información, consulte Acceso a las características de Amazon ECS con la configuración de la cuenta.
- Su tarea está utilizando el modo de red `awsvpc`.
- La VPC y la subred están configuradas para IPv6. La configuración incluye las interfaces de red creadas en la subred especificada. Para obtener más información sobre cómo configurar la VPC

para el modo de pila doble, consulte Migración a IPv6 y Modificación del atributo de direcciones IPv6 de su subred en la Guía del usuario de Amazon VPC.

Asignación de los puertos del contenedor de Amazon ECS a la interfaz de red de la instancia de EC2

El modo de red `host` solo se admite para las tareas de Amazon ECS alojadas en instancias de Amazon EC2. No es compatible cuando se utiliza Fargate en Amazon ECS.

El modo de red `host` es el modo de red más básico compatible con Amazon ECS. Con el modo host, la red del contenedor está vinculada directamente al host subyacente que está ejecutando el contenedor.

Suponga que está ejecutando un contenedor de Node.js con una aplicación Express que escucha en un puerto `3000` similar a la que se muestra en el diagrama anterior. Cuando se utiliza el modo de red `host`, el contenedor recibe tráfico en el puerto 3000 mediante la dirección IP de la instancia de Amazon EC2 del host subyacente. Le recomendamos que no utilice este modo.

El uso de este modo de red presenta importantes inconvenientes. No puede ejecutar más de una instancia de una tarea en cada host. Esto se debe a que solo la primera tarea puede vincularse al puerto requerido en la instancia de Amazon EC2. Tampoco hay forma de volver a asignar un puerto de contenedor cuando se utiliza el modo de red `host`. Por ejemplo, si una aplicación necesita escuchar un número de puerto concreto, no puede volver a asignar el número de puerto directamente. En su lugar, debe administrar cualquier conflicto de puertos cambiando la configuración de la aplicación.

El uso del modo de red `host` también tiene consecuencias en la seguridad. Este modo permite que los contenedores se hagan pasar por el host y que los contenedores se conecten a los servicios de red de bucle invertido privados del host.

Uso de la red virtual de Docker para las tareas de Linux de Amazon ECS

El modo de red `bridge` solo se admite para las tareas de Amazon ECS alojadas en instancias de Amazon EC2.

Con el modo `bridge`, utiliza un puente de red virtual para crear una capa entre el host y la red del contenedor. De esta forma, puede crear asignaciones de puertos que reasignen un puerto de host a un puerto de contenedor. Las asignaciones pueden ser estáticas o dinámicas.

Con una asignación de puertos estática, puede definir de forma explícita qué puerto de host desea asignar a un puerto de contenedor. En el ejemplo anterior, el puerto `80` del host se asigna al puerto `3000` del contenedor. Para comunicarse con la aplicación en contenedor, debe enviar el tráfico al puerto `80` a la dirección IP de la instancia de Amazon EC2. Desde la perspectiva de la aplicación en contenedor, se ve el tráfico entrante en el puerto `3000`.

Si solo desea cambiar el puerto de tráfico, las asignaciones de puertos estáticas son adecuadas. Sin embargo, esto sigue teniendo la misma desventaja que usar el modo de red `host`. No puede ejecutar más de una instancia de una tarea en cada host. Esto se debe a que una asignación de puertos estática solo permite asignar un único contenedor al puerto 80.

Para resolver este problema, considere la posibilidad de utilizar el modo de red `bridge` con una asignación dinámica de puertos, como se muestra en el siguiente diagrama.

Al no especificar un puerto de host en la asignación de puertos, puede hacer que Docker elija un puerto aleatorio y no utilizado del rango de puertos efímeros y lo asigne como puerto de host público del contenedor. Por ejemplo, a la aplicación Node.js que escucha en el puerto `3000` del contenedor se le puede asignar un puerto aleatorio con un número alto, como `47760` en el host de Amazon EC2. Esto significa que puede ejecutar varias copias de ese contenedor en el host. Además, a cada contenedor se le puede asignar su propio puerto en el host. Cada copia del contenedor recibe tráfico en el puerto `3000`. Sin embargo, los clientes que envían tráfico a estos contenedores utilizan los puertos de host asignados aleatoriamente.

Amazon ECS ayuda a realizar un seguimiento de los puertos asignados de forma aleatoria para cada tarea. Para ello, actualiza automáticamente los grupos de destino del equilibrador de carga y la detección de servicios de AWS Cloud Map para disponer de la lista de puertos y direcciones IP de las tareas. Esto facilita el uso de los servicios que funcionan en modo `bridge` con puertos dinámicos.

Sin embargo, una desventaja de usar el modo de red `bridge` es que es difícil bloquear las comunicaciones entre servicios. Como los servicios pueden asignarse a cualquier puerto aleatorio y no utilizado, es necesario abrir rangos de puertos amplios entre los hosts. Sin embargo, no es fácil crear reglas específicas para que un servicio en particular solo pueda comunicarse con otro servicio

específico. Los servicios no tienen puertos específicos para utilizarlos en las reglas de red de los grupos de seguridad.

Opciones de redes de tareas de Amazon ECS para el tipo de lanzamiento de Fargate

De forma predeterminada, a todas las tareas de Amazon ECS alojadas en Fargate se les proporciona una interfaz de red elástica (ENI) con una dirección IP privada principal. Cuando se utiliza una subred pública, es posible asignar opcionalmente una dirección IP pública a la ENI de la tarea. Si la VPC está configurada para el modo de pila doble y utiliza una subred con un bloque de CIDR IPv6, la ENI de la tarea también recibe una dirección IPv6. Una tarea solo puede tener una ENI asociada a ella por vez. Los contenedores que pertenecen a la misma tarea también pueden comunicarse a través de la interfaz `localhost`. Para obtener más información acerca de las VPC y subredes, consulte Funcionamiento de Amazon VPC en la Guía del usuario de Amazon VPC.

Para que una tarea alojada en Fargate extraiga una imagen de contenedor, la tarea debe tener una ruta a Internet. A continuación, se describe cómo puede comprobar que su tarea tiene una ruta a Internet.

- Cuando se utiliza una subred pública, se puede asignar una dirección IP pública a la ENI de la tarea.
- Cuando se utiliza una subred privada, la subred puede tener una gateway NAT conectada.
- Cuando se utilizan imágenes de contenedor alojadas en Amazon ECR, se puede configurar Amazon ECR para que utilice un punto de conexión de VPC de interfaz y la extracción de imágenes se realiza en la dirección IPv4 privada de la tarea. Para obtener más información, consulte Puntos de conexión de VCP de la interfaz de Amazon ECR (AWS PrivateLink) en la Guía del usuario de Amazon Elastic Container Registry.

Dado que cada tarea obtiene su propia ENI, puede utilizar características de integración en red, como los registros de flujo de VPC, para monitorear el tráfico entrante y saliente de las tareas. Para obtener más información, consulte Logs de flujo de VPC en la Guía del usuario de Amazon VPC.

También puede aprovechar AWS PrivateLink. Puede configurar un punto de conexión de interfaz de VPC para poder acceder a las API de Amazon ECS a través de direcciones IP privadas. AWS PrivateLink restringe todo el tráfico de red entre su VPC y Amazon ECS a la red de Amazon. No necesita una gateway de Internet, un dispositivo NAT ni una gateway privada virtual. Para obtener

más información, consulte Puntos de conexión de VPC de tipo interfaz de Amazon ECR (AWS PrivateLink).

Para ver ejemplos de cómo utilizar el recurso `NetworkConfiguration` con AWS CloudFormation, consulte the section called “Plantillas de AWS CloudFormation ejemplo”.

Las ENI que se crean están completamente administradas por AWS Fargate. Además, existe una política de IAM asociada que se utiliza para conceder permisos a Fargate. En el caso de las tareas que utilizan la versión `1.4.0` de la plataforma de Fargate o una posterior, la tarea recibe una única ENI (que se conoce como la ENI de la tarea) y todo el tráfico de red fluye a través de esa ENI dentro de la VPC. Este tráfico se registra en los registros de flujo de la VPC. En el caso de las tareas que utilizan la versión `1.3.0` de la plataforma de Fargate o una anterior, además de la ENI de la tarea, la tarea también recibe otra ENI distinta de propiedad de Fargate que se utiliza para cierto tráfico de red que no puede verse en los registros de flujo de la VPC. En la tabla que se muestra a continuación, se describe el comportamiento del tráfico de red y la política de IAM requerida para cada versión de la plataforma.

Acción	Flujo de tráfico con la versión **`1.3.0`** y anterior de la plataforma Linux	Flujo de tráfico con la versión **`1.4.0`** de la plataforma Linux	Flujo de tráfico con la versión **`1.0.0`** de la plataforma Windows	Permiso de IAM
Recuperación de credenciales de inicio de sesión de Amazon ECR	ENI de propiedad de Fargate	ENI de la tarea	ENI de la tarea	Rol de IAM de ejecución de tareas
Extracción de imágenes	ENI de la tarea	ENI de la tarea	ENI de la tarea	Rol de IAM de ejecución de tareas
Enviar registros a través de un controlador de registro	ENI de la tarea	ENI de la tarea	ENI de la tarea	Rol de IAM de ejecución de tareas

Acción	Flujo de tráfico con la versión **1.3.0** y anterior de la plataforma Linux	Flujo de tráfico con la versión **1.4.0** de la plataforma Linux	Flujo de tráfico con la versión **1.0.0** de la plataforma Windows	Permiso de IAM
Envío de registros a través de FireLens para Amazon ECS	ENI de la tarea	ENI de la tarea	ENI de la tarea	Rol de IAM para la tarea
Recuperación de secretos de Secrets Manager o Systems Manager	ENI de propiedad de Fargate	ENI de la tarea	ENI de la tarea	Rol de IAM de ejecución de tareas
Tráfico del sistema de archivos de Amazon EFS	No disponible	ENI de la tarea	ENI de la tarea	Rol de IAM para la tarea
Tráfico de la aplicación	ENI de la tarea	ENI de la tarea	ENI de la tarea	Rol de IAM para la tarea

Consideraciones

Tenga en cuenta lo siguiente al utilizar la integración en red de las tareas.

- El rol vinculado al servicio de Amazon ECS se requiere para proporcionar a Amazon ECS los permisos para realizar llamadas a otros servicios de AWS de en su nombre. Este rol se crea al crear un clúster, o bien al crear o actualizar un servicio en la AWS Management Console. Para obtener más información, consulte Uso de roles vinculados al servicio para Amazon ECS. También puede crear el rol vinculado a un servicio con el siguiente comando de la AWS CLI:

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- Amazon ECS rellena el nombre de host de la tarea con un nombre de host DNS que proporciona Amazon cuando las opciones `enableDnsHostnames` y `enableDnsSupport` están habilitadas en la VPC. Si estas opciones no están habilitadas, el nombre de host DNS de la tarea se establece en un nombre de host aleatorio. Para obtener más información acerca de la configuración de DNS de una VPC, consulte Utilización de DNS con su VPC en la Guía del usuario de Amazon VPC.
- Solo se pueden especificar hasta 16 subredes y 5 grupos de seguridad para `awsVpcConfiguration`. Para obtener más información, consulte AwsVpcConfiguration en la Referencia de la API de Amazon Elastic Container Service.
- No es posible separar manualmente ni modificar las ENI que Fargate crea y asocia. De este modo, se evita la eliminación accidental de una ENI asociada a una tarea en ejecución. Para liberar las ENI de una tarea, detenga la tarea.
- Si se actualiza una subred de la VPC para cambiar el conjunto de opciones DHCP que utiliza, no es posible aplicar estos cambios también a las tareas existentes que usan la VPC. Inicie tareas nuevas, que recibirán la nueva configuración para migrar sin problemas mientras prueba el nuevo cambio y, a continuación, detenga las anteriores, si no es necesario realizar ninguna reversión.
- Lo siguiente se aplica a las tareas que se ejecutan en la versión `1.4.0` o posterior de la plataforma Fargate para Linux o `1.0.0` para Windows. Las tareas que se ejecuten en subredes de doble pila se les asignará una dirección IPv4 e IPv6.
- En el caso de las tareas que utilizan la versión `1.4.0` o posterior de Linux o `1.0.0` de Windows de la plataforma, las ENI de la tarea admiten tramas gigantes. Las interfaces de red se configuran con una unidad de transmisión máxima (MTU), que es el tamaño de la carga útil más grande que cabe en una sola trama. Cuanto mayor sea la MTU, mayor será la carga de la aplicación que cabe en una sola trama, lo que reduce la sobrecarga por trama y aumenta la eficiencia. Esta compatibilidad con las tramas gigantes reduce la sobrecarga cuando la ruta de red entre la tarea y el destino admite el uso de estas tramas.
- Los servicios con tareas que utilizan el tipo de lanzamiento de Fargate solo admiten el equilibrador de carga de aplicación y el equilibrador de carga de red. No se admite el equilibrador de carga clásico. Cuando crea grupos de destino, debe elegir `ip` como tipo de destino, no `instance`. Para obtener más información, consulte Uso del equilibrador de carga para distribuir el tráfico de servicio de Amazon ECS.

Utilización de una VPC en modo de pila doble

Cuando se utiliza una VPC en modo de pila doble, las tareas se pueden comunicar mediante IPv4, IPv6 o ambos. Las direcciones IPv4 e IPv6 son independientes entre sí. Por lo tanto, debe configurar

el enrutamiento y la seguridad de su VPC de forma individual para IPv4 e IPv6. Para obtener más información acerca de la configuración de la VPC para el modo de pila doble, consulte Migración a IPv6 en la Guía del usuario de Amazon VPC.

Si se cumplen las siguientes condiciones, a las tareas de Amazon ECS alojadas en Fargate se les asigna una dirección IPv6:

- La configuración de la cuenta `dualStackIPv6` de Amazon ECS está activada (`enabled`) para la entidad principal de IAM que lanza las tareas en la región en la que las va a lanzar. Esta configuración solo se puede modificar con la API o AWS CLI. Tiene la opción de activar esta configuración para una entidad principal de IAM específica de su cuenta o para toda su cuenta mediante la configuración predeterminada de su cuenta. Para obtener más información, consulte Acceso a las características de Amazon ECS con la configuración de la cuenta.
- La VPC y la subred están habilitadas para IPv6. Para obtener más información acerca de cómo configurar la VPC para el modo de pila doble, consulte Migración a IPv6 en la Guía del usuario de Amazon VPC.
- La subred está habilitada para la asignación automática de direcciones IPv6. Para obtener más información sobre cómo configurar la subred, consulte Modificar el atributo de direccionamiento IPv6 de la subred en la Guía del usuario de Amazon VPC.
- La tarea o servicio utiliza la versión `1.4.0` o posterior para Linux de la plataforma de Fargate.

Las tareas de Amazon ECS alojadas en Fargate a las que se les haya asignado una dirección IPv6 pueden acceder a Internet siempre y cuando la VPC esté configurada con una puerta de enlace de Internet o una puerta de enlace de Internet de solo salida. Las puertas de enlace NAT no son necesarias. Para obtener más información, consulte Gateways de Internet y Gateways de Internet de solo salida en la Guía del usuario de Amazon VPC.

Opciones de almacenamiento para las tareas de Amazon ECS

Amazon ECS ofrece opciones de almacenamiento de datos flexibles, rentables y de uso fácil según sus necesidades. Amazon ECS admite las opciones siguientes de volumen de datos para contenedores:

Volumen de datos	Tipos de lanzamiento compatibles	Sistemas operativos compatibles	Persistencia del almacenamiento	Casos de uso
Amazon Elastic Block Store (Amazon EBS)	Fargate, Amazon EC2	Linux	Se puede mantener cuando se adjunta a una tarea independiente. Efímero cuando se adjunta a una tarea que mantiene un servicio.	Los volúmenes de Amazon EBS proporcionan almacenamiento en bloques rentable, duradero y de alto rendimiento para cargas de trabajo en contenedores con uso intensivo de datos. Entre los casos de uso más comunes se incluyen cargas de trabajo transaccionales, como bases de datos, escritorios virtuales y volúmenes raíz, y cargas de trabajo con un rendimiento intensivo, como las cargas de trabajo de procesamiento de registros y ETL. Para

Volumen de datos	Tipos de lanzamiento compatibles	Sistemas operativos compatibles	Persistencia del almacenamiento	Casos de uso
				obtener más información, consulte Uso de volúmenes de Amazon EBS con Amazon ECS.

Volumen de datos	Tipos de lanzamiento compatibles	Sistemas operativos compatibles	Persistencia del almacenamiento	Casos de uso
Amazon Elastic File System (Amazon EFS)	Fargate, Amazon EC2	Linux	Persistente	Los volúmenes de Amazon EFS proporcionan almacenamiento compartido de archivos sencillo, escalable y persistente para utilizarlo con las tareas de Amazon ECS. Dicho almacenam iento aumenta y se reduce automáticamente a medida que agrega o elimina archivos. Los volúmenes de Amazon EFS admiten la simultaneidad y son útiles para las aplicaciones en contenedores que se escalan horizontalmente y necesitan funcionalidades de almacenam iento como

Volumen de datos	Tipos de lanzamiento compatibles	Sistemas operativos compatibles	Persistencia del almacenamiento	Casos de uso
				baja latencia, rendimiento alto y coherencia de lectura tras escritura. Entre los casos de uso más comunes se incluyen las cargas de trabajo como el análisis de datos, el procesamiento multimedia, la administración de contenido y los servidore s web. Para obtener más información, consulte Uso de volúmenes de Amazon EFS con Amazon ECS.

Volumen de datos	Tipos de lanzamiento compatibles	Sistemas operativos compatibles	Persistencia del almacenamiento	Casos de uso
Amazon FSx para Windows File Server	Amazon EC2	Windows	Persistente	Los volúmenes FSx para Windows File Server proporcionan servidores de archivos de Windows completamente administrados que pueden utilizarse para aprovisionar las tareas de Windows que necesitan almacenamiento de archivos persistente, distribuido, compartido y estático. Entre los casos de uso más comunes se incluyen aplicaciones de .NET que pueden requerir carpetas locales, como el almacenamiento persistente para guardar

Volumen de datos	Tipos de lanzamiento compatibles	Sistemas operativos compatibles	Persistencia del almacenamiento	Casos de uso
				los resultados de las aplicaciones. Amazon FSx para Windows File Server ofrece una carpeta local en el contenedor que permite la lectura y escritura de varios contenedores en el mismo sistema de archivos respaldado por un recurso compartido de archivos SMB. Para obtener más información, consulte Uso de volúmenes de FSx para Windows File Server con Amazon ECS.

Volumen de datos	Tipos de lanzamiento compatibles	Sistemas operativos compatibles	Persistencia del almacenamiento	Casos de uso
Volúmenes de Docker	Amazon EC2	Windows, Linux	Persistente	Los volúmenes de Docker son una caracterí stica del tiempo de ejecución de contenedores de Docker que permite a los contenedores conservar los datos mediante el montaje de un directorio desde el sistema de archivos del host. Los controladores de volúmenes de Docker (también se les conocen como complemen tos) se utilizan para integrar los volúmenes de contenedores con sistemas de almacenam iento externos. Los volúmenes de Docker se pueden administr

Volumen de datos	Tipos de lanzamiento compatibles	Sistemas operativos compatibles	Persistencia del almacenamiento	Casos de uso
				ar mediante controladores de terceros o mediante el controlad or integrado `local`. Entre los casos de uso más comunes de los volúmenes de Docker se incluyen proporcionar volúmenes de datos persisten tes o compartir volúmenes en distintas ubicaciones de contenedores diferentes en la misma instancia de contenedo r. Para obtener más informaci ón, consulte Uso de volúmenes de Docker con Amazon ECS.

Volumen de datos	Tipos de lanzamiento compatibles	Sistemas operativos compatibles	Persistencia del almacenamiento	Casos de uso
Montajes de enlace	Fargate, Amazon EC2	Windows, Linux	Efímero	Los montajes de unión consisten en un archivo o directorio del host, por ejemplo, una instancia de Amazon EC2 o AWS Fargate, que se monta en un contenedor. Entre los casos de uso más comunes de los montajes de unión se incluyen compartir un volumen de un contenedor de origen con otros contenedores en la misma tarea o montar un volumen del host o un volumen vacío en uno o varios contenedores. Para obtener más informaci ón, consulte Uso

Volumen de datos	Tipos de lanzamiento compatibles	Sistemas operativos compatibles	Persistencia del almacenamiento	Casos de uso
				de montajes de unión con Amazon ECS.

Uso de volúmenes de Amazon EBS con Amazon ECS

Los volúmenes de Amazon Elastic Block Store (Amazon EBS) proporcionan almacenamiento en bloque de alta disponibilidad, rentable, duradero y de alto rendimiento para las cargas de trabajo con un uso intensivo de datos. Los volúmenes de Amazon EBS se pueden utilizar con las tareas de Amazon ECS para las aplicaciones de alto rendimiento y con un uso intensivo de transacciones.

Durante el lanzamiento de una tarea independiente, puede proporcionar la configuración que se utilizará para adjuntar un volumen de EBS a la tarea. Durante la creación o la actualización del servicio, puede proporcionar la configuración que se utilizará para adjuntar un volumen de EBS por tarea a cada tarea administrada por el servicio de Amazon ECS.

Al proporcionar la configuración del volumen en el momento del lanzamiento y no en la definición de la tarea, se crean definiciones de tareas que no se limitan a un tipo de volumen de datos específico ni a una configuración de volumen de EBS específica. A continuación, puede reutilizar las definiciones de tareas en distintos tiempos de ejecución. Por ejemplo, durante la implementación, puede proporcionar un mayor rendimiento para las cargas de trabajo de producción que para los entornos de preproducción.

Amazon ECS administra los volúmenes de Amazon EBS que se adjuntan a las tareas de Amazon ECS en su nombre. Los volúmenes se pueden cifrar con claves AWS Key Management Service (AWS KMS) para proteger los datos. Puede configurar volúmenes nuevos y vacíos para adjuntarlos o puede utilizar instantáneas para cargar los datos de los volúmenes existentes. Cuando utiliza instantáneas para configurar los volúmenes, puede especificar un valor de `volumeInitializationRate`, en MiB/s, que indica la velocidad la que se obtienen los datos de la instantánea para crear volúmenes que se inicialicen por completo en un período de tiempo predecible. Para obtener más información sobre la inicialización de volúmenes, consulte Inicialización de volúmenes de Amazon EBS en la Guía del usuario de Amazon EBS. Para obtener más información sobre la configuración de los volúmenes de Amazon EBS, consulte Aplazamiento de la configuración del volumen a la hora de lanzamiento en la definición de la tarea de Amazon ECS

y Especificación de la configuración del volumen de Amazon EBS durante la implementación de Amazon ECS.

Para supervisar el rendimiento del volumen, también puede utilizar las métricas de Amazon CloudWatch. Para obtener más información acerca de las métricas de Amazon ECS correspondientes a los volúmenes de Amazon EBS, consulte Métricas de Amazon ECS CloudWatch y Amazon ECS Container Insights metrics.

Se admite adjuntar un volumen de Amazon EBS a una tarea en todas las Regiones de AWS comerciales y de China que admiten Amazon ECS.

Para obtener más información sobre los volúmenes de Amazon EBS, consulte Amazon EBS volumes en la Guía del usuario de Amazon EBS.

Sistemas operativos y tipos de inicializaciones compatibles

En la siguiente tabla se proporcionan las configuraciones de sistema operativo y el tipo de lanzamientos compatibles.

Tipo de lanzamiento	Linux	Windows
Fargate	Los volúmenes de Amazon EBS son compatibles con la versión de la plataforma `1.4.0` o posterior (Linux). Para obtener más informaci ón, consulte Versiones de la plataforma Fargate para Amazon ECS.	No compatible
EC2	Tareas alojadas en instancia s de Linux basadas en Nitro con imágenes de máquina de Amazon (AMI) optimizad as para Amazon ECS. Para más información acerca de los tipos de instancias, consulte Instance types en la Guía del usuario de Amazon EC2.	Tareas alojadas en instancia s de Linux basadas en Nitro con imágenes de máquina de Amazon (AMI) optimizad as para Amazon ECS. Para más información acerca de los tipos de instancias, consulte Instance types en la Guía del usuario de Amazon EC2.

Tipo de lanzamiento	Linux	Windows
	Los volúmenes de Amazon EBS son compatibles con una AMI optimizada para ECS `20231219` o posterior. Para obtener más informaci ón, consulte Recuperación de los metadatos de la AMI optimizada para Amazon ECS.	Los volúmenes de Amazon EBS son compatibles con una AMI optimizada para ECS `20241017` o posterior. Para obtener más informaci ón, consulte Recuperación de los metadatos de la AMI de Windows optimizada para Amazon ECS.

Consideraciones

Al utilizar los volúmenes de Amazon EBS, tenga en cuenta lo siguiente:

- No puede configurar los volúmenes de Amazon EBS para adjuntarlos a las tareas de Amazon ECS de tipo de lanzamiento Fargate en la zona de disponibilidad `use1-az3`.
- No se admite el tipo de volumen magnético (`standard`) de Amazon EBS para las tareas de Fargate. Para obtener más información sobre los volúmenes de Amazon EBS, consulte Amazon EBS volumes en la Guía del usuario de Amazon EC2.
- Se requiere un rol de IAM en la infraestructura de Amazon ECS al crear un servicio o una tarea independiente que consiste en configurar un volumen en el momento de la implementación. Puede adjuntar la política de IAM `AmazonECSInfrastructureRolePolicyForVolumes` administrada por AWS al rol, o puede utilizar la política administrada como guía para crear y adjuntar su propia política con los permisos que cumplan sus necesidades específicas. Para obtener más información, consulte Rol de IAM de infraestructura de Amazon ECS.
- Puede adjuntar como máximo un volumen de Amazon EBS a cada tarea de Amazon ECS y debe ser un volumen nuevo. No se puede adjuntar un volumen existente de Amazon EBS a una tarea. Sin embargo, puede configurar un volumen nuevo de Amazon EBS en el momento de la implementación mediante la instantánea de un volumen existente.
- Para utilizar los volúmenes de Amazon EBS con los servicios de Amazon ECS, el controlador de implementación debe ser “ECS”. Cuando se utiliza este controlador de implementación, se admiten estrategias de implementación azul/verde y continua.
- Para que un contenedor de la tarea escriba en el volumen de Amazon EBS montado, debe ejecutar el contenedor como usuario raíz.

- Amazon ECS agrega automáticamente las etiquetas reservadas `AmazonECSCreated` y `AmazonECSManaged` al volumen adjunto. Si elimina estas etiquetas del volumen, Amazon ECS no podrá administrar el volumen en su nombre. Para obtener más información acerca del etiquetado de volúmenes de Amazon EBS, consulte Tagging Amazon EBS volumes. Para obtener más información acerca del etiquetado de recursos de Amazon ECS, consulte Tagging your Amazon ECS resources.
- No se admite el aprovisionamiento de volúmenes a partir de una instantánea de un volumen de Amazon EBS que contiene particiones.
- Los volúmenes adjuntos a tareas administradas por un servicio no se conservan y siempre se eliminan al terminar la tarea.
- No puede configurar los volúmenes de Amazon EBS para adjuntarlos a las tareas de Amazon ECS que se ejecutan en AWS Outposts.

Aplazamiento de la configuración del volumen a la hora de lanzamiento en la definición de la tarea de Amazon ECS

Para configurar un volumen de Amazon EBS para adjuntarlo a la tarea, debe indicar la configuración del punto de montaje en la definición de la tarea y asignar un nombre al volumen. También debe establecer `configuredAtLaunch` en `true` porque los volúmenes de Amazon EBS no se pueden configurar para adjuntarlos en la definición de la tarea. En su lugar, los volúmenes de Amazon EBS se configuran para que se adjunten durante la implementación.

Para registrar la definición de la tarea mediante la AWS Command Line Interface (AWS CLI), guarde la plantilla como archivo JSON y, luego, pase el archivo como entrada para el comando `register-task-definition`.

Para crear y registrar una definición de tarea mediante la AWS Management Console, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

La definición de la tarea siguiente muestra la sintaxis de los objetos `mountPoints` y `volumes` en la definición de la tarea. Para más información acerca de los parámetros de la definición de la tarea, consulte Parámetros en la definición de tareas de Amazon ECS para el tipo de lanzamiento de Fargate. Para utilizar este ejemplo, reemplace los *`user input placeholders`* con su propia información.

Linux

```
{
```

```
    "family": "mytaskdef",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/nginx/nginx:latest",
            "networkMode": "awsvpc",
           "portMappings": [
                {
                    "name": "nginx-80-tcp",
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp",
                    "appProtocol": "http"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEBSVolume",
                    "containerPath": "/mount/ebs",
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEBSVolume",
            "configuredAtLaunch": true
        }
    ],
    "requiresCompatibilities": [
        "FARGATE", "EC2"
    ],
    "cpu": "1024",
    "memory": "3072",
    "networkMode": "awsvpc"
}
```

Windows

```
{
    "family": "mytaskdef",
     "memory": "4096",
```

```
     "cpu": "2048",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["EC2"]
    "containerDefinitions": [
        {
             "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 443,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEBSVolume",
                    "containerPath": "drive:\ebs",
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEBSVolume",
            "configuredAtLaunch": true
        }
```

```
    ],
    "requiresCompatibilities": [
        "FARGATE", "EC2"
    ],
    "cpu": "1024",
    "memory": "3072",
    "networkMode": "awsvpc"
}
```

`mountPoints`

Tipo: matriz de objetos

Obligatorio: no

Puntos de montaje para los volúmenes de datos del contenedor. Este parámetro asigna a `Volumes` en la API create-container de Docker y la opción `--volume` a docker run.

Los contenedores de Windows pueden montar directorios completos en la misma unidad que `$env:ProgramData`. Los contenedores de Windows no pueden montar directorios en una unidad diferente y los puntos de montaje no se pueden utilizar entre unidades. Debe especificar los puntos de montaje para adjuntar un volumen de Amazon EBS directamente a una tarea de Amazon ECS.

`sourceVolume`

Tipo: cadena

Obligatorio: sí, si se utilizan `mountPoints`.

El nombre del volumen a montar.

`containerPath`

Tipo: cadena

Obligatorio: sí, si se utilizan `mountPoints`.

La ruta del contenedor donde se montará el volumen.

`readOnly`

Tipo: Booleano

Obligatorio: no

Si este valor es `true`, el acceso del contenedor al volumen es de solo lectura. Si este valor es `false`, el contenedor puede escribir en el volumen. El valor predeterminado es `false`.

Para las tareas que se ejecutan en instancias de EC2 que ejecutan el sistema operativo Windows, deje el valor predeterminado de `false`.

`name`

Tipo: cadena

Requerido: no

El nombre del volumen. Se admiten hasta 255 letras (mayúsculas y minúsculas), números, guiones (`-`) y caracteres de subrayado (`_`). Se hace referencia a este nombre en el parámetro `sourceVolume` del objeto `mountPoints` de la definición de contenedor.

`configuredAtLaunch`

Tipo: Booleano

Obligatorio: sí, cuando quiera adjuntar un volumen de EBS directamente a una tarea.

Indica si un volumen se puede configurar durante el lanzamiento. Cuando se establece en `true`, puede configurarlo se ejecuta una tarea independiente o cuando se crea o actualiza un servicio. Cuando se establece en `false`, no podrá proporcionar otra configuración de volumen en la definición de la tarea. Este parámetro debe proporcionarse y establecerse en `true` para configurar un volumen de Amazon EBS para adjuntarlo a una tarea.

Datos cifrados almacenados en volúmenes de Amazon EBS adjuntos a tareas de Amazon ECS

Puede utilizar AWS Key Management Service (AWS KMS) para crear y administrar claves criptográficas que protejan los datos. Los volúmenes de Amazon EBS se cifran en reposo mediante AWS KMS keys. Se cifran los tipos de datos siguientes:

- Datos almacenados en reposo en el volumen
- E/S de disco
- Instantáneas creadas a partir del volumen
- Volúmenes nuevos creados a partir de las instantáneas cifradas

Los volúmenes de Amazon EBS adjuntos a las tareas se pueden cifrar mediante una Clave administrada de AWS predeterminada con el alias `alias/aws/ebs` o una clave simétrica

administrada por el cliente especificada en la configuración del volumen. Las Claves administradas por AWS predeterminadas son exclusivas de cada Cuenta de AWS por Región de AWS y se crean automáticamente. Para crear una clave simétrica administrada por el cliente, siga los pasos que se indican en la sección Creating symmetric encryption KMS keys en AWS KMS Developer Guide.

Puede configurar el cifrado de Amazon EBS de manera predeterminada para que todos los volúmenes nuevos creados y adjuntos a una tarea en una Región de AWS específica se cifren mediante la clave de KMS que configure para su cuenta. Para obtener más información acerca del cifrado de Amazon EBS y el cifrado de manera predeterminada, consulte Cifrado de Amazon EBS en la Guía del usuario de Amazon EBS.

También puede configurar el cifrado a nivel de clúster de Amazon ECS para el almacenamiento administrado de Amazon ECS al crear o actualizar un clúster. El cifrado a nivel de clúster se puede utilizar para cifrar todos los volúmenes de Amazon EBS adjuntos a las tareas que se ejecutan en un clúster específico mediante la clave de KMS especificada en el nivel de clúster. Para obtener más información sobre la configuración del cifrado a nivel de clúster, consulte ManagedStorageConfiguration en la Referencia de la API de Amazon ECS.

Puede configurar cualquier combinación de estas claves. El orden de prioridad de las claves de KMS es el siguiente:

1. La clave de KMS especificada en la configuración del volumen. Al especificar una clave de KMS en la configuración del volumen, esta anula la clave predeterminada de Amazon EBS y cualquier clave de KMS que se especifique a nivel de clúster.
2. La clave de KMS especificada a nivel de clúster. Cuando especifica una clave de KMS para el cifrado a nivel de clúster del almacenamiento administrado de Amazon ECS, esta anula el cifrado predeterminado de Amazon EBS, pero no anula ninguna clave de KMS que se especifique en la configuración del volumen.
3. Cifrado predeterminado de Amazon EBS. El cifrado predeterminado se aplica cuando no se especifica una clave de KMS a nivel de clúster ni una clave en la configuración del volumen. Si activa el cifrado de Amazon EBS de forma predeterminada, el valor predeterminado es la clave de KMS que especifique para el cifrado predeterminado. De lo contrario, el valor predeterminado es la Clave administrada de AWS con el alias `alias/aws/ebs`.

> **Note**
>
> Si establece `encrypted` como `false` en su configuración de volumen, no especifica ninguna clave de KMS a nivel de clúster y activa el cifrado de Amazon EBS de forma

predeterminada, el volumen seguirá estando cifrado con la clave especificada para el cifrado de Amazon EBS predeterminado.

Política de claves de KMS administradas por el cliente

Para cifrar un volumen de EBS adjunto a la tarea mediante una clave administrada por el cliente, debe configurar su política de claves de KMS para garantizar que el rol de IAM que utiliza para la configuración del volumen tenga los permisos necesarios para utilizar la clave. La política de claves debe incluir los permisos `kms:CreateGrant` y `kms:GenerateDataKey*`. Los permisos `kms:ReEncryptTo` y `kms:ReEncryptFrom` son necesarios para cifrar los volúmenes que se crean mediante instantáneas. Si quiere configurar y cifrar solo los volúmenes nuevos y vacíos para adjuntarlos, puede excluir los permisos `kms:ReEncryptTo` y `kms:ReEncryptFrom`.

En el fragmento de código siguiente de JSON se muestran la instrucciones de la política de claves que puede adjuntar a la política de claves de KMS. El uso de estas instrucciones proporcionará acceso para que Amazon ECS utilice la clave para cifrar el volumen de EBS. Para utilizar las instrucciones de la política de ejemplo, sustituya *`user input placeholders`* por su propia información. Como siempre, configure únicamente los permisos que necesite.

```
{
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": "kms:DescribeKey",
      "Resource":"*"
    },
    {
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": [
      "kms:GenerateDataKey*",
      "kms:ReEncryptTo",
      "kms:ReEncryptFrom"
      ],
      "Resource":"*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "aws_account_id",
          "kms:ViaService": "ec2.region.amazonaws.com"
        },
        "ForAnyValue:StringEquals": {
```

```
          "kms:EncryptionContextKeys": "aws:ebs:id"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": "kms:CreateGrant",
      "Resource":"*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "aws_account_id",
          "kms:ViaService": "ec2.region.amazonaws.com"
        },
        "ForAnyValue:StringEquals": {
          "kms:EncryptionContextKeys": "aws:ebs:id"
        },
        "Bool": {
          "kms:GrantIsForAWSResource": true
        }
      }
    }
```

Para más información acerca de los permisos y las políticas de claves, consulte Key policies in AWS KMS y AWS KMS permissions en la Guía para desarrolladores de AWS KMS. Para solucionar los problemas de la conexión de volúmenes de EBS relacionados con los permisos de claves, consulte Solución de problemas de conexión de volúmenes de Amazon EBS a las tareas de Amazon ECS .

Especificación de la configuración del volumen de Amazon EBS durante la implementación de Amazon ECS

Tras registrar una definición de la tarea con el parámetro `configuredAtLaunch` establecido en `true`, puede configurar un volumen de Amazon EBS durante la implementación cuando ejecute una tarea independiente o cuando cree o actualice un servicio.

Para configurar un volumen, puede utilizar las API de Amazon ECS o pasar un archivo JSON como entrada para los comandos siguientes de la AWS CLI:

- `run-task` para ejecutar una tarea de ECS independiente.
- `start-task` para ejecutar una tarea de ECS independiente en una instancia de contenedor específica. Este comando no se aplica a las tareas del tipo de lanzamiento de Fargate.

- `create-service` para crear un nuevo servicio de ECS.
- `update-service` para actualizar un servicio existente.

> **Note**
>
> Para que un contenedor de la tarea escriba en el volumen de Amazon EBS montado, debe ejecutar el contenedor como usuario raíz.

También puede configurar un volumen de Amazon EBS mediante AWS Management Console. Para obtener más información, consulte Ejecución de una aplicación como tarea de Amazon ECS, Creación de una implementación de actualización continua de Amazon ECS y Actualización de un servicio de Amazon ECS.

El siguiente fragmento de código JSON muestra todos los parámetros de un volumen de Amazon EBS que se pueden configurar durante la implementación. Para utilizar estos parámetros en la configuración el volumen, sustituya *`user input placeholders`* por su propia información. Para más información acerca de estos parámetros, consulte Volume configurations.

```
"volumeConfigurations": [
        {
            "name": "ebs-volume",
            "managedEBSVolume": {
                "encrypted": true,
                "kmsKeyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
                "volumeType": "gp3",
                "sizeInGiB": 10,
                "snapshotId": "snap-12345",
                "volumeInitializationRate":100,
                "iops": 3000,
                "throughput": 125,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "key1",
                                "value": "value1"
                            }
```

```
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "arn:aws:iam::1111222333:role/ecsInfrastructureRole",
                 "terminationPolicy": {
                    "deleteOnTermination": true//can't be configured for service-
managed tasks, always true
                },
                "filesystemType": "ext4"
            }
        }
    ]
```

> **Important**
>
> Asegúrese de que la propiedad `volumeName` que indicó en la configuración sea la misma que la propiedad `volumeName` que indicó en la definición de la tarea.

Para obtener información acerca de la comprobación del estado conexión de volúmenes, consulte Solución de problemas de conexión de volúmenes de Amazon EBS a las tareas de Amazon ECS . Para obtener información acerca del rol de AWS Identity and Access Management (IAM) de la infraestructura de Amazon ECS necesaria para adjuntar volúmenes de EBS, consulte Rol de IAM de infraestructura de Amazon ECS.

A continuación, se muestran ejemplos de fragmentos de código de JSON que muestran la configuración de los volúmenes de Amazon EBS. Estos ejemplos se pueden utilizar si se guardan los fragmentos de código en archivos JSON y si se pasan como parámetros (con el parámetro `--cli-input-json file://filename`) para los comandos de la AWS CLI. Reemplace los *`user input placeholders`* con su propia información.

Configuración de un volumen para una tarea independiente

En el siguiente fragmento de código se muestra la sintaxis para configurar los volúmenes de Amazon EBS para adjuntarlos a una tarea independiente. En el siguiente fragmento de código de JSON se muestra la sintaxis para configurar los valores `volumeType`, `sizeInGiB`, `encrypted` y `kmsKeyId`. La configuración indicada en el archivo JSON se utiliza para crear y adjuntar un volumen de EBS a la tarea independiente.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "volumeConfigurations": [
        {
            "name": "datadir",
            "managedEBSVolume": {
                "volumeType": "gp3",
                "sizeInGiB": 100,
                "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
                "encrypted": true,
                "kmsKeyId":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        }
   ]
}
```

Configuración de un volumen durante la creación del servicio

En el siguiente fragmento de código se muestra la sintaxis para configurar los volúmenes de Amazon EBS para adjuntarlos a tareas administradas por un servicio. Los volúmenes se obtienen de la instantánea especificada mediante el parámetro `snapshotId` a una velocidad de 200 MiB/s. La configuración indicada en el archivo JSON se utiliza para crear y adjuntar un volumen de EBS a cada tarea administrada por el servicio.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
              "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
              "snapshotId": "snap-12345",
              "volumeInitializationRate": 200
            }
        }
   ]
}
```

Configuración de un volumen durante la actualización del servicio

El siguiente fragmento de código JSON muestra la sintaxis para actualizar un servicio que anteriormente no tenía los volúmenes de Amazon EBS configurados para adjuntarlos a las tareas. Debe proporcionar el ARN de una revisión de la definición de las tareas con el parámetro `configuredAtLaunch` establecido en `true`. En el siguiente fragmento de código de JSON se muestra la sintaxis para configurar los valores `volumeType`, `sizeInGiB`, `throughput`, `iops` y `filesystemType`. Esta configuración se utiliza para crear y adjuntar un volumen de EBS a cada tarea administrada por el servicio.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "service": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
              "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
               "volumeType": "gp3",
                "sizeInGiB": 100,
                 "iops": 3000,
                "throughput": 125,
                "filesystemType": "ext4"
            }
        }
   ]
}
```

Configuración de un servicio para que deje de utilizar los volúmenes de Amazon EBS

El siguiente fragmento de código JSON muestra la sintaxis para actualizar un servicio para que deje de utilizar los volúmenes de Amazon EBS. Debe proporcionar el ARN de una definición de tareas con el parámetro `configuredAtLaunch` establecido en `false` o una definición de tareas sin el parámetro `configuredAtLaunch`. También debe proporcionar un objeto `volumeConfigurations` vacío.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
```

```
    "service": "mysvc",
    "desiredCount": 2,
    "volumeConfigurations": []
}
```

Política de finalización para volúmenes de Amazon EBS

Cuando termina una tarea de Amazon ECS, dicho servicio utiliza el valor `deleteOnTermination` para determinar si se debe eliminar el volumen de Amazon EBS asociado a la tarea terminada. De manera predeterminada, los volúmenes de EBS asociados a las tareas se eliminan al finalizar la tarea. En el caso de las tareas independientes, puede cambiar esta configuración para conservar el volumen al terminar la tarea.

> **Note**
>
> Los volúmenes adjuntos a tareas administradas por un servicio no se conservan y siempre se eliminan al terminar la tarea.

Etiquetar volúmenes de Amazon EBS

Puede etiquetar los volúmenes de Amazon EBS mediante el objeto `tagSpecifications`. Con el objeto, puede proporcionar sus propias etiquetas y establecer la propagación de las etiquetas a partir de la definición de la tarea o del servicio, en función de si el volumen está adjunto a una tarea independiente o a una tarea de un servicio. La cantidad máxima de etiquetas que puede adjuntarse a un volumen es 50.

> **Important**
>
> Amazon ECS adjunta automáticamente las etiquetas reservadas `AmazonECSCreated` y `AmazonECSManaged` a un volumen de Amazon EBS. Esto significa que puede controlar la conexión de un máximo de 48 etiquetas adicionales a un volumen. Estas etiquetas adicionales pueden ser etiquetas definidas por el usuario, administradas por ECS o propagadas.

Si quiere agregar etiquetas administradas por Amazon ECS al volumen, debe establecer `enableECSManagedTags` en `true` en la llamada de `UpdateService`, `CreateService`, `RunTask`

o `StartTask`. Si activa las etiquetas administradas por Amazon ECS, este servicio etiquetará el volumen automáticamente con información del clúster y del servicio (`aws:ecs:`*`clusterName`* y `aws:ecs:`*`serviceName`*). Para obtener más información acerca del etiquetado de recursos de Amazon ECS, consulte Tagging your Amazon ECS resources.

El siguiente fragmento de código JSON muestra la sintaxis para etiquetar cada volumen de Amazon EBS adjunto a cada tarea de un servicio con una etiqueta definida por el usuario. Para utilizar este ejemplo y crear un servicio, sustituya *`user input placeholders`* por su propia información.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
   "enableECSManagedTags": true,
   "volumeConfigurations": [
        {
            "name": "datadir",
            "managedEBSVolume": {
                "volumeType": "gp3",
                "sizeInGiB": 100,
                 "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "key1",
                                "value": "value1"
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn":"arn:aws:iam:1111222333:role/ecsInfrastructureRole",
                "encrypted": true,
                "kmsKeyId":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        }
   ]
}
```

> **Important**
>
> Debe indicar un tipo de recurso `volume` para etiquetar los volúmenes de Amazon EBS.

Rendimiento de los volúmenes de Amazon EBS para las tareas bajo demanda de Fargate

El IOPS y el rendimiento de los volúmenes de referencia de Amazon EBS disponibles para una tarea bajo demanda de Fargate varían en función del total de unidades de CPU que solicite para la tarea. Si solicita 0,25, 0,5 o 1 unidad de CPU virtual (vCPU) para la tarea de Fargate, recomendamos configurar un volumen SSD de uso general (`gp2` o `gp3`) o un volumen de unidad de disco duro (HDD) (`st1` o `sc1`). Si solicita más de 1 vCPU para la tarea de Fargate, se aplicarán los siguientes límites de rendimiento básico a un volumen de Amazon EBS adjunto a la tarea. Es posible que obtenga temporalmente un rendimiento de EBS superior a los límites siguientes. Sin embargo, recomendamos planificar la carga de trabajo en función de estos límites.

Unidades de CPU solicitadas (en vCPU)	IOPS de referencia de Amazon EBS (E/S de 16 KiB)	Rendimiento de referencia de Amazon EBS (en MiBps, E/S de 128 KiB)	Ancho de banda de referencia (en Mbps)
2	3000	75	360
4	5 000	120	1.150
8	10 000	250	2.300
16	15.000	500	4500

> **Note**
>
> Al configurar un volumen de Amazon EBS para adjuntarlo a una tarea de Fargate, el límite de rendimiento de Amazon EBS de la tarea de Fargate se comparte entre el almacenamiento efímero de la tarea y el volumen adjunto.

Rendimiento de los volúmenes de Amazon EBS para tareas de EC2

Amazon EBS proporciona tipos de volúmenes, que difieren en cuanto a rendimiento y precio, para que pueda adaptar el rendimiento y el costo del almacenamiento a las necesidades de las aplicaciones. Para obtener información sobre el rendimiento, incluidas las IOPS por volumen y el rendimiento por volumen, consulte Tipos de volumen de Amazon EBS en la Guía del usuario de Amazon Elastic Block Store.

Solución de problemas de conexión de volúmenes de Amazon EBS a las tareas de Amazon ECS

Es posible que deba solucionar los problemas de conexión de los volúmenes de Amazon EBS a las tareas de Amazon ECS, o comprobarla.

Compruebe el estado de conexión de volúmenes

Puede utilizar la AWS Management Console para ver el estado de conexión de un volumen de Amazon EBS a una tarea de Amazon ECS. Si la tarea se inicia y se produce un error al adjuntar el volumen, también verá un motivo de estado que podrá utilizar para solucionar el problema. El volumen creado se eliminará y la tarea se detendrá. Para más información acerca de los motivos de estado, consulte Motivos del estado de la conexión de volúmenes de Amazon EBS a las tareas de Amazon ECS.

Para ver el estado de conexión de un volumen y el motivo del estado mediante la consola

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clústeres, elija el clúster en el que se ejecuta la tarea. Se abrirá la página de detalles del clúster.
3. En la página de detalles del clúster, elija la pestaña Tareas.
4. Elija la tarea para ver el estado de conexión de volúmenes. Puede que tenga que utilizar Filtrar estado deseado y elegir Detenido si la tarea que quiere examinar se ha detenido.
5. En la página de detalles de la tarea, elija la pestaña Volúmenes. Podrá ver el estado de la conexión de Amazon EBS en Estado de la conexión. Si el volumen no se puede conectar a la tarea, puede elegir el estado en Estado de la conexión para mostrar la causa del error.

También puede ver el estado de la conexión del volumen de una tarea y el motivo del estado asociado mediante la API DescribeTasks.

Errores en las tareas y los servicios

Es posible que se produzcan errores en el servicio o en las tareas que no sean específicos de los volúmenes de Amazon EBS y que puedan afectar a la conexión de volúmenes. Para obtener más información, consulte

- Mensajes de eventos de servicio
- Códigos de error de tareas detenidas
- Motivos de los errores de la API

Motivos del estado de la conexión de volúmenes de Amazon EBS a las tareas de Amazon ECS

Utilice la siguiente referencia para solucionar los problemas que puedan surgir como motivos de estado en la AWS Management Console cuando configure los volúmenes de Amazon EBS para adjuntarlos a las tareas de Amazon ECS. Para obtener más información sobre la ubicación de estos motivos de estado, consulte Compruebe el estado de conexión de volúmenes.

ECS no pudo suponer el rol de la infraestructura de ECS configurado "arn:aws:iam::*111122223333*:role/*ecsInfrastructureRole*". Compruebe que el rol que se va a transferir tenga la relación de confianza adecuada con Amazon ECS.

Este motivo de estado aparece en los escenarios siguientes.

- Usted proporciona un rol de IAM sin adjuntar la política de confianza necesaria. Amazon ECS no puede acceder al rol de IAM de la infraestructura de Amazon ECS que proporciona si el rol no tiene la política de confianza necesaria. La tarea puede bloquearse en el estado `DEPROVISIONING`. Para más información acerca de la política de confianza necesaria, consulte Rol de IAM de infraestructura de Amazon ECS.
- El usuario de IAM no tiene permiso para transferir el rol de infraestructura de Amazon ECS a Amazon ECS. La tarea puede bloquearse en el estado `DEPROVISIONING`. Para evitar este problema, puede adjuntar el permiso `PassRole` a su usuario. Para obtener más información, consulte Rol de IAM de infraestructura de Amazon ECS.
- Su rol de IAM no tiene los permisos necesarios para adjuntar los volúmenes de Amazon EBS. La tarea puede bloquearse en el estado `DEPROVISIONING`. Para más información acerca de los permisos específicos necesarios para adjuntar los volúmenes de Amazon EBS a las tareas, consulte Rol de IAM de infraestructura de Amazon ECS.

> **Note**
>
> Es posible que también vea este mensaje de error debido a un retraso en la propagación de los roles. Si volver a usar el rol después de esperar unos minutos no soluciona el problema, es posible que la política de confianza del rol este mal configurada.

ECS no pudo configurar el volumen de EBS. Se encontró IdempotentParameterMismatch"; "El token de cliente que ha proporcionado está asociado a un recurso que ya se eliminó. Utilice otro token de cliente".

Los siguientes escenarios de la clave de AWS KMS pueden provocar la aparición de un mensaje `IdempotentParameterMismatch`:

- Indica un ARN, identificador o alias de la clave de KMS que no es válido. En este escenario, puede parecer que la tarea se ha iniciado correctamente, pero finalmente se produce un error porque AWS autentica la clave de KMS de forma asíncrona. Para más información, consulte Amazon EBS encryption en Amazon EC2 User Guide.
- Proporciona una clave administrada por el cliente que carece de los permisos que permiten que el rol de IAM de la infraestructura de Amazon ECS utilice la clave para el cifrado. Para evitar problemas de permisos relacionados con la política de claves, consulte la política de claves de AWS KMS de ejemplo en Data encryption for Amazon EBS volumes.

Puede configurar Amazon EventBridge para enviar eventos de volumen de Amazon EBS y eventos de cambio de estado de las tareas de Amazon ECS a un destino, como los grupos de Amazon CloudWatch. A continuación, puede utilizar estos eventos para identificar el problema específico relacionado con las claves administradas por el cliente que afectó a la conexión del volumen. Para obtener más información, consulte

- ¿Cómo se puede crear un grupo de registro de CloudWatch para utilizarlo como destino de una regla de EventBridge? en AWS re:Post.
- Task state change events.
- Eventos de Amazon EventBridge para Amazon EBS en la Guía del usuario de Amazon EBS.

Se agotó el tiempo de espera de ECS al configurar la conexión del volumen de EBS a la tarea.

Los escenarios siguientes de formato del sistema de archivos dan lugar a este mensaje.

- El formato del sistema de archivos que indique durante la configuración no es compatible con el sistema operativo de la tarea.

- Usted configura la creación de un volumen de Amazon EBS a partir de una instantánea y el formato del sistema de archivos de la instantánea no es compatible con el sistema operativo de la tarea. En el caso de los volúmenes creados a partir de una instantánea, debe especificar el mismo tipo de sistema de archivos que utilizaba el volumen cuando se creó la instantánea.

Puede utilizar los registros del agente del contenedor de Amazon ECS para solucionar este mensaje en la tareas del tipo de lanzamiento de Amazon EC2. Para más información, consulte Amazon ECS log file locations y Amazon ECS log collector.

Uso de volúmenes de Amazon EFS con Amazon ECS

Amazon Elastic File System (Amazon EFS) proporciona almacenamiento de archivos sencillo y escalable para usarlo con tareas de Amazon ECS. Con Amazon EFS, la capacidad de almacenamiento es elástica. Aumenta y disminuye automáticamente a medida que se agregan o eliminan archivos. Las aplicaciones disponen del almacenamiento que necesitan, cuando lo necesitan.

Puede utilizar sistemas de archivos de Amazon EFS con Amazon ECS para exportar los datos del sistema de archivos a través de la flota de instancias de contenedor. De esa forma, las tareas tienen acceso al mismo almacenamiento persistente, con independencia de la instancia en la que aterricen. Las definiciones de tareas deben hacer referencia a montajes de volúmenes en la instancia de contenedor para utilizar el sistema de archivos.

Para ver un tutorial, consulte Configuración de sistemas de archivos de Amazon EFS para Amazon ECS mediante la consola.

Consideraciones

Al utilizar volúmenes de Amazon EFS, tenga en cuenta lo siguiente:

- Para las tareas que utilizan el tipo de lanzamiento de EC2, se agregó compatibilidad con el sistema de archivos de Amazon EFS como vista previa pública en la AMI optimizada para Amazon ECS versión `20191212` con el agente de contenedor versión 1.35.0. Sin embargo, el sistema de archivos de Amazon EFS está disponible con carácter general a partir de la versión `20200319` de la AMI optimizada para Amazon ECS con el agente de contenedor versión 1.38.0, que contenía las características de punto de acceso de Amazon EFS y autorización de IAM. Recomendamos utilizar la AMI optimizada para Amazon ECS versión `20200319` o una posterior para usar estas características. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.

> **Note**
>
> Si crea su propia AMI, debe usar el agente de contenedor 1.38.0 o posterior, `ecs-init` versión 1.38.0-1 o una posterior, y ejecutar los siguientes comandos en su instancia de Amazon EC2 para habilitar el complemento de volúmenes de Amazon ECS. Los comandos dependen de si utiliza Amazon Linux 2 o Amazon Linux como imagen base.
>
> Amazon Linux 2
>
> ```
> yum install amazon-efs-utils
> systemctl enable --now amazon-ecs-volume-plugin
> ```
>
> Amazon Linux
>
> ```
> yum install amazon-efs-utils
> sudo shutdown -r now
> ```

- Para las tareas que están alojadas en Fargate, los sistemas de archivos de Amazon EFS son compatibles con la versión 1.4.0 o una posterior (Linux) de la plataforma. Para obtener más información, consulte Versiones de la plataforma Fargate para Amazon ECS.
- Cuando se utilizan volúmenes de Amazon EFS para tareas alojadas en Fargate, Fargate crea un contenedor supervisor que es responsable de administrar el volumen de Amazon EFS. El contenedor supervisor utiliza una pequeña cantidad de la memoria de la tarea y de CPU. El contenedor supervisor puede verse al consultar la versión 4 del punto de conexión de metadatos de la tarea. Además, está visible en CloudWatch Container Insights (Información de contenedores de CloudWatch) como el nombre del contenedor `aws-fargate-supervisor`. Para más información al utilizar el tipo de lanzamiento de Amazon EC2, consulte Versión 4 del punto de conexión de metadatos de tareas de Amazon ECS. Para más información al utilizar el tipo de lanzamiento de Fargate, consulte Versión 4 del punto de conexión de los metadatos de tareas de Amazon ECS para tareas en Fargate.
- No se admite la utilización de volúmenes de Amazon EFS ni la especificación de un `EFSVolumeConfiguration` en instancias externas.
- Se recomienda establecer el parámetro `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` del archivo de configuración del agente en un valor inferior al predeterminado (aproximadamente 1 hora). Este cambio ayuda a evitar que caduquen las credenciales de montaje de EFS y permite

limpiar los montajes que no están en uso. Para más información, consulte Configuración del agente de contenedor de Amazon ECS.

Uso de puntos de acceso de Amazon EFS

Los puntos de acceso de Amazon EFS son puntos de entrada específicos de la aplicación a un sistema de archivos de EFS para administrar el acceso de las aplicaciones a conjuntos de datos compartidos. Para obtener más información acerca de los puntos de acceso de Amazon EFS y cómo controla el acceso a ellos, consulte Uso de puntos de acceso de Amazon EFS en la Guía del usuario de Amazon Elastic File System.

Los puntos de acceso pueden imponer una identidad de usuario, incluidos los grupos POSIX del usuario, para todas las solicitudes del sistema de archivos que se realizan a través del punto de acceso. Los puntos de acceso también pueden aplicar un directorio raíz diferente para el sistema de archivos. Esto permite que los clientes solo puedan acceder a los datos del directorio especificado o de sus subdirectorios.

> **Note**
>
> Cuando se crea un punto de acceso EFS, especifique una ruta en el sistema de archivos para que sirva como directorio raíz. Cuando se hace referencia al sistema de archivos de EFS con un ID de punto de acceso en la definición de tareas de Amazon ECS, el directorio raíz se debe omitir o establecer en `/`, lo que aplicará la ruta establecida en el punto de acceso de EFS.

Puede utilizar un rol de IAM para la tarea de Amazon ECS y exigir que determinadas aplicaciones utilicen un punto de acceso específico. Al combinar políticas de IAM con puntos de acceso, puede proporcionar acceso seguro a conjuntos de datos específicos para sus aplicaciones. Para obtener más información acerca de cómo utilizar los roles de IAM, consulte Rol de IAM de tarea de Amazon ECS.

Prácticas recomendadas para utilizar los volúmenes de Amazon EFS con Amazon ECS

Tome nota de las siguientes de prácticas recomendadas cuando utilice Amazon EFS con Amazon ECS.

Controles de seguridad y acceso para los volúmenes de Amazon EFS

Amazon EFS ofrece características de control de acceso que puede utilizar para garantizar que los datos almacenados en un sistema de archivos de Amazon EFS estén seguros y solo se pueda acceder a ellos desde las aplicaciones que los necesiten. Para proteger los datos, habilite el cifrado en reposo y en tránsito. Para obtener más información, consulte Cifrado de datos en Amazon EFS en la Guía del usuario de Amazon Elastic File System.

Además del cifrado de datos, también puede utilizar Amazon EFS para restringir el acceso a un sistema de archivos. Hay tres formas de implementar el control de acceso en EFS.

- Grupos de seguridad: con los objetivos de montaje de Amazon EFS, puede configurar un grupo de seguridad que se utilice para permitir y denegar el tráfico de red. Puede configurar el grupo de seguridad adjunto a Amazon EFS para permitir el tráfico de NFS (puerto 2049) desde el grupo de seguridad conectado a las instancias de Amazon ECS o, si utiliza el modo de red `awsvpc`, desde la tarea de Amazon ECS.
- IAM: puede restringir el acceso a un sistema de archivos de Amazon EFS mediante IAM. Cuando se configuran, las tareas de Amazon ECS requieren un rol de IAM para acceder al sistema de archivos a fin de montar un sistema de archivos de EFS. Para más información, consulte Uso de IAM para controlar el acceso a los datos del sistema de archivos en la Guía del usuario de Amazon Elastic File System.

  Las políticas de IAM también pueden imponer condiciones predefinidas, como exigir a un cliente que utilice TLS al conectarse a un sistema de archivos de Amazon EFS. Para más información, consulte Amazon EFS condition keys for clients en Amazon Elastic File System User Guide.
- Puntos de acceso de Amazon EFS: los puntos de acceso de Amazon EFS son puntos de entrada específicos de la aplicación a un sistema de archivos de Amazon EFS. Puede utilizar los puntos de acceso para imponer una identidad de usuario, incluidos los grupos POSIX del usuario, para todas las solicitudes del sistema de archivos que se hacen a través del punto de acceso. Los puntos de acceso también pueden aplicar un directorio raíz diferente para el sistema de archivos. Esto permite que los clientes solo puedan acceder a los datos del directorio especificado o de sus subdirectorios.

Políticas de IAM

Puede utilizar políticas de IAM para controlar el acceso al sistema de archivos de Amazon EFS.

Puede especificar las siguientes acciones para clientes que acceden a un sistema de archivos mediante una política de sistema de archivos.

Acción	Descripción
`elasticfilesystem:ClientMount`	Proporciona acceso de solo lectura a un sistema de archivos.
`elasticfilesystem:ClientWrite`	Proporciona permisos de escritura en un sistema de archivos.
`elasticfilesystem:ClientRootAccess`	Proporciona la capacidad de utilizar el usuario raíz al acceder a un sistema de archivos.

Debe especificar cada acción en una política. Las políticas pueden definirse de las siguientes maneras:

- Basado en cliente: adjunte la política al rol de tareas.

  Defina la opción de autorización de IAM al crear la definición de la tarea.
- Basado en recursos: adjunte la política al sistema de archivos de Amazon EFS.

  Si la política basada en recursos no existe, de forma predeterminada, al crear el sistema de archivos, el acceso se concede a todas las entidades principales (*).

Al configurar la opción de autorización de IAM, combinamos la política asociada al rol de la tarea y la política basada en recursos de Amazon EFS. La opción de autorización de IAM transfiere la identidad de la tarea (el rol de la tarea) con la política a Amazon EFS. Esto permite que la política basada en recursos de Amazon EFS tenga contexto para el rol o usuario de IAM especificado en la política. Si no establece la opción, la política de nivel de recursos de Amazon EFS identifica al usuario de IAM como “anónimo”.

Considere la posibilidad de implementar los tres controles de acceso en un sistema de archivos de Amazon EFS para obtener la máxima seguridad. Por ejemplo, puede configurar el grupo de seguridad adjunto a un punto de montaje de Amazon EFS para que solo permita la entrada de tráfico de NFS desde un grupo de seguridad asociado a la instancia de contenedor o tarea de Amazon ECS. Además, puede configurar que Amazon EFS requiera un rol de IAM para acceder al sistema

de archivos, incluso si la conexión se origina en un grupo de seguridad permitido. Por último, puede utilizar los puntos de acceso de Amazon EFS para aplicar los permisos de usuario de POSIX e indicar los directorios raíz de las aplicaciones.

El siguiente fragmento de código de la definición de tareas muestra cómo montar un sistema de archivos de Amazon EFS mediante un punto de acceso.

```
"volumes": [
    {
      "efsVolumeConfiguration": {
        "fileSystemId": "fs-1234",
        "authorizationConfig": {
          "accessPointId": "fsap-1234",
          "iam": "ENABLED"
        },
        "transitEncryption": "ENABLED",
        "rootDirectory": ""
      },
      "name": "my-filesystem"
    }
]
```

Desempeño de los volúmenes de Amazon EFS

Amazon EFS ofrece dos modos de rendimiento: Uso general y E/S máx. Uso general es adecuado para las aplicaciones sensibles a la latencia, como los sistemas de administración de contenido y las herramientas de CI/CD. Por el contrario, los sistemas de archivos de E/S máx. son adecuados para las cargas de trabajo como el análisis de datos, el procesamiento multimedia y el machine learning. Estas cargas de trabajo deben hacer operaciones en paralelo desde cientos o incluso miles de contenedores y requieren el mayor rendimiento agregado e IOPS posibles. Para más información, consulte Amazon EFS performance modes en Amazon Elastic File System User Guide.

Algunas cargas de trabajo sensibles a la latencia requieren los niveles de E/S más altos proporcionados por el modo de rendimiento de E/S máximo y la latencia más baja proporcionada por el modo de rendimiento de uso general. Para este tipo de carga de trabajo, recomendamos crear varios sistemas de archivos en modo de desempeño de uso general. De ese modo, puede distribuir la carga de trabajo de la aplicación entre todos estos sistemas de archivos, siempre que la carga de trabajo y las aplicaciones lo admitan.

Rendimiento de los volúmenes de Amazon EFS

Todos los sistemas de archivos de Amazon EFS tienen un rendimiento medido asociado que se determina mediante la cantidad del rendimiento aprovisionado para los sistemas de archivos que utilizan el rendimiento aprovisionado o la cantidad de datos almacenados en la clase de almacenamiento EFS Standard o One Zone para los sistemas de archivos que utilizan rendimiento por ráfagas. Para más información, consulte Understanding metered throughput en Amazon Elastic File System User Guide.

El modo de rendimiento predeterminado de los sistemas de archivos de Amazon EFS es el modo por ráfagas. Con el modo por ráfagas, el rendimiento disponible para un sistema de archivos aumenta o disminuye a medida que el sistema de archivos crece. Dado que las cargas de trabajo basadas en archivos suelen tener picos, lo que requiere altos niveles de rendimiento durante periodos y bajos niveles de rendimiento el resto del tiempo, Amazon EFS se ha diseñado para permitir altos niveles de rendimiento durante periodos de tiempo. Además, dado que muchas cargas de trabajo son de lectura intensiva, las operaciones de lectura se miden en una proporción de 1:3 con respecto a otras operaciones de NFS (como la escritura).

Todos los sistemas de archivos de Amazon EFS ofrecen un rendimiento de referencia coherente de 50 MB/s por cada TB de almacenamiento de Amazon EFS Standard o Amazon EFS One Zone. Todos los sistemas de archivos (con independencia de su tamaño), pueden transmitir por ráfagas hasta 100 MB/s. Los sistemas de archivos con más de 1 TB de almacenamiento EFS Standard o EFS One Zone pueden alcanzar los 100 MB/s por cada TB. Como las operaciones de lectura se miden en una proporción de 1:3, puede generar hasta 300 MiB/s por cada TiB de rendimiento de lectura. A medida que agrega los datos al sistema de archivos, el rendimiento máximo disponible para el sistema de archivos se escala lineal y automáticamente con el almacenamiento en la clase de almacenamiento de Amazon EFS Standard. Si es necesario un rendimiento superior al que se puede lograr con la cantidad de datos almacenados, puede configurar el rendimiento aprovisionado en función de la cantidad específica que requiera la carga de trabajo.

El rendimiento del sistema de archivos se comparte entre todas las instancias de Amazon EC2 conectadas a un sistema de archivos. Por ejemplo, un sistema de archivos de 1 TB que puede transmitir por ráfagas hasta 100 MB/s de rendimiento puede generar 100 MB/s desde una sola instancia de Amazon EC2, cada una de las cuales puede generar 10 MB/s. Para obtener más información, consulte Rendimiento de Amazon EFS en la Guía del usuario de Amazon Elastic File System.

Optimización de costos de los volúmenes de Amazon EFS

Amazon EFS simplifica el escalado del almacenamiento. Los sistemas de archivos de Amazon EFS crecen automáticamente a medida que se agregan más datos. Sobre todo con el modo Rendimiento por ráfagas de Amazon EFS, el rendimiento de Amazon EFS se escala cuando aumenta el tamaño del sistema de archivos en la clase de almacenamiento estándar. Para mejorar el rendimiento sin pagar un costo adicional por el rendimiento aprovisionado en un sistema de archivos de EFS, puede compartir un sistema de archivos de Amazon EFS con varias aplicaciones. Con puntos de acceso de Amazon EFS, puede implementar el aislamiento del almacenamiento en sistemas de archivos de Amazon EFS compartidos. De este modo, aunque las aplicaciones sigan compartiendo el mismo sistema de archivos, no podrán acceder a los datos a menos que usted lo autorice.

A medida que sus datos crecen, Amazon EFS ayuda a mover automáticamente los archivos de acceso poco frecuente a una clase de almacenamiento inferior. La clase de almacenamiento Standard-Infrequent Access (IA) de Amazon EFS Standard reduce los costos de almacenamiento de los archivos a los que no se accede todos los días. Todo ello, sin que se afecte a la alta disponibilidad, alta durabilidad, elasticidad y acceso al sistema de archivos POSIX que proporciona Amazon EFS. Para obtener más información, consulte Clases de almacenamiento de EFS en la Guía del usuario de Amazon Elastic File System.

Considere la posibilidad de utilizar las políticas de ciclo de vida de Amazon EFS para ahorrar dinero de forma automática al mover los archivos de acceso poco frecuente al almacenamiento de acceso poco frecuente de Amazon EFS. Para obtener más información, consulte Amazon EFS lifecycle management (Administración del ciclo de vida de Amazon EFS) en la Guía del usuario de Amazon Elastic File System.

Al crear un sistema de archivos de Amazon EFS, puede elegir si Amazon EFS replica los datos en varias zonas de disponibilidad (estándar) o los almacena de forma redundante en una única zona de disponibilidad. La clase de almacenamiento Amazon EFS One Zone puede reducir los costos de almacenamiento en un margen significativo en comparación con las clases de almacenamiento Amazon EFS Standard. Considere la posibilidad de utilizar la clase de almacenamiento Amazon EFS One Zone para las cargas de trabajo que no requieren resiliencia multi-AZ. Para reducir aún más el costo del almacenamiento de Amazon EFS One Zone, mueva los archivos a los que se accede con poca frecuencia a Amazon EFS One Zone de acceso poco frecuente. Para obtener más información, consulte Acceso poco frecuente de Amazon EFS.

Protección de los datos de volúmenes de Amazon EFS

Amazon EFS almacena los datos de manera redundante en varias zonas de disponibilidad de los sistemas de archivos mediante las clases de almacenamiento estándar. Si selecciona las clases de almacenamiento Amazon EFS One Zone, los datos se almacenan de manera redundante en una única zona de disponibilidad. Además, Amazon EFS se ha diseñado para ofrecer una durabilidad del 99,999999999 % (11 9) durante un año concreto.

Como ocurre con cualquier entorno, se recomienda disponer de una copia de seguridad y crear medidas de protección contra la eliminación accidental. En el caso de los datos de Amazon EFS, esa práctica recomendada incluye una copia de seguridad que funcione y se pruebe periódicamente con AWS Backup. Los sistemas de archivos que utilizan las clases de almacenamiento Amazon EFS One Zone están configurados para hacer copias de seguridad automáticas de los archivos de manera predeterminada al crear el sistema de archivos, a menos que decida deshabilitar esta funcionalidad. Para obtener más información, consulte Realización de copias de seguridad de sistemas de archivos EFS en la Guía del usuario de Amazon Elastic File System.

Especificación de un sistema de archivos de Amazon EFS en la definición de tareas de Amazon ECS

Para usar volúmenes del sistema de archivos de Amazon EFS para sus contenedores, debe especificar las configuraciones de volumen y punto de montaje en su definición de tarea. El siguiente fragmento de JSON de definición de tareas muestra la sintaxis de los objetos `volumes` y `mountPoints` para un contenedor.

```
{
    "containerDefinitions": [
        {
            "name": "container-using-efs",
            "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
            "entryPoint": [
                "sh",
                "-c"
            ],
            "command": [
                "ls -la /mount/efs"
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEfsVolume",
                    "containerPath": "/mount/efs",
                    "readOnly": true
```

```
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEfsVolume",
            "efsVolumeConfiguration": {
                "fileSystemId": "fs-1234",
                "rootDirectory": "/path/to/my/data",
                "transitEncryption": "ENABLED",
                "transitEncryptionPort": integer,
                "authorizationConfig": {
                    "accessPointId": "fsap-1234",
                    "iam": "ENABLED"
                }
            }
        }
    ]
}
```

`efsVolumeConfiguration`

Tipo: objeto

Obligatorio: no

Este parámetro se especifica cuando se usan volúmenes de Amazon EFS.

`fileSystemId`

Tipo: cadena

Obligatorio: sí

El ID del sistema de archivos de Amazon EFS que se va a usar.

`rootDirectory`

Tipo: cadena

Obligatorio: no

Directorio del sistema de archivos de Amazon EFS que se va a montar como directorio raíz dentro del host. Si se omite este parámetro, se utiliza la raíz del volumen de Amazon EFS. Si se especifica `/`, se obtiene el mismo efecto que si se omite este parámetro.

> **Important**
>
> Si se especifica un punto de acceso de EFS en el `authorizationConfig`, se debe omitir el parámetro del directorio raíz o establecerlo en `/`, lo que aplicará la ruta establecida en el punto de acceso de EFS.

`transitEncryption`

Tipo: cadena

Valores válidos: `ENABLED` | `DISABLED`

Obligatorio: no

Especifica si se habilita el cifrado para los datos en tránsito de Amazon EFS entre el host de Amazon ECS y el servidor de Amazon EFS. Si se utiliza la autorización de IAM en Amazon EFS, el cifrado en tránsito debe estar habilitado. Si se omite este parámetro, se usa el valor predeterminado de `DISABLED`. Para obtener más información, consulte Cifrado de datos en tránsito en la Guía del usuario de Amazon Elastic File System.

`transitEncryptionPort`

Tipo: entero

Obligatorio: no

El puerto que se utilizará al enviar datos cifrados entre el host de Amazon ECS y el servidor de Amazon EFS. Si no se especifica un puerto de cifrado en tránsito, se emplea la estrategia de selección de puertos que utiliza el ayudante de montaje de Amazon EFS. Para obtener más información, consulte Ayudante de montaje de EFS en la Guía del usuario de Amazon Elastic File System.

`authorizationConfig`

Tipo: objeto

Obligatorio: no

Los detalles de configuración de autorización en el sistema de archivos de Amazon EFS.

`accessPointId`

Tipo: cadena

Requerido: no

ID de punto de acceso que se va a utilizar. Si se especifica un punto de acceso, el valor del directorio raíz en `efsVolumeConfiguration` se debe omitir o establecer en `/`, lo que aplica la ruta establecida en el punto de acceso EFS. Si se utiliza un punto de acceso, el cifrado de tránsito debe estar habilitado en el `EFSVolumeConfiguration`. Para obtener más información, consulte Trabajo con puntos de acceso de Amazon EFS en la Guía del usuario de Amazon Elastic File System.

`iam`

Tipo: cadena

Valores válidos: `ENABLED` | `DISABLED`

Obligatorio: no

Especifica si utilizar el rol de IAM de tarea de Amazon ECS definido en una definición de tareas al montar el sistema de archivos de Amazon EFS. Si está habilitado, el cifrado de tránsito debe estar habilitado en el `EFSVolumeConfiguration`. Si se omite este parámetro, se usa el valor predeterminado de `DISABLED`. Para obtener más información, consulte Roles de IAM para las tareas.

Configuración de sistemas de archivos de Amazon EFS para Amazon ECS mediante la consola

Obtenga información sobre cómo utilizar los sistemas de archivos de Amazon Elastic File System (Amazon EFS) con Amazon ECS.

Paso 1: Crear un clúster de Amazon ECS

Siga estos pasos para crear un clúster de Amazon ECS.

Para crear un nuevo clúster (consola de Amazon ECS)

Antes de empezar, asigne el permiso de IAM correspondiente. Para obtener más información, consulte the section called “Ejemplos de clústeres de Amazon ECS”.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, seleccione la región a utilizar.
3. En el panel de navegación, seleccione Clusters (Clústeres).
4. En la página Clusters (Clústeres), elija Create Cluster (Crear clúster).
5. En Configuración de clúster, para Nombre del clúster, ingrese `EFS-tutorial`.
6. (Opcional) Para cambiar la VPC y las subredes donde se inician sus tareas y servicios, en Networking (Redes), realice cualquiera de las siguientes operaciones:
   - Para eliminar una subred, en Subnets (Subredes), elija Xpara cada subred que desea eliminar.
   - Para cambiar a una VPC distinta de la VPC predeterminada, enVPC, elija una VPC existente y, a continuación, en Subnets (Subredes), seleccione cada subred.
7. Para agregar instancias de Amazon EC2 al clúster, expanda Infraestructura y, a continuación, seleccione Instancias de Amazon EC2. A continuación, configure el grupo de Auto Scaling que actúa como proveedor de capacidad:
   - Para crear un grupo de Auto Scaling, desde Auto Scaling group (ASG) (Grupo de Auto Scaling), seleccione Create new group (Crear nuevo grupo) y, a continuación, proporcione los siguientes detalles sobre el grupo:
     - En Sistema operativo/arquitectura, elija Amazon Linux 2.
     - En EC2 instance type (Tipo de instancia EC2), elija `t2.micro`.

       Para Par de clave de SSH, elija el par que demuestre su identidad cuando se conecta a la instancia.
     - En Capacidad, escriba 1.
8. Seleccione Crear.

Paso 2: Crear un grupo de seguridad para las instancias de Amazon EC2 y el sistema de archivos de Amazon EFS

En este paso, se crea un grupo de seguridad para las instancias de Amazon EC2 que permiten el tráfico de red entrante en el puerto 80 y para el sistema de archivos de Amazon EFS que permite obtener acceso de entrada desde las instancias de contenedor.

Cree un grupo de seguridad para las instancias de Amazon EC2 con las siguientes opciones:

- Nombre del grupo de seguridad: un nombre único para el grupo de seguridad.
- VPC: la VPC que identificó anteriormente para el clúster.
- Regla de entrada
  - Tipo: HTTP
  - Fuente: 0.0.0.0/0.

Cree un grupo de seguridad para el sistema de archivos de Amazon EFS con las siguientes opciones:

- Nombre del grupo de seguridad: un nombre único para el grupo de seguridad. Por ejemplo, `EFS-access-for-sg-dc025fa2`.
- VPC: la VPC que identificó anteriormente para el clúster.
- Regla de entrada
  - Tipo: NFS
  - Fuente: personalizada con el ID del grupo de seguridad que creó para las instancias.

Para obtener información sobre cómo crear un grupo de seguridad, consulte Creación de un grupo de seguridad para instancias de Amazon EC2 en la Guía del usuario de Amazon EC2.

Paso 3: Crear un sistema de archivos de Amazon EFS

En este paso, se crea un sistema de archivos de Amazon EFS.

Para crear un sistema de archivos de Amazon EFS para tareas de Amazon ECS.

1. Abra la consola de Amazon Elastic File System en https://console.aws.amazon.com/efs/.
2. Seleccione Create file system (Crear sistema de archivos).
3. Introduzca un nombre para el sistema de archivos y, a continuación, elija la VPC en la que están alojadas las instancias de contenedor. De forma predeterminada, cada subred de la VPC especificada recibe un destino de montaje que utiliza el grupo de seguridad predeterminado para dicha VPC. A continuación, seleccione Personalizar.

> **Note**
>
> Este tutorial asume que su sistema de archivos de Amazon EFS, el clúster de Amazon ECS, las instancias de contenedor y las tareas deben estar en la misma VPC. Para

más información sobre cómo montar un sistema de archivos desde una VPC diferente, consulte Walkthrough: Mount a file system from a different VPC en la Guía del usuario de Amazon EFS.

4. En la página de Configuración del sistema de archivos, configure los ajustes opcionales y, a continuación, en Configuración de rendimiento, elija el modo de rendimiento Por ráfagas para el sistema de archivos. Una vez que haya configurado los ajustes, seleccione Siguiente.

   a. Añada etiquetas para su sistema de archivos. Este paso es opcional. Por ejemplo, es posible especificar un nombre único para el sistema de archivos al escribirlo en la columna Value situada junto a la clave Name.

   b. (Opcional) Habilite la administración del ciclo de vida para ahorrar dinero en el almacenamiento al que se accede con poca frecuencia. Para obtener más información, consulte Administración del ciclo de vida de EFS en la Guía del usuario de Amazon Elastic File System.

   c. (Opcional) Habilite el cifrado. Seleccione la casilla de verificación para habilitar el cifrado del sistema de archivos de Amazon EFS en reposo.

5. En la página Acceso a la red, en Montar objetivos, reemplace la configuración del grupo de seguridad existente para cada zona de disponibilidad con el grupo de seguridad que creó para el sistema de archivos en Paso 2: Crear un grupo de seguridad para las instancias de Amazon EC2 y el sistema de archivos de Amazon EFS y, a continuación, seleccione Siguiente.

6. No necesita configurar la Política del sistema de archivos para este tutorial, por lo que puede omitir la sección seleccionando Siguiente.

7. Revise las opciones del sistema de archivos y elija Crear para completar el proceso.

8. Desde la pantalla Sistemas de archivos, registre el ID del sistema de archivos. En el siguiente paso, hará referencia a este valor en la definición de tareas de Amazon ECS.

Paso 4: Agregar contenido al sistema de archivos de Amazon EFS

En este paso, se va a montar el sistema de archivos de Amazon EFS en una instancia de Amazon EC2 y a agregar contenido en ella. Esto se hace para realizar pruebas en este tutorial e ilustrar la naturaleza persistente de los datos. Por lo general, cuando se utiliza estas características, se cuenta con una aplicación u otro método de escritura de datos en el sistema de archivos de Amazon EFS.

Para crear una instancia de Amazon EC2 y montar el sistema de archivos de Amazon EFS

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. Elija Iniciar instancia.
3. En Imágenes de aplicaciones y sistema operativo (Imagen de máquina de Amazon), seleccione la AMI de Linux 2 de Amazon (HVM).
4. En Tipo de instancia, mantenga el tipo de instancia predeterminado, `t2.micro`.
5. En Par de claves (inicio de sesión), seleccione un par de claves para el acceso SSH a la instancia.
6. En Configuración de red, seleccione la VPC que especificó en el sistema de archivos de Amazon EFS y el clúster de Amazon ECS. Seleccione una subred y el grupo de seguridad de la instancia que se creó en Paso 2: Crear un grupo de seguridad para las instancias de Amazon EC2 y el sistema de archivos de Amazon EFS. Configure el grupo de seguridad de la instancia. Asegúrese de que Asignar automáticamente IP pública esté habilitado.
7. En Configurar almacenamiento, pulse el botón Editar para los sistemas de archivos y, a continuación, elija EFS. Seleccione el sistema de archivos que creó en Paso 3: Crear un sistema de archivos de Amazon EFS. Si lo desea, puede cambiar el punto de montaje o dejar el valor predeterminado.

   > **Important**
   >
   > Debe seleccionar una subred antes de poder agregar un sistema de archivos a la instancia.

8. Desactive Crear y adjuntar grupos de seguridad automáticamente. Deje seleccionada la otra casilla de verificación. Elija Agregar sistema de archivos compartidos.
9. En Advanced Details (Detalles avanzados), asegúrese de que el script de datos del usuario se rellene automáticamente con los pasos de montaje del sistema de archivos de Amazon EFS.
10. En Resumen, asegúrese de que el Número de instancias sea 1. Seleccione Iniciar instancia.
11. En la página Lanzar una instancia, seleccione Ver todas las instancias para ver el estado de las instancias. Al principio, el estado del Estado de instancia es `PENDING`. Una vez que el estado cambie a `RUNNING` y la instancia supere todas las comprobaciones de estado, la instancia estará lista para su uso.

Ahora, se conecta a la instancia de Amazon EC2 y agrega contenido al sistema de archivos de Amazon EFS.

Para conectarse a la instancia de Amazon EC2 y agregar contenido al sistema de archivos de Amazon EFS

1. SSH a la instancia de Amazon EC2 que creó. Para obtener más información, consulte Conexión a la instancia de Linux mediante SSH en la Guía del usuario de Amazon EC2.
2. Desde la ventana del terminal, ejecute el comando df -T para comprobar que el sistema de archivos de Amazon EFS esté montado. En el siguiente resultado, hemos resaltado el montaje del sistema de archivos de Amazon EFS.

```
$ df -T
Filesystem     Type            1K-blocks    Used        Available Use% Mounted on
devtmpfs       devtmpfs           485468       0           485468   0% /dev
tmpfs          tmpfs              503480       0           503480   0% /dev/shm
tmpfs          tmpfs              503480     424           503056   1% /run
tmpfs          tmpfs              503480       0           503480   0% /sys/fs/
cgroup
/dev/xvda1     xfs               8376300 1310952          7065348  16% /
127.0.0.1:/    nfs4     9007199254739968       0 9007199254739968   0% /mnt/efs/fs1
tmpfs          tmpfs              100700       0           100700   0% /run/
user/1000
```

3. Vaya al directorio en el que está montado el sistema de archivos de Amazon EFS. En el ejemplo anterior, es `/mnt/efs/fs1`.
4. Cree un archivo denominado `index.html` con el siguiente contenido:

```
<html>
    <body>
        <h1>It Works!</h1>
        <p>You are using an Amazon EFS file system for persistent container
 storage.</p>
    </body>
</html>
```

Paso 5: Crear una definición de tarea

La siguiente definición de tarea crea un volumen de datos llamado `efs-html`. El contenedor `nginx` monta el volumen de datos host en raíz de NGINX, `/usr/share/nginx/html`.

Para crear una nueva definición de tareas utilizando la consola de Amazon ECS

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, elija Task Definitions (Definiciones de tareas).
3. Elija Create new task definition (Crear nueva definición de tarea) y Create new task definition with JSON (Crear nueva definición de tarea con JSON).
4. En el cuadro editor de JSON, copie y pegue el siguiente texto JSON, sustituyendo `fileSystemId` por el ID del sistema de archivos de Amazon EFS.

```
{
    "containerDefinitions": [
        {
            "memory": 128,
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "mountPoints": [
                {
                    "containerPath": "/usr/share/nginx/html",
                    "sourceVolume": "efs-html"
                }
            ],
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest"
        }
    ],
    "volumes": [
        {
            "name": "efs-html",
            "efsVolumeConfiguration": {
                "fileSystemId": "fs-1324abcd",
                "transitEncryption": "ENABLED"
            }
        }
    ],
    "family": "efs-tutorial",
    "executionRoleArn":"arn:aws:iam::111122223333:role/ecsTaskExecutionRole"
```

```
}
```

> **Note**
>
> Para el rol de IAM de ejecución de tareas de Amazon ECS no es necesario ningún permiso específico relacionado con Amazon EFS para montar un sistema de archivos de Amazon EFS. De manera predeterminada, si no existe la política basada en recursos, al crear el sistema de archivos, el acceso se concede a todas las entidades principales (*). El rol de tarea de Amazon ECS solo es necesario si la “autorización de IAM de EFS” está habilitada en la definición de tareas de Amazon ECS. Cuando está habilitada, la identidad del rol de la tarea debe poder acceder al sistema de archivos de Amazon EFS en la política basada en recursos de Amazon EFS, y el acceso anónimo debe estar inhabilitado.

5. Seleccione Crear.

Paso 6: Ejecutar una tarea y ver los resultados

Ahora que se creó el sistema de archivos de Amazon EFS y hay contenido web para que el contenedor de NGINX lo sirva, puede ejecutar una tarea mediante la definición de tareas que creó. Los servidores web de NGINX dan servicio a su sencilla página HTML. Si actualiza el contenido en el sistema de archivos de Amazon EFS, esos cambios se propagan a cualquier contenedor que también haya montado ese sistema de archivos.

La tarea se ejecuta en la subred que definió para el clúster.

Para ejecutar una tarea y ver los resultados mediante la consola

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), seleccione el clúster que va a ejecutar la tarea independiente.

   Determine el recurso desde el que lanza el servicio.

Para iniciar un servicio desde	Pasos	
Clústeres	a. En la página Clusters (Clústeres), seleccione el	

Para iniciar un servicio desde	Pasos
	clúster que va a crear el servicio. b. En la pestaña Tasks (Tareas), elija Run new task (Ejecutar nueva tarea).
Tipo de lanzamiento	a. En la página Task (Tarea), seleccione la definición de tarea. b. Si hay más de una revisión, selecciónela. c. Elija Create (Crear), Run task (Ejecutar tarea).

3. (Opcional) Elija cómo se distribuye la tarea programada en su infraestructura de clúster. Expanda Compute configuration (Configuración de computación) y, a continuación, haga lo siguiente:

Método de distribución	Pasos
Tipo de lanzamiento	a. En la sección Compute options (Opciones de computación), seleccion e Lauch type (Tipo de lanzamiento). b. En Tipo de lanzamiento, elija EC2.

4. En Application type (Tipo de aplicación), elija Task (Tarea).
5. En Definición de tarea, elija la definición de tarea `efs-tutorial` que creó anteriormente.
6. En Tareas deseadas, ingrese 1.
7. Seleccione Crear.

8. En la página Clúster, elija Infraestructura.
9. En Instancias de contenedor, elija la instancia de contenedor a la que se va a conectar.
10. En la página Instancia de contenedor, en Redes registre la IP pública para la instancia.
11. Abra un navegador e ingrese la dirección IP pública. Debería ver el siguiente mensaje:

```
It works!
You are using an Amazon EFS file system for persistent container storage.
```

> **Note**
>
> Si no ve el mensaje, asegúrese de que el grupo de seguridad de la instancia de contenedor permita el tráfico de red entrante en el puerto 80 y que el grupo de seguridad del sistema de archivos permita el acceso entrante desde la instancia de contenedor.

Uso de volúmenes de FSx para Windows File Server con Amazon ECS

FSx para Windows File Server proporciona servidores de archivos de Windows completamente administrados y respaldados por un sistema de archivos de Windows. Cuando se utiliza FSx for Windows File Server junto con ECS, las tareas de Windows se pueden aprovisionar con almacenamiento de archivos estático, compartido, distribuido y persistente. Para obtener más información, consulte ¿Qué es FSx for Windows File Server?.

> **Note**
>
> Las instancias EC2 que utilizan la AMI completa de Windows Server 2016 optimizada para Amazon ECS no admiten volúmenes de tareas de ECS de FSx for Windows File Server. No se pueden utilizar los volúmenes de FSx para Windows File Server en contenedores de Windows en la configuración de Fargate. En su lugar, puede modificar los contenedores para montarlos al inicio.

Puede utilizar FSx for Windows File Server para implementar cargas de trabajo de Windows que requieren acceso a almacenamiento externo compartido, almacenamiento regional de alta disponibilidad o almacenamiento de alto rendimiento. Puede montar uno o varios volúmenes del sistema de archivos de FSx para Windows File Server en un contenedor de Amazon ECS que se ejecute en una instancia de Windows de Amazon ECS. Puede compartir volúmenes del sistema de

archivos de FSx para Windows File Server entre varios contenedores de Amazon ECS en una sola tarea de Amazon ECS.

Para habilitar el uso de FSx for Windows File Server con ECS, incluya el ID del sistema de archivos FSx for Windows File Server y la información relacionada en una definición de tareas. Esto se muestra en el siguiente fragmento JSON de definición de tareas de ejemplo. Antes de crear y ejecutar una definición de tareas, necesita lo siguiente.

- Una instancia de EC2 de Windows ECS unida a un dominio válido. Puede alojarla AWS Directory Service for Microsoft Active Directory, Active Directory en las instalaciones o Active Directory con alojamiento propio en Amazon EC2.
- Un parámetro de Systems Manager o secreto de AWS Secrets Manager que contenga las credenciales que se utilizan para unir el dominio de Active Directory y asociar el sistema de archivos de FSx para Windows File Server. Las credenciales son el nombre y la contraseña que especificó al crear el Active Directory.

Para ver un tutorial relacionado, consulte Obtenga información sobre cómo configurar FSx para sistemas de archivos de Windows File Server para Amazon ECS..

Consideraciones

Tenga en cuenta lo siguiente al utilizar volúmenes de FSx for Windows File Server:

- FSx for Windows File Server con Amazon ECS solo admite instancias de Amazon EC2 de Windows. No se admiten instancias Linux Amazon EC2.
- FSx for Windows File Server con Amazon ECS no admite AWS Fargate.
- FSx for Windows File Server con Amazon ECS y modo de red `awsvpc` requiere la versión `1.54.0` o posterior del agente de contenedor.
- Para una tarea de Amazon ECS, se pueden utilizar 23 letras de unidad, como máximo. A cada tarea con un volumen de FSx for Windows File Server se le asigna una letra de unidad.
- De forma predeterminada, el tiempo de limpieza de los recursos de tareas es de tres horas una vez finalizada la tarea. Incluso si no hay tareas que lo utilicen, una asignación de archivos creado por una tarea persiste durante tres horas. Para configurar el tiempo de limpieza predeterminado, se puede usar la variable de entorno de Amazon ECS `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS.

- Por lo general, las tareas solo se ejecutan en la misma VPC que el sistema de archivos FSx for Windows File Server. Sin embargo, es posible que exista compatibilidad entre VPC si hay una conectividad de red establecida entre la VPC del clúster de Amazon ECS y el sistema de archivos FSx for Windows File Server mediante interconexión de VPC.
- Para controlar el acceso a un sistema de archivos FSx for Windows File Server en el nivel de red, se configuran los grupos de seguridad de VPC. Solo las tareas alojadas en instancias de EC2 unidas al dominio de Active Directory con grupos de seguridad de Active Directory correctamente configurados podrán acceder al uso compartido de archivos de FSx para Windows File Server. Si los grupos de seguridad están mal configurados, Amazon ECS no lanza la tarea y se muestra el mensaje de error siguiente: “`unable to mount file system` *`fs-id`*”.
- FSx for Windows File Server está integrado con AWS Identity and Access Management (IAM) para controlar las acciones que los usuarios y grupos de IAM pueden realizar en los recursos específicos de FSx for Windows File Server. Con autorización de cliente, los clientes pueden definir roles de IAM que permiten o deniegan el acceso a sistemas de archivos FSx for Windows File Server específicos, que opcionalmente requieren acceso de solo lectura y que opcionalmente permiten o no permiten el acceso raíz al sistema de archivos desde el cliente. Para obtener más información, consulte Seguridad en la Guía del usuario de Windows para Amazon FSx.

Prácticas recomendadas para el uso de FSx para Windows File Server con Amazon ECS

Tome nota de las siguientes de prácticas recomendadas cuando utilice FSx para Windows File Server con Amazon ECS.

Controles de seguridad y acceso de FSx para Windows File Server

FSx para Windows File Server ofrece las siguientes características de control de acceso que puede utilizar para garantizar que los datos almacenados en un sistema de archivos de FSx para Windows File Server estén seguros y solo se pueda acceder a ellos desde las aplicaciones que los necesiten.

Cifrado de datos para volúmenes de FSx para Windows File Server

FSx para Windows File Server admite dos formas de cifrado para sistemas de archivos. Estas son el cifrado de datos en tránsito y cifrado en reposo. El cifrado de los datos en tránsito se admite en los recursos compartidos de archivos que están asignados en una instancia de contenedor compatible con el protocolo SMB 3.0 o posterior. El cifrado de los datos en reposo se activa de forma automática al crear un sistema de archivos Amazon FSx. Amazon FSx cifra de manera automática los datos en tránsito con el cifrado SMB, cuando el usuario accede al sistema de archivos, sin necesidad de

modificar las aplicaciones. Para más información, consulte Data encryption in Amazon FSx en la Guía del usuario de Amazon FSx para Windows File Server.

Uso de las ACL de Windows para el control de acceso a carpetas

La instancia de Amazon EC2 de Windows accede a los recursos compartidos de archivos de Amazon FSx con las credenciales de Active Directory. Utiliza las listas de control de acceso (ACL) estándar de Windows para tener un control de acceso detallado a archivos y carpetas. Puede crear varias credenciales, cada una para una carpeta específica del recurso compartido que se asigne a una tarea específica.

En el ejemplo siguiente, la tarea tiene acceso a la carpeta `App01` mediante una credencial guardada en Secrets Manager. Su nombre de recurso de Amazon (ARN) es `1234`.

```
"rootDirectory": "\\path\\to\\my\\data\App01",
"credentialsParameter": "arn-1234",
"domain": "corp.fullyqualified.com",
```

En otro ejemplo, una tarea tiene acceso a la carpeta `App02` mediante una credencial guardada en Secrets Manager. Su ARN es `6789`.

```
"rootDirectory": "\\path\\to\\my\\data\App02",
"credentialsParameter": "arn-6789",
"domain": "corp.fullyqualified.com",
```

Especificación de un sistema de archivos de FSx para Windows File Server en la definición de tareas de Amazon ECS

Para usar volúmenes del sistema de archivos de FSx for Windows File Server para sus contenedores, especifique las configuraciones de volumen y punto de montaje en la definición de tarea. El siguiente fragmento de JSON de definición de tareas muestra la sintaxis de los objetos `volumes` y `mountPoints` para un contenedor.

```
{
    "containerDefinitions": [
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [],
```

```
            "command": ["New-Item -Path C:\\fsx-windows-dir\\index.html -ItemType file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>It Works!</h2> <p>You are using Amazon
 FSx for Windows File Server file system for persistent container storage.</p>' -
Force"],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": false,
            "name": "container1",
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ]
        },
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [
                {
                    "hostPort": 443,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": ["Remove-Item -Recurse C:\\inetpub\\wwwroot\\* -Force; Start-
Sleep -Seconds 120; Move-Item -Path C:\\fsx-windows-dir\\index.html -Destination C:\
\inetpub\\wwwroot\\index.html -Force; C:\\ServiceMonitor.exe w3svc"],
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ],
            "cpu": 512,
            "memory": 256,
```

```
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": true,
            "name": "container2"
        }
    ],
    "family": "fsx-windows",
    "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
    "volumes": [
        {
            "name": "fsx-windows-dir",
            "fsxWindowsFileServerVolumeConfiguration": {
                "fileSystemId": "fs-0eeb5730b2EXAMPLE",
                "authorizationConfig": {
                    "domain": "example.com",
                    "credentialsParameter": "arn:arn-1234"
                },
                "rootDirectory": "share"
            }
        }
    ]
}
```

`FSxWindowsFileServerVolumeConfiguration`

Tipo: objeto

Obligatorio: no

Este parámetro se especifica cuando se utiliza el sistema de archivos FSx for Windows File Server para el almacenamiento de tareas.

`fileSystemId`

Tipo: cadena

Obligatorio: sí

ID del sistema de archivos FSx for Windows File Server que se va a utilizar.

`rootDirectory`

Tipo: cadena

Obligatorio: sí

Directorio dentro del sistema de archivos de FSx for Windows File Server que se va a montar como directorio raíz dentro del host.

`authorizationConfig`

`credentialsParameter`

Tipo: cadena

Obligatorio: sí

Opciones de credenciales de autorización:

- Nombre de recurso de Amazon (ARN) de un secreto de Secrets Manager.
- Nombre de recurso de Amazon (ARN) de un parámetro de Systems Manager.

`domain`

Tipo: cadena

Obligatorio: sí

Nombre de dominio completo alojado por un directorio de AWS Directory Service for Microsoft Active Directory(AWS Managed Microsoft AD) o un directorio de Active Directory de EC2 con alojamiento propio.

Métodos para almacenar credenciales de volúmenes de FSx para Windows File Server

Existen dos métodos diferentes de almacenamiento de credenciales que se utilizan con el parámetro de credenciales.

- AWS Secrets Manager secret

  Esta credencial se puede crear en la consola de AWS Secrets Manager en la categoría Other type of secret (Otro tipo de secreto). Agregue una fila para cada par clave/valor, nombre de usuario/administrador y contraseña/*contraseña*.

- Parámetro de Systems Manager

  Para crear esta credencial en la consola de parámetros de Systems Manager, ingrese texto en el formulario que se muestra en el siguiente fragmento de código de ejemplo.

```
{
  "username": "admin",
  "password": "password"
```

```
}
```

El `credentialsParameter` del parámetro `FSxWindowsFileServerVolumeConfiguration` de la definición de tarea contiene el ARN secreto o el ARN del parámetro de Systems Manager. Para obtener más información, consulte¿Qué es AWS Secrets Manager? en la Guía del usuario de Secrets Manager y Parameter Store de Systems Manager en la Guía del usuario de Systems Manager.

Obtenga información sobre cómo configurar FSx para sistemas de archivos de Windows File Server para Amazon ECS.

Obtenga información sobre cómo lanzar una instancia de Windows optimizada para Amazon ECS que aloje un sistema de archivos de FSx para Windows File Server y contenedores que puedan acceder al sistema de archivos. Para hacerlo, primero debe crear un Microsoft Active Directory AWS Directory Service administrado por AWS. A continuación, cree un sistema de archivos de FSx para Windows File Server y un clúster con una instancia de Amazon EC2 y una definición de tareas. Configure la definición de tareas de los contenedores para que utilicen el sistema de archivos FSx for Windows File Server. Por último, pruebe el sistema de archivos.

Cada vez que se lanza o se elimina el sistema de archivos Active Directory o FSx for Windows File Server, el proceso tarda entre 20 y 45 minutos. Prepárese para dedicar al menos 90 minutos a completar el tutorial o complételo en varias sesiones.

Requisitos previos para el tutorial

- Un usuario administrativo. Consulte Configuración para utilizar Amazon ECS.
- (Opcional) Un par de claves `PEM` para conectarse a la instancia EC2 de Windows a través del acceso RDP. Para obtener información acerca de cómo crear pares de claves, consulte Pares de claves de Amazon EC2 e instancias de Amazon EC2 en la Guía del usuario de Amazon EC2.
- Una VPC con al menos una subred pública y una subred privada, y un grupo de seguridad. Puede utilizar la VPC predeterminada. No necesita una gateway ni un dispositivo NAT. AWS Directory Service no admite la traducción de direcciones de red (NAT) en Active Directory. Para que esto funcione, Active Directory, el sistema de archivos FSx para Windows File Server, el clúster de ECS y la instancia de ECS deben ubicarse dentro de la VPC. Para obtener más información sobre las VPC y Active Directories, consulte Crear una VPC y Requisitos previos para crear un AWS Managed Microsoft AD.

- Los permisos ecsInstanceRole y ecsTaskExecutionRole de IAM están asociados a la cuenta. Estos roles vinculados a servicios permiten que los servicios realicen llamadas a la API y accedan a los contenedores, los secretos, los directorios y los servidores de archivos en su nombre.

Paso 1: Crear roles de IAM de acceso

Cree un clúster desde la AWS Management Console.

1. Consulte Rol de IAM de instancia de contenedor de Amazon ECS para comprobar si dispone de un ecsInstanceRole y, si no dispone de ninguno, ver cómo puede crear uno.
2. Recomendamos personalizar las políticas de roles con permisos mínimos en un entorno de producción real. Para poder trabajar con este tutorial, compruebe que la siguiente política administrada por AWS esté asociada a ecsInstanceRole. Si la política aún no está asociada, asóciela ahora.

   - AmazonEC2ContainerServiceforEC2Role
   - AmazonSSMManagedInstanceCore
   - AmazonSSMDirectoryServiceAccess

   Para asociar una política administrada por AWS

   a. Abra la consola de IAM.
   b. Seleccione Roles en el panel de navegación.
   c. Elija un Rol administrado de AWS.
   d. Elija Permisos y Asociar políticas.
   e. Para reducir la lista de políticas disponibles a asociar, utilice la función Filter (Filtro).
   f. Seleccione la política pertinente y, a continuación, elija Attach Policy (Asociar política).
3. Consulte Rol de IAM de ejecución de tareas de Amazon ECS para comprobar si dispone de un ecsTaskExecutionRole y, si no dispone de ninguno, ver cómo puede crear uno.

   Recomendamos personalizar las políticas de roles con permisos mínimos en un entorno de producción real. Para poder trabajar con este tutorial, compruebe que las siguientes políticas administradas por AWS estén asociadas a ecsTaskExecutionRole. Si las políticas no están asociadas, asócielas ahora. Para asociar las políticas administradas por AWS, utilice el procedimiento indicado en la sección anterior.

- SecretsManagerReadWrite
- AmazonFSxReadOnlyAccess
- AmazonSSMReadOnlyAccess
- AmazonECSTaskExecutionRolePolicy

Paso 2: Crear Windows Active Directory (AD)

1. Siga los pasos que se indican en Creación del directorio de Microsoft AD administrado por AWS en la Guía de administración de AWS Directory Service. Utilice la VPC que haya designado para este tutorial. En el paso 3 de Creación de AWS Managed Microsoft AD, guarde el nombre de usuario y la contraseña del administrador para utilizarlos en el siguiente paso. Además, tenga en cuenta el nombre de DNS del directorio calificado completo para futuros pasos. Puede completar el siguiente paso mientras se crea Active Directory.
2. Cree un secreto de AWS Secrets Manager para utilizarlo en los siguientes pasos. Para obtener más información, consulte Introducción a Secrets Manager en la Guía del usuario de AWS Secrets Manager.

   a. Abra la consola de Secrets Manager.

   b. Haga clic en Store a new secret (Almacenar un nuevo secreto).

   c. Seleccione Other type of secrets (Otro tipo de secretos).

   d. En Secret key/value (Clave/valor secreto), cree una clave **`username`**con el valor **`admin`** en la primera fila. Haga clic en.+ Add a row (+ Agregar una fila).

   e. En la nueva fila, cree una clave **`password`**. Para el valor, escriba la contraseña que ingresó en el paso 3 de Creación del Directorio de AD administrado por AWS.

   f. Haga clic en el botón Next (Siguiente).

   g. Proporcione un nombre y una descripción para el secreto. Haga clic en Next (Siguiente).

   h. Haga clic en Next (Siguiente). Haga clic en Store (Almacenar).

   i. En la lista que aparece en la página Secrets (Secretos), haga clic en el secreto que acaba de crear.

   j. Guarde el ARN del nuevo secreto para utilizarlo en los pasos siguientes.

   k. Puede continuar con el paso siguiente mientras se crea Active Directory.

Paso 3: Comprobar y actualizar el grupo de seguridad

En este paso, compruebe y actualice las reglas del grupo de seguridad que está utilizando. Para eso, puede utilizar el grupo de seguridad predeterminado que se creó para la VPC.

Verifique y actualice el grupo de seguridad.

Debe crear o editar el grupo de seguridad para que envíe datos a través de los puertos, que se describen en Grupos de seguridad de Amazon VPC en la Guía del usuario de FSx for Windows File Server. Para hacerlo, puede crear la regla de entrada del grupo de seguridad que se muestra en la primera fila de la siguiente tabla de reglas de entrada. Esta regla permite el tráfico entrante procedente de las interfaces de red (y las instancias asociadas) asignadas al mismo grupo de seguridad. Todos los recursos en la nube que cree están dentro de la misma VPC y asociados al mismo grupo de seguridad. Por lo tanto, esta regla permite que el tráfico se envíe a través del sistema de archivos FSx for Windows File Server, Active Directory y la instancia de ECS, según corresponda. Las otras reglas de entrada permiten que el tráfico atienda al sitio web y al acceso RDP para conectarse a la instancia de ECS.

En la tabla siguiente, se muestran las reglas de entrada del grupo de seguridad requeridas para este tutorial.

Tipo	Protocolo	Intervalo de puertos	Origen
Todo el tráfico	Todos	Todos	`sg-securitygroup`
HTTPS	TCP	443	0.0.0.0/0
RDP	TCP	3389	Dirección IP de la computadora portátil

En la tabla siguiente, se muestran las reglas de salida del grupo de seguridad requeridas para este tutorial.

Tipo	Protocolo	Rango de puerto	Destino

Tipo	Protocolo	Rango de puerto	Destino
Todo el tráfico	Todos	Todos	0.0.0.0/0

1. Abra la consola de EC2 y seleccione Security Groups (Grupos de seguridad) en el menú de la izquierda.
2. En la lista de grupos de seguridad que se muestra ahora, seleccione la casilla de verificación situada a la izquierda del grupo de seguridad que está utilizando para este tutorial.

   Se muestran los detalles del grupo de seguridad.
3. Para editar las reglas de entrada y salida, seleccione las pestañas Inbound rules (Reglas de entrada) o Outbound rules (Reglas de salida) y elija los botones Edit inbound rules (Editar reglas de entrada) o Edit outbound rules (Editar reglas de salida). Edite las reglas para que coincidan con las que se muestran en las tablas anteriores. Después de crear la instancia de EC2 más adelante en este tutorial, edite el origen RDP de la regla de entrada con la dirección IP pública de la instancia de EC2 como se describe en Conectarse a una instancia de Windows mediante RDP en la Guía del usuario de Amazon EC2.

Paso 4: Crear un sistema de archivos FSx for Windows File Server

Después de comprobar y actualizar el grupo de seguridad y de crear Active Directory y dejarlo en estado activo, cree el sistema de archivos FSx for Windows File Server en la misma VPC en la que se encuentra Active Directory. Siga estos pasos para crear un sistema de archivos FSx for Windows File Server para las tareas de Windows.

Cree el primer sistema de archivos.

1. Abra la consola de Amazon FSx.
2. En el panel, elija Create file system (Crear sistema de archivos) para iniciar el asistente de creación de sistemas de archivos.
3. En la página Seleccionar tipo de sistema de archivos, elija FSx para Windows File Server y, a continuación, elija Siguiente. Aparece la página Crear sistema de archivos.
4. En la sección File system details (Detalles del sistema de archivos), proporcione un nombre para el sistema de archivos. Dar nombre a los sistemas de archivos hace que sea más fácil encontrarlos y administrarlos. Puede usar hasta 256 caracteres Unicode. Los caracteres

permitidos son letras, números, espacios y los caracteres especiales signo más (+), signo menos (-), signo igual (=), punto (.), guion bajo (_), dos puntos (:) y barra inclinada (/).

5. En Deployment type (Tipo de implementación), elija Single-AZ (Zona de disponibilidad única) para implementar un sistema de archivos que se implemente en una sola zona de disponibilidad. Single-AZ 2 es la última generación de sistemas de archivos de zona de disponibilidad única y admite almacenamiento SSD y HDD.
6. En Storage type (Tipo de almacenamiento), elija HDD.
7. En Storage capacity (Capacidad de almacenamiento), ingrese la capacidad de almacenamiento mínima.
8. En el campo Throughput capacity (Capacidad de rendimiento), mantenga la configuración predeterminada.
9. En la sección Network & security (Red y seguridad), elija la misma Amazon VPC que eligió para el directorio AWS Directory Service.
10. En VPC Security Groups (Grupos de seguridad de la VPC), elija el grupo de seguridad que verificó en el Paso 3: Comprobar y actualizar el grupo de seguridad.
11. En Autenticación de Windows, elija AWS Managed Microsoft Active Directory y, a continuación, elija el directorio AWS Directory Service de la lista.
12. En Encryption (Cifrado), conserve el valor predeterminado aws/fsx (default) en el campo Encryption key (Clave de cifrado).
13. En Maintenance preferences (Preferencias de mantenimiento), conserve la configuración predeterminada.
14. Haga clic en el botón Next (Siguiente).
15. Revise la configuración del sistema de archivos que se muestra en la página Create File System (Crear sistema de archivos). Para su referencia, anote qué configuración del sistema de archivos puede modificar después de crear el sistema de archivos. Seleccione Crear sistema de archivos.
16. Anote el ID del sistema de archivos. Lo utilizará en un paso posterior.

    Vaya a los pasos siguientes para crear un clúster y una instancia EC2 mientras se crea el sistema de archivos FSx for Windows File Server.

Paso 5: Crear un clúster de Amazon ECS

Crear un clúster mediante la consola clásica de Amazon ECS

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.

2. En la barra de navegación, seleccione la región a utilizar.
3. En el panel de navegación, seleccione Clusters (Clústeres).
4. En la página Clusters (Clústeres), elija Create Cluster (Crear clúster).
5. En Configuración de clúster, en Nombre del clúster, ingrese windows-fsx-cluster.
6. Expanda Infraestructura, desactive AWS Fargate (sin servidor) y, a continuación, seleccione Instancias de Amazon EC2.
   - Para crear un grupo de Auto Scaling, desde Auto Scaling group (ASG) (Grupo de Auto Scaling), seleccione Create new group (Crear nuevo grupo) y, a continuación, proporcione los siguientes detalles sobre el grupo:
     - En Sistema operativo/arquitectura, elija Windows Server 2019 Core.
     - Para el Tipo de instancia de EC2, seleccione t2.medium o t2.micro.
7. Seleccione Crear.

Paso 6: Crear una instancia de Amazon EC2 optimizada para Amazon ECS

Cree una instancia de contenedor de Amazon ECS para Windows.

Para crear una instancia de Amazon ECS

1. Utilice el comando `aws ssm get-parameters` para recuperar el nombre de la AMI de la región que aloja la VPC. Para obtener más información, consulte Recuperación de los metadatos de la AMI optimizada para Amazon ECS.
2. Utilice la consola de Amazon EC2 para lanzar la instancia.
   a. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
   b. En la barra de navegación, seleccione la región a utilizar.
   c. En el panel de EC2, elija Launch Instance (Lanzar instancia).
   d. En Name (Nombre), escriba un nombre único.
   e. En Imágenes de aplicaciones y SO (Imagen de máquina de Amazon), en el campo Buscar, introduzca la AMI que recuperó.
   f. En Tipo de instancia, seleccione t2.medium o t2.micro.
   g. En Key pair (login) (Par de claves [inicio de sesión]), elija un par de claves. Si no especifica ningún para de clave

h. En Configuración de red, para VPC y Subred, elija su VPC y una subred pública.

i. En Network settings (Configuración de red), para Security groups (Grupo de seguridad), seleccione un grupo de seguridad existente o cree uno nuevo. Asegurarse de que el grupo de seguridad que elija tenga las reglas de entrada y salida definidas en Requisitos previos para el tutorial

j. En Network settings (Configuración de red), para Auto-assign Public IP (Asignar automáticamente una IP pública), selecciona Enable (Activar).

k. Expanda Detalles avanzados y, a continuación, en Directorio de unión a dominios, seleccione el ID del Active Directory que creó. Este dominio opcional se une al AD cuando se lanza la instancia EC2.

l. En Advanced details (Detalles avanzados), en IAM instance profile (Perfil de instancia de IAM), elija ecsInstanceRole.

m. Configure su instancia de contenedor de Amazon ECS con los siguientes datos de usuario. En Advanced details (Detalles avanzados), pegue el siguiente script en el campo User data (Datos de usuario), reemplazando *cluster_name* con el nombre de su clúster.

```
<powershell>
Initialize-ECSAgent -Cluster windows-fsx-cluster -EnableTaskIAMRole
</powershell>
```

n. Cuando esté listo, seleccione el campo de confirmación y después elija Launch Instances.

o. Verá una página de confirmación que indicará que la instancia se está lanzando. Elija View instances para cerrar la página de confirmación y volver a la consola.

3. Abra la consola en https://console.aws.amazon.com/ecs/v2.

4. En el panel de navegación, seleccione Clústeres y, luego, seleccione windows-fsx-cluster.

5. Seleccione la pestaña Infraestructura y compruebe que la instancia se haya registrado en el clúster windows-fsx-cluster.

Paso 7: Registrar una definición de tareas de Windows

Antes de poder ejecutar los contenedores de Windows en el clúster de Amazon ECS, debe registrar una definición de tarea. El siguiente ejemplo de definición de tareas muestra una página web sencilla. La tarea lanza dos contenedores que tienen acceso al sistema de archivos FSx. El primer contenedor escribe un archivo HTML en el sistema de archivos. El segundo contenedor descarga el archivo HTML del sistema de archivos y atiende la página web.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, elija Task Definitions (Definiciones de tareas).
3. Elija Create new task definition (Crear nueva definición de tarea) y Create new task definition with JSON (Crear nueva definición de tarea con JSON).
4. En el cuadro del editor JSON, sustituya los valores del rol de ejecución de tareas y los detalles del sistema de archivos FSx y, a continuación, elija Guardar.

```
{
    "containerDefinitions": [
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [],
            "command": ["New-Item -Path C:\\fsx-windows-dir\\index.html -ItemType
 file -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body
 {margin-top: 40px; background-color: #333;} </style> </head><body> <div
 style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1> <h2>It
 Works!</h2> <p>You are using Amazon FSx for Windows File Server file system for
 persistent container storage.</p>' -Force"],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": false,
            "name": "container1",
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ]
        },
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [
                {
```

```
                    "hostPort": 443,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": ["Remove-Item -Recurse C:\\inetpub\\wwwroot\\* -Force;
 Start-Sleep -Seconds 120; Move-Item -Path C:\\fsx-windows-dir\\index.html -
Destination C:\\inetpub\\wwwroot\\index.html -Force; C:\\ServiceMonitor.exe
 w3svc"],
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": true,
            "name": "container2"
        }
    ],
    "family": "fsx-windows",
    "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
    "volumes": [
        {
            "name": "fsx-windows-dir",
            "fsxWindowsFileServerVolumeConfiguration": {
                "fileSystemId": "fs-0eeb5730b2EXAMPLE",
                "authorizationConfig": {
                    "domain": "example.com",
                    "credentialsParameter": "arn:arn-1234"
                },
                "rootDirectory": "share"
            }
        }
    ]
}
```

Paso 8: Ejecutar una tarea y ver los resultados

Antes de ejecutar la tarea, compruebe que el estado del sistema de archivos FSx for Windows File Server sea Available (Disponible). Una vez que esté disponible, puede ejecutar una tarea mediante la definición de tareas que creó. La tarea comienza por crear contenedores que mezclan un archivo HTML entre ellos mediante el sistema de archivos. Después de la mezcla, un servidor web atiende la página HTML simple.

> **Note**
>
> Es posible que no pueda conectarse al sitio web desde una VPN.

Ejecute una tarea y observe los resultados con la consola de Amazon ECS.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clústeres y, luego, seleccione windows-fsx-cluster.
3. Elija la pestaña de Tareas y, a continuación, Ejecutar nueva tarea.
4. En Launch Type, elija EC2.
5. En configuración de la implementación, en Definición de tareas, elija fsx-windows y, a continuación, elija Crear.
6. Cuando el estado de la tarea sea EN EJECUCIÓN, haga clic en el ID de la tarea.
7. En Contenedores, cuando el estado del contenedor1 sea DETENIDO, seleccione el contenedor 2 para ver los detalles del contenedor.
8. En Detalles del contenedor para container2, seleccione Enlaces de red y, a continuación, haga clic en la dirección IP externa asociada al contenedor. Se abrirá el navegador y mostrará el siguiente mensaje.

```
Amazon ECS Sample App
It Works!
You are using Amazon FSx for Windows File Server file system for persistent
 container storage.
```

> **Note**
>
> Puede que transcurran unos minutos hasta que se muestre el mensaje. Si no ve este mensaje después de unos minutos, compruebe que no se esté ejecutando en una VPN y

asegúrese de que el grupo de seguridad de la instancia de contenedor permita el tráfico HTTP de red entrante en el puerto 443.

Paso 9: limpiar

> **Note**
>
> El proceso de eliminación del sistema de archivos FSx for Windows File Server o el AD tarda entre 20 y 45 minutos. Debe esperar hasta que se hayan completado las operaciones de eliminación del sistema de archivos FSx for Windows File Server antes de iniciar las operaciones de eliminación de AD.

Elimine el sistema de archivos de FSx para Windows File Server.

1. Abra la consola de Amazon FSx.
2. Elija el botón de la radio a la izquierda del sistema de archivos FSx para Windows File Server que acaba de crear.
3. Elija Acciones.
4. Seleccione Delete file system (Eliminar sistema de archivos).

Eliminar AD.

1. Abra la consola de AWS Directory Service.
2. Haga clic en el botón de radio situado a la izquierda del AD que acaba de crear.
3. Elija Acciones.
4. Seleccione Delete directory (Eliminar directorio).

Eliminar el clúster.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clústeres y, luego, seleccione windows-fsx-cluster.
3. Seleccione Delete cluster (Eliminar clúster).
4. Escriba la frase y, a continuación, elija Eliminar.

Termine la instancia de EC2.

1. Abra la consola de Amazon EC2.
2. En el menú de la izquierda, seleccione Instances (Instancias).
3. Marque la casilla de verificación situada a la izquierda de la instancia de EC2 que creó.
4. Haga clic en el botón Estado de instancia y en Terminar instancia.

Elimine el secreto.

1. Abra la consola de Secrets Manager.
2. Seleccione el secreto que creó para este ejercicio.
3. Haga clic en Actions (Acciones).
4. Seleccione Delete secret (Eliminar secreto).

Uso de volúmenes de Docker con Amazon ECS

Cuando se utilizan volúmenes de Docker, se puede usar el controlador `local` integrado o un controlador de volumen de terceros. Los volúmenes de Docker los administra Docker, y se crea un directorio en `/var/lib/docker/volumes` en la instancia de contenedor que contiene los datos del volumen.

Para utilizar volúmenes de Docker, especifique `dockerVolumeConfiguration` en su definición de tarea. Para obtener más información, consulte Volúmenes en la documentación de Docker.

Algunos casos de uso comunes de volúmenes de Docker son los siguientes:

- Proporcionar volúmenes de datos persistentes para su uso con contenedores
- Compartir un volumen de datos definido en distintas ubicaciones de distintos contenedores en la misma instancia de contenedor
- Definir un volumen de datos no persistentes vacío y montarlo en varios contenedores dentro de la misma tarea
- Proporcionar un volumen de datos a la tarea que está administrado por un controlador de terceros

Consideraciones sobre el uso de volúmenes de Docker

Al utilizar volúmenes de Amazon Docker, tenga en cuenta lo siguiente:

- Los volúmenes de Docker solo se admiten cuando se utiliza el tipo de lanzamiento de EC2 o instancias externas.
- Los contenedores de Windows solo admiten el uso del controlador `local`.
- Si se utiliza un controlador de terceros, asegúrese de que está instalado y activo en la instancia de contenedor antes de iniciar el agente de contenedor. Si el controlador de terceros no está activo antes de iniciar el agente, puede reiniciar el agente de contenedor con uno de los siguientes comandos:
  - Para la AMI de Amazon Linux 2 optimizada para Amazon ECS:

    ```
    sudo systemctl restart ecs
    ```

  - Para la AMI de Amazon Linux optimizada para Amazon ECS:

    ```
    sudo stop ecs && sudo start ecs
    ```

Para obtener información sobre cómo especificar un volumen de Docker en una definición de tarea, consulte Especificación de un volumen de Docker en la definición de tareas de Amazon ECS.

Especificación de un volumen de Docker en la definición de tareas de Amazon ECS

Para que los contenedores puedan utilizar volúmenes de datos, debe especificar las configuraciones del volumen y el punto de montaje en su definición de tarea. En esta sección se describe la configuración de volumen para un contenedor. Para las tareas que usan un volumen de Docker, especifique `dockerVolumeConfiguration`. Para las tareas que usan un volumen de host de montaje vinculado, especifique `host` y, si lo desea, `sourcePath`.

El siguiente JSON de definición de tareas muestra la sintaxis de los objetos `volumes` y `mountPoints` para un contenedor.

```
{
    "containerDefinitions": [
        {
            "mountPoints": [
                {
                    "sourceVolume": "string",
                    "containerPath": "/path/to/mount_volume",
                    "readOnly": boolean
                }
            ]
```

```
        }
    ],
    "volumes": [
        {
            "name": "string",
            "dockerVolumeConfiguration": {
                "scope": "string",
                "autoprovision": boolean,
                "driver": "string",
                "driverOpts": {
                    "key": "value"
                },
                "labels": {
                    "key": "value"
                }
            }
        }
    ]
}
```

`name`

Tipo: cadena

Requerido: no

El nombre del volumen. Se admiten hasta 255 letras (mayúsculas y minúsculas), números, guiones (`-`) y caracteres de subrayado (_). Se hace referencia a este nombre en el parámetro `sourceVolume` del objeto `mountPoints` de la definición de contenedor.

`dockerVolumeConfiguration`

Type: objeto de DockerVolumeConfiguration

Obligatorio: no

Este parámetro se especifica cuando se usan volúmenes de Docker. Los volúmenes de Docker se admiten solo cuando se ejecutan tareas en instancias de EC2. Los contenedores de Windows admiten solo el uso del controlador `local`. Para utilizar montajes vinculados, especifique `host` en su lugar.

`scope`

Tipo: cadena

Valores válidos: `task` | `shared`

Obligatorio: no

El ámbito del volumen de Docker, que determina su ciclo de vida. Los volúmenes de Docker con un ámbito de `task` se aprovisionan automáticamente cuando se inicia la tarea y se destruyen cuando la tarea se detiene. Los volúmenes de Docker cuyo ámbito es `shared` se conservan una vez detenida la tarea.

`autoprovision`

Tipo: Booleano

Valor predeterminado: `false`

Obligatorio: no

Si este valor es `true`, el volumen de Docker se crea si aún no existe. Este campo se usa solo si `scope` es `shared`. Si el `scope` es `task`, este parámetro debe omitirse.

`driver`

Tipo: cadena

Requerido: no

El controlador del volumen de Docker que se va a usar. El valor de controlador debe coincidir con el nombre del controlador proporcionado por Docker, ya que se utiliza para la colocación de tareas. Si el controlador se instaló mediante la CLI del complemento de Docker, utilice `docker plugin ls` para recuperar el nombre de controlador de la instancia de contenedor. Si el controlador se instaló con otro método, utilice la detección de complementos de Docker para recuperar el nombre del controlador.

`driverOpts`

Tipo: cadena

Requerido: no

Un mapa de las opciones específicas del controlador de Docker que se deben transferir. Este parámetro se corresponde con `DriverOpts` en la sección Crear un volumen de Docker.

`labels`

Tipo: cadena

Requerido: no

Metadatos personalizados que se añaden al volumen de Docker.

`mountPoints`

Tipo: matriz de objetos

Obligatorio: no

Puntos de montaje para los volúmenes de datos del contenedor. Este parámetro asigna a `Volumes` en la API create-container de Docker y la opción `--volume` a docker run.

Los contenedores de Windows pueden montar directorios completos en la misma unidad que `$env:ProgramData`. Los contenedores de Windows no pueden montar directorios en una unidad diferente y los puntos de montaje no se pueden utilizar entre unidades. Debe especificar los puntos de montaje para adjuntar un volumen de Amazon EBS directamente a una tarea de Amazon ECS.

`sourceVolume`

Tipo: cadena

Obligatorio: sí, si se utilizan `mountPoints`.

El nombre del volumen a montar.

`containerPath`

Tipo: cadena

Obligatorio: sí, si se utilizan `mountPoints`.

La ruta del contenedor donde se montará el volumen.

`readOnly`

Tipo: Booleano

Obligatorio: no

Si este valor es `true`, el acceso del contenedor al volumen es de solo lectura. Si este valor es `false`, el contenedor puede escribir en el volumen. El valor predeterminado es `false`.

Para las tareas que se ejecutan en instancias de EC2 con sistema operativo Windows, deje el valor predeterminado `false`.

Ejemplos de volúmenes de Docker para Amazon ECS

Los siguientes ejemplos muestran cómo proporcionar almacenamiento efímero para un contenedor, cómo proporcionar un volumen compartido para varios contenedores y cómo proporcionar almacenamiento persistente de NFS para un contenedor.

Para proporcionar almacenamiento efímero para un contenedor con un volumen de Docker

En este ejemplo, un contenedor utiliza un volumen de datos vacío que se elimina después de finalizar la tarea. Como caso de uso de ejemplo, podría tener un contenedor que necesita obtener acceso a una ubicación de almacenamiento de archivos scratch durante una tarea. Esta tarea se puede lograr utilizando un volumen de Docker.

1. En sección de definición de tarea de `volumes`, defina un volumen de datos con los valores `name` y `DockerVolumeConfiguration`. En este ejemplo, especifique el ámbito como `task` para que el volumen se elimine una vez que se detenga la tarea y utilice el controlador `local` integrado.

```
"volumes": [
    {
        "name": "scratch",
        "dockerVolumeConfiguration" : {
            "scope": "task",
            "driver": "local",
            "labels": {
                "scratch": "space"
            }
        }
    }
]
```

2. En la sección `containerDefinitions`, defina un contenedor con valores `mountPoints` que hagan referencia al nombre del volumen definido y el valor `containerPath` en el que desea montar el volumen en el contenedor.

```
"containerDefinitions": [
    {
        "name": "container-1",
        "mountPoints": [
            {
              "sourceVolume": "scratch",
              "containerPath": "/var/scratch"
```

```
            }
        ]
    }
]
```

Para proporcionar almacenamiento persistente para varios contenedores con un volumen de Docker

En este ejemplo, desea usar un volumen compartido para varios contenedores y desea que se conserve una vez que se detenga cualquiera de las tareas que lo usa. El controlador `local` integrado se está utilizando. Esto es para que el volumen siga vinculado al ciclo de vida de la instancia de contenedor.

1. En sección de definición de tarea de `volumes`, defina un volumen de datos con los valores `name` y `DockerVolumeConfiguration`. En este ejemplo, especifique un alcance `shared` para que el volumen persista, establezca el aprovisionamiento automático en `true`. Esto es para que el volumen se cree para su uso. A continuación, utilice también el controlador `local` integrado.

```
"volumes": [
    {
        "name": "database",
        "dockerVolumeConfiguration" : {
            "scope": "shared",
            "autoprovision": true,
            "driver": "local",
            "labels": {
                "database": "database_name"
            }
        }
    }
]
```

2. En la sección `containerDefinitions`, defina un contenedor con valores `mountPoints` que hagan referencia al nombre del volumen definido y el valor `containerPath` en el que desea montar el volumen en el contenedor.

```
"containerDefinitions": [
    {
        "name": "container-1",
        "mountPoints": [
```

```
        {
          "sourceVolume": "database",
          "containerPath": "/var/database"
        }
      ]
    },
    {
      "name": "container-2",
      "mountPoints": [
        {
          "sourceVolume": "database",
          "containerPath": "/var/database"
        }
      ]
    }
  ]
```

Para proporcionar almacenamiento persistente de NFS para un contenedor con un volumen de Docker

En este ejemplo, un contenedor utiliza un volumen de datos de NFS que se monta automáticamente cuando inicia la tarea y se desmonta cuando la tarea finaliza. Utiliza el controlador integrado `local` de Docker. Un caso de uso de ejemplo es que puede tener un almacenamiento de NFS local y necesita acceder a él desde una tarea de ECS Anywhere. Esto se puede lograr con un volumen de Docker con la opción de controlador de NFS.

1. En sección de definición de tarea de `volumes`, defina un volumen de datos con los valores `name` y `DockerVolumeConfiguration`. En este ejemplo, especifique un alcance `task` para que el volumen se desmonte cuando finaliza la tarea. Utilice el controlador `local` y configure `driverOpts` con las opciones `type`, `device` y o en consecuencia. Sustituya `NFS_SERVER` por el punto de conexión del servidor de NFS.

```
"volumes": [
       {
           "name": "NFS",
           "dockerVolumeConfiguration" : {
               "scope": "task",
               "driver": "local",
               "driverOpts": {
                   "type": "nfs",
                   "device": "$NFS_SERVER:/mnt/nfs",
```

```
                    "o": "addr=$NFS_SERVER"
                }
            }
        }
    ]
```

2. En la sección `containerDefinitions`, defina un contenedor con valores `mountPoints` que hagan referencia al nombre del volumen definido y el valor `containerPath` en el que desea montar el volumen en el contenedor.

```
"containerDefinitions": [
        {
            "name": "container-1",
            "mountPoints": [
                {
                  "sourceVolume": "NFS",
                  "containerPath": "/var/nfsmount"
                }
            ]
        }
    ]
```

Uso de montajes de unión con Amazon ECS

Con los montajes de unión, se monta un archivo o directorio de un host en un contenedor, como, por ejemplo, una instancia de Amazon EC2. Se admiten montajes de enlace para tareas que están alojadas en instancias de Fargate y Amazon EC2. Los montajes de unión están vinculados al ciclo de vida del contenedor que los utiliza. Una vez que se detienen todos los contenedores que utilizan un montaje de enlace, como cuando se detiene una tarea, se eliminan los datos. En el caso de las tareas que están alojadas en instancias de Amazon EC2, para vincular los datos al ciclo de vida de la instancia de Amazon EC2 del host, se puede especificar un `host` y un valor de `sourcePath` opcional en la definición de tareas. Para obtener más información, consulte Montajes vinculados en la documentación de Docker.

A continuación, se indican algunos casos de uso comunes de los montajes de enlace.

- Para proporcionar un volumen de datos vacío y montarlo en uno o más contenedores.
- Para montar un volumen de datos del host en uno o más contenedores.

- Para compartir un volumen de datos de un contenedor de origen con otros contenedores de la misma tarea.
- Para exponer una ruta de acceso y su contenido desde un Dockerfile a uno o más contenedores.

Consideraciones acerca del uso de los montajes de enlace

Al utilizar montajes de enlace, tenga en cuenta lo siguiente.

- De manera predeterminada, las tareas alojadas en AWS Fargate que utilizan la versión de la plataforma `1.4.0` o una posterior (Linux) o `1.0.0` o una posterior (Windows) reciben un mínimo de 20 GiB de almacenamiento efímero para los montajes de unión. Para aumentar la cantidad total de almacenamiento efímero, hasta un máximo de 200 GiB, indique el parámetro `ephemeralStorage` en la definición de tareas.
- Para exponer los archivos de un Dockerfile a un volumen de datos cuando se ejecuta una tarea, el plano de datos de Amazon ECS busca una directiva `VOLUME`. Si la ruta absoluta especificada en la directiva `VOLUME` es la misma que la `containerPath` especificada en la definición de tarea, los datos de la ruta de la directiva `VOLUME` se copian en el volumen de datos. En el siguiente ejemplo de Dockerfile, un archivo con el nombre `examplefile` en el directorio `/var/log/exported` se escribe en el host y, a continuación, se monta dentro del contenedor.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
RUN mkdir -p /var/log/exported
RUN touch /var/log/exported/examplefile
VOLUME ["/var/log/exported"]
```

  De forma predeterminada, los permisos de los volúmenes se establecen en`0755` y el propietario como `root`. Puede personalizar estos permisos en el Dockerfile. En el siguiente ejemplo, se define al propietario del directorio como `node`.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
RUN yum install -y shadow-utils && yum clean all
RUN useradd node
RUN mkdir -p /var/log/exported && chown node:node /var/log/exported
RUN touch /var/log/exported/examplefile
USER node
VOLUME ["/var/log/exported"]
```

- Para las tareas que están alojadas en instancias de Amazon EC2, cuando no se especifica el valor de `host` ni de `sourcePath`, es el daemon de Docker el que administra el montaje de enlace. Cuando ningún contenedor hace referencia a este montaje de enlace, el servicio de limpieza de tareas del agente de contenedor de Amazon ECS lo elimina finalmente. De forma predeterminada, esto sucede tres horas después de que el contenedor se cierre. Sin embargo, es posible configurar esta duración con la variable de agente `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS. Si necesita que estos datos persistan más allá del ciclo de vida del contenedor, especifique un valor de `sourcePath` para el montaje de enlace.

Especificación de un montaje de unión en la definición de tareas de Amazon ECS

En el siguiente fragmento de JSON de definición de tareas, se muestra la sintaxis de los objetos `volumes`, `mountPoints` y `ephemeralStorage` para una definición de las tareas de Amazon ECS alojadas en instancias de Amazon EC2 o Fargate.

```
{
   "family": "",
   ...
   "containerDefinitions" : [
      {
         "mountPoints" : [
            {
               "containerPath" : "/path/to/mount_volume",
               "sourceVolume" : "string"
            }
          ],
          "name" : "string"
       }
    ],
    ...
    "volumes" : [
       {
          "name" : "string"
       }
    ],
    "ephemeralStorage": {
    "sizeInGiB": integer
    }
}
```

Para las tareas de Amazon ECS que están alojadas en instancias de Amazon EC2, puede utilizar el parámetro `host` opcional y un `sourcePath` al especificar los detalles del volumen de tareas. Cuando se especifica, vincula el montaje de enlace al ciclo de vida de la tarea en lugar del contenedor.

```
"volumes" : [
    {
        "host" : {
            "sourcePath" : "string"
        },
        "name" : "string"
    }
]
```

A continuación, se describe en más detalle cada uno de los parámetros de la definición de tarea.

`name`

Tipo: cadena

Requerido: no

El nombre del volumen. Se admiten hasta 255 letras (mayúsculas y minúsculas), números, guiones (`-`) y caracteres de subrayado (`_`). Se hace referencia a este nombre en el parámetro `sourceVolume` del objeto `mountPoints` de la definición de contenedor.

`host`

Obligatorio: no

El parámetro `host` se utiliza para vincular el ciclo de vida del montaje de enlace a la instancia de Amazon EC2 del host, en lugar de a la tarea, y donde se almacena. Si el parámetro `host` está vacío, entonces el daemon de Docker asigna una ruta de host a su volumen de datos, pero no se garantiza que los datos persistan después de que los contenedores asociados dejen de funcionar.

Los contenedores de Windows pueden montar directorios completos en la misma unidad que `$env:ProgramData`.

> **Note**
>
> El parámetro `sourcePath` se admite solo cuando se utilizan las tareas que se alojan en instancias de Amazon EC2.

`sourcePath`

Tipo: cadena

Requerido: no

Cuando utilice el parámetro `host`, especifique una `sourcePath` para declarar la ruta de la instancia de Amazon EC2 del host que se presenta al contenedor. Si este parámetro está vacío, el daemon de Docker asigna una ruta de host. Si el parámetro `host` contiene una ubicación de archivos `sourcePath`, el volumen de datos persiste en la ubicación especificada en la instancia de Amazon EC2 del host hasta que la elimine manualmente. Si el valor `sourcePath` no existe en la instancia de Amazon EC2 del host, el daemon de Docker lo crea. Si la ubicación existe, el contenido de la carpeta de la ruta de origen se exporta.

`mountPoints`

Tipo: matriz de objetos

Obligatorio: no

Puntos de montaje para los volúmenes de datos del contenedor. Este parámetro asigna a `Volumes` en la API create-container de Docker y la opción `--volume` a docker run.

Los contenedores de Windows pueden montar directorios completos en la misma unidad que `$env:ProgramData`. Los contenedores de Windows no pueden montar directorios en una unidad diferente y los puntos de montaje no se pueden utilizar entre unidades. Debe especificar los puntos de montaje para adjuntar un volumen de Amazon EBS directamente a una tarea de Amazon ECS.

`sourceVolume`

Tipo: cadena

Obligatorio: sí, si se utilizan `mountPoints`.

El nombre del volumen a montar.

`containerPath`

Tipo: cadena

Obligatorio: sí, si se utilizan `mountPoints`.

La ruta del contenedor donde se montará el volumen.

`readOnly`

Tipo: Booleano

Obligatorio: no

Si este valor es `true`, el acceso del contenedor al volumen es de solo lectura. Si este valor es `false`, el contenedor puede escribir en el volumen. El valor predeterminado es `false`.

Para las tareas que se ejecutan en instancias de EC2 con sistema operativo Windows, deje el valor predeterminado `false`.

`ephemeralStorage`

Tipo: objeto

Obligatorio: no

Cantidad de almacenamiento efímero que se asignará a la tarea. Este parámetro se utiliza para expandir la cantidad total de almacenamiento efímero disponible, más allá de la cantidad predeterminada, para las tareas alojadas en AWS Fargate que utilizan la versión `1.4.0` o posterior (Linux) o `1.0.0` o posterior (Windows) de la plataforma.

Se puede utilizar la CLI de Copilot, CloudFormation, el SDK de AWS o la CLI para especificar almacenamiento efímero para un montaje de enlace.

Ejemplos de montajes vinculados para Amazon ECS

En los siguientes ejemplos se cubren los casos de uso más comunes de montaje vinculado para los contenedores.

Para asignar una mayor cantidad de espacio de almacenamiento efímero a una tarea de Fargate

Para las tareas de Amazon ECS alojadas en Fargate que utilizan la versión `1.4.0` o posterior (Linux) o `1.0.0` (Windows) de la plataforma, se puede asignar una cantidad mayor de almacenamiento

efímero que la predeterminada para que la utilicen los contenedores de la tarea. Este ejemplo se puede incorporar a los otros ejemplos para asignar un almacenamiento más efímero para sus tareas de Fargate.

- En la definición de tarea, defina un objeto `ephemeralStorage`. El valor de `sizeInGiB` se expresa en GiB y debe ser un número entero entre `21` y `200`.

```
"ephemeralStorage": {
    "sizeInGiB": integer
}
```

Para proporcionar un volumen de datos vacío para uno o más contenedores

En algunos casos, es posible que desee proporcionar a los contenedores de una tarea algo de espacio de almacenamiento temporal. Por ejemplo, podría tener dos contenedores de base de datos que deben acceder a la misma ubicación de almacenamiento de archivos scratch durante una tarea. Esto se puede lograr con un montaje de enlace.

1. En la sección de definición de tarea `volumes`, defina un montaje vinculado con el nombre `database_scratch`.

```
  "volumes": [
    {
      "name": "database_scratch"
    }
  ]
```

2. En la sección `containerDefinitions`, cree las definiciones de contenedor de base de datos. Esto es para que monten el volumen.

```
"containerDefinitions": [
    {
      "name": "database1",
      "image": "my-repo/database",
      "cpu": 100,
      "memory": 100,
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "database_scratch",
```

```
          "containerPath": "/var/scratch"
        }
      ]
    },
    {
      "name": "database2",
      "image": "my-repo/database",
      "cpu": 100,
      "memory": 100,
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "database_scratch",
          "containerPath": "/var/scratch"
        }
      ]
    }
  ]
```

Para exponer una ruta y su contenido en un Dockerfile a un contenedor

En este ejemplo, tiene un Dockerfile que escribe los datos que va a montar dentro de un contenedor. Este ejemplo funciona para tareas que están alojadas en instancias de Fargate o Amazon EC2.

1. Cree un Dockerfile. En el siguiente ejemplo, se utiliza la imagen de contenedor pública de Amazon Linux 2 y se crea un archivo con el nombre `examplefile` en el directorio `/var/log/exported` que queremos montar dentro del contenedor. La directiva `VOLUME` debe especificar una ruta absoluta.

   ```
   FROM public.ecr.aws/amazonlinux/amazonlinux:latest
   RUN mkdir -p /var/log/exported
   RUN touch /var/log/exported/examplefile
   VOLUME ["/var/log/exported"]
   ```

   De forma predeterminada, los permisos de los volúmenes se establecen en`0755` y el propietario como `root`. Estos permisos se pueden cambiar en el Dockerfile. En el ejemplo siguiente, el propietario del directorio `/var/log/exported` se establece como `node`.

   ```
   FROM public.ecr.aws/amazonlinux/amazonlinux:latest
   RUN yum install -y shadow-utils && yum clean all
   ```

```
RUN useradd node
RUN mkdir -p /var/log/exported && chown node:node /var/log/exported
USER node
RUN touch /var/log/exported/examplefile
VOLUME ["/var/log/exported"]
```

2. En la sección de definición de tarea de `volumes`, defina un volumen con el nombre `application_logs`.

```
"volumes": [
  {
    "name": "application_logs"
  }
]
```

3. En la sección `containerDefinitions`, cree las definiciones de contenedor de aplicaciones. Esto es para que monten el almacenamiento. El valor de `containerPath` debe coincidir con la ruta absoluta especificada en la directiva `VOLUME` del Dockerfile.

```
"containerDefinitions": [
  {
    "name": "application1",
    "image": "my-repo/application",
    "cpu": 100,
    "memory": 100,
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "application_logs",
        "containerPath": "/var/log/exported"
      }
    ]
  },
  {
    "name": "application2",
    "image": "my-repo/application",
    "cpu": 100,
    "memory": 100,
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "application_logs",
        "containerPath": "/var/log/exported"
```

```
        }
      ]
    }
  ]
```

Para proporcionar un volumen de datos vacío para un contenedor que está vinculado al ciclo de vida de la instancia de Amazon EC2 del host

Para las tareas que están alojadas en instancias de Amazon EC2, se pueden utilizar montajes de enlace y vincular los datos al ciclo de vida de la instancia de Amazon EC2 del host. Para ello, se utiliza el parámetro `host` y se especifica un valor de `sourcePath`. Cualquier archivo que exista en `sourcePath` se presenta a los contenedores con el valor `containerPath`. Cualquier archivo que se escriba en el valor `containerPath`, se escribe en el valor `sourcePath` en la instancia de Amazon EC2 del host.

> **Important**
>
> Amazon ECS no sincroniza el almacenamiento en todas las instancias de Amazon EC2. Las tareas que utilizan almacenamiento persistente se pueden colocar en cualquier instancia de Amazon EC2 del clúster que tenga capacidad disponible. Si las tareas requieren almacenamiento persistente después de detenerse y reiniciarse, especifique siempre la misma instancia de Amazon EC2 en el momento de lanzar la tarea con el comando start-task de la AWS CLI. También puede utilizar volúmenes de Amazon EFS para el almacenamiento persistente. Para obtener más información, consulte Uso de volúmenes de Amazon EFS con Amazon ECS.

1. En la sección de la definición de tarea `volumes`, defina un montaje vinculado con los valores de `name` y `sourcePath`. En el ejemplo siguiente, la instancia de Amazon EC2 del host contiene datos de `/ecs/webdata` que va a montar dentro del contenedor.

```
"volumes": [
  {
    "name": "webdata",
    "host": {
      "sourcePath": "/ecs/webdata"
    }
  }
```

```
]
```

2. En la sección `containerDefinitions`, defina un contenedor con un valor de `mountPoints` que haga referencia al nombre del montaje de enlace y el valor de `containerPath` en el que va a realizar el montaje de enlace en el contenedor.

```
"containerDefinitions": [
  {
    "name": "web",
    "image": "public.ecr.aws/docker/library/nginx:latest",
    "cpu": 99,
    "memory": 100,
    "portMappings": [
      {
        "containerPort": 80,
        "hostPort": 80
      }
    ],
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "webdata",
        "containerPath": "/usr/share/nginx/html"
      }
    ]
  }
]
```

Para montar un volumen definido en varios contenedores en diferentes ubicaciones

Puede definir un volumen de datos en una definición de tarea y montarlo en diferentes ubicaciones en distintos contenedores. Por ejemplo, el contenedor del host tiene una carpeta de datos del sitio web en `/data/webroot`. Es posible que desee montar ese volumen de datos como de solo lectura en dos servidores web diferentes que tengan diferentes raíces de documentos.

1. En la sección de definición de tarea de `volumes`, defina un volumen de datos con el nombre `webroot` y la ruta de origen `/data/webroot`.

```
"volumes": [
  {
    "name": "webroot",
```

```
      "host": {
        "sourcePath": "/data/webroot"
      }
    }
  ]
```

2. En la sección `containerDefinitions`, defina un contenedor para cada servidor web con valores de `mountPoints` que permitan asociar el volumen de `webroot` con el valor de `containerPath` apuntando a la raíz de documentos para ese contenedor.

```
  "containerDefinitions": [
    {
      "name": "web-server-1",
      "image": "my-repo/ubuntu-apache",
      "cpu": 100,
      "memory": 100,
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80
        }
      ],
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "webroot",
          "containerPath": "/var/www/html",
          "readOnly": true
        }
      ]
    },
    {
      "name": "web-server-2",
      "image": "my-repo/sles11-apache",
      "cpu": 100,
      "memory": 100,
      "portMappings": [
        {
          "containerPort": 8080,
          "hostPort": 8080
        }
      ],
      "essential": true,
```

```
      "mountPoints": [
        {
          "sourceVolume": "webroot",
          "containerPath": "/srv/www/htdocs",
          "readOnly": true
        }
      ]
    }
  ]
```

Para montar volúmenes desde otro contenedor con **`volumesFrom`**

Para las tareas alojadas en instancias de Amazon EC2, puede definir uno o más volúmenes en un contenedor y, a continuación, utilizar el parámetro `volumesFrom` en una definición de contenedor diferente dentro de la misma tarea para montar todos los volúmenes de `sourceContainer` en sus puntos de montaje definidos originalmente. El parámetro `volumesFrom` se aplica a volúmenes definidos en la definición de tarea, y a los que están integrados en la imagen con un Dockerfile.

1. (Opcional) Para compartir un volumen integrado en una imagen, utilice la instrucción `VOLUME` del Dockerfile. En el siguiente ejemplo de Dockerfile se utiliza una imagen `httpd` y, a continuación, se agrega un volumen y se lo monta en `dockerfile_volume` en la raíz de documentos de Apache. Es la carpeta que utiliza el servidor web `httpd`.

   ```
   FROM httpd
   VOLUME ["/usr/local/apache2/htdocs/dockerfile_volume"]
   ```

   Puede crear una imagen con este Dockerfile y enviarla a un repositorio, como Docker Hub, y utilizarla en su definición de tarea. La imagen de ejemplo `my-repo/httpd_dockerfile_volume` que se utiliza en los pasos siguientes se diseñó con el Dockerfile anterior.
2. Cree una definición de tarea que defina los otros volúmenes y puntos de montaje para los contenedores. En esta sección de ejemplo `volumes`, se crea un volumen vacío llamado `empty`, que será administrado por el daemon de Docker. También hay un volumen de host definido denominado `host_etc`. Exporta la carpeta `/etc` en la instancia de contenedor del host.

   ```
   {
     "family": "test-volumes-from",
     "volumes": [
       {
   ```

```
      "name": "empty",
      "host": {}
    },
    {
      "name": "host_etc",
      "host": {
        "sourcePath": "/etc"
      }
    }
  ],
```

En la sección de definiciones de contenedor, cree un contenedor para montar los volúmenes definidos anteriormente. En este ejemplo, el contenedor `web` monta los volúmenes `empty` y `host_etc`. Este es el contenedor que utiliza la imagen creada con un volumen en el Dockerfile.

```
"containerDefinitions": [
    {
      "name": "web",
      "image": "my-repo/httpd_dockerfile_volume",
      "cpu": 100,
      "memory": 500,
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80
        }
      ],
      "mountPoints": [
        {
          "sourceVolume": "empty",
          "containerPath": "/usr/local/apache2/htdocs/empty_volume"
        },
        {
          "sourceVolume": "host_etc",
          "containerPath": "/usr/local/apache2/htdocs/host_etc"
        }
      ],
      "essential": true
    },
```

Cree otro contenedor que utiliza `volumesFrom` para montar todos los volúmenes que están asociados con el contenedor `web`. Todos los volúmenes del contenedor `web` se montan también

en el contenedor `busybox`. Esto incluye el volumen especificado en el Dockerfile que se utilizó para crear la imagen `my-repo/httpd_dockerfile_volume`.

```
    {
      "name": "busybox",
      "image": "busybox",
      "volumesFrom": [
        {
          "sourceContainer": "web"
        }
      ],
      "cpu": 100,
      "memory": 500,
      "entryPoint": [
        "sh",
        "-c"
      ],
      "command": [
        "echo $(date) > /usr/local/apache2/htdocs/empty_volume/date && echo $(date)
 > /usr/local/apache2/htdocs/host_etc/date && echo $(date) > /usr/local/apache2/
htdocs/dockerfile_volume/date"
      ],
      "essential": false
    }
  ]
}
```

Cuando esta tarea se ejecuta, los dos contenedores montan los volúmenes y el `command` en el contenedor `busybox` escribe la fecha y la hora en un archivo. Este archivo se denomina `date` en cada una de las carpetas de volumen. Las carpetas se pueden ver en el sitio web mostrado por el contenedor `web`.

> **Note**
>
> Como el contenedor `busybox` ejecuta un comando rápido y, a continuación, se cierra, debe establecerse como `"essential": false` en la definición de contenedor. De no ser así, detiene toda la tarea cuando se cierra.

Administración del espacio de memoria de intercambio de contenedores en Amazon ECS

Con Amazon ECS, es posible controlar el uso del espacio de memoria de intercambio en las instancias de Amazon EC2 basadas en Linux en el nivel de contenedor. Con la configuración de intercambio por contenedor, cada contenedor dentro de una definición de tareas puede tener el intercambio habilitado o deshabilitado. Para aquellos que lo tienen habilitado, es posible limitar la cantidad máxima de espacio de intercambio que se utiliza. Por ejemplo, los contenedores de latencia crítica pueden tener el intercambio deshabilitado. En cambio, los contenedores con una gran demanda de memoria transitoria pueden tener activado el intercambio para reducir las posibilidades de errores de memoria insuficiente cuando el contenedor opera con carga.

La configuración de intercambio de un contenedor se administra mediante los siguientes parámetros de definición de contenedor:

`maxSwap`

La cantidad total de memoria de intercambio (en MiB) que puede utilizar un contenedor. Este parámetro se traduce en la opción `--memory-swap` de docker run donde el valor es la suma de la memoria del contenedor más el valor de `maxSwap`.

Si se especifica un valor `maxSwap` para `0`, el contenedor no utiliza el intercambio. Los valores aceptados son `0` o cualquier entero positivo. Si se omite el parámetro `maxSwap`, el contenedor utiliza la configuración de intercambio de la instancia de contenedor en la que se está ejecutando. Debe establecerse un valor de `maxSwap` para el parámetro `swappiness`.

`swappiness`

Puede utilizar esta opción para ajustar el comportamiento de intercambio de memoria de un contenedor. Con un valor `swappiness` de `0`, no se produce el intercambio a menos que sea necesario. Un valor `swappiness` de `100` aumenta al máximo el intercambio de páginas. Los valores aceptados son números enteros comprendidos entre `0` y `100`. Si no se especifica el parámetro `swappiness`, se utiliza el valor predeterminado de `60`. Si no se especifica ningún valor para `maxSwap`, este parámetro se omite. Este parámetro se corresponde con la opción `--memory-swappiness` con docker run.

En el siguiente ejemplo, se proporciona la sintaxis JSON.

```
"containerDefinitions": [{
```

```
        ...
        "linuxParameters": {
            "maxSwap": integer,
            "swappiness": integer
        },
        ...
}]
```

Consideraciones

Tenga en cuenta lo siguiente cuando utilice una configuración de intercambio por contenedor.

- El espacio de intercambio debe estar habilitado y asignado a la instancia de Amazon EC2 que aloja las tareas para que las utilicen los contenedores. De forma predeterminada, las AMI optimizadas para Amazon ECS no tienen habilitado el intercambio. Debe habilitar el intercambio en la instancia para utilizar esta característica. Para obtener más información, consulte Volúmenes de intercambio de almacén de instancias en la Guía del usuario de Amazon EC2 o ¿Cómo asigno memoria para que funcione como espacio de intercambio en una instancia de Amazon EC2?.
- Los parámetros de definición de contenedores de espacio de intercambio solo se admiten para las definiciones de tareas que especifican el tipo de lanzamiento de EC2. Estos parámetros no se admiten en las definiciones de tareas destinadas únicamente al uso de Amazon ECS en Fargate.
- Esta característica solo se admite para los contenedores de Linux. Los contenedores de Windows actualmente no se admiten.
- Si en una definición de tareas se omiten los parámetros de definición de contenedor `maxSwap` y `swappiness`, cada contenedor tiene un valor `swappiness` predeterminado de `60`. Además, el uso total de intercambio se limita al doble de la memoria del contenedor.
- Si utiliza tareas en Amazon Linux 2023, no se admite el parámetro `swappiness`.

Diferencias en la definición de tareas de Amazon ECS para el tipo de lanzamiento de Fargate

Para utilizar Fargate, se debe configurar la definición de tareas para utilizar el tipo de lanzamiento de Fargate. Hay algunas consideraciones adicionales a la hora de utilizar Fargate.

Parámetros de definición de tarea

Las tareas que utilizan el tipo de lanzamiento Fargate no admiten todos los parámetros de definición de tareas de Amazon ECS que están disponibles. Algunos parámetros directamente no son compatibles, y otros se comportan de forma distinta para tareas de Fargate.

Los siguientes parámetros de definición de tareas no son válidos en tareas de Fargate:

- `disableNetworking`
- `dnsSearchDomains`
- `dnsServers`
- `dockerSecurityOptions`
- `extraHosts`
- `gpu`
- `ipcMode`
- `links`
- `placementConstraints`
- `privileged`
- `maxSwap`
- `swappiness`

Los siguientes parámetros de definición de tareas son válidos en tareas de Fargate, pero presentan limitaciones que se deben tener en cuenta:

- `linuxParameters`: al especificar opciones específicas de Linux que se aplican al contenedor, la única capacidad que se puede agregar en `capabilities` es `CAP_SYS_PTRACE`. No se admiten los parámetros `devices`, `sharedMemorySize` y `tmpfs`. Para obtener más información, consulte Parámetros de Linux.
- `volumes`: las tareas de Fargate solo admiten volúmenes de host de montaje vinculado, por lo que no se admite el parámetro `dockerVolumeConfiguration`. Para obtener más información, consulte Volúmenes.
- `cpu`: para contenedores Windows en AWS Fargate, el valor no puede ser inferior a 1 vCPU.
- `networkConfiguration`: las tareas de Fargate siempre utilizan el modo de red `awsvpc`.

A fin de garantizar que la definición de tareas sea válida para su utilización con Fargate, puede especificar lo siguiente al registrar la definición de tareas:

- En la AWS Management Console, en el campo Requires Compatibilities (Requiere compatibilidades), especifique `FARGATE`.
- En la AWS CLI, especifique la opción `--requires-compatibilities`.
- En la API de Amazon ECS, especifique el indicador `requiresCompatibilities`.

Arquitecturas y sistemas operativos

Al configurar una definición de tarea y contenedor para AWS Fargate, debe especificar el sistema operativo que ejecuta el contenedor. Se admiten los siguientes sistemas operativos para AWS Fargate:

- Amazon Linux 2

> **Note**
>
> Los contenedores de Linux utilizan únicamente el kernel y la configuración del kernel del sistema operativo del host. Por ejemplo, la configuración del kernel incluye los controles del sistema `sysctl`. Se puede crear una imagen de contenedor de Linux a partir de una imagen base que contenga los archivos y programas de cualquier distribución de Linux. Si la arquitectura de la CPU coincide, puede ejecutar contenedores desde cualquier imagen de contenedor de Linux en cualquier sistema operativo.

- Windows Server 2019 Full
- Windows Server 2019 Core
- Windows Server 2022 Full
- Windows Server 2022 Core

Cuando ejecuta contenedores de Windows en AWS Fargate, debe tener la arquitectura de CPU X86_64.

Cuando ejecuta contenedores Linux en AWS Fargate, puede utilizar la arquitectura de CPU X86_64 o la arquitectura ARM64 para las aplicaciones basadas en ARM. Para obtener más información, consulte the section called “Definiciones de tareas para cargas de trabajo de ARM de 64 bits”.

Memoria y CPU de tarea

Las definiciones de tareas de Amazon ECS para AWS Fargate requieren que especifique la CPU y la memoria en el nivel de tarea. La mayoría de casos de uso solo se cumplen especificando estos recursos en el nivel de tarea. En la siguiente tabla se muestran las combinaciones válidas de CPU y memoria de nivel de tarea. Puede indicar los valores de memoria en la definición de tarea como una cadena en MiB o GB. Por ejemplo, puede especificar un valor de memoria `3072` en MiB o `3 GB` en GB. Puede indicar los valores de la CPU en el archivo JSON como una cadena en unidades de CPU o CPU virtuales (vCPU). Por ejemplo, puede especificar un valor de CPU `1024` en unidades de CPU o `1 vCPU` en vCPU.

Valor de CPU	Valor de memoria	Sistemas operativos admitidos por AWS Fargate
256 (0,25 vCPU)	512 MiB, 1 GB, 2 GB	Linux
512 (0,5 vCPU)	1 GB, 2 GB, 3 GB, 4 GB	Linux
1024 (1 vCPU)	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB	Linux, Windows
2048 (2 vCPU)	Entre 4 GB y 16 GB en incrementos de 1 GB	Linux, Windows
4096 (4 vCPU)	Entre 8 GB y 30 GB en incrementos de 1 GB	Linux, Windows
8192 (8 vCPU) **Note** Esta opción requiere una plataforma Linux `1.4.0` o posterior.	Entre 16 GB y 60 GB en incrementos de 4 GB	Linux
16 384 (16 vCPU)	Entre 32 GB y 120 GB en incrementos de 8 GB	Linux

Valor de CPU	Valor de memoria	Sistemas operativos admitidos por AWS Fargate
Note Esta opción requiere una plataforma Linux `1.4.0` o posterior.		

Integración en red de las tareas

Las tareas de Amazon ECS para AWS Fargate requieren el modo de red `awsvpc`, que proporciona a cada tarea una interfaz de red elástica. Cuando se ejecuta una tarea o se crea un servicio con este modo de red, debe especificar una o más subredes para asociar la interfaz de red y uno o más grupos de seguridad para aplicarlo a la interfaz de red.

Si va a usar subredes públicas, decida si desea proporcionar una dirección IP pública para la interfaz de red. Para que las tareas de Fargate de una subred pública extraigan imágenes de contenedor, es necesario asignar una dirección IP pública a la interfaz de red elástica de la tarea con una ruta a Internet o una gateway NAT que pueda dirigir las solicitudes a Internet. Para que las tareas de Fargate de una subred privada extraigan imágenes de contenedor, debe haber una gateway NAT en la subred para dirigir las solicitudes a Internet. Cuando aloja las imágenes de contenedor en Amazon ECR, puede configurar Amazon ECR para que utilice un punto de enlace de la VPC de interfaz. En este caso, la dirección IPv4 privada de la tarea se utiliza para extraer la imagen. Para obtener más información acerca de los puntos de conexión de la interfaz de Amazon ECR, consulte Puntos de conexión de VCP de la interfaz de Amazon ECR (AWS PrivateLink) en la Guía del usuario de Amazon Elastic Container Registry.

A continuación, se muestra un ejemplo de la sección `networkConfiguration` de un servicio de Fargate:

```
"networkConfiguration": {
   "awsvpcConfiguration": {
      "assignPublicIp": "ENABLED",
      "securityGroups": [ "sg-12345678" ],
      "subnets": [ "subnet-12345678" ]
   }
```

```
}
```

Límites de recursos de tareas

Las definiciones de tareas de Amazon ECS para contenedores Linux en AWS Fargate admiten el parámetro `ulimits` para definir los límites de recursos que se van a establecer en un contenedor.

Las definiciones de tareas de Amazon ECS para Windows en AWS Fargate admiten el parámetro `ulimits` para definir los límites de recursos que se van a establecer en un contenedor.

Las tareas de Amazon ECS alojadas en Fargate utilizan los valores límite de recursos predeterminados que establece el sistema operativo, a excepción del parámetro límite de recursos `nofile`. El límite de recursos `nofile` define una restricción en el número de archivos abiertos que puede utilizar un contenedor. En Fargate, el límite flexible `nofile` predeterminado es `65535` y el límite invariable es `65535`. Puede establecer los valores de ambos límites en un valor máximo de `1048576`.

El siguiente es un fragmento de código de definición de tareas de ejemplo que muestra cómo definir un límite `nofile` personalizado que se ha duplicado:

```
"ulimits": [
    {
       "name": "nofile",
       "softLimit": 2048,
       "hardLimit": 8192
    }
]
```

Para obtener más información acerca de los otros límites de recursos que se pueden ajustar, consulte Límites de recursos.

Registro

Registro de eventos

Amazon ECS registra las acciones que realiza en EventBridge. Puede utilizar eventos de Amazon ECS para EventBridge con el fin de recibir notificaciones casi en tiempo real sobre el estado actual de los clústeres, servicios y tareas de Amazon ECS. Además, puede automatizar acciones para responder a estos eventos. Para obtener más información, consulte Automaticización de las respuestas a los errores de Amazon ECS mediante EventBridge.

Registro del ciclo de vida de tareas

Las tareas que se ejecutan en Fargate publican marcas temporales para realizar un seguimiento de la tarea a través de los estados de su ciclo de vida. Puede ver las marcas temporales en los detalles de la tarea en la AWS Management Console y describiendo la tarea en la AWS CLI y en los SDK. Por ejemplo, puedes utilizar las marcas de temporales para evaluar cuánto tiempo ha dedicado la tarea a descargar las imágenes del contenedor y decidir si debe optimizar el tamaño de las imágenes del contenedor o utilizar índices Seekable OCI. Para obtener más información acerca de las prácticas de imágenes de contenedor, consulte Prácticas recomendadas para las imágenes de contenedores de Amazon ECS.

Registro de la aplicación

Las definiciones de tareas de Amazon ECS para AWS Fargate admiten los controladores de registros `awslogs`, `splunk` y `awsfirelens` para la configuración de registros.

El controlador de registros `awslogs` configura las tareas de Fargate para que envíen información de registro a Amazon CloudWatch Logs. A continuación se muestra un fragmento de definición de tarea donde se configura el controlador de registros `awslogs`:

```
"logConfiguration": {
   "logDriver": "awslogs",
   "options": {
      "awslogs-group" : "/ecs/fargate-task-definition",
      "awslogs-region": "us-east-1",
      "awslogs-stream-prefix": "ecs"
   }
}
```

Para obtener más información acerca de la utilización del controlador de registros `awslogs` en una definición de tareas para que envíe sus registros de contenedor a CloudWatch Logs, consulte Envío de registros de Amazon ECS a CloudWatch .

Para obtener más información acerca de cómo utilizar el controlador de registros de `awsfirelens` en una definición de tarea, consulte Envío de registros de Amazon ECS a un servicio de AWS o AWS Partner.

Para obtener más información acerca de cómo utilizar el controlador de registros de `splunk` en una definición de tarea, consulte Controlador de registros de `splunk`.

Almacenamiento de tareas

Para las tareas de Amazon ECS alojadas en Fargate, se admiten los siguientes tipos de almacenamiento:

- Los volúmenes de Amazon EBS proporcionan almacenamiento en bloques rentable, duradero y de alto rendimiento para cargas de trabajo en contenedores con uso intensivo de datos. Para obtener más información, consulte Uso de volúmenes de Amazon EBS con Amazon ECS.
- Volúmenes de Amazon EFS para almacenamiento persistente. Para obtener más información, consulte Uso de volúmenes de Amazon EFS con Amazon ECS.
- Montajes vinculados para almacenamiento efímero. Para obtener más información, consulte Uso de montajes de unión con Amazon ECS.

Carga diferida de imágenes de contenedores mediante Seekable OCI (SOCI)

Las tareas de Amazon ECS en Fargate que utilizan la versión de la plataforma Linux `1.4.0` pueden utilizar Seekable OCI (SOCI) para iniciar las tareas con mayor rapidez. Con los SOCI, los contenedores solo tardan unos segundos en extraer la imagen antes de empezar, lo que proporciona tiempo para configurar el entorno y crear instancias de la aplicación mientras la imagen se descarga en segundo plano. Esto se denomina carga diferida. Cuando Fargate inicia una tarea de Amazon ECS, Fargate detecta automáticamente si existe un índice SOCI para una imagen de la tarea e inicia el contenedor sin esperar a que se descargue la imagen completa.

En el caso de los contenedores que se ejecutan sin índices SOCI, las imágenes del contenedor se descargan completamente antes de iniciar el contenedor. Sucede lo mismo en todas las otras versiones de la plataforma de Fargate y en las AMI optimizadas para Amazon ECS en instancias de Amazon ECS.

Seekable OCI (SOCI) es una tecnología de código abierto desarrollada por AWS que puede lanzar contenedores más rápido al cargar la imagen del contenedor en diferido. SOCI funciona creando un índice (índice SOCI) de los archivos dentro de una imagen de contenedor existente. Este índice ayuda a lanzar los contenedores con mayor rapidez, lo que permite extraer un archivo individual de una imagen del contenedor antes de descargar la imagen completa. El índice SOCI debe almacenarse como un artefacto en el mismo repositorio de la imagen en el registro del contenedor. Solo debe utilizar índices SOCI de fuentes confiables, ya que el índice es la fuente autorizada del contenido de la imagen. Para obtener más información, consulte Introducción a Seekable OCI para imágenes de contenedores de carga diferida.

Los clientes que deseen utilizar SOCI solo podrán utilizar el manifiesto v2 de índices SOCI. Los clientes que hayan utilizado anteriormente SOCI en Fargate pueden seguir utilizando el manifiesto v1 de índices SOCI; sin embargo, les recomendamos encarecidamente que migren al manifiesto v2 de índices SOCI. El manifiesto v2 de índices SOCI crea una relación explícita entre las imágenes de los contenedores y sus índices SOCI para garantizar una implementación coherente.

Consideraciones

Si desea que Fargate utilice un índice SOCI para cargar imágenes de contenedores en diferido en una tarea, tenga en cuenta lo siguiente:

- Solo las tareas que se ejecutan en la versión de la plataforma de Linux `1.4.0` pueden usar índices SOCI. No se admiten tareas que ejecutan contenedores de Windows en Fargate.
- Se admiten tareas que se ejecutan en las arquitecturas de CPU X86_64 o ARM64.
- Las imágenes del contenedor de la definición de la tarea deben almacenarse en un registro de imágenes compatible. A continuación se enumeran los registros compatibles:
  - Registros privados de Amazon ECR
- Solo se admiten las imágenes de contenedor que utilizan compresión gzip o que no están comprimidas. No se admiten las imágenes de contenedor que utilizan compresión zstd.
- En el caso del manifiesto v2 de índices SOCI, al generar un manifiesto de índice SOCI, se modifica el manifiesto de la imagen del contenedor al agregar una anotación para el índice SOCI. Esto resulta en un nuevo resumen de la imagen del contenedor. El contenido de las capas del sistema de archivos de las imágenes del contenedor no cambia.
- En el caso del manifiesto v2 de índices SOCI, si la imagen del contenedor ya se ha almacenado en el repositorio de imágenes del contenedor, después de generar un índice SOCI, es necesario volver a enviar la imagen del contenedor. El reenvío de una imagen de contenedor no aumentará los costos de almacenamiento mediante la duplicación de las capas del sistema de archivos, solo se cargará un nuevo archivo de manifiesto.
- Recomendamos que pruebe la carga diferida con imágenes de contenedores con un tamaño superior a 250 MiB comprimido. Es menos probable que se reduzca el tiempo de carga de imágenes más pequeñas.
- Como la carga diferida puede cambiar el tiempo que tardan en empezar las tareas, es posible que deba que cambiar varios tiempos de espera, como el período de gracia de las comprobaciones de estado de Elastic Load Balancing.
- Si desea evitar que la imagen de un contenedor se cargue de manera diferida, la imagen del contenedor deberá ser reenviada sin un índice SOCI adjunto.

Crear un índice Seekable OCI

Para que una imagen de un contenedor se cargue de forma progresiva, es necesario crear un índice SOCI (un archivo de metadatos) y almacenarlo en el repositorio de imágenes del contenedor junto con la imagen del contenedor. Para crear y enviar un índice SOCI, puede utilizar la herramienta de código abierto soci-snapshotter CLI en GitHub. O bien, puede implementar CloudFormation AWS SOCI Index Builder. Se trata de una solución sin servidor que crea y envía automáticamente un índice SOCI cuando se envía una imagen de contenedor a Amazon ECR. Para más información sobre la solución y los pasos de instalación, consulte CloudFormation AWS SOCI Index Builder en GitHub. CloudFormation AWS SOCI Index Builder es una manera más sencilla de automatizar la introducción a SOCI, mientras que la herramienta SOCI de código abierto ofrece más flexibilidad en cuanto a la generación de índices y permite integrar la generación de índices en los procesos de integración continua y entrega continua (CI/CD).

> **Note**
>
> Para crear el índice SOCI para una imagen, esta debe existir en el almacén de imágenes containerd del `soci-snapshotter`de la computadora en ejecución. Si la imagen está en el almacén de imágenes de Docker, esta no se puede encontrar.

Verificar que una tarea utilizó la carga diferida

Para verificar que una tarea se cargó en diferido mediante SOCI, compruebe el punto de conexión de los metadatos de la tarea desde dentro de la tarea. Cuando ejecuta una consulta a la versión 4 del punto de conexión de los metadatos de la tarea, hay un campo de `Snapshotter` en la ruta predeterminada para el contenedor desde el que se ejecuta la consulta. Además, hay campos de `Snapshotter` para cada contenedor en la ruta `/task`. El valor predeterminado de este campo es `overlayfs`, y este campo se establece en `soci` si se utiliza SOCI. Para verificar que una imagen de contenedor tiene un manifiesto v2 de índices SOCI adjunto, puede recuperar el índice de imágenes de Amazon ECR mediante la AWS CLI.

```
IMAGE_REPOSITORY=r
IMAGE_TAG=latest

aws ecr batch-get-image \
    --repository-name=$IMAGE_REPOSITORY \
    --image-ids imageTag=$IMAGE_TAG \
```

```
    --query 'images[0].imageManifest' --output text | jq -r '.manifests[] |
 select(.artifactType=="application/vnd.amazon.soci.index.v2+json")'
```

Para comprobar que una imagen de contenedor tiene un manifiesto v1 de índices SOCI adjunto, puede utilizar la API de referencia de OCI.

```
ACCOUNT_ID=111222333444
AWS_REGION=us-east-1
IMAGE_REPOSITORY=nginx-demo
IMAGE_TAG=latest
IMAGE_DIGEST=$(aws ecr describe-images --repository-name $IMAGE_REPOSITORY --image-ids
 imageTag=$IMAGE_TAG --query 'imageDetails[0].imageDigest' --output text)
ECR_PASSWORD=$(aws ecr get-login-password)

curl \
    --silent \
    --user AWS:$ECR_PASSWORD \
    https://$ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/v2/$IMAGE_REPOSITORY/
referrers/$IMAGE_DIGEST?artifactType=application%2Fvnd.amazon.soci.index.v1%2Bjson | jq
 -r '.'
```

Diferencias en la definición de tareas de Amazon ECS para instancias de EC2 que ejecutan Windows

Las tareas que se ejecutan en instancias de Windows de EC2 no admiten todos los parámetros de definición de tareas de Amazon ECS disponibles. Algunos parámetros directamente no son compatibles, y otros se comportan de manera distinta.

Los siguientes parámetros de la definición de tareas no se admiten para las definiciones de tareas de Windows de Amazon EC2:

- `containerDefinitions`
  - `disableNetworking`
  - `dnsServers`
  - `dnsSearchDomains`
  - `extraHosts`
  - `links`
  - `linuxParameters`

  - `privileged`
  - `readonlyRootFilesystem`
  - `user`
  - `ulimits`
- `volumes`
  - `dockerVolumeConfiguration`
- `cpu`

  Le recomendamos que especifique la CPU de nivel de contenedor para los contenedores de Windows.
- `memory`

  Le recomendamos que especifique la memoria de nivel de contenedor para los contenedores de Windows.
- `proxyConfiguration`
- `ipcMode`
- `pidMode`
- `taskRoleArn`

  Los roles de IAM para las tareas en instancias de Windows de EC2 requieren una configuración adicional, pero gran parte de esta configuración es similar a la de los roles de IAM para las tareas en instancias de contenedor de Linux. Para obtener más información, consulte the section called “ Configuración adicional de las instancias de Amazon EC2 de Windows”.

Creación de una definición de tareas de Amazon ECS mediante la consola

Cree una definición de tarea de forma que pueda definir la aplicación que se ejecuta como tarea o servicio.

Al crear una definición de tarea para el tipo de lanzamiento externo, debe crear la definición de tarea mediante el editor JSON y establecer el parámetro `requireCapabilities` en `EXTERNAL`.

Para crear una definición de tarea, utilice la consola o especifique un archivo JSON.

Validación de JSON

El editor JSON de la consola de Amazon ECS valida lo siguiente en el archivo JSON:

- El archivo es un archivo JSON válido.
- El archivo no contiene claves extrañas.
- El archivo contiene el parámetro `familyName`.
- Hay por lo menos una entrada en `containerDefinitions`.

Pilas de AWS CloudFormation

El siguiente comportamiento se aplica a las definiciones de tareas que se crearon en la nueva consola de Amazon ECS antes del 12 de enero de 2023.

Al crear una definición de tareas, la consola de Amazon ECS crea automáticamente una pila de CloudFormation cuyo nombre comienza por `ECS-Console-V2-TaskDefinition-`. Si utilizó la AWS CLI o el AWS SDK para anular el registro de la definición de tareas, debe eliminar manualmente la pila de la definición de tareas. Para obtener más información, consulte Elimina una pila en la Guía del usuario de AWS CloudFormation.

Para las definiciones de tareas creadas después del 12 de enero de 2023, no se crea automáticamente una pila de CloudFormation.

Procedimiento

Amazon ECS console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, elija Task Definitions (Definiciones de tareas).
3. En el menú Crear una nueva definición de tarea, elija Crear una nueva definición de tarea.
4. Para Task definition family (Familia de definiciones de tareas), especifique un nombre único para la definición de tareas.
5. En Tipo de lanzamiento, elija el entorno de la aplicación. El valor predeterminado de la consola es AWS Fargate (que es sin servidor). Amazon ECS utiliza este valor para la validación y garantizar que los parámetros de la definición de tareas sean válidos para el tipo de infraestructura.

6. Para Operating system/Architecture (Arquitectura y sistema operativo), elija el sistema operativo y la arquitectura de CPU para la tarea.

   Para ejecutar la tarea en una arquitectura ARM de 64 bits, elija Linux/ARM64. Para obtener más información, consulte the section called “Plataforma de tiempo de ejecución”.

   Para ejecutar sus tareas de AWS Fargate en contenedores de Windows, elija un sistema operativo compatible con Windows. Para obtener más información, consulte the section called “Arquitecturas y sistemas operativos”.

7. En Task size (Tamaño de tarea), elija los valores de CPU y memoria que desea reservar para la tarea. El valor de CPU se especifica como vCPU y la memoria se especifica como GB.

   Para las tareas alojadas en Fargate, en la siguiente tabla, se muestran las combinaciones de CPU y memoria válidas.

Valor de CPU	Valor de memoria	Sistemas operativos admitidos por AWS Fargate
256 (0,25 vCPU)	512 MiB, 1 GB, 2 GB	Linux
512 (0,5 vCPU)	1 GB, 2 GB, 3 GB, 4 GB	Linux
1024 (1 vCPU)	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB	Linux, Windows
2048 (2 vCPU)	Entre 4 GB y 16 GB en incrementos de 1 GB	Linux, Windows
4096 (4 vCPU)	Entre 8 GB y 30 GB en incrementos de 1 GB	Linux, Windows
8192 (8 vCPU) Note Esta opción requiere una	Entre 16 GB y 60 GB en incrementos de 4 GB	Linux

Valor de CPU	Valor de memoria	Sistemas operativos admitidos por AWS Fargate
plataforma Linux `1.4.0` o posterior.		
16 384 (16 vCPU) Note Esta opción requiere una plataforma Linux `1.4.0` o posterior.	Entre 32 GB y 120 GB en incrementos de 8 GB	Linux

En el caso de las tareas que usan EC2 o el tipo de lanzamiento externo, los valores de CPU para tareas admitidos están entre 128 unidades de CPU (0,125 vCPU) y 196 608 unidades de CPU (192 vCPU).

Para indicar el valor de memoria en GB, ingrese GB después del valor. Por ejemplo, para establecer Memory value en 3 GB, ingrese 3GB.

> **Note**
> Los parámetros de CPU y memoria de nivel de tarea se omiten para los contenedores de Windows.

8. Para Network mode (Modo de red), elija el modo de red que desea utilizar. El valor predeterminado es el modo awsvpc. Para obtener más información, consulte redes de tareas de Amazon ECS.

   Si elige Puente, en Asignaciones de puertos, para Puerto del host, ingrese el número de puerto en la instancia de contenedor que se debe reservar para el contenedor.

9. (Opcional) Amplíe la sección Roles de tareas para configurar los roles de AWS Identity and Access Management (IAM) de la tarea:

a. En Task role (Rol de la tarea), elija el rol de IAM para asignar a la tarea. Un rol de IAM de tarea proporciona permisos para los contenedores de una tarea para llamar a las operaciones de la API de AWS.

b. En Rol de ejecución de tareas, elija rol.

   Para obtener más información sobre cuándo utilizar el rol de ejecución de tareas, consulte the section called “Rol de IAM de ejecución de tareas”. Si no necesita el rol, elija Ninguno.

10. (Opcional) Amplíe la sección Ubicación de tareas para agregar restricciones de ubicación. Las restricciones de ubicación de tareas permiten filtrar las instancias de contenedores utilizadas para la ubicación de las tareas mediante atributos integrados o personalizados.

11. (Opcional) Amplíe la sección Inyección de errores para habilitar la inyección de errores. La inyección de errores permite probar cómo la aplicación responde a determinados escenarios de deterioro.

12. Para definir a cada contenedor en su definición de tareas, siga los pasos que se describen a continuación.

    a. En Name (Nombre), escriba un nombre para el contenedor.

    b. En Image URI (URI de imagen), ingrese la imagen que se va a usar para iniciar un contenedor. Las imágenes del registro de Galería pública de Amazon ECR se pueden especificar solo mediante el uso del nombre de registro público de Amazon ECR. Por ejemplo, si se indica `public.ecr.aws/ecs/amazon-ecs-agent:latest`, se utiliza el contenedor de Amazon Linux alojado en Galería pública de Amazon ECR. Para todos los demás repositorios, indique el repositorio mediante los formatos `repository-url/image:tag` o `repository-url/image@digest`.

    c. Si la imagen se encuentra en un registro privado ajeno a Amazon ECR, en Registro privado, active la Autenticación del registro privado. A continuación, en nombre o ARN de Secrets Manager, introduzca el nombre de recurso de Amazon (ARN) del secreto.

    d. En Contenedor esencial, si la definición de tarea tiene dos o varios contenedores definidos, puede especificar si el contenedor debe considerarse esencial. Cuando un contenedor está marcado como Esencial, si se detiene, la tarea se detiene. Cada definición de tarea debe contener al menos un contenedor esencial.

    e. Las asignaciones de puertos permiten a los contenedores acceder a puertos en el host para enviar o recibir tráfico. En Port mappings (Asignaciones de puertos), realice una de las siguientes operaciones:

- Cuando usa el modo de red awsvpc, en Container port (Puerto del contenedor) y Protocol (Protocolo), elija la asignación de puertos que se va a usar para el contenedor.
- Cuando usa el modo de red bridge (puente), en Container port (Puerto del contenedor) y Protocol (Protocolo), elija la asignación de puertos que se va a usar para el contenedor.

Elija Add more port mappings (Agregar más asignaciones de puertos) para especificar asignaciones de puertos de contenedores adicionales.

f. Para conceder al contenedor acceso de solo lectura a su sistema de archivos raíz, en Sistema de archivos raíz de solo lectura, seleccione Solo lectura.

g. (Opcional) Para definir los límites de CPU, GPU y memoria a nivel de contenedor que sean diferentes de los valores a nivel de tarea incluidos en Resource allocation limits, haga lo siguiente:

- En CPU, ingrese el número de unidades de CPU que el agente de contenedor de Amazon ECS reserva para el contenedor.
- En GPU, ingrese el número de unidades de GPU para la instancia de contenedor.

  Una instancia de Amazon EC2 compatible con GPU tiene 1 unidad de GPU por cada GPU. Para obtener más información, consulte the section called “Definiciones de tareas para cargas de trabajo de GPU”.
- En Límite estricto de memoria, ingrese la cantidad de memoria, en GB, para presentarla al contenedor. Si el contenedor intenta superar el límite duro, se cancela el contenedor.
- El daemon de Docker 20.10.0 o posterior reserva un mínimo de 6 mebibytes (MiB) de memoria para un contenedor, por tanto no indique menos de 6 MiB de memoria para los contenedores.

  El daemon de Docker 19.03.13-ce o anterior reserva un mínimo de 4 MiB de memoria para un contenedor, por tanto no indique menos de 4 MiB de memoria para los contenedores.
- En Límite flexible de memoria, ingrese el límite flexible (en GB) de memoria que reservar para el contenedor.

Cuando la memoria del sistema está en conflicto, Docker intenta mantener la memoria del contenedor en este límite flexible. Si no especifica memoria de nivel de tarea, debe especificar un entero distinto de cero para el Límite duro de memoria y el Límite flexible de memoria, o para ambos. Si especifica ambos, el Límite duro de memoria debe ser mayor que el Límite flexible de memoria.

Esta característica no es compatible con los contenedores de Windows.

h. (Opcional) Expanda la sección Variables de entorno para especificar variables de entorno que se van a inyectar en el contenedor. Puede indicar variables de entorno individualmente mediante pares clave-valor o de forma masiva si indica un archivo de variable de entorno alojado en un bucket de Amazon S3. Para obtener información sobre cómo dar formato a un archivo de variable de entorno, consulte Transferencia de una variable de entorno individual a un contenedor de Amazon ECS.

   Cuando especifique una variable de entorno para el almacenamiento de secretos, en Clave, ingrese el nombre del secreto. A continuación, en ValueFrom, indique el ARN completo del secreto de Almacén de parámetros de Systems Manager o del secreto de Secrets Manager.

i. (Opcional) Seleccione la opción Use log collection (Utilizar colección de registros) para especificar una configuración de registro. Para cada controlador de registro disponible, hay opciones de controladores de registro que se deben especificar. La opción predeterminada envía registros de contenedor a Registros de Amazon CloudWatch. Las demás opciones del controlador de registros se configuran mediante AWS FireLens. Para obtener más información, consulte Envío de registros de Amazon ECS a un servicio de AWS o AWS Partner.

   A continuación, se describe con más detalle cada uno de los destinos de registro de contenedor.

   - Amazon CloudWatch: configure la tarea para enviar registros de contenedor a CloudWatch Logs. Se proporcionan las opciones de controlador de registro predeterminadas que crean un grupo de registros de CloudWatch en su nombre. Para especificar otro nombre de grupo de registros, cambie los valores de las opciones del controlador.
   - Exportar registros a Splunk: configure la tarea para enviar los registros del contenedor al controlador de Splunk que envía los registros a un servicio remoto. Debe ingresar

la URL del servicio web de Splunk. El token de Splunk se indica como una opción secreta, ya que puede tratarse como información confidencial.

- Exportar registros a Amazon Data Firehose: configure la tarea para enviar registros de contenedor a Firehose. Se proporcionan las opciones de controlador de registro predeterminadas, que envían registros a un flujo de entrega de Firehose. Para especificar un nombre de flujo de entrega distinto, cambie los valores de las opciones del controlador.
- Exportar registros a Amazon Kinesis Data Streams: configure la tarea para enviar registros de contenedores a Kinesis Data Streams. Se proporcionan las opciones de controlador de registro predeterminadas que envían registros a un flujo de Kinesis Data Streams. Para especificar otro nombre de transmisión, cambie los valores de las opciones del controlador.
- Exportar registros a Amazon OpenSearch Service: configure la tarea para enviar registros de contenedor a un dominio de OpenSearch Service. Se deben proporcionar las opciones del controlador de registros.
- Exportar registros a Amazon S3: configure la tarea para enviar registros de contenedor a un bucket de Amazon S3. Se proporcionan las opciones de controlador de registro predeterminadas, pero debe especificar un nombre de bucket de Amazon S3 válido.

j. (Opcional) Configure parámetros de contenedor adicionales.

Para configurar esta opción	Haga lo siguiente	
Política de reinicio Estas opciones definen una política de reinicio para reiniciar un contenedor al salir.	Amplíe la Política de reinicio y, luego, configure los elementos siguientes: • Para habilitar una política de reinicio para el contenedor, activr Habilitar la política de reinicio. • En el caso de los códigos de salida ignorados, especifique una lista de códigos de salida de contenedores de enteros separados por comas. Si cuando sale el contenedor lo hace con alguno de los códigos de salida especificados, Amazon ECS no intentará reiniciarlo. Si no hay especificación alguna, Amazon ECS no ignorará ningún código de salida. • En el caso de Período de intento de restablec imiento, especifique un período de tiempo	

Para configurar esta opción	Haga lo siguiente	
	entero, en segundos, durante el que debe funcionar el contenedor antes de que se pueda intentar un reinicio en caso de que se produzca una salida. Amazon ECS puede intentar reiniciar un contenedor solo una vez por cada período de intento de restablec imiento de segundos. Si no se especifica nada, el contenedor debe funcionar durante 300 segundos antes de que se pueda intentar un reinicio.	

Para configurar esta opción	Haga lo siguiente	
HealthCheck Estos son los comandos que determinan si un contenedor está en buen estado. Para obtener más información, consulte Determine el estado de las tareas de Amazon ECS mediante comprobaciones de estado de los contenedores.	Expanda HealthCheck y, a continuación, configure los siguientes elementos: • En Command (Comando), introduzca una lista de comandos separados por comas. Puede comenzar los comandos con `CMD` para ejecutar los argumentos del comando directamente, o por `CMD-SHELL` para ejecutar el comando con el intérprete de comandos predeterminado del contenedor. Si no se especifica ninguno, se utiliza `CMD`. • En Interval (Intervalo), introduzca el número de segundos entre cada comprobación de estado. Los valores válidos se encuentran entre 5 y 30. • En Timeout (Tiempo de espera), ingrese el periodo (en segundos) que hay	

Para configurar esta opción	Haga lo siguiente	
	que esperar para que una comprobación de estado se realice correctamente antes de que se considere un error. Los valores válidos se encuentran entre 2 y 60. • En Start period (Periodo de inicio), introduzca el periodo de tiempo (en segundos) que hay que esperar para que un contenedor se inicie antes de que se ejecuten los comandos de comprobación de estado. Los valores válidos se encuentran entre 0 y 300. • En Retries (Reintentos), introduzca el número de veces que desea volver a intentar los comandos de comprobación de estado cuando se produzca un error. Los valores válidos se encuentran entre 1 y 10.	

Para configurar esta opción	Haga lo siguiente	
Solicitud de inicio de dependencias Esta opción define las dependencias para el inicio y apagado del contenedor. Un contenedor puede contener varias dependencias.	Expanda Ordenación de la dependencia de inicio y, a continuación, configure lo siguiente: a. Seleccione Agregar dependencia de contenedor. b. En Contenedor, elija el contenedor. c. En Condición, elija la condición de dependencia de inicio. Para agregar una dependencia adicional, elija Agregar dependencia de contenedor.	

Para configurar esta opción	Haga lo siguiente	
Tiempos de espera de contenedor Estas opciones determina n cuándo iniciar y detener un contenedor.	Expanda Tiempos de espera del contenedor y, a continuación, configure lo siguiente: • Para configurar el tiempo de espera antes de dejar de resolver dependencias para un contenedor, en Tiempo de espera de inicio, ingrese el número de segundos. • Para configurar el tiempo de espera antes de que el contenedor se detenga si no sale normalmente por sí mismo, en Tiempo de espera de inicio, ingrese el número de segundos.	

Para configurar esta opción	Haga lo siguiente	
Configuraciones de red del contenedor Estas opciones determina n si se deben utilizar las redes dentro de un contenedor.	Expanda Configuraciones de red del contenedor y, a continuación, configure lo siguiente: • Para deshabilitar las redes de contenedores, seleccione Desactivar las redes. • Para configurar las direcciones IP de los servidores DNS que se presentan en el contenedor, en Servidores DNS, introduzca la dirección IP de cada servidor en una línea independi ente. • Para configurar que los dominios DNS busquen los nombres de host incomplet os que se presentan en el contenedor, en Dominios de búsqueda de DNS, ingrese cada dominio en una línea independiente.	

Para configurar esta opción	Haga lo siguiente	
	El patrón es **`^[a-zA-Z0-9-.]{0,253}[a-zA-Z0-9]$`** . • Para configurar el nombre de host del contenedor, en Nombre de host, introduzca el nombre goat del contenedor. • Para agregar los nombres de host y las asignaciones de direcciones IP que se adjuntan al archivo `/etc/hosts` del contenedor, elija Agregar host adicional y, a continuación, en Nombre de host y Dirección IP, introduzca el nombre del host y la dirección IP.	

Para configurar esta opción	Haga lo siguiente	
Configuración de Docker Estos anulan los valores de Dockerfile.	Amplíe Configuración de Docker y, luego, configure los elementos siguientes: • En Comando, ingrese un comando ejecutable para un contenedor. Este parámetro se corresponde con `Cmd` en la sección Create a container de la API remota de Docker y la opción `COMMAND` a `docker run`. Este parámetro anula la instrucción de `CMD` de un Dockerfile. • En Punto de entrada, ingrese el ENTRYPOINT de Docker que se transfiere al contenedor. Este parámetro se corresponde con `Entrypoint` en la sección Create a container de la API remota de Docker y la opción `--entrypoint` a `docker run`. Este parámetro	

Para configurar esta opción	Haga lo siguiente	
	anula la instrucción de `ENTRYPOINT` de un Dockerfile. • En Directorio de trabajo, ingrese el directorio en el que el contenedor ejecutará cualquier punto de entrada e instrucción de comando proporcionados. Este parámetro se corresponde con `WorkingDir` en la sección Create a container de la API remota de Docker y la opción `--workdir` a `docker run`. Este parámetro anula la instrucción de `WORKDIR` de un Dockerfile.	

Para configurar esta opción	Haga lo siguiente
Límites de recursos (Ulimits) Estos valores sobrescriben la configuración predeterminada de la cuota de recursos para el sistema operativo. Este parámetro se corresponde con `Ulimits` en la sección Crear un contenedor de la API remota de Docker y con la opción `--ulimit` de docker run.	Amplíe Límites de recursos (ulimits) y, luego, elija Agregar ulimit. En Nombre del límite, elija el límite. A continuación, en Límite flexible y Límite invariable, introduzca los valores. Para agregar más ulimits, elija Agregar ulimit.
etiquetas de Docker Esta opción agrega metadatos al contenedor. Este parámetro se corresponde con `Labels` en la sección Crear un contenedor de la API remota de Docker y con la opción `--label` de docker run.	Amplíe Etiquetas de Docker, elija Agregar par clave-valor y, luego, ingrese la clave y el valor. Para agregar más etiquetas de Docker, seleccione Agregar par clave-valor.

k. (Opcional) Elija Add more containers (Agregar más contenedores) para agregar contenedores adicionales a la definición de tareas.

13. (Opcional) La sección Almacenamiento se utiliza para ampliar la cantidad de almacenamiento efímero de las tareas alojadas en Fargate. También puede utilizar esta sección para agregar una configuración de volumen de datos para la tarea.

- Para ampliar el almacenamiento efímero disponible más allá del valor predeterminado de 20 gibibytes (GiB) para las tareas de Fargate, en Amount (Cantidad), ingrese un valor de hasta 200 GiB.

14. (Opcional) Para agregar una configuración de volumen de datos para la definición de tareas, elija Agregar volumen y, a continuación, siga estos pasos.

    a. En Volume name (Nombre del volumen), ingrese un nombre para el volumen de datos. El nombre del volumen de datos se utiliza al crear un punto de montaje de contenedor.

    b. En Configuración de volumen, seleccione si quiere configurar el volumen al crear la definición de la tarea o durante la implementación.

    > **Note**
    >
    > Entre los volúmenes que se pueden configurar al crear una definición de tareas se incluyen Montaje de enlace, Docker, Amazon EFS y Amazon FSx para Windows File Server. Entre los volúmenes que se pueden configurar en el momento de la implementación, al ejecutar una tarea o al crear o actualizar un servicio, se incluye Amazon EBS.

    c. En Tipo de volumen, seleccione un tipo de volumen compatible con el tipo de configuración que seleccionó y, luego, configure el tipo de volumen.

Tipo de volumen	Pasos	
Montaje de unión	a. Elija Add mount point (Agregar punto de montaje) y, a continuación, configure lo siguiente: • Para Container (Contenedor), elija el contenedor para el punto de montaje.	

Tipo de volumen	Pasos	
	• Para Source volume (Volumen de origen), elija el volumen de datos que desea montar en el contenedor. • En Container path (Ruta del contenedor), ingrese la ruta del contenedor en el cual montar el volumen. • En Solo lectura, seleccione si el contenedor tiene acceso de solo lectura al volumen. b. Para agregar puntos de montaje adicionales, utilice Add mount point (Agregar punto de montaje).	

Tipo de volumen	Pasos	
EFS	a. En File system ID (ID del sistema de archivos) , seleccione el ID del sistema de archivos de Amazon EFS. b. (Opcional) En Root directory (Directorio raíz), ingrese el directori o dentro del archivo de Amazon EFS que se va a montar como directorio raíz dentro del host. Si se omite este parámetro, se utiliza la raíz del volumen de Amazon EFS. Si piensa utilizar un punto de acceso de EFS, deje este campo en blanco. c. (Opcional) En Access point (Punto de acceso), elija el ID del punto de acceso que desee utilizar. d. (Opcional) Para cifrar los datos entre el sistema de archivos de Amazon EFS y el host de Amazon ECS, o para utilizar el rol de ejecución de	

Tipo de volumen	Pasos	
	tareas al montar el volumen, elija Advanced configurations (Configuraciones avanzadas) y, a continuación, configure lo siguiente: • Para cifrar los datos entre el sistema de archivos de Amazon EFS y el host de Amazon ECS, seleccione Transit encryption (Cifrado de tránsito) y, a continuación, introduzca el Port (Puerto) que se utilizará al enviar datos cifrados entre el host de Amazon ECS y el servidor de Amazon EFS. Si no se especifica un puerto de cifrado en tránsito, se emplea la estrategia de selección de puertos que utiliza el ayudante de montaje de Amazon EFS. Para obtener más información, consulte Ayudante de montaje de EFS en la Guía del usuario de Amazon Elastic File System.	

Tipo de volumen	Pasos	
	• Para utilizar el rol de IAM de tarea de Amazon ECS definido en una definición de tarea al montar el sistema de archivos de Amazon EFS, seleccione IAM authorization (Autorización de IAM). e. Elija Add mount point (Agregar punto de montaje) y, a continuación, configure lo siguiente: • Para Container (Contenedor), elija el contenedor para el punto de montaje. • Para Source volume (Volumen de origen), elija el volumen de datos que desea montar en el contenedor. • En Container path (Ruta del contenedor), ingrese la ruta del contenedor en el cual montar el volumen.	

Tipo de volumen	Pasos	
	• En Solo lectura, seleccione si el contenedor tiene acceso de solo lectura al volumen. f. Para agregar puntos de montaje adicionales, utilice Add mount point (Agregar punto de montaje).	

Tipo de volumen	Pasos	
Docker	a. En Controlador, ingrese la configuración del volumen de Docker. Los contenedores de Windows admiten solo el uso del controlador local. Para utilizar montajes vinculados, especifique un host. b. En Scope (Alcance), elija el ciclo de vida del volumen. • Para que el ciclo de vida dure cuando la tarea se inicie y se detenga, elija Task (Tarea). • Para que el volumen persista después de que se detenga la tarea, elija Shared (Compartido). c. Elija Add mount point (Agregar punto de montaje) y, a continuación, configure lo siguiente: • Para Container (Contenedor), elija	

Tipo de volumen	Pasos	
	el contenedor para el punto de montaje. • Para Source volume (Volumen de origen), elija el volumen de datos que desea montar en el contenedor. • En Container path (Ruta del contenedor), ingrese la ruta del contenedor en el cual montar el volumen. • En Solo lectura, seleccione si el contenedor tiene acceso de solo lectura al volumen. d. Para agregar puntos de montaje adicionales, utilice Add mount point (Agregar punto de montaje).	

Tipo de volumen	Pasos	
FSx para Windows File Server	a. En ID del sistema de archivos, elija el ID del sistema de archivos de FSx for Windows File Server. b. En Directorio raíz, ingrese el directorio del sistema de archivos de FSx para Windows File Server que se va a montar como directorio raíz dentro del host. c. En Parámetro de credenciales, elija cómo se almacenarán las credenciales. • Para utilizar AWS Secrets Manager, ingrese el nombre de recurso de Amazon (ARN) de un secreto de Secrets Manager. • Para utilizar AWS Systems Manager, ingrese el nombre de recurso de Amazon (ARN) de un parámetro de Systems Manager.	

Tipo de volumen	Pasos	
	d. En Dominio, ingrese el nombre de dominio completo que aloja un directorio de AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) o un directorio de EC2 de Active Directory con alojamiento propio. e. Elija Add mount point (Agregar punto de montaje) y, a continuación, configure lo siguiente: • Para Container (Contenedor), elija el contenedor para el punto de montaje. • Para Source volume (Volumen de origen), elija el volumen de datos que desea montar en el contenedor. • En Container path (Ruta del contenedor), ingrese la ruta del	

Tipo de volumen	Pasos	
	contenedor en el cual montar el volumen. • En Solo lectura, seleccione si el contenedor tiene acceso de solo lectura al volumen. f. Para agregar puntos de montaje adicionales, utilice Add mount point (Agregar punto de montaje).	

Tipo de volumen	Pasos	
Amazon EBS	a. Elija Add mount point (Agregar punto de montaje) y, a continuación, configure lo siguiente: • Para Container (Contenedor), elija el contenedor para el punto de montaje. • Para Source volume (Volumen de origen), elija el volumen de datos que desea montar en el contenedor. • En Container path (Ruta del contenedor), ingrese la ruta del contenedor en el cual montar el volumen. • En Solo lectura, seleccione si el contenedor tiene acceso de solo lectura al volumen. b. Para agregar puntos de montaje adicionales, utilice Add mount	

Tipo de volumen	Pasos	
	point (Agregar punto de montaje).	

15. Para agregar un volumen desde otro contenedor, seleccione Agregar volumen desde y, a continuación, configure lo siguiente:
    - En Contenedor, elija el contenedor.
    - En Origen, elija el contenedor que tiene el volumen que desea montar.
    - En Solo lectura, seleccione si el contenedor tiene acceso de solo lectura al volumen.
16. (Opcional) Para configurar los valores de seguimiento y recopilación de métricas de la aplicación mediante la integración de AWS Distro for OpenTelemetry, expanda Supervisión y, luego, seleccione Utilizar recopilación de métricas para recopilar y enviar las métricas de las tareas a Amazon CloudWatch o Amazon Managed Service para Prometheus. Cuando se selecciona esta opción, Amazon ECS crea un sidecar de contenedor de AWS Distro for OpenTelemetry que está preconfigurado para enviar las métricas de la aplicación. Para obtener más información, consulte Correlacionar el rendimiento de las aplicaciones de Amazon ECS mediante métricas de aplicaciones.

    a. Cuando se selecciona Amazon CloudWatch, las métricas de aplicaciones personalizadas se enrutan a CloudWatch como métricas personalizadas. Para obtener más información, consulte Exportación de métricas de aplicaciones a Amazon CloudWatch.

    > **Important**
    >
    > Al exportar métricas de aplicaciones a Amazon CloudWatch, la definición de tarea requiere un rol de IAM de tarea con los permisos necesarios. Para obtener más información, consulte Permisos de IAM necesarios para la integración de AWS Distro for OpenTelemetry Amazon CloudWatch.

    b. Cuando se selecciona Amazon Managed Service for Prometheus (Prometheus libraries instrumentation) (Amazon Managed Service for Prometheus [instrumentación de bibliotecas Prometheus]), las métricas de CPU, memoria, red y almacenamiento de nivel de tarea y las métricas de aplicaciones personalizadas se enrutan a Amazon

Managed Service for Prometheus. En Punto de enlace de escritura remota del espacio del espacio de trabajo, ingrese la URL de punto de conexión de escritura remota del espacio de trabajo de Prometheus. En Objetivo de raspado, ingrese el host y el puerto que el recopilador de AWS Distro for OpenTelemetry puede utilizar para extraer los datos de métricas. Para obtener más información, consulte Exportación de métricas de aplicaciones a Amazon Managed Service for Prometheus.

> **Important**
>
> Al exportar métricas de aplicaciones a Amazon Managed Service for Prometheus, la definición de tarea requiere un rol de IAM de tarea con los permisos necesarios. Para obtener más información, consulte Permisos de IAM necesarios para la integración de AWS Distro for OpenTelemetry con Amazon Managed Service for Prometheus.

c. Cuando selecciona Amazon Managed Service para Prometheus (instrumentación de OpenTelemetry), las métricas de CPU, memoria, red y almacenamiento de nivel de tarea y las métricas de aplicaciones personalizadas se dirigen a Amazon Managed Service para Prometheus. En Punto de enlace de escritura remota del espacio del espacio de trabajo, ingrese la URL de punto de conexión de escritura remota del espacio de trabajo de Prometheus. Para obtener más información, consulte Exportación de métricas de aplicaciones a Amazon Managed Service for Prometheus.

> **Important**
>
> Al exportar métricas de aplicaciones a Amazon Managed Service for Prometheus, la definición de tarea requiere un rol de IAM de tarea con los permisos necesarios. Para obtener más información, consulte Permisos de IAM necesarios para la integración de AWS Distro for OpenTelemetry con Amazon Managed Service for Prometheus.

17. (Opcional) Expanda la sección Tags (Etiquetas) para agregar etiquetas, como pares clave-valor, a la definición de tarea.

    - [Agregar una etiqueta] Seleccione Add tag (Agregar etiqueta), y, a continuación, haga lo siguiente:
      - En Clave, escriba el nombre de la clave.

    - En Valor, escriba el valor de la clave.
  - [Eliminar una etiqueta] Junto a la etiqueta, elija Remove tag (Quitar etiqueta).
18. Elija Crear para registrar la definición de tarea.

Amazon ECS console JSON editor

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, elija Task Definitions (Definiciones de tareas).
3. Elija Crear una nueva definición de tarea y Crear una nueva definición de tarea con JSON.
4. En el cuadro del editor de JSON, edite su archivo JSON,

   El JSON debe pasar las comprobaciones de validación especificadas en the section called "Validación de JSON".
5. Seleccione Crear.

Actualización de una definición de tareas de Amazon ECS mediante la consola

Una revisión de definición de tareas es una copia de la definición de tarea actual con los nuevos valores de parámetro que sustituyen a los existentes. Todos los parámetros que no modifica se encuentran en la nueva revisión.

Para actualizar una definición de tarea, cree una revisión de definición de tarea. Si la definición de tarea se utiliza en un servicio, debe actualizar ese servicio para usar la definición de tarea actualizada.

Al crear una revisión, puede modificar las siguientes propiedades de contenedor y de entorno.

- ID de imagen de contenedor
- Mapeos de puertos
- Variables de entorno
- Tamaño de tarea
- Tamaño del contenedor
- Rol de la tarea
- Rol de ejecución de tareas

- Volúmenes y puntos de montaje de contenedores
- Registro privado

Validación de JSON

El editor JSON de la consola de Amazon ECS valida lo siguiente en el archivo JSON:

- El archivo es un archivo JSON válido
- El archivo no contiene claves extrañas
- El archivo contiene el parámetro `familyName`
- Hay por lo menos una entrada en `containerDefinitions`

Procedimiento

Amazon ECS console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, seleccione la Región que contiene la definición de tarea.
3. En el panel de navegación, elija Task Definitions (Definiciones de tareas).
4. Elija la definición de tareas.
5. Seleccione la revisión de las definiciones de tareas y, a continuación, elija Crear nueva revisión y Crear nueva revisión.
6. En la página Create new task definition revision (Crear nueva revisión de definición de tarea), realice cambios. Por ejemplo, para cambiar las definiciones de contenedor existentes (como la imagen de contenedor, los límites de memoria, o las asignaciones de puertos), seleccione el contenedor, realice los cambios.
7. Verifique la información y, luego, seleccione Actualizar.
8. Si la definición de tarea se utiliza en un servicio, actualice su servicio con la definición de tarea actualizada. Para obtener más información, consulte Actualización de un servicio de Amazon ECS.

Amazon ECS console JSON editor

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.

2. En el panel de navegación, elija Task Definitions (Definiciones de tareas).
3. Elija Create new revision (Crear nueva revisión) y Create new revision with JSON (Crear nueva revisión con JSON).
4. En el cuadro del editor de JSON, edite su archivo JSON,

   El JSON debe pasar las comprobaciones de validación especificadas en the section called "Validación de JSON".
5. Seleccione Crear.

Anulación del registro de la revisión de una definición de tareas de Amazon ECS mediante la consola

Puede anular el registro de la revisión de la definición de la tarea para que ya no se muestre en las llamadas a la API de `ListTaskDefinition` ni en la consola cuando quiera ejecutar una tarea o actualizar un servicio.

Cuando se cancela el registro de una revisión de definición de tarea, se marca de inmediato como `INACTIVE`. Las tareas y servicios existentes que hacen referencia a una revisión de la definición de tarea `INACTIVE` continúan ejecutándose sin interrupciones. Para ajustar la escala de los servicios existentes que hacen referencia a una revisión de la definición de tarea `INACTIVE`, puede modificar el recuento deseado del servicio.

No se puede utilizar una revisión de la definición de tarea `INACTIVE` para ejecutar nuevas tareas ni crear nuevos servicios. Tampoco se puede actualizar un servicio existente para hacer referencia a una revisión de la definición de tarea `INACTIVE` (aunque podría haber una ventana de hasta 10 minutos después de la anulación del registro en la que estas restricciones aún no hayan surtido efecto).

> **Note**
>
> Cuando se anulan los registros de todas las revisiones de una familia de tareas, la familia de definición de tareas se traslada a la lista `INACTIVE`. Al agregar una nueva revisión de una definición de tarea `INACTIVE`, la familia de definiciones de tareas vuelve a aparecer en la lista `ACTIVE`.
> En este momento, las revisiones de la definición de tarea `INACTIVE` permanecen visibles en su cuenta indefinidamente. Sin embargo, este comportamiento está sujeto a cambio en el

futuro. Por lo tanto, no debe confiar en que las revisiones de la definición de tarea `INACTIVE` persistan más allá del ciclo de vida de las tareas y servicios asociados.

Pilas de AWS CloudFormation

El siguiente comportamiento se aplica a las definiciones de tareas que se crearon en la nueva consola de Amazon ECS antes del 12 de enero de 2023.

Al crear una definición de tareas, la consola de Amazon ECS crea automáticamente una pila de CloudFormation cuyo nombre comienza por `ECS-Console-V2-TaskDefinition-`. Si utilizó la AWS CLI o el AWS SDK para anular el registro de la definición de tareas, debe eliminar manualmente la pila de la definición de tareas. Para obtener más información, consulte Elimina una pila en la Guía del usuario de AWS CloudFormation.

Para las definiciones de tareas creadas después del 12 de enero de 2023, no se crea automáticamente una pila de CloudFormation.

Procedimiento

Para anular el registro de una nueva definición de tareas (consola de Amazon ECS)

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, seleccione la región que contiene la definición de tarea.
3. En el panel de navegación, elija Task Definitions (Definiciones de tareas).
4. En la página Task Definitions (Definiciones de tareas), elija la familia de definiciones de tareas que contiene una o más revisiones para las que desea anular el registro.
5. En la página Nombre de la definición de tarea, seleccione las revisiones que desee eliminar y, a continuación, elija Acciones, Anular registro.
6. Verifique la información en la ventana Deregister (Anular registro) y elija Deregister (Anular registro) para finalizar.

Eliminación de una revisión de definición de tareas de Amazon ECS con la consola

Cuando ya no necesite una revisión de definición de tarea específica en Amazon ECS, puede eliminarla.

Cuando se elimina una revisión de definición de tareas, de inmediato pasa del estado `INACTIVE` al `DELETE_IN_PROGRESS`. Las tareas y los servicios existentes que hacen referencia a una revisión de definición de tareas en estado `DELETE_IN_PROGRESS` continúan ejecutándose sin interrupciones.

No se puede utilizar una revisión de definición de tareas en estado `DELETE_IN_PROGRESS` para ejecutar nuevas tareas ni crear nuevos servicios. Tampoco se puede actualizar un servicio existente para hacer referencia a una revisión de definición de tareas en estado `DELETE_IN_PROGRESS`.

Al eliminar todas las revisiones de la definición de tareas `INACTIVE`, el nombre de la definición de tareas no se muestra en la consola ni se devuelve en la API. Si una revisión de definición de tareas tiene el estado `DELETE_IN_PROGRESS`, el nombre de la definición de tareas se muestra en la consola y se devuelve en la API. Amazon ECS retiene el nombre de la definición de tarea y la revisión se incrementa la próxima vez que cree una definición de tarea con ese nombre.

Recursos de Amazon ECS que pueden bloquear una eliminación

Una solicitud de eliminación de definición de tareas no se completará cuando haya recursos de Amazon ECS que dependan de la revisión de la definición de tareas. Los siguientes recursos pueden impedir que se elimine una definición de tareas:

- Tareas independientes de Amazon ECS: la definición de la tarea es necesaria para que la tarea se mantenga en buen estado.
- Tareas de servicio de Amazon ECS: la definición de la tarea es necesaria para que la tarea se mantenga en buen estado.
- Implementaciones de servicio y conjuntos de tareas de Amazon ECS: la definición de la tarea es necesaria cuando se inicia un evento de escalado para una implementación o conjunto de tareas de Amazon ECS.

Si la definición de la tarea permanece en el estado `DELETE_IN_PROGRESS`, puede utilizar la consola o la AWS CLI para identificar y, a continuación, detener los recursos que bloquean la eliminación de la definición de la tarea.

Eliminación de la definición de tareas después de eliminar el recurso bloqueado

Las siguientes reglas se aplican después de eliminar los recursos que bloquean la eliminación de la definición de tarea:

- Tareas de Amazon ECS: la eliminación de la definición de tareas puede tardar hasta 1 hora en completarse una vez detenida la tarea.

- Implementaciones de servicio y conjuntos de tareas de Amazon ECS: la eliminación de la definición de tareas puede tardar hasta 24 horas en completarse una vez que se haya eliminado la implementación o el conjunto de tareas.

Procedimiento

Para eliminar definiciones de tareas (consola de Amazon ECS)

Debe anular el registro de la revisión de la definición de tareas antes de eliminarla. Para obtener más información, consulte the section called “Anulación del registro de la revisión de una definición de tareas con la consola”.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, seleccione la región que contiene la definición de tarea.
3. En el panel de navegación, elija Task Definitions (Definiciones de tareas).
4. En la página Definiciones de tareas, elija la familia de definiciones de tareas que contenga una o más revisiones que desee eliminar.
5. En la página Nombre de la definición de tarea, seleccione las revisiones que quiere eliminar y, a continuación, elija Acciones, Eliminar.

   Si la opción Eliminar no está disponible, debe anular el registro de la definición de tarea.
6. Compruebe la información en el cuadro de confirmación Eliminar y, luego, elija Eliminar para finalizar.

Casos de uso de definiciones de tareas de Amazon ECS

Obtenga más información sobre cómo escribir definiciones de tareas para varios servicios y características de AWS.

En función de la carga de trabajo, hay parámetros determinados de la definición de tareas que deben configurarse. Además, para el tipo de lanzamiento de EC2, debe elegir instancias específicas que están diseñadas para la carga de trabajo.

Temas

- Definiciones de tareas de Amazon ECS para cargas de trabajo de GPU
- Definiciones de tareas de Amazon ECS para cargas de trabajo de transcodificación de video

- Definiciones de tareas de Amazon ECS para cargas de trabajo de machine learning de AWS Neuron
- Definiciones de tareas de Amazon ECS para instancias de aprendizaje profundo
- Definiciones de tareas de Amazon ECS para cargas de trabajo de ARM de 64 bits
- Envío de registros de Amazon ECS a CloudWatch
- Envío de registros de Amazon ECS a un servicio de AWS o AWS Partner
- Uso de imágenes de contenedor que no sean de AWS en Amazon ECS
- Reinicio de contenedores individuales en tareas de Amazon ECS con políticas de reinicio de contenedores
- Transferencia de datos confidenciales a un contenedor de Amazon ECS

Definiciones de tareas de Amazon ECS para cargas de trabajo de GPU

Amazon ECS admite cargas de trabajo que utilizan unidades GPU cuando se crean clústeres con instancias de contenedor que admiten GPU. Las instancias de contenedor de Amazon EC2 basadas en GPU que utilizan los tipos de instancia p2, p3, p5, g3, g4 y g5 proporcionan acceso a las GPU NVIDIA. Para obtener más información, consulte Instancias de computación acelerada de Linux en la Guía de tipos de instancias de Amazon EC2.

Amazon ECS proporciona una AMI optimizada para GPU que está preconfigurada con controladores de kernel de NVIDIA y un tiempo de ejecución de GPU de Docker. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.

Puede designar un número de GPU en su definición de tareas para la ubicación de tareas en el nivel de contenedor. Amazon ECS programa las tareas de acuerdo con las instancias de contenedor que admiten GPU disponibles y fija las GPU físicas en los contenedores correspondientes para conseguir un rendimiento óptimo.

Se admiten los siguientes tipos de instancias de Amazon EC2 basadas en GPU. Para obtener más información, consulte Instancias P2 de Amazon EC2, Instancias P3 de Amazon EC2, Instancias P4d de Amazon EC2, Instancias P5 de Amazon EC2, Instancias G3 de Amazon EC2, Instancias G4 de Amazon EC2, Instancias G5 de Amazon EC2, Instancias G6 de Amazon EC2 e Instancias G6e de Amazon EC2.

Tipo de instancia	GPU	Memoria de GPU (GiB)	vCPU	Memoria (GiB)
p3.2xlarge	1	16	8	61
p3.8xlarge	4	64	32	244
p3.16xlarge	8	128	64	488
p3dn.24xlarge	8	256	96	768
p4d.24xlarge	8	320	96	1152
p5.48xlarge	8	640	192	2048
g3s.xlarge	1	8	4	30,5
g3.4xlarge	1	8	16	122
g3.8xlarge	2	16	32	244
g3.16xlarge	4	32	64	488
g4dn.xlarge	1	16	4	16
g4dn.2xlarge	1	16	8	32
g4dn.4xlarge	1	16	16	64
g4dn.8xlarge	1	16	32	128
g4dn.12xlarge	4	64	48	192
g4dn.16xlarge	1	16	64	256
g5.xlarge	1	24	4	16
g5.2xlarge	1	24	8	32
g5.4xlarge	1	24	16	64
g5.8xlarge	1	24	32	128

Tipo de instancia	GPU	Memoria de GPU (GiB)	vCPU	Memoria (GiB)
g5.16xlarge	1	24	64	256
g5.12xlarge	4	96	48	192
g5.24xlarge	4	96	96	384
g5.48xlarge	8	192	192	768
g6.xlarge	1	24	4	16
g6.2xlarge	1	24	8	32
g6.4xlarge	1	24	16	64
g6.8xlarge	1	24	32	128
g6.16.xlarge	1	24	64	256
g6.12xlarge	4	96	48	192
g6.24xlarge	4	96	96	384
g6.48xlarge	8	192	192	768
g6.metal	8	192	192	768
gr6.4xlarge	1	24	16	128
g6e.xlarge	1	48	4	32
g6e.2xlarge	1	48	8	64
g6e.4xlarge	1	48	16	128
g6g.8xlarge	1	48	32	256
g6e16.xlarge	1	48	64	512
g6e12.xlarge	4	192	48	384

Tipo de instancia	GPU	Memoria de GPU (GiB)	vCPU	Memoria (GiB)
g6e24.xlarge	4	192	96	768
g6e48.xlarge	8	384	192	1536
gr6.8xlarge	1	24	32	256

Puede recuperar el ID de Imagen de máquina de Amazon (AMI) de las AMI optimizadas para Amazon ECS al consultar la API de Parameter Store de AWS Systems Manager. Al utilizar este parámetro, no necesita buscar de manera manual los ID de la AMI optimizada para Amazon ECS. Para obtener más información acerca de la API de Systems Manager Parameter Store, consulte GetParameter. El usuario que utiliza debe tener el permiso de IAM `ssm:GetParameter` para recuperar los metadatos de la AMI optimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

Consideraciones

> **Note**
>
> La compatibilidad con el tipo de familia de instancias g2 ha quedado obsoleta.
> El tipo de familia de instancias p2 es compatible solo con versiones anteriores a `20230912` de la AMI de Amazon ECS optimizada para GPU. Si necesita seguir usando instancias p2, consulte Qué hacer si necesita una instancia P2.
> Las actualizaciones locales de los controladores NVIDIA/CUDA en estos dos tipos de familia de instancias pueden provocar posibles errores en la carga de trabajo de la GPU.

Le recomendamos que tenga en cuenta lo siguiente antes de comenzar a trabajar con GPU en Amazon ECS.

- Sus clústeres pueden contener una combinación de instancias de contenedor habilitadas para GPU y no habilitadas para GPU.

- Puede ejecutar cargas de trabajo de GPU en instancias externas. Cuando se registra una instancia externa en el clúster, asegúrese de que la marca `--enable-gpu` se incluya en el script de instalación. Para obtener más información, consulte Registro de una instancia externa en un clúster de Amazon ECS.
- Debe establecer `ECS_ENABLE_GPU_SUPPORT` en `true` en el archivo de configuración del agente. Para obtener más información, consulte the section called "Configuración del agente de contenedor".
- Cuando ejecuta una tarea o crea un servicio, puede utilizar los atributos de tipo de instancia al configurar las restricciones de colocación de tareas para determinar las instancias de contenedor que se lanzan en la tarea. Esto le permite utilizar sus recursos de manera más eficiente. Para obtener más información, consulte Cómo coloca Amazon ECS las tareas en las instancias de contenedor.

  El siguiente ejemplo lanza una tarea en una instancia de contenedor `g4dn.xlarge` en el clúster predeterminado.

  ```
  aws ecs run-task --cluster default --task-definition ecs-gpu-task-def \
      --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
    g4dn.xlarge" --region us-east-2
  ```

- Para cada contenedor que tiene un requisito de recursos de GPU especificado en la definición de contenedor, Amazon ECS establece el tiempo de ejecución del contenedor en el tiempo de ejecución del contenedor de NVIDIA.
- Para que el tiempo de ejecución del contenedor NVIDIA funcione correctamente, es preciso establecer algunas variables de entorno en el contenedor. Para obtener una lista de estas variables de entorno, consulte Specialized Configurations with Docker. Amazon ECS establece el valor de las variables de entorno `NVIDIA_VISIBLE_DEVICES` en una lista de los ID de dispositivo de GPU que Amazon ECS asigna al contenedor. Amazon ECS no establece las demás variables de entorno necesarias. Por tanto, asegúrese de que la imagen del contenedor las establezca o que estén configuradas en la definición de contenedor.
- La familia del tipo de instancias p5 es compatible con la versión `20230929` y versiones posteriores de la AMI de Amazon ECS optimizada para GPU.
- La familia de tipo de instancias g4 es compatible con la versión `20230913` y versiones posteriores de la AMI de Amazon ECS optimizada para GPU. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS. No es admitido en el flujo de trabajo de Create Cluster (Crear clúster) en la consola de Amazon ECS. Para utilizar estos tipos de instancias, debe utilizar

la consola de Amazon EC2, la AWS CLI o la API, y registrar manualmente las instancias en el clúster.

- El tipo de instancias p4d.24xlarge solo funciona con CUDA 11 o posterior.
- La AMI de Amazon ECS optimizada para GPU está habilitada para IPv6, lo que provoca problemas cuando se utiliza `yum`. Para resolverlo, puede configurar que `yum` utilice IPv4 con el siguiente comando.

```
echo "ip_resolve=4" >> /etc/yum.conf
```

- Cuando crea una imagen de contenedor que no utiliza las imágenes base NVIDIA/CUDA, debe establecer la variable de tiempo de ejecución de contenedor `NVIDIA_DRIVER_CAPABILITIES` en uno de los siguientes valores:
  - `utility,compute`
  - `all`

  Para obtener información acerca de cómo establecer la variable, consulte Control del tiempo de ejecución del contenedor NVIDIA en el sitio web de NVIDIA.
- Las GPU no son compatibles con los contenedores de Windows.

Lanzamiento de una instancia de contenedor de GPU para Amazon ECS

Para utilizar una instancia de GPU en Amazon ECS, debe crear una plantilla de lanzamiento, un archivo de datos de usuario y lanzar la instancia.

A continuación, puede ejecutar una tarea que utilice una definición de tarea configurada para la GPU.

Uso de una plantilla de inicialización

Puede crear una plantilla de lanzamiento.

- Cree una plantilla de lanzamiento que utilice el identificador de AMI de GPU optimizado para Amazon ECS para la AMI. Para obtener información sobre cómo crear una plantilla de lanzamiento, consulte Create a new launch template using parameters you define en la Guía del usuario de Amazon EC2.

  Utilice el ID de AMI del paso anterior para la imagen de máquina de Amazon. Para información sobre cómo especificar el identificador de AMI con el parámetro de Systems Manager, consulte

Specify a Systems Manager parameter in a launch template en la Guía del usuario de Amazon EC2.

Agregue lo siguiente a Datos de usuario en la plantilla de lanzamiento. Sustituya *cluster-name* por el nombre de su clúster.

```
#!/bin/bash
echo ECS_CLUSTER=cluster-name >> /etc/ecs/ecs.config;
echo ECS_ENABLE_GPU_SUPPORT=true >> /etc/ecs/ecs.config
```

Utilizar AWS CLI

Puede utilizar la AWS CLI para iniciar una instancia de contenedor.

1. Cree un archivo denominado `userdata.toml`. Este archivo se utiliza para los datos de usuario de la instancia. Sustituya *cluster-name* por el nombre de su clúster.

```
#!/bin/bash
echo ECS_CLUSTER=cluster-name >> /etc/ecs/ecs.config;
echo ECS_ENABLE_GPU_SUPPORT=true >> /etc/ecs/ecs.config
```

2. Ejecute el siguiente comando para obtener el identificador de la IAM de la GPU. Utilice esto en el siguiente paso.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

3. Ejecute el siguiente comando para lanzar una instancia de la GPU. Recuerde reemplazar los siguientes parámetros:

   - Sustituya la *subred* por el ID de la subred pública o privada en la que se lanzará la instancia.
   - Sustituya *gpu_ami* por el identificador de la AMI del paso anterior.
   - Sustituya *t3.large* por el tipo de instancia que desee usar.
   - Sustituya *región* por su código de región.

```
aws ec2 run-instances --key-name ecs-gpu-example \
   --subnet-id subnet \
   --image-id gpu_ami \
```

```
    --instance-type t3.large \
    --region region \
    --tag-specifications 'ResourceType=instance,Tags=[{Key=GPU,Value=example}]' \
    --user-data file://userdata.toml \
    --iam-instance-profile Name=ecsInstanceRole
```

4. Ejecute el siguiente comando para comprobar que la instancia de contenedor está registrada en el clúster. Al ejecutar este comando, recuerde reemplazar los siguientes parámetros:

   - Sustituya *clúster* por el nombre del clúster.
   - Sustituya *región* por el código de región.

```
aws ecs list-container-instances --cluster cluster-name --region region
```

Especificación de GPU en una definición de tareas de Amazon ECS

Para utilizar las GPU en una instancia de contenedor y el tiempo de ejecución de GPU de Docker, asegúrese de designar el número de GPU que requiere el contenedor en la definición de tareas. Cuando haya contenedores que admiten GPU, el agente de contenedores de Amazon ECS fijará el número deseado de GPU físicas en el contenedor correspondiente. El número de unidades GPU reservadas para todos los contenedores de una tarea no puede superar el número de GPU disponibles en la instancia de contenedor en la que se lanza la tarea. Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

> **Important**
>
> Si los requisitos de GPU no se especifican en la definición de tareas, la tarea utilizará el tiempo de ejecución predeterminado de Docker.

A continuación, se muestra el formato JSON de los requisitos de GPU en una definición de tareas:

```
{
  "containerDefinitions": [
     {
        ...
        "resourceRequirements" : [
            {
               "type" : "GPU",
```

```
                "value" : "2"
            }
        ],
    },
...
}
```

El ejemplo siguiente muestra la sintaxis de un contenedor Docker que especifica un requisito de GPU. Este contenedor utiliza dos GPU, ejecuta la utilidad `nvidia-smi` y, luego, se cierra.

```
{
  "containerDefinitions": [
    {
      "memory": 80,
      "essential": true,
      "name": "gpu",
      "image": "nvidia/cuda:11.0.3-base",
      "resourceRequirements": [
         {
           "type":"GPU",
           "value": "2"
         }
      ],
      "command": [
        "sh",
        "-c",
        "nvidia-smi"
      ],
      "cpu": 100
    }
  ],
  "family": "example-ecs-gpu"
}
```

Uso compartido de GPU

Si quiere compartir las GPU, tiene que configurar lo siguiente:

1. Elimine los requisitos de recursos de GPU de las definiciones de tareas para que Amazon ECS no reserve ninguna GPU que deba compartirse.
2. Agregue los siguientes datos de usuario a las instancias cuando quiera compartir las GPU. Esto hará que nvidia sea el tiempo de ejecución predeterminado del contenedor de Docker en la

instancia de contenedor, de modo que todos los contenedores de Amazon ECS puedan usar las GPU. Para obtener más información acerca de los scripts de datos de usuario, consulte Ejecución de comandos al lanzar una instancia de EC2 con la entrada de datos de usuario en la Guía del usuario de Amazon EC2.

```
const userData = ec2.UserData.forLinux();
 userData.addCommands(
 'sudo rm /etc/sysconfig/docker',
 'echo DAEMON_MAXFILES=1048576 | sudo tee -a /etc/sysconfig/docker',
 'echo OPTIONS="--default-ulimit nofile=32768:65536 --default-runtime nvidia" |
 sudo tee -a /etc/sysconfig/docker',
 'echo DAEMON_PIDFILE_TIMEOUT=10 | sudo tee -a /etc/sysconfig/docker',
 'sudo systemctl restart docker',
);
```

3. Configure la variable de entorno `NVIDIA_VISIBLE_DEVICES` en el contenedor. Para hacerlo, especifique la variable de entorno en la definición de tareas. Para obtener información sobre los valores válidos, consulte la Enumeración de GPU en el sitio de documentación de NVIDIA.

Qué hacer si necesita una instancia P2

Si necesita utilizar la instancia P2, puede usar una de las siguientes opciones para continuar usando las instancias.

Debe modificar los datos de usuario de la instancia para ambas opciones. Para obtener más información acerca de los scripts de datos de usuario, consulte Ejecución de comandos al lanzar una instancia de EC2 con la entrada de datos de usuario en la Guía del usuario de Amazon EC2.

Utilizar la última AMI optimizada para GPU compatible

Puede usar la versión `20230906` de la AMI optimizada para GPU y agregar lo siguiente a los datos de usuario de la instancia.

Sustituya cluster-name por el nombre de su clúster.

```
#!/bin/bash
echo "exclude=*nvidia* *cuda*" >> /etc/yum.conf
echo "ECS_CLUSTER=cluster-name" >> /etc/ecs/ecs.config
```

Utilizar la última AMI optimizada para GPU y actualizar los datos de usuario

Puede agregar lo siguiente a los datos de usuario de la instancia. Esto desinstala los controladores Nvidia 535/Cuda12.2 y, a continuación, instala los controladores Nvidia 470/Cuda11.4 y corrige la versión.

```
#!/bin/bash
yum remove -y cuda-toolkit* nvidia-driver-latest-dkms*
tmpfile=$(mktemp)
cat >$tmpfile <<EOF
[amzn2-nvidia]
name=Amazon Linux 2 Nvidia repository
mirrorlist=\$awsproto://\$amazonlinux.\$awsregion.\$awsdomain/\$releasever/amzn2-
nvidia/latest/\$basearch/mirror.list
priority=20
gpgcheck=1
gpgkey=https://developer.download.nvidia.com/compute/cuda/repos/rhel7/
x86_64/7fa2af80.pub
enabled=1
exclude=libglvnd-*
EOF

mv $tmpfile /etc/yum.repos.d/amzn2-nvidia-tmp.repo
yum install -y system-release-nvidia cuda-toolkit-11-4 nvidia-driver-latest-
dkms-470.182.03
yum install -y libnvidia-container-1.4.0 libnvidia-container-tools-1.4.0 nvidia-
container-runtime-hook-1.4.0 docker-runtime-nvidia-1

echo "exclude=*nvidia* *cuda*" >> /etc/yum.conf
nvidia-smi
```

Crear su propia AMI optimizada para GPU compatible con P2

Puede crear su propia AMI optimizada para la GPU de Amazon ECS personalizada que sea compatible con las instancias P2 y, a continuación, lanzar las instancias P2 mediante la AMI.

1. Ejecute el siguiente comando para clonar el `amazon-ecs-ami repo`.

   ```
   git clone https://github.com/aws/amazon-ecs-ami
   ```

2. Configure el agente de Amazon ECS requerido y las versiones de la AMI de Amazon Linux de origen en `release.auto.pkrvars.hcl` o `overrides.auto.pkrvars.hcl`.
3. Ejecute el siguiente comando para crear una AMI de EC2 privada compatible con P2.

Sustituya la región por la región de la instancia.

```
REGION=region make al2keplergpu
```

4. Utilice la AMI con los siguientes datos de usuario de la instancia para conectarse al clúster de Amazon ECS.

   Sustituya cluster-name por el nombre de su clúster.

```
#!/bin/bash
echo "ECS_CLUSTER=cluster-name" >> /etc/ecs/ecs.config
```

Definiciones de tareas de Amazon ECS para cargas de trabajo de transcodificación de video

Para utilizar cargas de trabajo de transcodificación de video en Amazon ECS, registre las instancias VT1 de Amazon EC2. Una vez registradas estas instancias, puede ejecutar cargas de trabajo de transcodificación de video en directo preprocesadas como tareas en Amazon ECS. Las instancias VT1 de Amazon EC2 utilizan tarjetas de transcodificación multimedia Xilinx U30 para acelerar las cargas de trabajo de transcodificación de video en directo preprocesadas.

> **Note**
>
> Para obtener instrucciones sobre cómo ejecutar cargas de trabajo de transcodificación de video en contenedores que no sean de Amazon ECS, consulte la documentación de Xilinx.

Consideraciones

Antes de comenzar a implementar instancias VT1 en Amazon ECS, considere lo siguiente:

- Los clústeres pueden contener instancias VT1 y no VT1 combinadas.
- Necesita una aplicación Linux que utilice tarjetas de transcodificación multimedia Xilinx U30 con códecs AVC (H.264) y HEVC (H.265) acelerados.

> **Important**
>
> Es posible que las aplicaciones que utilizan otros códecs no tengan un rendimiento mejorado en las instancias VT1.

- En una tarjeta U30 solo se puede ejecutar una tarea de transcodificación. Cada tarjeta tiene dos dispositivos asociados a ella. Puede ejecutar tantas tareas de transcodificación como tarjetas haya para cada instancia VT1.
- Cuando cree un servicio o ejecute una tarea independiente, puede utilizar los atributos de tipo de instancia al configurar las restricciones de ubicación de tareas. Esto garantiza que la tarea se lance en la instancia de contenedor que especifique. Al hacerlo, puede asegurarse de que utiliza los recursos de forma eficaz y de que las tareas de las cargas de trabajo de transcodificación de video se encuentran en las instancias VT1. Para obtener más información, consulte Cómo coloca Amazon ECS las tareas en las instancias de contenedor.

  En el ejemplo siguiente, se ejecuta una tarea en una instancia `vt1.3xlarge` del clúster `default`.

  ```
  aws ecs run-task \
       --cluster default \
       --task-definition vt1-3xlarge-xffmpeg-processor \
       --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
   vt1.3xlarge"
  ```

- Configura un contenedor para que utilice la tarjeta U30 específica disponible en la instancia de contenedor del host. Para ello, utilice el parámetro `linuxParameters` y especifique los detalles del dispositivo. Para obtener más información, consulte Requisitos de definición de tareas.

Uso de una AMI VT1

Tiene dos opciones para ejecutar una AMI en Amazon EC2 para instancias de contenedores de Amazon ECS. La primera opción es utilizar la AMI oficial de Xilinx en AWS Marketplace. La segunda opción es crear su propia AMI desde el repositorio de muestra.

- Xilinx ofrece AMI en AWS Marketplace.
- Amazon ECS proporciona un repositorio de muestra que puede utilizar para crear una AMI para cargas de trabajo de transcodificación de video. Esta AMI viene con los controladores Xilinx

U30. Puede encontrar el repositorio que contiene scripts de Packer en GitHub. Para obtener más información sobre Packer, consulte la documentación de Packer.

Requisitos de definición de tareas

Para ejecutar contenedores de transcodificación de video en Amazon ECS, la definición de tareas debe contener una aplicación de transcodificación de video que utilice los códecs H.264/AVC y H.265/HEVC acelerados. Para crear una imagen de contenedor, siga los pasos que se indican en el GitHub de Xilinx.

La definición de tareas debe ser específica del tipo de instancia. Los tipos de instancias son 3xlarge, 6xlarge y 24xlarge. Debe configurar un contenedor para que utilice los dispositivos Xilinx U30 específicos disponibles en la instancia de contenedor del host. Para ello, utilice el parámetro `linuxParameters`. En la tabla que se muestra a continuación se detallan las tarjetas y los SoC de dispositivos específicos de cada tipo de instancia.

Tipo de instancia	vCPU	RAM (GiB)	Tarjetas aceleradoras U30	Dispositivos SoC XCU30 direccionables	Rutas del dispositivo
vt1.3xlarge	12	24	1	2	`/dev/dri/renderD128 ,/dev/dri/renderD129`
vt1.6xlarge	24	48	2	4	`/dev/dri/renderD128 ,/dev/dri/renderD129 ,/dev/dri/renderD130 ,/dev/`

Tipo de instancia	vCPU	RAM (GiB)	Tarjetas aceleradoras U30	Dispositivos SoC XCU30 direccionables	Rutas del dispositivo
					`dri/renderD131`

Tipo de instancia	vCPU	RAM (GiB)	Tarjetas aceleradoras U30	Dispositivos SoC XCU30 direccion ables	Rutas del dispositivo
vt1.24xlarge	96	182	8	16	`/dev/dri/renderD128 ,/dev/dri/renderD129 ,/dev/dri/renderD130 ,/dev/dri/renderD131 ,/dev/dri/renderD132 ,/dev/dri/renderD133 ,/dev/dri/renderD134 ,/dev/dri/renderD135 ,/dev/dri/renderD136 ,/dev/dri/renderD137 ,/dev/dri/`

Tipo de instancia	vCPU	RAM (GiB)	Tarjetas aceleradoras U30	Dispositivos SoC XCU30 direccionables	Rutas del dispositivo
					`renderD138 ,/dev/dri/renderD139 ,/dev/dri/renderD140 ,/dev/dri/renderD141 ,/dev/dri/renderD142 ,/dev/dri/renderD143`

> **Important**
>
> Si la definición de tareas enumera los dispositivos que la instancia de EC2 no tiene, la tarea no se ejecuta. Cuando se produce·un·error·en la tarea, aparece el siguiente mensaje de error en `stoppedReason`: `CannotStartContainerError: Error response from daemon: error gathering device information while adding custom device "/dev/dri/renderD130": no such file or directory`.

Especificación de la transcodificación de video en una definición de tareas de Amazon ECS

En el siguiente ejemplo, se proporciona la sintaxis que se utiliza para la definición de tareas de un contenedor de Linux en Amazon EC2. Esta definición de tareas se usa para imágenes de contenedor que se crean siguiendo el procedimiento proporcionado en la documentación de Xilinx. Si utiliza este ejemplo, reemplace `image` con su propia imagen y copie los archivos de video en la instancia del directorio `/home/ec2-user`.

vt1.3xlarge

1. Cree un archivo de texto denominado `vt1-3xlarge-ffmpeg-linux.json` con el siguiente contenido.

```
{
    "family": "vt1-3xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.3xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
                "devices": [
                    {
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
```

```
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
            "name": "xilinix-xffmpeg"
        }
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Registre la definición de tareas.

```
aws ecs register-task-definition --family vt1-3xlarge-xffmpeg-processor --cli-
input-json file://vt1-3xlarge-xffmpeg-linux.json --region us-east-1
```

vt1.6xlarge

1. Cree un archivo de texto denominado `vt1-6xlarge-ffmpeg-linux.json` con el siguiente contenido.

```
{
    "family": "vt1-6xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.6xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
                "devices": [
                    {
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
```

```
                    {
                        "containerPath": "/dev/dri/renderD130",
                        "hostPath": "/dev/dri/renderD130",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD131",
                        "hostPath": "/dev/dri/renderD131",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
            "name": "xilinix-xffmpeg"
        }
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Registre la definición de tareas.

```
aws ecs register-task-definition --family vt1-6xlarge-xffmpeg-processor --cli-
input-json file://vt1-6xlarge-xffmpeg-linux.json --region us-east-1
```

vt1.24xlarge

1. Cree un archivo de texto denominado `vt1-24xlarge-ffmpeg-linux.json` con el siguiente contenido.

```
{
    "family": "vt1-24xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.24xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
                "devices": [
                    {
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
```

```
                "permissions": [
                    "read",
                    "write"
                ]
            },
            {
                "containerPath": "/dev/dri/renderD130",
                "hostPath": "/dev/dri/renderD130",
                "permissions": [
                    "read",
                    "write"
                ]
            },
            {
                "containerPath": "/dev/dri/renderD131",
                "hostPath": "/dev/dri/renderD131",
                "permissions": [
                    "read",
                    "write"
                ]
            },
            {
                "containerPath": "/dev/dri/renderD132",
                "hostPath": "/dev/dri/renderD132",
                "permissions": [
                    "read",
                    "write"
                ]
            },
            {
                "containerPath": "/dev/dri/renderD133",
                "hostPath": "/dev/dri/renderD133",
                "permissions": [
                    "read",
                    "write"
                ]
            },
            {
                "containerPath": "/dev/dri/renderD134",
                "hostPath": "/dev/dri/renderD134",
                "permissions": [
                    "read",
                    "write"
                ]
```

```
                    },
                    {
                        "containerPath": "/dev/dri/renderD135",
                        "hostPath": "/dev/dri/renderD135",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD136",
                        "hostPath": "/dev/dri/renderD136",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD137",
                        "hostPath": "/dev/dri/renderD137",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD138",
                        "hostPath": "/dev/dri/renderD138",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD139",
                        "hostPath": "/dev/dri/renderD139",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD140",
                        "hostPath": "/dev/dri/renderD140",
```

```
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD141",
                        "hostPath": "/dev/dri/renderD141",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD142",
                        "hostPath": "/dev/dri/renderD142",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD143",
                        "hostPath": "/dev/dri/renderD143",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
            "name": "xilinix-xffmpeg"
        }
```

```
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Registre la definición de tareas.

```
aws ecs register-task-definition --family vt1-24xlarge-xffmpeg-processor --cli-
input-json file://vt1-24xlarge-xffmpeg-linux.json --region us-east-1
```

Definiciones de tareas de Amazon ECS para cargas de trabajo de machine learning de AWS Neuron

Puede registrar instancias Trn1 de Amazon EC2, Inf1 de Amazon EC2 e Inf2 de Amazon EC2 en los clústeres para cargas de trabajo de machine learning.

Las instancias Trn1 de Amazon EC2 funcionan con chips de AWS Trainium. Estas instancias proporcionan capacitación de alto rendimiento y bajo costo para el machine learning en la nube. Puede entrenar un modelo de inferencia de machine learning mediante un marco de machine learning con AWS Neuron en una instancia de Trn1. A continuación, puede ejecutar el modelo en una instancia de Inf1 o de Inf2 para usar la aceleración de los chips de AWS Inferentia.

Las instancias Inf1 e Inf2 de Amazon EC2 funcionan con chips de AWS Inferentia, que proporcionan un alto rendimiento y una inferencia de menor costo en la nube.

Los modelos de machine learning se implementan en contenedores mediante AWS Neuron, que es un kit de desarrollo de software (SDK) especializado. SDK consiste en un compilador, tiempo de ejecución y herramientas de perfilado que optimizan el rendimiento de machine learning de los chips de AWS. AWS Neuron admite marcos de machine learning populares como TensorFlow, PyTorch y Apache MXNet.

Consideraciones

Antes de comenzar a implementar Neuron en Amazon ECS, tenga en cuenta lo siguiente:

- Los clústeres pueden contener una combinación de instancias Trn1, Inf1, Inf2 y otras instancias.

- Necesita una aplicación de Linux en un contenedor que use un marco de machine learning compatible con AWS Neuron.

> **Important**
>
> Es posible que las aplicaciones que utilizan otros marcos no tengan un rendimiento mejorado en las instancias Trn1, Inf1 e Inf2.

- Solo se puede ejecutar una tarea de inferencia o entrenamiento de inferencias en cada chip AWS Trainium o AWS Inferentia. Para Inf1, cada chip tiene 4 NeuronCore. Para Trn1 e Inf2, cada chip tiene 2 NeuronCore. Puede ejecutar tantas tareas como chips haya para cada instancia de Trn1, Inf1 e Inf2.
- Cuando cree un servicio o ejecute una tarea independiente, puede utilizar los atributos de tipo de instancia al configurar las restricciones de ubicación de tareas. Esto garantiza que la tarea se lance en la instancia de contenedor que especifique. Al hacerlo, puede optimizar la utilización general de los recursos y garantizar que las tareas de las cargas de trabajo de inferencia se encuentren en las instancias Trn1, Inf1 e Inf2. Para obtener más información, consulte Cómo coloca Amazon ECS las tareas en las instancias de contenedor.

  En el ejemplo siguiente, se ejecuta una tarea en una instancia `Inf1.xlarge` del clúster `default`.

```
aws ecs run-task \
     --cluster default \
     --task-definition ecs-inference-task-def \
     --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
 Inf1.xlarge"
```

- Los requisitos de recursos de Neuron no se pueden definir en una definición de tareas. En su lugar, configure un contenedor para que use chips de AWS Trainium o de AWS Inferentia específicos disponibles en la instancia de contenedor del host. Para ello, use el parámetro `linuxParameters` y especifique los detalles del dispositivo. Para obtener más información, consulte Requisitos de definición de tareas.

Uso de la AMI de Amazon Linux 2023 (Neuron) optimizada para Amazon ECS

Amazon ECS proporciona una AMI optimizada para Amazon ECS que se basa en Amazon Linux 2023 para cargas de trabajo de AWS Trainium y AWS Inferentia. Viene con los controladores AWS

Neuron y el tiempo de ejecución para Docker. Esta AMI facilita la ejecución de cargas de trabajo de inferencia de machine learning en Amazon ECS.

Recomendamos utilizar la AMI de Amazon Linux 2023 (Neuron) optimizada para Amazon ECS cuando se lanzan las instancias Trn1, Inf1 e Inf2 de Amazon EC2.

Puede recuperar la AMI actual de Amazon Linux 2023 (Neuron) optimizada para Amazon ECS a través de la AWS CLI con el siguiente comando.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/neuron/
recommended
```

La AMI de Amazon Linux 2023 (Neuron) optimizada para Amazon ECS se admite en las regiones siguientes:

- Este de EE. UU. (Norte de Virginia)
- Este de EE. UU. (Ohio)
- Oeste de EE. UU. (Norte de California)
- Oeste de EE. UU. (Oregón)
- Asia-Pacífico (Mumbai)
- Asia-Pacífico (Osaka)
- Asia-Pacífico (Seúl)
- Asia-Pacífico (Tokio)
- Asia-Pacífico (Singapur)
- Asia-Pacífico (Sídney)
- Canadá (centro)
- Europa (Fráncfort)
- Europa (Irlanda)
- Europa (Londres)
- Europa (París)
- Europa (Estocolmo)
- América del Sur (São Paulo)

Uso de la AMI de Amazon Linux 2 (Neuron) optimizada para Amazon ECS

Amazon ECS proporciona una AMI optimizada para Amazon ECS basada en Amazon Linux 2 para cargas de trabajo de AWS Trainium y AWS Inferentia. Viene con los controladores AWS Neuron y el tiempo de ejecución para Docker. Esta AMI facilita la ejecución de cargas de trabajo de inferencia de machine learning en Amazon ECS.

Recomendamos utilizar la AMI de Amazon Linux 2 (Neuron) optimizada para Amazon ECS cuando se lanzan las instancias Trn1, Inf1 e Inf2 de Amazon EC2.

Puede recuperar la AMI actual de Amazon Linux 2 (Neuron) optimizada para Amazon ECS a través de la AWS CLI con el siguiente comando.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/inf/
recommended
```

La AMI de Amazon Linux 2 (Neuron) optimizada para Amazon ECS se admite en las regiones siguientes:

- Este de EE. UU. (Norte de Virginia)
- Este de EE. UU. (Ohio)
- Oeste de EE. UU. (Norte de California)
- Oeste de EE. UU. (Oregón)
- Asia-Pacífico (Mumbai)
- Asia-Pacífico (Osaka)
- Asia-Pacífico (Seúl)
- Asia-Pacífico (Tokio)
- Asia-Pacífico (Singapur)
- Asia-Pacífico (Sídney)
- Canadá (centro)
- Europa (Fráncfort)
- Europa (Irlanda)
- Europa (Londres)
- Europa (París)

- Europa (Estocolmo)
- América del Sur (São Paulo)

Requisitos de definición de tareas

Para implementar Neuron en Amazon ECS, la definición de tareas debe contener la definición de contenedor correspondiente a un contenedor prefabricado que atienda al modelo de inferencia para TensorFlow. Este es proporcionado por AWS Deep Learning Containers. Dentro del contenedor, se incluye el tiempo de ejecución de AWS Neuron y la aplicación TensorFlow Serving. Al iniciarse, este contenedor obtiene su modelo de Amazon S3, lanza Neuron TensorFlow Serving con el modelo guardado y espera las solicitudes de predicción. En el ejemplo a continuación, la imagen de contenedor contiene TensorFlow 1.15 y Ubuntu 18.04. Se conserva una lista completa de Deep Learning Containers prefabricados optimizados para Neuron en GitHub. Para obtener más información, consulte Utilizar AWS Neuron TensorFlow Serving.

```
763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-inference-neuron:1.15.4-neuron-
py37-ubuntu18.04
```

Como alternativa, puede crear su propia imagen de contenedor de sidecar de Neuron. Para obtener más información, consulte Tutorial: Neuron TensorFlow Serving en la Guía para desarrolladores de AWS Deep Learning AMIs.

La definición de la tarea debe ser específica para un único tipo de instancia. Debe configurar un contenedor para que use los dispositivos AWS Trainium o AWS Inferentia específicos disponibles en la instancia de contenedor del host. Para ello, utilice el parámetro `linuxParameters`. En la tabla que se muestra a continuación se detallan los chips específicos de cada tipo de instancia.

Tipo de instancia	vCPU	RAM (GiB)	Chips de aceleradores AWS ML	Rutas del dispositivo
trn1.2xlarge	8	32	1	`/dev/neuron0`
trn1.32xlarge	128	512	16	`/dev/neuron0 ,/dev/neuron1 ,/`

Tipo de instancia	vCPU	RAM (GiB)	Chips de aceleradores AWS ML	Rutas del dispositivo
				`dev/neuron2 , /dev/neuron3 , /dev/neuron4 , /dev/neuron5 , /dev/neuron6 , /dev/neuron7 , /dev/neuron8 , /dev/neuron9 , /dev/neuron10 , /dev/neuron11 , /dev/neuron12 , /dev/neuron13 , /dev/neuron14 , /dev/neuron15`
inf1.xlarge	4	8	1	`/dev/neuron0`
inf1.2xlarge	8	16	1	`/dev/neuron0`
inf1.6xlarge	24	48	4	`/dev/neuron0 , /dev/neuron1 , /dev/neuron2 , /dev/neuron3`

Tipo de instancia	vCPU	RAM (GiB)	Chips de aceleradores AWS ML	Rutas del dispositivo
inf1.24xlarge	96	192	16	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,/dev/neuron6 ,/dev/neuron7 ,/dev/neuron8 ,/dev/neuron9 ,/dev/neuron10 ,/dev/neuron11 ,/dev/neuron12 ,/dev/neuron13 ,/dev/neuron14 ,/dev/neuron15`
inf2.xlarge	8	16	1	`/dev/neuron0`
inf2.8xlarge	32	64	1	`/dev/neuron0`

Tipo de instancia	vCPU	RAM (GiB)	Chips de aceleradores AWS ML	Rutas del dispositivo
inf2.24xlarge	96	384	6	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,`
inf2.48xlarge	192	768	12	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,/dev/neuron6 ,/dev/neuron7 ,/dev/neuron8 ,/dev/neuron9 ,/dev/neuron10 ,/dev/neuron11`

Especificación de machine learning de AWS Neuron en una definición de tareas de Amazon ECS

A continuación, se muestra un ejemplo de definición de tarea de Linux para `inf1.xlarge` en el que se muestra la sintaxis que se va a usar.

```
{
    "family": "ecs-neuron",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == inf1.xlarge"
        }
    ],
    "executionRoleArn": "${YOUR_EXECUTION_ROLE}",
    "containerDefinitions": [
        {
            "entryPoint": [
                "/usr/local/bin/entrypoint.sh",
                "--port=8500",
                "--rest_api_port=9000",
                "--model_name=resnet50_neuron",
                "--model_base_path=s3://amzn-s3-demo-bucket/resnet50_neuron/"
            ],
            "portMappings": [
                {
                    "hostPort": 8500,
                    "protocol": "tcp",
                    "containerPort": 8500
                },
                {
                    "hostPort": 8501,
                    "protocol": "tcp",
                    "containerPort": 8501
                },
                {
                    "hostPort": 0,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "linuxParameters": {
                "devices": [
                    {
```

```
                        "containerPath": "/dev/neuron0",
                        "hostPath": "/dev/neuron0",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ],
                "capabilities": {
                    "add": [
                        "IPC_LOCK"
                    ]
                }
            },
            "cpu": 0,
            "memoryReservation": 1000,
            "image": "763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-
inference-neuron:1.15.4-neuron-py37-ubuntu18.04",
            "essential": true,
            "name": "resnet50"
        }
    ]
}
```

Definiciones de tareas de Amazon ECS para instancias de aprendizaje profundo

Para utilizar cargas de trabajo de aprendizaje profundo en Amazon ECS, registre las instancias DL1 de Amazon EC2 en los clústeres. Las instancias DL1 de Amazon EC2 funcionan con aceleradores Gaudi de Habana Labs (una empresa de Intel). Utilice el SDK de Habana SynapseAI para conectarse a los aceleradores Gaudi de Habana. El SDK admite los marcos de machine learning populares TensorFlow y PyTorch.

Consideraciones

Antes de comenzar a implementar instancias DL1 en Amazon ECS, tenga en cuenta lo siguiente:

- Los clústeres pueden contener instancias DL1 y no DL1 combinadas.
- Cuando crea un servicio o ejecuta una tarea independiente, puede utilizar los atributos de tipo de instancias concretamente al configurar las restricciones de ubicación de tareas para asegurarse de que la tarea se lance en la instancia de contenedor que especifique. Al hacerlo, se asegura

de que los recursos se utilizan de manera eficaz y de que las tareas de las cargas de trabajo de aprendizaje profundo se encuentran en las instancias DL1. Para obtener más información, consulte Cómo coloca Amazon ECS las tareas en las instancias de contenedor.

En el ejemplo siguiente, se ejecuta una tarea en una instancia `dl1.24xlarge` del clúster `default`.

```
aws ecs run-task \
     --cluster default \
     --task-definition ecs-dl1-task-def \
     --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
 dl1.24xlarge"
```

Uso de una AMI DL1

Tiene tres opciones para ejecutar una AMI en instancias DL1 de Amazon EC2 para Amazon ECS:

- AMI de AWS Marketplace proporcionadas por Habana aquí.
- AMI de aprendizaje profundo de Habana proporcionadas por Amazon Web Services. Como no está incluido, debe instalar el agente de contenedor de Amazon ECS por separado.
- Utilice Packer para crear una AMI personalizada proporcionada por el repositorio de GitHub. Para obtener más información, consulte la documentación de Packer.

Especificación del aprendizaje profundo en una definición de tareas de Amazon ECS

Para ejecutar contenedores de aprendizaje profundo acelerados Gaudi de Habana en Amazon ECS, la definición de tareas debe contener la definición de contenedor de un contenedor prefabricado que atienda el modelo de aprendizaje profundo de TensorFlow o PyTorch utilizando el software SynapseAI de Habana proporcionado por AWS Deep Learning Containers.

La siguiente imagen de contenedor contiene TensorFlow 2.7.0 y Ubuntu 20.04. En GitHub se conserva una lista completa de contenedores de aprendizaje profundo prefabricados optimizados para aceleradores Gaudi de Habana. Para obtener más información, consulte Habana Training Containers (Contenedores de capacitación de Habana).

```
763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training-habana:2.7.0-hpu-py38-
synapseai1.2.0-ubuntu20.04
```

A continuación, se muestra un ejemplo de definición de tareas de contenedores Linux en Amazon EC2 en el que se muestra la sintaxis que se va a utilizar. En este ejemplo se utiliza una imagen que contiene la herramienta de interfaz de administración del sistema de Habana Labs (HL-SMI) que se encuentra aquí: `vault.habana.ai/gaudi-docker/1.1.0/ubuntu20.04/habanalabs/tensorflow-installer-tf-cpu-2.6.0:1.1.0-614`

```
{
    "family": "dl-test",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == dl1.24xlarge"
        }
    ],
    "networkMode": "host",
    "cpu": "10240",
    "memory": "1024",
    "containerDefinitions": [
        {
            "entryPoint": [
                "sh",
                "-c"
            ],
            "command": ["hl-smi"],
            "cpu": 8192,
            "environment": [
                {
                    "name": "HABANA_VISIBLE_DEVICES",
                    "value": "all"
                }
            ],
            "image": "vault.habana.ai/gaudi-docker/1.1.0/ubuntu20.04/habanalabs/
tensorflow-installer-tf-cpu-2.6.0:1.1.0-614",
            "essential": true,
            "name": "tensorflow-installer-tf-hpu"
        }
    ]
```

```
}
```

Definiciones de tareas de Amazon ECS para cargas de trabajo de ARM de 64 bits

Amazon ECS admite el uso de aplicaciones ARM de 64 bits. Puede ejecutar las aplicaciones en la plataforma con la tecnología de los procesadores de AWS Graviton. Es adecuada para una gran variedad de cargas de trabajo. Entre ellas se incluyen cargas de trabajo tales como servidores de aplicaciones, microservicios, computación de alto rendimiento, inferencia de machine learning basada en CPU, codificación de video, automatización de diseño electrónico, juegos, bases de datos de código abierto y cachés en memoria.

Consideraciones

Antes de comenzar a implementar definiciones de tareas que utilizan la arquitectura de ARM de 64 bits, tenga en cuenta lo siguiente:

- Las aplicaciones pueden utilizar los tipos de lanzamiento de Fargate o EC2.
- Las aplicaciones solo pueden utilizar el sistema operativo Linux.
- Para el tipo Fargate, las aplicaciones deben utilizar la versión de plataforma Fargate `1.4.0` o posterior.
- Las aplicaciones pueden utilizar Fluent Bit o CloudWatch para supervisión.
- Para el tipo de lanzamiento de Fargate, las siguientes Regiones de AWS no admiten cargas de trabajo ARM de 64 bits:
  - Este de EE. UU. (Norte de Virginia), la zona de disponibilidad `use1-az3`
- Para el tipo de lanzamiento de Amazon EC2, consulte lo siguiente para comprobar que la región en la que se encuentra admite el tipo de instancia que desea utilizar:
  - Instancias M6g de Amazon EC2
  - Instancias T4g de Amazon EC2
  - Instancias C6g de Amazon EC2
  - Instancias R6gd de Amazon EC2
  - Instancias X2gd de Amazon EC2

  También puede utilizar el comando `describe-instance-type-offerings` de Amazon EC2 con un filtro para ver la oferta de instancias de su región.

```
aws ec2 describe-instance-type-offerings --filters Name=instance-
type,Values=instance-type --region region
```

En el siguiente ejemplo, se comprueba la disponibilidad del tipo de instancia M6 en la región Este de EE. UU. (Norte de Virginia) (us-east-1).

```
aws ec2 describe-instance-type-offerings --filters "Name=instance-type,Values=m6*" --
region us-east-1
```

Para obtener más información, consulte describe-instance-type-offerings en la Referencia de la línea de comandos de Amazon EC2.

Especificación de la arquitectura de ARM en la definición de tareas de Amazon ECS

Para utilizar la arquitectura de ARM, especifique `ARM64` para el parámetro de definición de tareas `cpuArchitecture`.

En el siguiente ejemplo, la arquitectura de ARM se especifica en una definición de tareas. Está en formato JSON.

```
{
    "runtimePlatform": {
        "operatingSystemFamily": "LINUX",
        "cpuArchitecture": "ARM64"
    },
...
}
```

En el siguiente ejemplo, una definición de tarea para la arquitectura de ARM muestra “hello world”.

```
{
 "family": "arm64-testapp",
 "networkMode": "awsvpc",
 "containerDefinitions": [
    {
        "name": "arm-container",
        "image": "public.ecr.aws/docker/library/busybox:latest",
        "cpu": 100,
        "memory": 100,
```

```
        "essential": true,
        "command": [ "echo hello world" ],
        "entryPoint": [ "sh", "-c" ]
    }
 ],
 "requiresCompatibilities": [ "EC2" ],
 "cpu": "256",
 "memory": "512",
 "runtimePlatform": {
        "operatingSystemFamily": "LINUX",
        "cpuArchitecture": "ARM64"
  },
 "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole"
}
```

Envío de registros de Amazon ECS a CloudWatch

Puede configurar los contenedores de las tareas para que envíen información de registro a CloudWatch Logs. Si utiliza el tipo de lanzamiento de Fargate para las tareas, puede ver los registros de los contenedores. Si utiliza el tipo de lanzamiento de EC2, esto le permite ver distintos registros desde los contenedores en una ubicación cómoda y evita que los registros de contenedor ocupen espacio en disco en sus instancias de contenedor.

> **Note**
>
> El tipo de información que registran los contenedores de la tarea depende en gran medida del comando `ENTRYPOINT`. De forma predeterminada, los registros que se capturan muestran la salida del comando que aparecería normalmente en un terminal interactivo si el contenedor se ejecutara localmente, que son los flujos de E/S `STDOUT` y `STDERR`. El controlador del registros `awslogs` simplemente transfiere estos registros de Docker a CloudWatch Logs. Para obtener más información acerca de cómo se procesan los registros de Docker, incluidas formas alternativas de capturar diferentes datos de archivo o flujos, consulte la página sobre cómo Ver los registros de un contenedor o servicio en la documentación de Docker.

Para enviar registros del sistema desde las instancias de contenedor de Amazon ECS a CloudWatch Logs, consulte Supervisar archivos de registro y Cuotas de registros de CloudWatch en la Guía del usuario de registros de Amazon CloudWatch.

Tipo de lanzamiento de Fargate

Si utiliza el tipo de lanzamiento de Fargate para las tareas, debe agregar los parámetros `logConfiguration` necesarios a la definición de tareas para activar el controlador de registros `awslogs`. Para obtener más información, consulte Definición de tareas de Amazon ECS de ejemplo: enrutar registros a CloudWatch.

Para el contenedor de Windows en Fargate, siga una de las siguientes opciones cuando alguno de los parámetros de definición de tareas tenga caracteres especiales como, por ejemplo, `& \ < > ^ |`:

- Agregue un carácter de escape (\) con comillas dobles alrededor de toda la cadena de parámetros.

  Ejemplo

  ```
  "awslogs-multiline-pattern": "\"^[|DEBUG|INFO|WARNING|ERROR\"",
  ```

- Agregue un carácter de escape (^) alrededor de cada carácter especial.

  Ejemplo

  ```
  "awslogs-multiline-pattern": "^^[^|DEBUG^|INFO^|WARNING^|ERROR",
  ```

Tipo de lanzamiento de EC2

Si utiliza el tipo de lanzamiento de EC2 para las tareas y desea activar el controlador de registros `awslogs`, las instancias de contenedor de Amazon ECS requieren al menos la versión 1.9.0 del agente de contenedor. Para obtener información acerca de cómo comprobar la versión del agente y actualizar a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS.

> **Note**
>
> Debe utilizar una AMI optimizada para Amazon ECS o una AMI personalizada con al menos la versión `1.9.0-1` del paquete `ecs-init`. Cuando utilice una AMI personalizada, debe especificar que el controlador del registro `awslogs` esté disponible en la instancia de Amazon EC2 al iniciar el agente mediante la siguiente variable de entorno en la instrucción docker run o el archivo de variables de entorno.

```
ECS_AVAILABLE_LOGGING_DRIVERS=["json-file","awslogs"]
```

Las instancias de contenedor de Amazon ECS también requieren el permiso `logs:CreateLogStream` y `logs:PutLogEvents` en el rol de IAM con el que se pueden lanzar las instancias de contenedor. Si creó el rol de instancia de contenedor de Amazon ECS antes de que se habilitara la compatibilidad con el controlador de registros `awslogs` en Amazon ECS, es posible que tenga que agregar este permiso. El `ecsTaskExecutionRole` se utiliza cuando se asigna a la tarea y probablemente contenga los permisos correctos. Para obtener información acerca del rol de ejecución de tareas, consulte Rol de IAM de ejecución de tareas de Amazon ECS. Si las instancias de contenedor utilizan la política de IAM administrada para instancias de contenedor, probablemente tengan los permisos correctos. Para obtener información acerca de la política de IAM administrada para las instancias de contenedor, consulte Rol de IAM de instancia de contenedor de Amazon ECS.

Definición de tareas de Amazon ECS de ejemplo: enrutar registros a CloudWatch

Antes de que los contenedores puedan enviar registros a CloudWatch, debe especificar el controlador de registros `awslogs` para los contenedores de la definición de tarea. Para obtener más información sobre los parámetros de registros, consulte Almacenamiento y registro.

El JSON de definición de tareas a continuación tiene un objeto `logConfiguration` especificado para cada contenedor. Uno es para el contenedor de WordPress que envía registros a un grupo de registro denominado `awslogs-wordpress`. El otro es para un contenedor MySQL que envía registros a un grupo de registro denominado `awslogs-mysql`. Ambos contenedores utilizan el prefijo de flujo de registros `awslogs-example`.

```
{
    "containerDefinitions": [
        {
            "name": "wordpress",
            "links": [
                "mysql"
            ],
            "image": "public.ecr.aws/docker/library/wordpress:latest",
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80
```

```
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-create-group": "true",
                    "awslogs-group": "awslogs-wordpress",
                    "awslogs-region": "us-west-2",
                    "awslogs-stream-prefix": "awslogs-example"
                }
            },
            "memory": 500,
            "cpu": 10
        },
        {
            "environment": [
                {
                    "name": "MYSQL_ROOT_PASSWORD",
                    "value": "password"
                }
            ],
            "name": "mysql",
            "image": "public.ecr.aws/docker/library/mysql:latest",
            "cpu": 10,
            "memory": 500,
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-create-group": "true",
                    "awslogs-group": "awslogs-mysql",
                    "awslogs-region": "us-west-2",
                    "awslogs-stream-prefix": "awslogs-example",
                    "mode": "non-blocking",
                    "max-buffer-size": "25m"
                }
            }
        }
    ],
    "family": "awslogs-example"
}
```

Pasos a seguir a continuación

- Si lo desea, puede establecer una política de retención para el grupo de registros mediante la AWS CLI de CloudWatch o la API. Para obtener más información, consulte put-retention-policy en la Referencia de comandos de la AWS Command Line Interface.
- Después de haber registrado una definición de tarea con el controlador de registros `awslogs` en una configuración de registros de definición de contenedor, puede ejecutar una tarea o crear un servicio con dicha definición de tarea para comenzar a enviar registros a CloudWatch Logs. Para obtener más información, consulte Ejecución de una aplicación como tarea de Amazon ECS y Creación de una implementación de actualización continua de Amazon ECS.

Envío de registros de Amazon ECS a un servicio de AWS o AWS Partner

Puede utilizar FireLens para Amazon ECS para utilizar parámetros de definición de tareas para dirigir registros a un servicio de AWS o a un destino de AWS Partner Network (APN) para el almacenamiento y el análisis de registros. La AWS Partner Network es una comunidad global de socios que aprovecha los programas, el conocimiento técnico y los recursos para crear, comercializar y vender ofertas para los clientes. Para obtener más información, consulte AWS Partner. FireLens trabaja con Fluentd y Fluent Bit. Proporcionamos AWS para la imagen de Fluent Bit o puede utilizar su propia imagen de Fluentd o Fluent Bit.

De forma predeterminada, Amazon ECS configura la dependencia del contenedor para que el contenedor Firelens se inicie antes que cualquier contenedor que lo utilice. El contenedor Firelens también se detiene después de que se detengan todos los contenedores que lo utilizan.

Al utilizar FireLens para Amazon ECS, tenga en cuenta lo siguiente:

- Se recomienda agregar `my_service_` al nombre del contenedor de registros para poder distinguir fácilmente los nombres de los contenedores en la consola.
- Amazon ECS agrega una dependencia de orden de contenedores inicial entre los contenedores de la aplicación y el contenedor de FireLens de manera predeterminada. Al especificar un orden de contenedores entre los contenedores de la aplicación y el contenedor de FireLens, se anula el orden de contenedores inicial predeterminado.
- FireLens para Amazon ECS es compatible con tareas que están alojadas en AWS Fargate en Linux y Amazon EC2 en Linux. Los contenedores de Windows no son compatibles con FireLens.

Para obtener información sobre cómo configurar el registro centralizado para contenedores de Windows, consulte Centralized logging for Windows containers on Amazon ECS using Fluent Bit (Registro centralizado para contenedores de Windows en Amazon ECS con Fluent Bit).

- Puede utilizar plantillas de AWS CloudFormation para configurar FireLens para Amazon ECS. Para obtener más información, consulte FirelensConfiguration de AWS::ECS::TaskDefinition en la Guía del usuario de AWS CloudFormation.
- FireLens escucha en el puerto `24224`, para asegurarse de que el direccionador de registros FireLens no sea accesible fuera de la tarea, no debe permitir la entrada de tráfico en el puerto `24224` en el grupo de seguridad que utiliza la tarea. Para tareas que utilizan el modo de red `awsvpc`, es el grupo de seguridad asociado a la tarea. Para tareas que utilizan el modo de red `host`, es el grupo de seguridad asociado a la instancia de Amazon EC2 que aloja la tarea. Para tareas que utilizan el modo de red `bridge`, no cree asignaciones de puertos que utilicen el puerto `24224`.
- Para las tareas que utilizan el modo de red `bridge`, el contenedor con la configuración de FireLens debe iniciarse antes de que se inicie cualquier contenedor de aplicación que se base en él. Para controlar el orden de inicio de los contenedores, utilice las condiciones de dependencia en la definición de la tarea. Para obtener más información, consulte Dependencia de contenedor.

> **Note**
>
> Si utiliza parámetros de condición de dependencia en definiciones de contenedor con una configuración de FireLens, asegúrese de que cada contenedor tenga un requisito de condición `HEALTHY` o `START`.

- Por defecto, FireLens agrega el nombre de definición de clúster y tarea y el nombre de recurso de Amazon (ARN) del clúster como claves de metadatos a los registros de contenedor de stdout/stderr. A continuación se muestra un ejemplo del formato de los metadatos.

```
"ecs_cluster": "cluster-name",
"ecs_task_arn": "arn:aws:ecs:region:111122223333:task/cluster-
name/f2ad7dba413f45ddb4EXAMPLE",
"ecs_task_definition": "task-def-name:revision",
```

Si no quiere que los metadatos estén en los registros, establezca `enable-ecs-log-metadata` en `false` en la `firelensConfiguration` de la definición de tareas.

```
"firelensConfiguration":{
   "type":"fluentbit",
   "options":{
      "enable-ecs-log-metadata":"false",
      "config-file-type":"file",
      "config-file-value":"/extra.conf"
}
```

Para utilizar esta característica, debe crear un rol de IAM para las tareas, que proporcione los permisos necesarios para utilizar los servicios de AWS que las tareas requieran. Por ejemplo, si un contenedor dirige los registros a Firehose, la tarea necesita permiso para llamar a la API `firehose:PutRecordBatch`. Para obtener más información, consulte Adición y eliminación de permisos de identidad de IAM en la Guía del usuario de IAM.

Es posible que la tarea también requiera el rol de ejecución de tareas de Amazon ECS en las condiciones que se describen a continuación. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

- Si la tarea está alojada en Fargate y se están extrayendo imágenes de contenedor de Amazon ECR o haciendo referencia a información confidencial de AWS Secrets Manager en la configuración de registro, debe incluir el rol de IAM de ejecución de tareas.
- Si utiliza un archivo de configuración personalizado alojado en Amazon S3, el rol de IAM de ejecución de tareas debe incluir el permiso `s3:GetObject`.

Para obtener información sobre cómo usar varios archivos de configuración con Amazon ECS, incluidos los archivos que usted aloja o los archivos en Amazon S3, consulte Init process for Fluent Bit on ECS, multi-config support.

Para obtener información acerca de las configuraciones de ejemplo, consulte Definición de tareas de Amazon ECS de ejemplo: enrutar registros a FireLens

Configuración de los registros de Amazon ECS para conseguir un alto rendimiento

Al crear una definición de tareas, puede indicar el número de líneas de registro que se almacenan en búfer en la memoria mediante la especificación del valor en el `log-driver-buffer-limit`. Para obtener más información, consulte Controladores de registro Fluentd en la documentación de Docker.

Utilice esta opción cuando haya un alto rendimiento, ya que Docker podría quedarse sin memoria de búfer y descartar mensajes de búfer para poder agregar nuevos mensajes.

Al utilizar FireLens para Amazon ECS con la opción de límite de búfer, tenga en cuenta lo siguiente:

- Esta opción se admite en el tipo de lanzamiento de Amazon EC2 y en el tipo de lanzamiento de Fargate con versión de plataforma `1.4.0` o posterior.
- La opción solo es válida cuando `logDriver` se establece en `awsfirelens`.
- El límite de búfer predeterminado es de `1048576` líneas de registro.
- El límite del búfer debe ser igual o mayor que `0` y menor que las líneas de registro `536870912`.
- La cantidad máxima de memoria utilizada para este búfer es el producto del tamaño de cada línea de registro y el tamaño del búfer. Por ejemplo, si las líneas de registro de la aplicación tienen un promedio de 2 KiB, un límite de búfer de 4096 utilizaría como máximo 8 MiB. La cantidad total de memoria asignada a nivel de tarea debe ser superior a la cantidad de memoria asignada a todos los contenedores, además del búfer de memoria del controlador de registro.

Cuando se especifica el controlador de registros `awsfirelens` en una definición de tarea, el agente de contenedor de Amazon ECS introduce las siguientes variables de entorno en el contenedor:

`FLUENT_HOST`

La dirección IP que está asignada al contenedor FireLens.

> **Note**
>
> Si utiliza el tipo de lanzamiento EC2 con el modo de red `bridge`, la variable de entorno `FLUENT_HOST` del contenedor de aplicaciones puede dejar de ser precisa tras reiniciar el contenedor del router de registro de FireLens (el contenedor con el objeto `firelensConfiguration` en su definición de contenedor). Esto se debe a que `FLUENT_HOST` es una dirección IP dinámica y puede cambiar tras un reinicio. El registro directo desde el contenedor de la aplicación en la dirección IP `FLUENT_HOST` puede empezar a fallar después de que la dirección cambie. Para obtener más información acerca de cómo reiniciar contenedores individuales, consulte Reinicio de contenedores individuales en tareas de Amazon ECS con políticas de reinicio de contenedores.

`FLUENT_PORT`

El puerto en el que se está escuchando el protocolo Fluent Forward.

Puede utilizar las variables de entorno `FLUENT_HOST` y `FLUENT_PORT` para iniciar sesión directamente en el enrutador de registros desde el código en lugar de pasar por `stdout`. Para obtener más información, consulte fluent-logger-golang en GitHub.

A continuación, se muestra la sintaxis para indicar la propiedad `log-driver-buffer-limit`. Sustituya `my_service_` por el nombre del servicio:

```
{
    "containerDefinitions": [
        {
            "name": "my_service_log_router",
            "image": "public.ecr.aws/aws-observability/aws-for-fluent-bit:stable",
            "cpu": 0,
            "memoryReservation": 51,
            "portMappings": [],
            "essential": true,
            "environment": [],
            "mountPoints": [],
            "volumesFrom": [],
            "user": "0",
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/ecs-aws-firelens-sidecar-container",
                    "mode": "non-blocking",
                    "awslogs-create-group": "true",
                    "max-buffer-size": "25m",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "firelens"
                },
                "secretOptions": []
            },
            "systemControls": [],
            "firelensConfiguration": {
                "type": "fluentbit"
            }
        },
        {
```

```
            "essential": true,
            "image": "public.ecr.aws/docker/library/httpd:latest",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "firehose",
                    "region": "us-west-2",
                    "delivery_stream": "my-stream",
                    "log-driver-buffer-limit": "51200"
                }
            },
            "dependsOn": [
                {
                    "containerName": "log_router",
                    "condition": "START"
                }
            ],
            "memoryReservation": 100
        }
    ]
}
```

AWS para repositorios de imágenes de Fluent Bit para Amazon ECS

AWS proporciona una imagen de Fluent Bit con complementos para Registros de CloudWatch y Firehose. Recomendamos usar Fluent Bit como router de registro porque tiene una tasa de utilización de recursos más baja que Fluentd. Para obtener más información, consulte CloudWatch Logs para Fluent Bit y Amazon Kinesis Firehose para Fluent Bit.

La imagen de AWS para Fluent Bit está disponible en la galería pública y en un repositorio de Amazon ECR en la mayoría de las Regiones de AWS para lograr alta disponibilidad.

Galería pública de Amazon ECR

La imagen de AWS para Fluent Bit está disponible en la galería pública de Amazon ECR. Esta es la ubicación recomendada para descargar la imagen de AWS para Fluent Bit, ya que es un repositorio público y está disponible para su uso desde todas las Regiones de AWS. Para obtener más información, consulte aws-por-fluent-bit en la galería pública de Amazon ECR.

Linux

La imagen Fluent Bit de AWS de la galería pública de Amazon ECR es compatible con el sistema operativo Amazon Linux con la arquitectura `ARM 64` o `x86-64`.

Para extraer la imagen de AWS para Fluent Bit de la galería pública de Amazon ECR, puede especificar la URL del repositorio con la etiqueta de imagen deseada. Para ver las etiquetas de imágenes disponibles, consulte la pestaña Image tags (Etiquetas de imágenes) en la galería pública de Amazon ECR.

A continuación, se muestra la sintaxis que se debe usar para la CLI de Docker.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:tag
```

Por ejemplo, para extraer la imagen de AWS para Fluent Bit más estable, puede utilizar este comando de la CLI de Docker.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:stable
```

> **Note**
>
> Se permiten extracciones sin autenticación, pero tienen un límite de tasa inferior al de las extracciones autenticadas. Para autenticar el uso de la cuenta de AWS antes de la extracción, utilice el comando a continuación.
>
> ```
> aws ecr-public get-login-password --region us-east-1 | docker login --username
> AWS --password-stdin public.ecr.aws
> ```

Windows

La imagen Fluent Bit de AWS de la galería pública de Amazon ECR es compatible con la arquitectura `AMD64` con los siguientes sistemas operativos:

- Windows Server 2022 Full
- Windows Server 2022 Core
- Windows Server 2019 Full
- Windows Server 2019 Core

Los contenedores de Windows que están en AWS Fargate no admiten FireLens.

Para extraer la imagen de AWS para Fluent Bit de la galería pública de Amazon ECR, puede especificar la URL del repositorio con la etiqueta de imagen deseada. Para ver las etiquetas de imágenes disponibles, consulte la pestaña Image tags (Etiquetas de imágenes) en la galería pública de Amazon ECR.

A continuación, se muestra la sintaxis que se debe usar para la CLI de Docker.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:tag
```

Por ejemplo, para extraer la imagen de AWS para Fluent Bit estable más nueva, puede utilizar este comando de la CLI de Docker.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:windowsservercore-
stable
```

> **Note**
>
> Se permiten extracciones sin autenticación, pero tienen un límite de tasa inferior al de las extracciones autenticadas. Para autenticar el uso de la cuenta de AWS antes de la extracción, utilice el comando a continuación.
>
> ```
> aws ecr-public get-login-password --region us-east-1 | docker login --username
> AWS --password-stdin public.ecr.aws
> ```

Amazon ECR

La imagen de AWS para Fluent Bit está disponible en Amazon ECR para lograr alta disponibilidad. Estas imágenes están disponibles en la mayoría de las Regiones de AWS, incluso en las de AWS GovCloud (US).

Linux

Para recuperar la URI de la imagen de AWS para Fluent Bit más estable, utilice el siguiente comando.

```
aws ssm get-parameters \
```

```
      --names /aws/service/aws-for-fluent-bit/stable \
      --region us-east-1
```

Para obtener una lista de todas las versiones de la imagen de AWS para Fluent Bit y consultar el parámetro del Parameter Store de Systems Manager, utilice el siguiente comando.

```
aws ssm get-parameters-by-path \
      --path /aws/service/aws-for-fluent-bit \
      --region us-east-1
```

Para consultar la imagen de AWS para Fluent Bit estable más nueva en una plantilla de AWS CloudFormation, puede hacer referencia al nombre del almacén de parámetros de Systems Manager. A continuación, se muestra un ejemplo:

```
Parameters:
  FireLensImage:
    Description: Fluent Bit image for the FireLens Container
    Type: AWS::SSM::Parameter::Value<String>
    Default: /aws/service/aws-for-fluent-bit/stable
```

Windows

Para recuperar la URI de la imagen de AWS para Fluent Bit más estable, utilice el siguiente comando.

```
aws ssm get-parameters \
      --names /aws/service/aws-for-fluent-bit/windowsservercore-stable \
      --region us-east-1
```

Para obtener una lista de todas las versiones de la imagen de AWS para Fluent Bit y consultar el parámetro del Parameter Store de Systems Manager, utilice el siguiente comando.

```
aws ssm get-parameters-by-path \
      --path /aws/service/aws-for-fluent-bit/windowsservercore \
      --region us-east-1
```

Para consultar la imagen de AWS para Fluent Bit más estable en una plantilla de AWS CloudFormation, puede hacer referencia al nombre del Parameter Store de Systems Manager. A continuación, se muestra un ejemplo:

```
Parameters:
  FireLensImage:
    Description: Fluent Bit image for the FireLens Container
    Type: AWS::SSM::Parameter::Value<String>
    Default: /aws/service/aws-for-fluent-bit/windowsservercore-stable
```

Definición de tareas de Amazon ECS de ejemplo: enrutar registros a FireLens

Para utilizar el enrutamiento de registros personalizado con FireLens, debe especificar lo siguiente en la definición de tareas:

- Un contenedor del enrutador de registros que incluye una configuración de FireLens. Recomendamos que el contenedor se marque como `essential`.
- Uno o varios contenedores de aplicaciones que incluyen una configuración de registro que especifica el controlador de registros `awsfirelens`.
- Un nombre de recurso de Amazon (ARN) de rol de IAM de tarea que incluya los permisos necesarios para que la tarea envíe los registros.

Al crear una nueva definición de tarea mediante la AWS Management Console, hay una sección de integración de FireLens que permite añadir fácilmente un contenedor de router de registros. Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

Amazon ECS convierte la configuración de registros y genera la configuración de salida de Fluentd o Fluent Bit. La configuración de salida se monta en el contenedor de enrutamiento de registros en `/fluent-bit/etc/fluent-bit.conf` para Fluent Bit y `/fluentd/etc/fluent.conf` para Fluentd.

> **Important**
>
> FireLens escucha en el puerto `24224`. Por lo tanto, para asegurarse de que el direccionador de registros FireLens no sea accesible fuera de la tarea, debe permitir la entrada de tráfico en el puerto `24224` en el grupo de seguridad que utiliza la tarea. Para tareas que utilizan el modo de red `awsvpc`, es el grupo de seguridad asociado a la tarea. Para tareas que utilizan el modo de red `host`, es el grupo de seguridad asociado a la instancia de Amazon EC2 que aloja la tarea. Para tareas que utilizan el modo de red `bridge`, no cree asignaciones de puertos que utilicen el puerto `24224`.

De forma predeterminada, Amazon ECS agrega campos adicionales a las entradas de registro, que ayudan a identificar su origen.

- `ecs_cluster`: el nombre del clúster del que forma parte la tarea.
- `ecs_task_arn`: nombre de recurso de Amazon (ARN) de la tarea de la que el contenedor forma parte.
- `ecs_task_definition`: el nombre y la revisión de la definición de tareas que utiliza la tarea.
- `ec2_instance_id`: el ID de la instancia de Amazon EC2 en la que está alojado el contenedor. Este campo solo es válido para las tareas que utilizan el tipo de lanzamiento EC2.

Puede establecer `enable-ecs-log-metadata` en `false` si no quiere los metadatos.

En el siguiente ejemplo de definición de tarea, se define un contenedor de enrutamiento de registros que utiliza Fluent Bit para enrutar sus registros a CloudWatch Logs. También define un contenedor de aplicaciones que utiliza una configuración de registros para dirigirlos a Amazon Data Firehose y establecer en 2 MiB la memoria que se utiliza para guardar en búfer los eventos.

> **Note**
>
> Para ver más ejemplos de definiciones de tareas, consulte Ejemplos de FireLens en Amazon ECS en GitHub.

```
{
  "family": "firelens-example-firehose",
  "taskRoleArn": "arn:aws:iam::123456789012:role/ecs_task_iam_role",
  "containerDefinitions": [
    {
            "name": "log_router",
            "image": "public.ecr.aws/aws-observability/aws-for-fluent-bit:stable",
            "cpu": 0,
            "memoryReservation": 51,
            "portMappings": [],
            "essential": true,
            "environment": [],
            "mountPoints": [],
            "volumesFrom": [],
            "user": "0",
            "logConfiguration": {
```

```
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/ecs-aws-firelens-sidecar-container",
                    "mode": "non-blocking",
                    "awslogs-create-group": "true",
                    "max-buffer-size": "25m",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "firelens"
                },
                "secretOptions": []
            },
            "systemControls": [],
            "firelensConfiguration": {
                "type": "fluentbit"
            }
        },
    {
      "essential": true,
      "image": "public.ecr.aws/docker/library/httpd:latest",
      "name": "app",
      "logConfiguration": {
        "logDriver": "awsfirelens",
        "options": {
          "Name": "firehose",
          "region": "us-west-2",
          "delivery_stream": "my-stream",
          "log-driver-buffer-limit": "2097152"
        }
      },
      "memoryReservation": 100
    }
  ]
}
```

Los pares de clave-valor especificados como opciones en el objeto `logConfiguration` se utilizan para generar la configuración de salida de Fluentd o Fluent Bit. A continuación, se muestra un ejemplo de código de una definición de salida de Fluent Bit.

```
[OUTPUT]
    Name   firehose
    Match  app-firelens*
    region us-west-2
```

```
    delivery_stream my-stream
```

> **Note**
>
> FireLens administra la configuración `match`. No se especifica la configuración de `match` en la definición de tarea.

Uso de un archivo de configuración personalizado

Puede especificar un archivo de configuración personalizado. El formato del archivo de configuración es el formato nativo del enrutador de registros que está utilizando. Para obtener más información, consulte Fluentd Config File Syntax y YAML Configuration.

En el archivo de configuración personalizado, en el caso de las tareas que utilizan el modo de red `bridge` o `awsvpc`, no establezca una entrada directa Fluentd o Fluent Bit a través de TCP porque FireLens la agrega a la configuración de entrada.

La configuración de FireLens debe incluir las siguientes opciones para especificar un archivo de configuración personalizado:

`config-file-type`

La ubicación de origen del archivo de configuración personalizado. Las opciones disponibles son `s3` o `file`.

> **Note**
>
> Las tareas que están alojadas en AWS Fargate solo admiten el tipo de archivo de configuración `file`.

`config-file-value`

El origen del archivo de configuración personalizado. Si se utiliza el tipo de archivo de configuración `s3`, el valor del archivo de configuración es el ARN completo del bucket y el archivo de Amazon S3. Si se utiliza el tipo de archivo de configuración `file`, el valor del archivo de configuración es la ruta completa del archivo de configuración que existe en la imagen del contenedor o en un volumen que se monta en el contenedor.

> **Important**
>
> Cuando se utilice un archivo de configuración personalizado, se debe especificar una ruta diferente a la que utiliza FireLens. Amazon ECS reserva la ruta de archivo `/fluent-bit/etc/fluent-bit.conf` para Fluent Bit y la `/fluentd/etc/fluent.conf` para Fluentd.

En el ejemplo siguiente se muestra la sintaxis necesaria al especificar una configuración personalizada.

> **Important**
>
> Para especificar un archivo de configuración personalizado alojado en Amazon S3, asegúrese de haber creado un rol de IAM de ejecución de tareas con los permisos adecuados.

A continuación se muestra la sintaxis necesaria al especificar una configuración personalizada.

```
{
  "containerDefinitions": [
    {
      "essential": true,
      "image": "906394416424.dkr.ecr.us-west-2.amazonaws.com/aws-for-fluent-
bit:stable",
      "name": "log_router",
      "firelensConfiguration": {
        "type": "fluentbit",
        "options": {
          "config-file-type": "s3 | file",
          "config-file-value": "arn:aws:s3:::amzn-s3-demo-bucket/fluent.conf
 | filepath"
        }
      }
    }
  ]
}
```

> **Note**
>
> Las tareas alojadas en AWS Fargate solo admiten el tipo de archivo de configuración `file`.

Uso de imágenes de contenedor que no sean de AWS en Amazon ECS

Utilice el registro privado para almacenar las credenciales en AWS Secrets Manager y hacer referencia a ellas en la definición de tarea. Esto proporciona una forma de hacer referencia a las imágenes de contenedores que existen en registros privados fuera de AWS que requieren autenticación en las definiciones de tareas. Esta función es compatible con tareas alojadas en Fargate, Amazon EC2 instances (Instancias de Amazon EC2) e instancias externas que utilizan Amazon ECS Anywhere.

> **Important**
>
> Si la definición de la tarea hace referencia a una imagen que está almacenada en Amazon ECR, este tema no es válido. Para obtener más información, consulte Utilización de imágenes de Amazon ECR con Amazon ECS en la Guía del usuario de Amazon Elastic Container Registry.

Para las tareas alojadas en instancias de Amazon EC2, esta característica requiere la versión `1.19.0` del agente de contenedor o una posterior. No obstante, recomendamos utilizar la versión del agente de contenedor más reciente. Para obtener información acerca de cómo comprobar la versión del agente y actualizar a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS.

Para las tareas alojadas en Fargate, esta función requiere la versión de plataforma `1.2.0` o posterior. Para obtener más información, consulta Versiones de la plataforma Fargate para Amazon ECS.

En la definición de contenedor, especifique el objeto `repositoryCredentials` con los detalles del secreto que ha creado. El secreto al que se hace referencia puede proceder de una Región de AWS o cuenta distintas de la tarea que lo utiliza.

> **Note**
>
> Cuando se utiliza la API de Amazon ECS, la AWS CLI o el SDK de AWS, si el secreto existe en la misma Región de AWS que la tarea que se va a lanzar, se puede utilizar el ARN completo o el nombre del secreto. Si el secreto existe en otra cuenta, debe especificarse el ARN completo del secreto. Cuando se utiliza la AWS Management Console, siempre debe especificarse el ARN completo del secreto.

A continuación, se incluye un fragmento de definición de tareas que muestra los parámetros necesarios:

Sustituya los siguientes parámetros:

- *private-repo* con el nombre de host del repositorio privado
- *private-image* con el nombre de la imagen
- *arn:aws:secretsmanager:region:aws_account_id:secret:secret_name* con el nombre de recurso de Amazon (ARN) secreto

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
        }
    }
]
```

> **Note**
>
> Otro método para habilitar la autenticación de registros privados consiste en utilizar las variables de entorno del agente de contenedor de Amazon ECS para la autenticación en registros privados. Este método solo se admite para tareas alojadas en Amazon EC2 instances (Instancias de Amazon EC2). Para obtener más información, consulte Configuración de instancias de contenedor de Amazon ECS para imágenes de Docker privadas .

Para utilizar el registro privado

1. La definición de tareas debe tener un rol de ejecución de tareas. Esto permite que el agente de contenedor extraiga la imagen del contenedor. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

   La autenticación de registro privado permite que sus tareas de Amazon ECS obtengan imágenes de contenedores de registros privados externos a AWS (como Docker Hub, Quay.io, o su propio registro privado) que requieren credenciales de autenticación. Esta característica utiliza Secrets Manager para almacenar de forma segura las credenciales de registro, que luego se referencian en la definición de la tarea mediante el parámetro `repositoryCredentials`.

   Para obtener más información sobre la configuración de la autenticación de registro privado, consulte Uso de imágenes de contenedor no AWS en Amazon ECS.

   Para proporcionar acceso a los secretos que contienen las credenciales de su registro privado, agregue los siguientes permisos como una política insertada al rol de ejecución de tareas. Para obtener más información, consulte Adición y eliminación de políticas de IAM.

   - `secretsmanager:GetSecretValue`: obligatorio para recuperar las credenciales de registro privado de Secrets Manager.
   - `kms:Decrypt`: obligatorio solo si el secreto utiliza una clave de KMS personalizada y no la clave predeterminada. Se debe agregar el nombre de recurso de Amazon (ARN) de la clave de personalizada como un recurso.

   Es siguiente es un ejemplo de política insertada que agrega los permisos.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [

 "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:secret_name",
                "arn:aws:kms:<region>:<aws_account_id>:key/key_id"
```

```
            ]
        }
    ]
}
```

2. Utilice AWS Secrets Manager para crear un secreto para sus credenciales de registros privados. Para obtener información acerca de la creación de un secreto, consulte Create an AWS Secrets Manager secret en la Guía del usuario de AWS Secrets Manager.

   Ingrese las credenciales de registros privados con el siguiente formato:

```
{
  "username" : "privateRegistryUsername",
  "password" : "privateRegistryPassword"
}
```

3. Registre una definición de tareas. Para obtener más información, consulte the section called "Creación de una definición de tareas con la consola".

Reinicio de contenedores individuales en tareas de Amazon ECS con políticas de reinicio de contenedores

Puede habilitar las políticas de reinicio para los contenedores esenciales y no esenciales definidas en la definición de su tarea para superar los errores transitorios con mayor rapidez y mantener la disponibilidad de las tareas. Al habilitar una política de reinicio para un contenedor, Amazon ECS puede reiniciar el contenedor si se cierra, sin necesidad de reemplazar la tarea.

Las políticas de reinicio no están habilitadas para los contenedores de forma predeterminada. Al habilitar una política de reinicio para un contenedor, puede especificar códigos de salida en los que no se reiniciará el contenedor. Pueden ser códigos de salida que indican que la realización ha sido correcta, como códigos de salida `0`, que no requieren un reinicio. También puede especificar durante cuánto tiempo debe funcionar correctamente un contenedor antes de intentar su reinicio. Para obtener más información sobre estos parámetros, consulte Política de reinicio. Para ver un ejemplo de definición de tarea que especifique estos valores, consulte Especificación de una política de reinicio de contenedor en una definición de tarea de Amazon ECS.

Puede utilizar el punto de conexión de metadatos de tareas de Amazon ECS o CloudWatch Container Insights para supervisar el número de veces que se ha reiniciado un contenedor. Para más información sobre el punto de conexión de los metadatos de la tarea, consulte Versión 4 del

punto de conexión de metadatos de tareas de Amazon ECS y Versión 4 del punto de conexión de los metadatos de tareas de Amazon ECS para tareas en Fargate. Para obtener más información sobre las métricas de Container Insights, consulte Métricas de Amazon ECS Container Insights en la Guía del usuario de Amazon CloudWatch.

Las políticas de reinicio del contenedor son compatibles con tareas alojadas en Fargate, Instancias de Amazon EC2 e instancias externas que utilizan Amazon ECS Anywhere.

Consideraciones

Tenga en cuenta lo siguiente antes de habilitar una política de reinicio para su contenedor:

- Las políticas de reinicio no son compatibles con contenedores de Windows en Fargate.
- Para las tareas alojadas en instancias de Amazon EC2, esta característica requiere la versión `1.86.0` del agente de contenedor o una posterior. No obstante, recomendamos utilizar la versión del agente de contenedor más reciente. Para obtener información acerca de cómo comprobar la versión del agente y actualizar a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS.
- Para las tareas alojadas en Fargate, esta función requiere la versión de plataforma `1.4.0` o posterior. Para obtener más información, consulta Versiones de la plataforma Fargate para Amazon ECS.
- Si utiliza el tipo de lanzamiento EC2 con el modo de red `bridge`, la variable de entorno `FLUENT_HOST` del contenedor de aplicaciones puede dejar de ser precisa tras reiniciar el contenedor del router de registro de Firelens (el contenedor con el objeto `firelensConfiguration` en su definición de contenedor). Esto se debe a que `FLUENT_HOST` es una dirección IP dinámica y puede cambiar tras un reinicio. El registro directo desde el contenedor de la aplicación en la dirección IP `FLUENT_HOST` puede empezar a fallar después de que la dirección cambie. Para obtener más información acerca de `FLUENT_HOST`, consulte Configuración de los registros de Amazon ECS para conseguir un alto rendimiento.
- El agente de Amazon ECS gestiona las políticas de reinicio de contenedores. Si, por alguna razón inesperada, el agente de Amazon ECS falla o deja de ejecutarse, el contenedor no se reiniciará.
- El período de intento de reinicio definido en su política determina el período de tiempo (en segundos) que debe funcionar el contenedor antes de que Amazon ECS lo reinicie.

Especificación de una política de reinicio de contenedor en una definición de tarea de Amazon ECS

Para especificar una política de reinicio para un contenedor en una definición de tarea, dentro de la definición del contenedor, se debe especificar el objeto `restartPolicy`. Para obtener más información sobre el objeto de nodo `restartPolicy`, consulte Política de reinicio.

A continuación, se muestra una definición de tarea de ejemplo con el tipo de lanzamiento de Fargate o contenedores Linux que configura un servidor web. La definición del contenedor incluye el objeto `restartPolicy`, y `enabled` establecido en verdadero para habilitar una política de reinicio para el contenedor. El contenedor debe funcionar durante 180 segundos antes del reinicio y no se reiniciará si sale con el código de salida `0`, lo que indica éxito.

```
{
  "containerDefinitions": [
    {
      "command": [
        "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</title>
 <style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
 <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
 ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && httpd-
foreground\""
      ],
      "entryPoint": ["sh", "-c"],
      "essential": true,
      "image": "public.ecr.aws/docker/library/httpd:2.4",
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/fargate-task-definition",
          "awslogs-region": "us-east-1",
          "awslogs-stream-prefix": "ecs"
        }
      },
      "name": "sample-fargate-app",
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80,
          "protocol": "tcp"
        }
```

```
      ],
      "restartPolicy": {
        "enabled": true,
        "ignoredExitCodes": [0],
        "restartAttemptPeriod": 180
      }
    }
  ],
  "cpu": "256",
  "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
  "family": "fargate-task-definition",
  "memory": "512",
  "networkMode": "awsvpc",
  "runtimePlatform": {
    "operatingSystemFamily": "LINUX"
  },
  "requiresCompatibilities": ["FARGATE"]
}
```

Después de haber registrado una definición de tarea con el objeto `restartPolicy` en una definición de contenedor, puede ejecutar una tarea o crear un servicio con dicha definición de tarea. Para obtener más información, consulte Ejecución de una aplicación como tarea de Amazon ECS y Creación de una implementación de actualización continua de Amazon ECS.

Transferencia de datos confidenciales a un contenedor de Amazon ECS

Puede transferir de forma segura datos confidenciales, como las credenciales de una base de datos, a su contenedor.

Las aplicaciones suelen utilizar secretos, como las claves de API y las credenciales de las bases de datos, para acceder a otros sistemas. Suelen consistir en un nombre de usuario y una contraseña, un certificado o una clave de API. El acceso a estos secretos debe restringirse a las entidades principales de IAM específicas que utilizan IAM e deben inyectarse en los contenedores durante el tiempo de ejecución.

Los secretos se pueden introducir sin problemas en los contenedores desde AWS Secrets Manager y Parameter Store de Amazon EC2 Systems Manager. Puede hacer referencia a estos secretos en su tarea mediante cualquiera de las siguientes formas.

1. Se hace referencia a ellos como variables de entorno que utilizan el parámetro de definición del contenedor de `secrets`.

2. Se hace referencia a ellos como `secretOptions` si su plataforma de registro requiriera autenticación. Para obtener más información, consulte las opciones de configuración del registro.
3. Se hace referencia a ellos como secretos y se extraen mediante imágenes que utilizan el parámetro de definición del contenedor `repositoryCredentials` si el registro del que se extrae el contenedor requiere autenticación. Utilice este método cuando extraiga imágenes de la galería pública de Amazon ECR. Para obtener más información, consulte Autenticación de registros privados para tareas.

Se recomienda que realice las siguientes acciones al configurar la administración de secretos.

Utilización del almacén de parámetros AWS Secrets Manager o AWS Systems Manager para el almacenamiento de materiales secretos

Debe almacenar de forma segura las claves de la API, las credenciales de bases de datos y otros materiales secretos en Secrets Manager o como un parámetro cifrado en el almacén de parámetros de Systems Manager. Estos servicios son similares porque ambos son almacenes clave-valor administrados que utilizan AWS KMS para cifrar información confidencial. Secrets Manager, sin embargo, también incluye la capacidad de rotar de forma automática los secretos, generar secretos aleatorios y compartirlos entre las cuentas. Para utilizar estas características, use Secrets Manager. De lo contrario, utilice parámetros cifrados en el Almacén de parámetros de Systems Manager.

> **Important**
>
> Si su secreto cambia, debe forzar una nueva implementación o iniciar una nueva tarea para recuperar el valor secreto más reciente. Para obtener más información, consulte los temas siguientes:
>
> - Tareas: detenga la tarea y, luego, iníciela. Para obtener más información, consulte Detención de una tarea de Amazon ECS y Ejecución de una aplicación como tarea de Amazon ECS.
> - Servicio: actualice el servicio y utilice la opción de forzar una nueva implementación. Para obtener más información, consulte Actualización de un servicio de Amazon ECS.

Recuperación de datos de un bucket cifrado de Amazon S3

Debe almacenar los secretos en un bucket cifrado de Amazon S3 y utilizar roles de tareas para restringir el acceso a esos secretos. Esto evita que los valores de las variables de entorno se filtre

de manera inadvertida en los registros y se revelen al ejecutar `docker inspect`. De este modo, la aplicación debe escribirse para leer el secreto del bucket de Amazon S3. Para obtener instrucciones, consulte Establecer el comportamiento predeterminado de cifrado del lado del servidor para buckets de Amazon S3.

Montar el secreto en un volumen utilizando un contenedor sidecar

Dado que existe un riesgo elevado de filtración de datos con las variables de entorno, debe utilizar un contenedor sidecar que lea sus secretos de AWS Secrets Manager y los escriba en un volumen compartido. Este contenedor puede ejecutarse y salir antes que el contenedor de la aplicación mediante pedidos de contenedores de Amazon ECS. De este modo, el contenedor de la aplicación monta posteriormente el volumen en el que se escribió el secreto. Al igual que el método de bucket de Amazon S3, la aplicación debe escribirse para leer el secreto del volumen compartido. Como el volumen se limita a la tarea, este se elimina automáticamente cuando la tarea se detiene. Para ver un ejemplo, consulte el proyecto task-def.json.

En Amazon EC2, el volumen en el que está escrito el secreto se puede cifrar con una clave administrada por el cliente de AWS KMS. En AWS Fargate, el almacenamiento por volumen se cifra automáticamente mediante una clave administrada por el servicio.

Transferencia de una variable de entorno individual a un contenedor de Amazon ECS

> **Important**
>
> Recomendamos almacenar la información confidencial en cualquiera de los secretos de AWS Secrets Manager o en los parámetros AWS Systems Manager Parameter Store. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.
>
> Las variables de entorno especificadas en la definición de tarea son aptas para que las lean todos los usuarios y roles a los que se les permite la acción `DescribeTaskDefinition` para la definición de tareas.

Puede transferir variables de entorno a sus contenedores de las siguientes maneras:

- Individualmente con el parámetro de definición de contenedor `environment`. Se asigna a la opción `--env` de docker container run.

- En bloque, mediante el parámetro de definición de contenedor de `environmentFiles` para enumerar uno o más archivos que contienen las variables de entorno. El archivo debe estar alojado en Amazon S3. Se asigna a la opción `--env-file` de docker run.

A continuación, se presenta un fragmento de una definición de tarea que muestra cómo especificar variables de entorno individuales.

```
{
    "family": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            ...
            "environment": [
                {
                    "name": "variable",
                    "value": "value"
                }
            ],
            ...
        }
    ],
    ...
}
```

Transferencia de variables de entorno a un contenedor de Amazon ECS

> **Important**
>
> Recomendamos almacenar la información confidencial en cualquiera de los secretos de AWS Secrets Manager o en los parámetros AWS Systems Manager Parameter Store. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.
>
> Los archivos de variables de entorno son objetos de Amazon S3 y se aplican todas las consideraciones de seguridad de Amazon S3.
>
> No puede utilizar el parámetro `environmentFiles` en los contenedores de Windows ni en los contenedores de Windows en Fargate.

Puede crear un archivo de variables de entorno y almacenarlo en Amazon S3 para pasar las variables de entorno a su contenedor.

Al especificar variables de entorno en un archivo, puede introducir variables de entorno en bloque. En la definición de contenedor, especifique el objeto `environmentFiles` con una lista de buckets de Amazon S3 con los archivos de variables de entorno.

Amazon ECS no aplica un límite de tamaño a las variables de entorno, pero un archivo de variables de entorno grande podría llenar el espacio en el disco. Cada tarea que utiliza un archivo de variables de entorno hace que se descargue una copia del archivo en el disco. Amazon ECS elimina el archivo como parte de la limpieza de tareas.

Para obtener información sobre las variables de entorno compatibles, consulte Parámetros de definición avanzada de contenedores: entorno.

Tenga en cuenta lo siguiente al especificar un archivo de variable de entorno en una definición de contenedor.

- Para las tareas de Amazon ECS en Amazon EC2, las instancias de contenedor requieren la versión `1.39.0` del agente de contenedor o una posterior para utilizar esta característica. Para obtener información acerca de cómo comprobar la versión del agente y actualizar a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS.
- Para realizar tareas de Amazon ECS en AWS Fargate, sus tareas deben utilizar la versión `1.4.0` de la plataforma o una posterior (para Linux) a fin de utilizar esta característica. Para obtener más información, consulte Versiones de la plataforma Fargate para Amazon ECS.

  Compruebe que la variable sea compatible con la plataforma del sistema operativo. Para obtener más información, consulte the section called “Definiciones de contenedores” y the section called “Otros parámetros de definición de tarea”.
- El archivo debe usar la extensión de archivo `.env` y la codificación UTF-8.
- Se requiere el rol de ejecución de tareas para utilizar esta característica con los permisos adicionales para Amazon S3. De este modo el agente de contenedor puede extraer el archivo de variables de entorno de Amazon S3. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.
- Hay un límite de 10 archivos por definición de tarea.
- Cada línea de un archivo de entorno debe contener una variable de entorno con el formato `VARIABLE=VALUE`. Los espacios o las comillas se incluyen como parte de los valores para los

archivos de Amazon ECS. Las líneas que comienzan por # se tratan como comentarios y se ignoran. Para obtener más información acerca de la sintaxis del archivo de variables de entorno, consulte Establecimiento de variables de entorno (-e, --env, --env-file) en la documentación de Docker.

A continuación, se presenta la sintaxis adecuada.

```
#This is a comment and will be ignored
VARIABLE=VALUE
ENVIRONMENT=PRODUCTION
```

- Si hay variables de entorno especificadas mediante el parámetro `environment` en una definición de contenedor, tienen prioridad sobre las variables incluidas en un archivo de entorno.
- Si se especifican varios archivos de entorno que contienen la misma variable, se procesan en orden de entrada. Esto significa que se utiliza el primer valor de la variable y se ignoran los valores posteriores de las variables duplicadas. Le recomendamos que utilice nombres de variables únicos.
- Si se especifica un archivo de entorno como anulación de un contenedor, se utiliza. Además, se omite cualquier otro archivo de entorno especificado en la definición de contenedor.
- Las siguientes reglas se aplican al tipo de lanzamiento de Fargate:
  - El archivo se gestiona como un archivo .env-file nativo de Docker.
  - Las definiciones de contenedor que hacen referencia a variables de entorno que están en blanco y almacenadas en Amazon S3 no aparecen en el contenedor.
  - No se admite la gestión del escape del intérprete de comandos.
  - El punto de entrada del contenedor interpreta los valores `VARIABLE`.

Ejemplo

A continuación, se presenta un fragmento de una definición de tarea que muestra cómo especificar un archivo de variable de entorno.

```
{
    "family": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            ...
```

```
            "environmentFiles": [
                {
                    "value": "arn:aws:s3:::amzn-s3-demo-
bucket/envfile_object_name.env",
                    "type": "s3"
                }
            ],
            ...
        }
    ],
    ...
}
```

Transmisión de secretos de Secrets Manager mediante programación en Amazon ECS

En lugar de codificar la información confidencial en texto sin formato en la aplicación, puede utilizar Secrets Manager para almacenar los datos confidenciales.

Recomendamos este método para recuperar datos confidenciales porque si el secreto de Secrets Manager se actualiza luego, la aplicación recuperará automáticamente la versión más reciente del secreto.

Cree un secreto en Secrets Manager. Después de crear un secreto de Secrets Manager, actualice el código de la aplicación para recuperar el secreto.

Revise las siguientes consideraciones antes de proteger datos confidenciales en Secrets Manager.

- Solo se admiten los secretos que almacenan datos de texto, que son secretos creados con el parámetro `SecretString` de la API CreateSecret. No son compatibles los secretos que almacenan datos binarios, que son secretos creados con el parámetro `SecretBinary` de la API CreateSecret.
- Utilice los puntos de conexión de VPC de interfaz para mejorar los controles de seguridad. Debe crear los puntos de conexión de VPC de interfaz para Secrets Manager. Para obtener información sobre el punto de conexión de VPC, consulte Crear puntos de conexión de VPC en la Guía del usuario de AWS Secrets Manager.
- La VPC que utiliza la tarea debe usar la resolución de DNS.
- En la definición de tareas se debe utilizar un rol de tareas con los permisos adicionales para Secrets Manager. Para obtener más información, consulte Rol de IAM de tarea de Amazon ECS.

Creación de secretos de Secrets Manager

Puede utilizar la consola de Secrets Manager para crear un secreto con su información confidencial. Para obtener información acerca de la creación de secretos, consulte Crear un secreto de AWS Secrets Manager en la Guía del usuario de AWS Secrets Manager.

Actualice la aplicación para que recupere secretos de Secrets Manager mediante programación

Puede recuperar secretos mediante una llamada a las API de Secrets Manager directamente desde la aplicación. Para información, consulte Retrieve secrets from AWS Secrets Manager en la Guía del usuario de AWS Secrets Manager.

Para recuperar los datos confidenciales almacenados en el AWS Secrets Manager, consulte Ejemplos de código para AWS Secrets Manager mediante SDK de AWS en la Biblioteca de ejemplos de código del SDK de AWS.

Transmisión de secretos de Almacén de parámetros de Systems Manager mediante programación en Amazon ECS

El almacén de parámetros de Systems Manager proporciona almacenamiento y administración seguros de secretos. Puede almacenar datos como contraseñas, cadenas de base de datos, ID de instancias de EC2 e ID de AMI y códigos de licencia como valores de parámetros, en lugar de integrar con codificación rígida esta información en la aplicación. Puede almacenar valores como texto sin formato o como datos cifrados.

Recomendamos este método para recuperar datos confidenciales porque si el parámetro del Almacén de parámetros de Systems Manager se actualiza posteriormente, la aplicación recuperará automáticamente la versión más reciente.

Revise las siguientes consideraciones antes de proteger datos confidenciales en el almacén de parámetros de Systems Manager.

- Solo se admiten secretos que almacenan datos de texto. No se admiten los secretos que almacenan datos binarios.
- Utilice los puntos de conexión de VPC de interfaz para mejorar los controles de seguridad.
- La VPC que utiliza la tarea debe usar la resolución de DNS.
- En el caso de las tareas que utilizan el tipo de lanzamiento de EC2, debe utilizar la variable `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` de configuración del agente de Amazon ECS para utilizar esta característica. Puede añadirla al archivo `/etc/ecs/ecs.config`

durante la creación de la instancia de contenedor o puede añadirla a una instancia existente y, a continuación, reiniciar el agente de ECS. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS.

- En la definición de tareas se debe utilizar un rol de tareas con los permisos adicionales para el almacén de parámetros de Systems Manager. Para obtener más información, consulte Rol de IAM de tarea de Amazon ECS.

Cree el parámetro de

Puede utilizar la consola de Systems Manager para crear un parámetro del almacén de parámetros de Systems Manager para la información confidencial. Para obtener más información, consulte Creación de un parámetro de Systems Manager (consola) o Creación de un parámetro de Systems Manager (AWS CLI) en la Guía del usuario de AWS Systems Manager.

Actualice la aplicación para que recupere secretos del almacén de parámetros de Systems Manager mediante programación

Para recuperar los datos confidenciales almacenados en el parámetro de Parameter Store de Systems Manager, consulte Ejemplos de código de Systems Manager mediante SDK de AWS en la Biblioteca de ejemplos de código del SDK de AWS.

Transmisión de secretos de Secrets Manager a través de variables de entorno de Amazon ECS

Cuando introduce un secreto como una variable de entorno, puede especificar el contenido completo de un secreto, una clave JSON específica dentro de un secreto. Este proceso ayuda a controlar la información confidencial expuesta al contenedor. Para obtener más información sobre el control de versiones secreto, consulte ¿Qué hay en un secreto de Secrets Manager? en la Guía del usuario de AWS Secrets Manager.

Al utilizar una variable de entorno para introducir un secreto de Secrets Manager en un contenedor, se debe tener en cuenta lo siguiente.

- Los datos confidenciales se inyectan en el contenedor al iniciar el contenedor. Si el secreto se actualiza posteriormente o se rota, el contenedor no recibirá automáticamente el valor actualizado. Debe lanzar una nueva tarea o, si su tarea forma parte de un servicio, puede actualizar el servicio y utilizar la opción Force new deployment (Forzar nueva implementación) para forzar que el servicio lance una nueva tarea.

- Las aplicaciones que se ejecutan en el contenedor, los registros del contenedor y las herramientas de depuración tienen acceso a las variables de entorno.
- Para las tareas de Amazon ECS alojadas en AWS Fargate, tenga en cuenta lo siguiente:
  - Para introducir el contenido completo de un secreto como variable de entorno o en una configuración de registro, debe usar la versión `1.3.0` de la plataforma o una posterior. Para obtener más información, consulta Versiones de la plataforma Fargate para Amazon ECS.
  - Para introducir una versión o clave JSON específica de un secreto como variable de entorno o en una configuración de registro, debe usar la versión `1.4.0` o posterior (Linux) o `1.0.0` (Windows) de la plataforma. Para obtener más información, consulta Versiones de la plataforma Fargate para Amazon ECS.
- Para las tareas de Amazon ECS alojadas en EC2, se debe tener en cuenta lo siguiente:
  - Para introducir un secreto utilizando una versión o clave JSON específica de un secreto, la instancia de contenedor debe tener la versión `1.37.0` del agente de contenedor o una posterior. No obstante, recomendamos utilizar la versión del agente de contenedor más reciente. Para obtener información sobre la comprobación de la versión del agente y la actualización a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS.

    Para introducir el contenido completo de un secreto como variable de entorno o un secreto en una configuración de registro, la instancia de contenedor debe tener la versión `1.22.0` del agente de contenedor o una posterior.
- Utilice los puntos de conexión de VPC de interfaz para mejorar los controles de seguridad y conectarse a Secrets Manager a través de una subred privada. Debe crear los puntos de conexión de VPC de interfaz para Secrets Manager. Para obtener información sobre el punto de conexión de VPC, consulte Crear puntos de conexión de VPC en la Guía del usuario de AWS Secrets Manager. Para obtener más información sobre el uso de Secrets Manager y Amazon VPC, consulte How to connect to Secrets Manager service within a Amazon VPC.
- Para las tareas de Windows configuradas para utilizar el controlador de registros `awslogs`, debe también establecer la variable de entorno `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE` en la instancia del contenedor. Utilice la siguiente sintaxis:

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster <cluster name> -EnableTaskIAMRole -LoggingDrivers
 '["json-file","awslogs"]'
</powershell>
```

- En la definición de tareas se debe utilizar un rol de ejecución de tareas con los permisos adicionales para Secrets Manager. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

Cree el secreto de AWS Secrets Manager

Puede utilizar la consola de Secrets Manager para crear un secreto con su información confidencial. Para obtener más información, consulte Crear un secreto de AWS Secrets Manager en la Guía del usuario de AWS Secrets Manager.

Adición de la variable de entorno a la definición del contenedor

Dentro de la definición de contenedor, puede especificar lo siguiente:

- El objeto `secrets` que contiene el nombre de la variable de entorno que se va a establecer en el contenedor
- Nombre de recurso de Amazon (ARN) del secreto de Secrets Manager
- Parámetros adicionales que contienen información confidencial que se debe presentar al contenedor

En el ejemplo siguiente, se muestra la sintaxis completa que se debe especificar para el secreto de Secrets Manager.

```
arn:aws:secretsmanager:region:aws_account_id:secret:secret-name:json-key:version-
stage:version-id
```

En la siguiente sección se describen los parámetros adicionales. Estos parámetros son opcionales, pero si no los utiliza, debe incluir los dos puntos y coma `:` para utilizar los valores predeterminados. A continuación se ofrecen ejemplos para obtener más contexto.

`json-key`

Especifica el nombre de la clave en un par clave-valor con el valor que desea establecer como valor de variable de entorno. Solo se admiten valores en formato JSON. Si no especifica una clave JSON, se usa el contenido completo del secreto.

`version-stage`

Especifica la etiqueta de ensayo de la versión de un secreto que desea utilizar. Si se especifica una etiqueta de ensayo de versión, no se puede especificar un ID de versión. Si no se especifica

ninguna etapa de versión, el comportamiento predeterminado consiste en recuperar el secreto con la etiqueta de ensayo `AWSCURRENT`.

Las etiquetas de ensayo se utilizan para realizar un seguimiento de las distintas versiones de un secreto cuando se actualizan o rotan. Cada versión de un secreto tiene una o varias etiquetas de ensayo y un ID.

`version-id`

Especifica el identificador único de la versión del secreto que desea utilizar. Si se especifica un ID de versión, no se puede especificar una etiqueta de ensayo de versión. Si no se especifica ningún ID de versión, el comportamiento predeterminado consiste en recuperar el secreto con la etiqueta de ensayo `AWSCURRENT`.

Los ID de versión se utilizan para realizar un seguimiento de las distintas versiones de un secreto cuando se actualizan o rotan. Cada versión de un secreto tiene un ID. Para obtener más información, consulte Términos y conceptos clave para AWS Secrets Manager en la Guía del usuario de AWS Secrets Manager.

Definiciones de contenedor de ejemplo

En los siguientes ejemplos, se muestran las formas en las que se pueden referenciar secretos de Secrets Manager en las definiciones de contenedor.

Example hacer referencia a un secreto completo

A continuación, se incluye un fragmento de código de una definición de tarea que muestra el formato cuando se referencia el texto completo de un secreto de Secrets Manager.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
    }]
  }]
}
```

Para acceder al valor de este secreto desde el contenedor, deberá llamar a `$environment_variable_name`.

Example referencia de secretos completos

A continuación, se incluye un fragmento de código de una definición de tarea que muestra el formato cuando se referencia el texto completo de múltiples secretos de Secrets Manager.

```
{
  "containerDefinitions": [{
     "secrets": [
      {
        "name": "environment_variable_name1",
         "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
      },
      {
        "name": "environment_variable_name2",
         "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
abcdef"
      },
      {
        "name": "environment_variable_name3",
        "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
ABCDEF"
      }
    ]
  }]
}
```

Para acceder al valor de este secreto desde el contenedor, deberá llamar a `$environment_variable_name1`, `$environment_variable_name2` y `$environment_variable_name3`.

Example hacer referencia a una clave específica dentro de un secreto

A continuación se muestra un ejemplo de salida de un comando get-secret-value que muestra el contenido de un secreto junto con la etiqueta de ensayo de versión y el ID de versión asociados con ella.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "VersionId": "871d9eca-18aa-46a9-8785-981ddEXAMPLE",
    "SecretString": "{\"username1\":\"password1\",\"username2\":\"password2\",
\"username3\":\"password3\"}",
```

```
    "VersionStages": [
        "AWSCURRENT"
    ],
    "CreatedDate": 1581968848.921
}
```

Haga referencia a una clave específica de la salida anterior en una definición de contenedor especificando el nombre de clave al final del ARN.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::"
    }]
  }]
}
```

Example hacer referencia a una versión de secreto específica

A continuación se muestra una salida de ejemplo de un comando describe-secret que muestra el contenido sin cifrar de un secreto junto con los metadatos de todas las versiones del secreto.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "Description": "Example of a secret containing application authorization data.",
    "RotationEnabled": false,
    "LastChangedDate": 1581968848.926,
    "LastAccessedDate": 1581897600.0,
    "Tags": [],
    "VersionIdsToStages": {
        "871d9eca-18aa-46a9-8785-981ddEXAMPLE": [
            "AWSCURRENT"
        ],
        "9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE": [
            "AWSPREVIOUS"
        ]
    }
}
```

Haga referencia a una etiqueta de ensayo de versión específica de la salida anterior en una definición de contenedor especificando el nombre de clave al final del ARN.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf::AWSPREVIOUS:"
    }]
  }]
}
```

Haga referencia a un ID de versión específico de la salida anterior en una definición de contenedor especificando el nombre de clave al final del ARN.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:::9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE"
    }]
  }]
}
```

Example hacer referencia a una clave específica y una etiqueta de ensayo de versión de un secreto

A continuación se muestra cómo hacer referencia tanto a una clave específica dentro de un secreto como a una etiqueta de ensayo de versión específica.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1:AWSPREVIOUS:"
    }]
  }]
}
```

Para especificar una clave y un ID de versión específicos, utilice la sintaxis siguiente.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE"
    }]
  }]
}
```

Para obtener información sobre cómo crear una definición de tareas con el secreto especificado en una variable de entorno, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

Transmisión de parámetros de Systems Manager a través de variables de entorno de Amazon ECS

Amazon ECS le permite ingresar información confidencial en los contenedores al almacenarla en parámetros de Almacén de parámetros de AWS Systems Manager y, a continuación, hacer referencia a ella en la definición de los contenedores.

Tenga en cuenta lo siguiente cuando utilice una variable de entorno para ingresar un secreto de Systems Manager en un contenedor.

- Los datos confidenciales se inyectan en el contenedor al iniciar el contenedor. Si el secreto se actualiza posteriormente o se rota, el contenedor no recibirá automáticamente el valor actualizado. Debe lanzar una nueva tarea o, si su tarea forma parte de un servicio, puede actualizar el servicio y utilizar la opción Force new deployment (Forzar nueva implementación) para forzar que el servicio lance una nueva tarea.
- Para las tareas de Amazon ECS alojadas en AWS Fargate, se debe tener en cuenta lo siguiente:
  - Para introducir el contenido completo de un secreto como variable de entorno o en una configuración de registro, debe usar la versión `1.3.0` de la plataforma o una posterior. Para obtener más información, consulta Versiones de la plataforma Fargate para Amazon ECS.
  - Para introducir una versión o clave JSON específica de un secreto como variable de entorno o en una configuración de registro, debe usar la versión `1.4.0` o posterior (Linux) o `1.0.0` (Windows) de la plataforma. Para obtener más información, consulta Versiones de la plataforma Fargate para Amazon ECS.
- Para las tareas de Amazon ECS alojadas en EC2, se debe tener en cuenta lo siguiente:

- Para introducir un secreto utilizando una versión o clave JSON específica de un secreto, la instancia de contenedor debe tener la versión `1.37.0` del agente de contenedor o una posterior. No obstante, recomendamos utilizar la versión del agente de contenedor más reciente. Para obtener información sobre la comprobación de la versión del agente y la actualización a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS.

  Para introducir el contenido completo de un secreto como variable de entorno o un secreto en una configuración de registro, la instancia de contenedor debe tener la versión `1.22.0` del agente de contenedor o una posterior.

- Utilice los puntos de conexión de VPC de interfaz para mejorar los controles de seguridad. Debe crear los puntos de conexión de VPC de interfaz para Systems Manager. Para obtener información sobre cómo crear un punto de conexión de VPC, consulte Mejora de la seguridad de las instancias de EC2 mediante puntos de conexión de VPC para Systems Manager en la Guía del usuario de AWS Systems Manager.
- En la definición de tareas se debe utilizar un rol de ejecución de tareas con los permisos adicionales para el almacén de parámetros de Systems Manager. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.
- Para las tareas de Windows configuradas para utilizar el controlador de registros `awslogs`, debe también establecer la variable de entorno `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE` en la instancia del contenedor. Utilice la siguiente sintaxis:

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster <cluster name> -EnableTaskIAMRole -LoggingDrivers
 '["json-file","awslogs"]'
</powershell>
```

Creación del parámetro de Systems Manager

Puede utilizar la consola de Systems Manager para crear un parámetro del almacén de parámetros de Systems Manager para la información confidencial. Para obtener más información, consulte Creación de un parámetro de Systems Manager (consola) o Creación de un parámetro de Systems Manager (AWS CLI) en la Guía del usuario de AWS Systems Manager.

Adición de la variable de entorno a la definición del contenedor

Dentro de la definición de contenedor en la definición de tareas, especifique `secrets` con el nombre de la variable de entorno que se va a establecer en el contenedor y el ARN completo del parámetro del almacén de parámetros de Systems Manager que contiene la información confidencial que se va a presentar al contenedor. Para obtener más información, consulte `secrets`.

A continuación, se incluye un fragmento de código de una definición de tarea que muestra el formato cuando se hace referencia a un parámetro del Parameter Store de Systems Manager. Si el parámetro del Parameter Store de Systems Manager está en la misma región que la tarea que se va a lanzar, se puede utilizar el ARN completo o el nombre del parámetro. Si el parámetro existe en una región distinta, debe especificar el ARN completo.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }]
  }]
}
```

Para obtener información sobre cómo crear una definición de tareas con el secreto especificado en una variable de entorno, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

Actualice la aplicación para que recupere secretos del almacén de parámetros de Systems Manager mediante programación

Para recuperar los datos confidenciales almacenados en el parámetro de Parameter Store de Systems Manager, consulte Ejemplos de código de Systems Manager mediante SDK de AWS en la Biblioteca de ejemplos de código del SDK de AWS.

Transmisión de secretos para la configuración de registro de Amazon ECS

Puede utilizar el parámetro `secretOptions` en `logConfiguration` para transferir los datos confidenciales que se utilizan para los registros.

Puede almacenar el secreto en Secrets Manager o Systems Manager.

Uso de Secrets Manager

En la definición de contenedor, al especificar una `logConfiguration`, puede especificar `secretOptions` con el nombre de la opción del controlador de registros para definir el contenedor y el ARN completo del secreto de Secrets Manager que contiene la información confidencial que se va a presentar al contenedor. Para obtener más información acerca de cómo crear secretos, consulte Creación de un AWS Secrets Manager.

A continuación, se incluye un fragmento de código de una definición de tarea que muestra el formato cuando se referencia un secreto de Secrets Manager.

```
{
  "containerDefinitions": [{
    "logConfiguration": [{
      "logDriver": "splunk",
      "options": {
        "splunk-url": "https://your_splunk_instance:8088"
      },
      "secretOptions": [{
        "name": "splunk-token",
        "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
      }]
    }]
  }]
}
```

Adición de la variable de entorno a la definición del contenedor

En la definición de contenedor, especifique `secrets` con el nombre de la variable de entorno que se va a establecer en el contenedor y el ARN completo del parámetro del Parameter Store de Systems Manager que contiene la información confidencial que se va a presentar al contenedor. Para obtener más información, consulte `secrets`.

A continuación, se incluye un fragmento de código de una definición de tarea que muestra el formato cuando se hace referencia a un parámetro del Parameter Store de Systems Manager. Si el parámetro del Parameter Store de Systems Manager está en la misma región que la tarea que se va a lanzar, se puede utilizar el ARN completo o el nombre del parámetro. Si el parámetro existe en una región distinta, debe especificar el ARN completo.

```
{
```

```
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }]
  }]
}
```

Para obtener información sobre cómo crear una definición de tareas con el secreto especificado en una variable de entorno, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

Uso de Systems Manager

Puede introducir información confidencial en una configuración de registros. En la definición de contenedor, al especificar una `logConfiguration`, puede especificar `secretOptions` con el nombre de la opción del controlador de registros que va a definir en el contenedor y el ARN completo del parámetro del Parameter Store de Systems Manager que contiene la información confidencial que se va a presentar al contenedor.

> **Important**
>
> Si el parámetro del Parameter Store de Systems Manager está en la misma región que la tarea que se va a lanzar, se puede utilizar el ARN completo o el nombre del parámetro. Si el parámetro existe en una región distinta, debe especificar el ARN completo.

A continuación, se incluye un fragmento de código de una definición de tarea que muestra el formato cuando se hace referencia a un parámetro del Parameter Store de Systems Manager.

```
{
  "containerDefinitions": [{
    "logConfiguration": [{
      "logDriver": "fluentd",
      "options": {
        "tag": "fluentd demo"
      },
      "secretOptions": [{
        "name": "fluentd-address",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter:/parameter_name"
```

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

Especificación de información confidencial mediante secretos de Secrets Manager en Amazon ECS

Amazon ECS permite introducir información confidencial en los contenedores, la almacena en secretos de AWS Secrets Manager y, a continuación, la referencia en la definición de contenedor. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.

Obtenga información sobre cómo crear un secreto de Secrets Manager, hacer referencia al secreto en una definición de tareas de Amazon ECS y, luego, comprobar que funciona al consultar la variable de entorno dentro de un contenedor que muestra el contenido del secreto.

Requisitos previos

En este tutorial se supone que los siguientes requisitos previos se han completado:

- Se han completado los pasos que se indican en Configuración para utilizar Amazon ECS.
- El usuario dispone de los permisos de IAM requeridos para crear los recursos de Secrets Manager y Amazon ECS.

Paso 1: Crear un secreto de Secrets Manager

Puede utilizar la consola de Secrets Manager para crear un secreto con su información confidencial. En este tutorial se creará un secreto básico para almacenar un nombre de usuario y una contraseña para referencia futura en un contenedor. Para obtener más información, consulte Creación de un secreto de AWS Secrets Manager en la Guía del usuario de AWS Secrets Manager.

Los pares clave/valor que se almacenarán en este secreto (key/value pairs to be stored in this secret) son el valor de la variable de entorno en el contenedor al final del tutorial.

Guarde el Secret ARN (ARN del secreto) para hacer referencia en su política de IAM de ejecución de tareas y la definición de tarea en pasos posteriores.

Paso 2: agregue los permisos secretos al rol de ejecución de tareas.

Para que Amazon ECS recupere la información confidencial de su secreto de Secrets Manager, debe contar con los permisos secretos para el rol de ejecución de tareas. Para obtener más información, consulte Permisos de Secrets Manager o Systems Manager.

Paso 3: Cree una definición de tarea

Puede utilizar la consola de Amazon ECS para crear una definición de tareas que haga referencia a un secreto de Secrets Manager.

Para crear una definición de tarea que especifique un secreto

Utilice la consola de IAM para actualizar el rol de ejecución de tareas con los permisos requeridos.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, elija Task Definitions (Definiciones de tareas).
3. Elija Create new task definition (Crear nueva definición de tarea) y Create new task definition with JSON (Crear nueva definición de tarea con JSON).
4. En el cuadro del editor JSON, ingrese el siguiente texto JSON de definición de tareas y asegúrese de especificar el ARN completo del secreto de Secrets Manager que creó en el paso 1 y el rol de ejecución de tareas que actualizó en el paso 2. Seleccione Save.
5.

```
{
    "executionRoleArn": "arn:aws:iam::aws_account_id:role/ecsTaskExecutionRole",
    "containerDefinitions": [
        {
            "entryPoint": [
                "sh",
                "-c"
            ],
            "portMappings": [
                {
                    "hostPort": 80,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
```

```
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
            ],
            "cpu": 10,
            "secrets": [
                {
                    "valueFrom":
 "arn:aws:secretsmanager:region:aws_account_id:secret:username_value",
                    "name": "username_value"
                }
            ],
            "memory": 300,
            "image": "public.ecr.aws/docker/library/httpd:2.4",
            "essential": true,
            "name": "ecs-secrets-container"
        }
    ],
    "family": "ecs-secrets-tutorial"
}
```

6. Seleccione Crear.

Paso 4: cree un clúster

Puede utilizar la consola de Amazon ECS para crear un clúster que contenga una instancia de contenedor en la que se va a ejecutar la tarea. Si tiene un clúster existente con al menos una instancia de contenedor registrada en ella, con los recursos disponibles para ejecutar una instancia de la definición de tarea creada para este tutorial, puede pasar al siguiente paso.

Para este tutorial, vamos a crear un clúster con una instancia de contenedor `t2.micro` mediante la AMI de Amazon Linux 2 optimizada para Amazon ECS.

Para obtener información sobre cómo crear un clúster para el tipo de lanzamiento de EC2, consulte the section called “Creación de un clúster para el tipo de lanzamiento de Amazon EC2”.

Paso 5: ejecute una tarea

Puede utilizar la consola de Amazon ECS para ejecutar una tarea mediante la definición de tareas creada. En este tutorial, ejecutaremos una tarea mediante el tipo de lanzamiento de EC2, utilizando el clúster que creamos en el paso anterior.

Para obtener información sobre cómo ejecutar una tarea, consulte the section called “Ejecución de una aplicación como tarea”.

Paso 6: Verificar

Puede verificar que todos los pasos se han completado correctamente y que la variable de entorno se ha creado correctamente en el contenedor con los pasos que se describen a continuación.

Para verificar que se ha creado la variable de entorno

1. Busque la dirección IP o DNS pública para su instancia de contenedor.
   a. Abra la consola en https://console.aws.amazon.com/ecs/v2.
   b. En el panel de navegación, elija Clústeres y, a continuación, elija el clúster que creó.
   c. Elija Infraestructura y, a continuación, elija la instancia de contenedor.
   d. Registre la IP pública o el DNS público para su instancia.
2. Si está utilizando un equipo con macOS o Linux, conéctese a la instancia con el siguiente comando, sustituyendo la ruta por su clave privada y la dirección pública para su instancia:

   ```
   $ ssh -i /path/to/my-key-pair.pem ec2-user@ec2-198-51-100-1.compute-1.amazonaws.com
   ```

   Para obtener más información acerca del uso de una computadora con Windows, consulte Conexión a la instancia de Linux con PuTTY en la Guía del usuario de Amazon EC2.

   > **Important**
   > Para obtener más información acerca de los problemas que pueden surgir al conectarse a la instancia, consulte Solucione el problema de conectar la instancia en la Guía del usuario de Amazon EC2.

3. Obtenga la lista de los contenedores que se ejecutan en la instancia. Anote el ID del contenedor `ecs-secrets-tutorial`.

   ```
   docker ps
   ```

4. Conéctese al contenedor `ecs-secrets-tutorial` con el ID de contenedor desde la salida del paso anterior.

```
docker exec -it container_ID /bin/bash
```

5. Utilice el comando `echo` para imprimir el valor de la variable del entorno.

```
echo $username_value
```

Si el tutorial no se ha realizado correctamente, debería ver el siguiente resultado:

```
password_value
```

> **Note**
>
> Si lo prefiere, puede mostrar todas las variables de entorno en su contenedor con el comando `env` (o `printenv`).

Paso 7: limpiar

Cuando termine este tutorial, debe limpiar los recursos asociados para evitar incurrir en cargos generados por recursos sin utilizar.

Para limpiar los recursos.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. En la página Clusters (Clústeres), elija el clúster.
4. Elija Delete cluster.
5. En el cuadro de confirmación, ingrese Eliminar ***cluster-name*** y, a continuación, elija Eliminar.
6. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
7. Seleccione Roles en el panel de navegación.
8. En la lista de roles, busque `ecsTaskExecutionRole` y selecciónelo.
9. Elija Permisos y, a continuación, elija la X junto a ECSSecretsTutorial. Elija Eliminar.
10. Abra la consola de Secrets Manager enhttps://console.aws.amazon.com/secretsmanager/.
11. Seleccione el secreto username_value creado y elija Actions (Acciones), Delete secret (Eliminar secreto).

Parámetros en la definición de tareas de Amazon ECS para el tipo de lanzamiento de Fargate

Las definiciones de tareas se dividen en partes independientes: la familia de tareas, el rol de tarea de AWS Identity and Access Management (IAM), el modo de red, las definiciones de contenedor, los volúmenes y los tipos de lanzamiento. En una definición de tareas, son necesarias las definiciones de familia y contenedor. En cambio, el rol de tarea, el modo de red, los volúmenes y el tipo de lanzamiento son opcionales.

Puede utilizar estos parámetros en un archivo JSON para configurar la definición de tarea.

A continuación, se muestran las descripciones de cada uno de los parámetros de definición de tareas para el tipo de lanzamiento de Fargate.

Familia

`family`

Tipo: cadena

Obligatorio: sí

Cuando se registra una definición de tarea, se le da una familia, que es similar a un nombre para varias versiones de la definición de tarea, especificado con un número de revisión. A la primera definición de tareas que se registra en una familia particular se le da una revisión de 1 y a cualquier definición de tarea registrada después se le da un número de revisión secuencial.

Tipos de lanzamiento

Cuando se registre una definición de tareas, puede especificar un tipo de lanzamiento con respecto al cual Amazon ECS debe validar la definición de tareas. Si la definición de tareas no se valida con respecto a las compatibilidades especificadas, se devuelve una excepción de cliente. Para obtener más información, consulte Tipos de lanzamiento de Amazon ECS.

El parámetro a continuación está permitido en una definición de tareas.

`requiresCompatibilities`

Tipo: matriz de cadenas

Obligatorio: no

Valores válidos: `FARGATE`

Los tipos de lanzamiento con respecto a los cuales se validó la definición de tareas. Esto inicia una comprobación para garantizar que todos los parámetros utilizados en la definición de tareas cumplan los requisitos del tipo de lanzamiento.

Rol de la tarea

`taskRoleArn`

Tipo: cadena

Requerido: no

Cuando se registra una definición de tareas, se puede proporcionar un rol de tarea para un rol de IAM que permita a los contenedores del permiso de tareas llamar en su nombre a las API de AWS especificadas en sus políticas asociadas. Para obtener más información, consulte Rol de IAM de tarea de Amazon ECS.

Rol de ejecución de tareas

`executionRoleArn`

Tipo: cadena

Obligatorio: condicional

El nombre de recurso de Amazon (ARN) del rol de ejecución de tarea que concede permiso al agente de contenedor de Amazon ECS para realizar llamadas a la API de AWS en su nombre.

> **Note**
>
> El rol de IAM de ejecución de tareas es necesario en función de los requisitos de la tarea. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

Modo de red

`networkMode`

Tipo: cadena

Obligatorio: sí

El modo de red de Docker que utilizar para los contenedores en la tarea. Para las tareas de Amazon ECS alojadas en Fargate, se requiere el modo de red `awsvpc`.

Plataforma de tiempo de ejecución

`operatingSystemFamily`

Tipo: cadena

Obligatorio: condicional

Predeterminado: LINUX

Este parámetro es necesario para las tareas de Amazon ECS que están alojadas en Fargate.

Cuando se registra una definición de tarea, especifica la familia de sistemas operativos.

Los valores válidos son `LINUX`, `WINDOWS_SERVER_2025_FULL`, `WINDOWS_SERVER_2025_CORE`, `WINDOWS_SERVER_2022_FULL`, `WINDOWS_SERVER_2022_CORE`, `WINDOWS_SERVER_2019_FULL` y `WINDOWS_SERVER_2019_CORE`.

Todas las definiciones de tareas que se utilizan en un servicio deben tener el mismo valor para este parámetro.

Cuando una definición de tarea forma parte de un servicio, este valor debe coincidir con el valor `platformFamily` de servicio.

`cpuArchitecture`

Tipo: cadena

Obligatorio: condicional

Predeterminado: X86_64

Si el parámetro se deja como `null`, el valor predeterminado se asigna automáticamente al iniciar una tarea alojada en Fargate.

Cuando se registra una definición de tarea, especifica la arquitectura de CPU. Los valores válidos son `X86_64` y `ARM64`.

Todas las definiciones de tareas que se utilizan en un servicio deben tener el mismo valor para este parámetro.

Si tiene tareas de Linux, puede establecer el valor en `ARM64`. Para obtener más información, consulte the section called "Definiciones de tareas para cargas de trabajo de ARM de 64 bits".

Tamaño de tarea

Cuando se registra una definición de tareas, puede especificar la cantidad total de CPU y memoria usada para la tarea. Es distinto de los valores `cpu` y `memory` en el nivel de definición de contenedor. Para las tareas que están alojadas en Fargate (tanto en Linux como en Windows), estos campos son obligatorios y se admiten valores específicos para `cpu` y `memory`.

El siguiente parámetro está permitido en una definición de tarea:

`cpu`

Tipo: cadena

Obligatorio: sí

> **Note**
>
> Los parámetros de CPU y memoria a nivel de tarea son necesarios y se utilizan para determinar el tipo y el tamaño de la instancia en la que se ejecutan las tareas. En el caso de las tareas de Windows, estos valores no se aplican en tiempo de ejecución, ya que Windows no tiene un mecanismo nativo que pueda imponer fácilmente límites de recursos colectivos a un grupo de contenedores. Si desea imponer límites de recursos, le recomendamos que utilice los recursos a nivel de contenedor para los contenedores de Windows.

El límite máximo de unidades de CPU que presentar para la tarea. Puede indicar los valores de la CPU en el archivo JSON como una cadena en unidades de CPU o CPU virtuales (vCPU). Por ejemplo, puede especificar un valor de CPU `1024` en unidades de CPU o `1 vCPU` en vCPU. Cuando se registra la definición de tarea, el valor de una CPU virtual se convierte en un entero que indica las unidades de CPU.

Este campo es obligatorio y debe utilizar uno de los siguientes valores, lo que determina el intervalo de valores válidos para el parámetro `memory`. En la siguiente tabla se muestran las combinaciones válidas de CPU y memoria de nivel de tarea.

Valor de CPU	Valor de memoria	Sistemas operativos admitidos por AWS Fargate
256 (0,25 vCPU)	512 MiB, 1 GB, 2 GB	Linux
512 (0,5 vCPU)	1 GB, 2 GB, 3 GB, 4 GB	Linux
1024 (1 vCPU)	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB	Linux, Windows
2048 (2 vCPU)	Entre 4 GB y 16 GB en incrementos de 1 GB	Linux, Windows
4096 (4 vCPU)	Entre 8 GB y 30 GB en incrementos de 1 GB	Linux, Windows
8192 (8 vCPU) **Note** Esta opción requiere una plataforma Linux `1.4.0` o posterior.	Entre 16 GB y 60 GB en incrementos de 4 GB	Linux
16 384 (16 vCPU)	Entre 32 GB y 120 GB en incrementos de 8 GB	Linux

Valor de CPU	Valor de memoria	Sistemas operativos admitidos por AWS Fargate
Note Esta opción requiere una plataforma Linux `1.4.0` o posterior.		

`memory`

Tipo: cadena

Obligatorio: sí

> **Note**
>
> Los parámetros de CPU y memoria a nivel de tarea son necesarios y se utilizan para determinar el tipo y el tamaño de la instancia en la que se ejecutan las tareas. En el caso de las tareas de Windows, estos valores no se aplican en tiempo de ejecución, ya que Windows no tiene un mecanismo nativo que pueda imponer fácilmente límites de recursos colectivos a un grupo de contenedores. Si desea imponer límites de recursos, le recomendamos que utilice los recursos a nivel de contenedor para los contenedores de Windows.

El límite máximo de memoria para presentar a la tarea. Puede especificar los valores de memoria en la definición de tareas como una cadena en mebibytes (MiB) o gigabytes (GB). Por ejemplo, puede especificar un valor de memoria `3072` en MiB o `3 GB` en GB. Cuando se registra la definición de tarea, el valor de GB se convierte en un entero que indica el número de MiB.

Este campo es obligatorio y debe utilizar uno de los siguientes valores, lo que determina el intervalo de valores válidos para el parámetro `cpu`:

Valor de memoria (en MiB, con un valor equivalente aproximado en GB)	Valor de CPU	Sistemas operativos compatibles con Fargate
512 (0,5 GB), 1024 (1 GB), 2048 (2 GB)	256 (0,25 vCPU)	Linux
1024 (1 GB), 2048 (2 GB), 3072 (3 GB), 4096 (4 GB)	512 (0,5 vCPU)	Linux
2048 (2 GB), 3072 (3 GB), 4096 (4GB), 5120 (5 GB), 6144 (6 GB), 7168 (7 GB), 8192 (8 GB)	1024 (1 vCPU)	Linux, Windows
Entre 4096 (4 GB) y 16384 (16 GB) en incrementos de 1024 (1 GB)	2048 (2 vCPU)	Linux, Windows
Entre 8192 (8 GB) y 30720 (30 GB) en incrementos de 1024 (1 GB)	4096 (4 vCPU)	Linux, Windows
Entre 16 GB y 60 GB en incrementos de 4 GB **Note** Esta opción requiere una plataforma Linux `1.4.0` o posterior.	8192 (8 vCPU)	Linux
Entre 32 GB y 120 GB en incrementos de 8 GB	16 384 (16 vCPU)	Linux

Valor de memoria (en MiB, con un valor equivalente aproximado en GB)	Valor de CPU	Sistemas operativos compatibles con Fargate
Note Esta opción requiere una plataforma Linux `1.4.0` o posterior.		

Definiciones de contenedores

Al registrar una definición de tarea, debe especificar una lista de definiciones de contenedor que se transfieren al daemon de Docker en una instancia de contenedor. Los siguientes parámetros están permitidos en una definición de contenedor.

Temas

- Parámetros de definición de contenedor estándar
- Parámetros de definición de contenedor avanzados
- Otros parámetros de definición de contenedor

Parámetros de definición de contenedor estándar

Los siguientes parámetros de definición de tarea son obligatorios o utilizados en la mayoría de definiciones de contenedor.

Temas

- Nombre
- Imagen
- Memoria
- Mapeos de puertos
- Credenciales de repositorio privado

Nombre

`name`

Tipo: cadena

Obligatorio: sí

El nombre de un contenedor. Se admiten hasta 255 letras (mayúsculas y minúsculas), números, guiones y caracteres de subrayado. Si está vinculando varios contenedores en una definición de tareas, el `name` de un contenedor se puede introducir en los `links` de otro contenedor. Esto sirve para conectar los contenedores.

Imagen

`image`

Tipo: cadena

Obligatorio: sí

La imagen que se utiliza para iniciar un contenedor. Esta cadena se transfiere directamente al daemon de Docker. De manera predeterminada, las imágenes del registro de Docker Hub están disponibles. También puede especificar otros repositorios con *`repository-url`*`/`*`image`*`:`*`tag`* o *`repository-url`*`/`*`image`*`@`*`digest`*. Se permiten hasta 255 letras (mayúsculas y minúsculas), números, guiones, caracteres de subrayado, comas, puntos, barras diagonales y signos numéricos. Este parámetro se corresponde con `Image` en el comando create-container de docker y el parámetro `IMAGE` del comando de docker run.

- Cuando se inicia una tarea nueva, el agente de contenedor de Amazon ECS extrae la última versión de la imagen y la etiqueta especificadas para que las utilice el contenedor. Sin embargo, las actualizaciones realizadas posteriormente en un repositorio de imágenes no se propagan a las tareas en ejecución.
- Cuando no se especifica una etiqueta ni un resumen en la ruta de imagen en la definición de la tarea, el agente de contenedor de Amazon ECS extrae la última versión de la imagen especificada.
- Sin embargo, las actualizaciones realizadas posteriormente en un repositorio de imágenes no se propagan a las tareas en ejecución.
- Se admiten las imágenes de registros privados. Para obtener más información, consulte Uso de imágenes de contenedor que no sean de AWS en Amazon ECS.

- Para especificar imágenes de los repositorios de Amazon ECR, se puede utilizar la convención de nomenclatura completa `registry/repository:tag` o `registry/repository@digest` (por ejemplo, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`* o *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`@`*`sha256:94afd1f2e64d908bc90dbca0035a5b567EXAMPLE`*).
- Las imágenes de los repositorios oficiales de Docker Hub utilizan un solo nombre (por ejemplo, `ubuntu` o `mongo`).
- Las imágenes de otros repositorios de Docker Hub se clasifican con un nombre de organización (por ejemplo, `amazon/amazon-ecs-agent`).
- Las imágenes de otros repositorios online se cualifican más con un nombre de dominio (por ejemplo, `quay.io/assemblyline/ubuntu`).

`versionConsistency`

Tipo: cadena

Valores válidos: `enabled|disabled`

Obligatorio: no

Especifica si Amazon ECS convertirá la etiqueta de imagen de contenedores proporcionada en la definición de contenedores en un resumen de imágenes. De manera predeterminada, el comportamiento es `enabled`. Si establece `disabled` como valor de un contenedor, Amazon ECS no convertirá la etiqueta de imagen del contenedor en un resumen y utilizará el URI de imagen original especificado en la definición del contenedor para la implementación. Para obtener más información acerca de la conversión de imagen de contenedor, consulte Resolución de imagen de contenedor.

Memoria

`memory`

Tipo: entero

Obligatorio: no

La cantidad (en MiB) de memoria que se presenta al contenedor. Si su contenedor intenta superar la memoria especificada aquí, el contenedor se cancela. La cantidad total de memoria reservada para todos los contenedores dentro de una tarea debe ser menor que el valor `memory`

de la tarea, si se especifica. Este parámetro se corresponde con `Memory` en el comando create-container de docker y la opción `--memory` se corresponde con docker run.

El daemon de Docker 20.10.0 o posterior reserva un mínimo de 6 MiB de memoria para un contenedor. Por lo tanto, no debe especificar menos de 6 MiB de memoria para los contenedores.

El daemon de Docker 19.03.13-ce o anterior reserva un mínimo de 4 MiB de memoria para un contenedor. Por lo tanto, no debe especificar menos de 4 MiB de memoria para los contenedores.

> **Note**
>
> Si intenta maximizar la utilización de los recursos proporcionando a las tareas la mayor cantidad de memoria posible para un tipo de instancia determinado, consulte Reserva de la memoria de instancias de contenedor de Linux de Amazon ECS.

`memoryReservation`

Tipo: entero

Obligatorio: no

El límite flexible (en MiB) de memoria que reservar para el contenedor. Cuando la memoria del sistema está en conflicto, Docker intenta mantener la memoria del contenedor en este límite flexible. Sin embargo, el contenedor puede utilizar más memoria cuando sea necesario. El contenedor puede utilizar hasta el límite invariable especificado con el parámetro de `memory` (si procede) o toda la memoria disponible en la instancia de contenedor, lo que ocurra primero. Este parámetro se corresponde con `MemoryReservation` en el comando create-container de docker y la opción `--memory-reservation` se corresponde con docker run.

Si no se especifica un valor de memoria de nivel de tarea, debe especificar un entero distinto de cero para `memory` o `memoryReservation`, o para ambos, en una definición de contenedor. Si especifica ambos, `memory` debe ser mayor que `memoryReservation`. Si especifica `memoryReservation`, el valor se resta de los recursos de memoria disponibles para la instancia de contenedor en la que se coloca el contenedor. De lo contrario, se utiliza el valor de `memory`.

Por ejemplo, supongamos que el contenedor utiliza normalmente 128 MiB de memoria, pero con ráfagas ocasionales de hasta 256 MiB durante períodos de tiempo breves. Puede establecer una `memoryReservation` de 128 MiB y un límite invariable de `memory` de 300 MiB. Esta configuración permite al contenedor reservar solo 128 MiB de memoria de los recursos restantes

en la instancia de contenedor. Al mismo tiempo, esta configuración también permite que el contenedor consuma más recursos de memoria en caso de ser necesario.

> **Note**
>
> Este parámetro no es compatible con los contenedores de Windows.

El daemon de Docker 20.10.0 o posterior reserva un mínimo de 6 MiB de memoria para un contenedor. Por lo tanto, no debe especificar menos de 6 MiB de memoria para los contenedores.

El daemon de Docker 19.03.13-ce o anterior reserva un mínimo de 4 MiB de memoria para un contenedor. Por lo tanto, no debe especificar menos de 4 MiB de memoria para los contenedores.

> **Note**
>
> Si intenta maximizar la utilización de los recursos proporcionando a las tareas la mayor cantidad de memoria posible para un tipo de instancia determinado, consulte Reserva de la memoria de instancias de contenedor de Linux de Amazon ECS.

Mapeos de puertos

`portMappings`

Tipo: matriz de objetos

Obligatorio: no

Las asignaciones de puertos exponen los puertos de red del contenedor al mundo exterior, lo que permite a los clientes acceder a su aplicación. También se utilizan para la comunicación entre contenedores dentro de la misma tarea.

Para las definiciones de tareas que utilizan el modo de red `awsvpc`, solo especifique `containerPort`. El `hostPort` siempre se ignora y el puerto del contenedor se asigna automáticamente a un puerto aleatorio con un número alto del host.

Las asignaciones de puertos en Windows utilizan la dirección de puerto de enlace `NetNAT` en lugar de `localhost`. No existe ningún bucle invertido para el mapeo de puertos en Windows, por lo que no puede acceder a un puerto mapeado del contenedor desde el propio host.

La mayoría de los campos de este parámetro (incluidos `containerPort`, `hostPort`, `protocol`) se corresponden con `PortBindings` en el comando create-container de docker y con la opción `--publish` para docker run. Si el modo de red de una definición de tareas se establece en `host`, los puertos de host deben estar sin definir o deben corresponder al puerto de contenedor en la asignación de puerto.

> **Note**
>
> Después de que una tarea alcanza el estado `RUNNING`, las asignaciones manuales y automáticas de puertos de contenedor y de host se pueden ver en las siguientes ubicaciones:
>
> - Consola: la sección Network Bindings (Conexiones de red) de la descripción de un contenedor para una tarea seleccionada.
> - AWS CLI: la sección `networkBindings` de la salida del comando describe-tasks.
> - API: la respuesta de `DescribeTasks`.
> - Metadatos: el punto de conexión de metadatos de la tarea.

`appProtocol`

Tipo: cadena

Requerido: no

El protocolo de aplicación que se utiliza para la asignación de puertos. Este parámetro solo se aplica a Service Connect. Recomendamos que defina este parámetro para que sea coherente con el protocolo que utiliza la aplicación. Si se establece este parámetro, Amazon ECS agrega la administración de la conexión específica del protocolo al proxy de Service Connect. Si se establece este parámetro, Amazon ECS agrega telemetría específica del protocolo en la consola de Amazon ECS y CloudWatch.

Si no establece un valor para este parámetro, se utilizará TCP. Sin embargo, Amazon ECS no agrega telemetría específica del protocolo para TCP.

Para obtener más información, consulte the section called “ Service Connect”.

Valores de protocolo válidos: `"http" | "http2" | "grpc"`

`containerPort`

Tipo: entero

Obligatorio: sí, si se utilizan `portMappings`.

El número de puerto en el contenedor que está vinculado al puerto de host especificado por el usuario o asignado automáticamente.

Para las tareas que utilizan el modo de red `awsvpc`, debe utilizar `containerPort` para especificar los puertos expuestos.

Para los contenedores de Windows en Fargate, no se puede utilizar el puerto 3150 para el `containerPort`. Esto se debe a que está reservado.

`containerPortRange`

Tipo: cadena

Requerido: no

El rango de números de puerto en el contenedor que está vinculado al rango de puertos de host asignado de manera dinámica.

Solo puede configurar este parámetro mediante la API `register-task-definition`. La opción está disponible en el parámetro `portMappings`. Para obtener más información, consulte register-task-definition en la Referencia de AWS Command Line Interface.

Las siguientes reglas se aplican al especificar un `containerPortRange`:

- Debe usar el modo de red `awsvpc`.
- Este parámetro está disponible tanto para sistemas operativos Windows como Linux.
- La instancia de contenedor debe tener al menos la versión 1.67.0 del agente de contenedor y al menos la versión 1.67.0-1 del paquete `ecs-init`
- Puede especificar 100 rangos de puertos como máximo por cada contenedor.
- No especifica un `hostPortRange`. El valor de `hostPortRange` se establece de la siguiente manera:
  - Para los contenedores de una tarea con el modo de red `awsvpc`, `hostPort` se establece en el mismo valor que `containerPort`. Se trata de una estrategia de asignación estática.
- Los valores válidos de `containerPortRange` se encuentran entre 1 y 65 535.

- Un puerto solo puede incluirse en una asignación de puertos por cada contenedor.
- No puede especificar rangos de puertos superpuestos.
- El primer puerto del rango debe ser menor que el último puerto del rango.
- Docker recomienda desactivar el docker-proxy en el archivo de configuración del daemon de Docker cuando haya una gran cantidad de puertos.

  Para más información, consulte Issue #11185 en GitHub.

  Para obtener información sobre cómo desactivar el docker-proxy en el archivo de configuración del daemon de Docker, consulte Daemon de Docker en la Guía para desarrolladores de Amazon ECS.

Puede llamar a DescribeTasks para ver el valor de `hostPortRange`, es decir, los puertos del host que están vinculados a los puertos del contenedor.

Los rangos de puertos no se incluyen en los eventos de tareas de Amazon ECS que se envían a EventBridge. Para obtener más información, consulte the section called “Automaticzación de las respuestas a los errores de Amazon ECS mediante EventBridge”.

`hostPortRange`

Tipo: string

Requerido: no

El rango de números de puerto del host que se usa con el enlace de red. Docker lo asigna y el agente de Amazon ECS lo entrega.

`hostPort`

Tipo: entero

Obligatorio: no

El número de puerto en la instancia de contenedor que reservar para el contenedor.

El valor de `hostPort` puede dejarse en blanco o debe ser el mismo valor que `containerPort`.

La versión 1.6.0 de Docker y posteriores para el rango de puertos efímeros predeterminado se puede ver en la instancia en `/proc/sys/net/ipv4/ip_local_port_range`. Si este parámetro de kernel no está disponible, se usa el intervalo de puertos efímeros

predeterminado, de `49153-65535`. No intente especificar un puerto de host en el rango de puertos efímeros. Esto se debe a que están reservados para la asignación automática. En general, los puertos bajo `32768` están fuera del rango de puertos efímeros.

Los puertos reservados predeterminados son el `22` para SSH; los puertos de Docker `2375` y `2376` y los puertos `51678-51680` del agente de contenedor de Amazon ECS. Cualquier puerto de host especificado por el usuario con anterioridad para una tarea en ejecución también se reserva mientras la tarea está en ejecución. Cuando se detiene una tarea, se libera el puerto del host. Los puertos reservados actuales se muestran en los `remainingResources` de la salida de describe-container-instances. Es posible que una instancia de contenedor tenga hasta 100 puertos reservados por vez, incluidos los puertos reservados predeterminados. Los puertos asignados automáticamente no se contabilizan para la cuota de 100 puertos reservados.

`name`

Tipo: cadena

Obligatorio: no, es obligatorio para configurar Service Connect y VPC Lattice en un servicio

El nombre que se utiliza para la asignación de puertos. Este parámetro solo se aplica a Service Connect y VPC Lattice. Este parámetro es el nombre que se utiliza en la configuración de Service Connect y VPC Lattice de un servicio.

Para obtener más información, consulte Uso de Service Connect para conectar los servicios de Amazon ECS con nombres abreviados.

En el siguiente ejemplo, se utilizan los dos campos obligatorios de Service Connect y VPC Lattice.

```
"portMappings": [
    {
        "name": string,
        "containerPort": integer
    }
]
```

`protocol`

Tipo: cadena

Requerido: no

El protocolo que se utiliza para la asignación de puertos. Los valores válidos son `tcp` y `udp`. El valor predeterminado es `tcp`.

> **Important**
>
> Solo `tcp` es compatible con Service Connect. Recuerde que `tcp` está implícito si este campo no está configurado.

Utilice la sintaxis a continuación para especificar un puerto de host.

```
"portMappings": [
    {
        "containerPort": integer,
        "hostPort": integer
    }
    ...
]
```

Use la siguiente sintaxis si desea asignar automáticamente un puerto de host.

```
"portMappings": [
    {
        "containerPort": integer
    }
    ...
]
```

Credenciales de repositorio privado

`repositoryCredentials`

Tipo: objeto de RepositoryCredentials

Obligatorio: no

Las credenciales del repositorio para la autenticación de registros privados.

Para obtener más información, consulte Uso de imágenes de contenedor que no sean de AWS en Amazon ECS.

`credentialsParameter`

Tipo: cadena

Obligatorio: sí, si se utilizan `repositoryCredentials`.

El nombre de recurso de Amazon (ARN) del secreto que contiene las credenciales del repositorio privado.

Para obtener más información, consulte Uso de imágenes de contenedor que no sean de AWS en Amazon ECS.

> **Note**
>
> Cuando se utiliza la API de Amazon ECS, la AWS CLI o los SDK de AWS, si el secreto existe en la misma región que la tarea que se va a lanzar, se puede utilizar el ARN completo o el nombre del secreto. Cuando se utiliza la AWS Management Console, debe especificar el ARN completo del secreto.

A continuación, se incluye un fragmento de definición de tareas que muestra los parámetros necesarios:

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
        }
    }
]
```

Parámetros de definición de contenedor avanzados

Los siguientes parámetros avanzados de definición de contenedor ofrecen capacidades ampliadas para el comando de docker run que se utiliza para lanzar contenedores en las instancias de contenedor de Amazon ECS.

Temas

- Política de reinicio
- Comprobación de estado
- Entorno
- Configuración de red
- Almacenamiento y registro
- Seguridad
- Límites de recursos
- Etiquetas de Docker

Política de reinicio

`restartPolicy`

La política de reinicio del contenedor y los parámetros de configuración asociados. Al configurar una política de reinicio para un contenedor, Amazon ECS puede reiniciar el contenedor sin necesidad de reemplazar la tarea. Para obtener más información, consulte Reinicio de contenedores individuales en tareas de Amazon ECS con políticas de reinicio de contenedores.

`enabled`

Tipo: Booleano

Obligatorio: sí

Especifica si una política de reinicio está habilitada para el contenedor.

`ignoredExitCodes`

Tipo: matriz de enteros

Obligatorio: no

Una lista de códigos de salida que Amazon ECS ignorará y no intentará reiniciar. Puede especificar un máximo de 50 códigos de salida de contenedor. De forma predeterminada, Amazon ECS no ignora ningún código de salida.

`restartAttemptPeriod`

Tipo: entero

Obligatorio: no

La cantidad de tiempo (en segundos) que debe ejecutarse el contenedor antes de que se pueda intentar un reinicio. Un contenedor solo se puede reiniciar una vez cada `restartAttemptPeriod` segundos. Si un contenedor no puede ejecutarse durante este período de tiempo y se cierra antes, no se reiniciará. Puede especificar un mínimo `restartAttemptPeriod` de 60 segundos y un `restartAttemptPeriod` máximo de 1.800 segundos. De forma predeterminada, un contenedor debe ejecutarse durante 300 segundos antes de poder reiniciarse.

Comprobación de estado

`healthCheck`

El parámetro de comprobación de estado del contenedor y los parámetros de configuración asociados para el contenedor. Para obtener más información, consulte Determine el estado de las tareas de Amazon ECS mediante comprobaciones de estado de los contenedores.

`command`

Una matriz de cadenas que representa el comando que ejecuta el contenedor para determinar si está en buen estado. La matriz de cadenas puede comenzar por `CMD` para ejecutar los argumentos del comando directamente, o por `CMD-SHELL` para ejecutar el comando con el shell predeterminado del contenedor. Si no se especifica ninguno, se utiliza `CMD`.

Al registrar una definición de tarea en la AWS Management Console, utilice una lista de comandos separados por comas. Estos comandos se convierten en una cadena una vez que se cree la definición de tareas. A continuación, se muestra un ejemplo de entrada de comprobación de estado.

```
CMD-SHELL, curl -f http://localhost/ || exit 1
```

Cuando registre una definición de tarea mediante el panel de JSON de la AWS Management Console, la AWS CLI o las API, incluya la lista de comandos entre corchetes. A continuación, se muestra un ejemplo de entrada de comprobación de estado.

```
[ "CMD-SHELL", "curl -f http://localhost/ || exit 1" ]
```

Un código de salida de 0, sin salida `stderr`, indica una ejecución correcta y cualquier código de salida distinto de cero indica un error.

`interval`

El periodo de tiempo (en segundos) entre cada comprobación de estado. Es posible especificar entre 5 y 300 segundos. El valor predeterminado es de 30 segundos.

`timeout`

El periodo de tiempo (en segundos) que hay que esperar para que una comprobación de estado se realice correctamente antes de que se considere un error. Puede especificar entre 2 y 60 segundos. El valor de predeterminado es de 5 segundos.

`retries`

Es el número de veces que se reintenta una comprobación de estado fallida antes de que se considere que el contenedor está en mal estado. Puede especificar entre 1 y 10 reintentos. El valor predeterminado es tres reintentos.

`startPeriod`

El periodo de gracia opcional dentro del cual se puede proporcionar tiempo a los contenedores para el arranque antes de que una comprobación de estado fallida se cuente para el número máximo de reintentos. Puede especificar un valor comprendido entre 0 y 300 segundos. De forma predeterminada, `startPeriod` está deshabilitado.

Si una comprobación de estado se realiza correctamente dentro del periodo indicado en `startPeriod`, se considera que el estado del contenedor es correcto y los errores posteriores se contabilizan al calcular el número máximo de intentos.

Entorno

`cpu`

Tipo: entero

Obligatorio: no

El número de unidades de `cpu` que el agente de contenedor de Amazon ECS reserva para el contenedor. En Linux, este parámetro se corresponde con `CpuShares` en la sección Crear un contenedor.

Este campo es opcional para las tareas que utilizan el tipo de lanzamiento de Fargate. La cantidad total de CPU reservada para todos los contenedores dentro de una tarea debe ser menor que el valor `cpu` de la tarea.

Los contenedores de Linux comparten unidades de CPU no asignadas con otros contenedores en la instancia de contenedor en la misma proporción que la cantidad asignada. Por ejemplo, supongamos que ejecuta una tarea de contenedor único en un tipo de instancia de un solo núcleo con 512 unidades de CPU especificadas para dicho contenedor. Además, esa tarea es la única que se ejecuta en la instancia de contenedor. En este ejemplo, el contenedor puede utilizar la cuota completa de 1024 unidades de CPU en un momento dado. Sin embargo, supongamos que lanzó otra copia de la misma tarea en esa instancia de contenedor. Cada tarea tiene garantizado un mínimo de 512 unidades de CPU cuando sea necesario. Del mismo modo, si el otro contenedor no utiliza la CPU restante, cada contenedor puede aumentar el uso de la CPU. Sin embargo, si ambas tareas estuvieran 100 % activas todo el tiempo, están limitadas a 512 unidades de CPU.

En las instancias de contenedor de Linux, el daemon de Docker en la instancia de contenedor utiliza el valor de CPU para calcular las proporciones de cuota de CPU relativas para los contenedores en ejecución. El valor de cuota de CPU válido mínimo que permite el kernel de Linux es 2 y el valor de cuota de CPU válido máximo que permite el kernel de Linux es 262144. Sin embargo, el parámetro de CPU no es obligatorio y puede utilizar valores de CPU por debajo de 2 y por encima de 262144 en las definiciones de contenedor. Para valores de CPU por debajo de 2 (incluido el valor nulo) y por encima de 262144, el comportamiento varía en función de la versión de agente de contenedor de Amazon ECS:

En las instancias de contenedor de Windows, la cuota de CPU se aplica como una cuota absoluta. Los contenedores de Windows solo tienen acceso a la cantidad de CPU especificada que se establece en la definición de tareas. Un valor de CPU nulo o cero se pasa a Docker como`0`. A continuación, Windows interpreta este valor como el 1 % de una CPU.

Para ver otros ejemplos, consulte Cómo administra Amazon ECS los recursos de CPU y memoria.

`gpu`

Este parámetro no es compatible con los contenedores que están alojados en Fargate.

Tipo: objeto ResourceRequirement

Obligatorio: no

El número de `GPU`s físicas que el agente de contenedor de Amazon ECS reserva para el contenedor. El número de GPU reservadas para todos los contenedores de una tarea no debe superar el número de GPU disponibles en la instancia de contenedor en la que se lanza la tarea. Para obtener más información, consulte Definiciones de tareas de Amazon ECS para cargas de trabajo de GPU.

`Elastic Inference accelerator`

Este parámetro no es compatible con los contenedores que están alojados en Fargate.

Tipo: objeto ResourceRequirement

Obligatorio: no

Para el tipo `InferenceAccelerator`, el `value` coincide con el `deviceName` para un `InferenceAccelerator` especificado en una definición de tareas. Para obtener más información, consulte the section called “Nombre del acelerador de Elastic Inference”.

`essential`

Tipo: Booleano

Obligatorio: no

Supongamos que el parámetro `essential` de un contenedor se marca como `true` y dicho contenedor falla o se detiene por algún motivo. A continuación, se detienen todos los demás contenedores que forman parte de la tarea. Si el parámetro `essential` de un contenedor se marca como `false`, su error no afecta al resto de los contenedores en una tarea. Si este parámetro se omite, se supone que un contenedor es esencial.

Todas las tareas deben tener al menos un contenedor esencial. Supongamos que tiene una aplicación compuesta por varios contenedores. Agrupe los contenedores que se utilizan para un fin común en componentes y sepárelos en diversas definiciones de tareas. Para obtener más información, consulte Diseño de la arquitectura de su aplicación para Amazon ECS.

```
"essential": true|false
```

`entryPoint`

> **Important**
>
> Las primeras versiones del agente de contenedor de Amazon ECS no tratan correctamente los parámetros `entryPoint`. Si tiene problemas al utilizar `entryPoint` , actualice el agente de contenedor o introduzca los comandos y argumentos como elementos de matriz de `command` en su lugar.

Tipo: matriz de cadenas

Obligatorio: no

El punto de entrada que se transfiere al contenedor.

```
"entryPoint": ["string", ...]
```

`command`

Tipo: matriz de cadenas

Obligatorio: no

El comando que se transfiere al contenedor. Este parámetro se corresponde con `Cmd` en el comando create-container y el parámetro `COMMAND` para ejecutar el docker. Si hay varios argumentos, asegúrese de que cada uno de ellos sea una cadena distinta en la matriz.

```
"command": ["string", ...]
```

`workingDirectory`

Tipo: cadena

Requerido: no

El directorio de trabajo para ejecutar los comandos dentro del contenedor. Este parámetro se corresponde con `WorkingDir` en la sección Crear un contenedor de la API remota de Docker y con la opción `--workdir` de docker run.

```
"workingDirectory": "string"
```

`environmentFiles`

No está disponible para los contenedores de Windows en Fargate.

Tipo: matriz de objetos

Obligatorio: no

Una lista de archivos que contienen las variables de entorno que transferir a un contenedor. Este parámetro se corresponde con la opción `--env-file` del comando de docker run.

Puede especificar hasta diez archivos de entorno. El archivo debe tener una extensión de archivo `.env`. Cada línea de un archivo de entorno contiene una variable de entorno con el formato `VARIABLE=VALUE`. Las líneas que comienzan por # se tratan como comentarios y se ignoran.

Si hay variables de entorno individuales especificadas en la definición del contenedor, tienen prioridad sobre las variables que contiene un archivo de entorno. Si se especifican varios archivos de entorno que contienen la misma variable, se procesan en orden descendente. Le recomendamos que utilice nombres de variables únicos. Para obtener más información, consulte Transferencia de una variable de entorno individual a un contenedor de Amazon ECS.

`value`

Tipo: cadena

Obligatorio: sí

Nombre de recurso de Amazon (ARN) del objeto Amazon S3 que contiene el archivo de variable de entorno.

`type`

Tipo: cadena

Obligatorio: sí

Tipo de archivo que se utilizará. El único valor admitido es `s3`.

`environment`

Tipo: matriz de objetos

Obligatorio: no

Las variables de entorno a transferir a un contenedor. Este parámetro se corresponde con `Env` en el comando create-container de docker y la opción `--env` con el comando de docker run.

> **Important**
>
> No es recomendable que utilice variables del entorno en texto sin formato para información confidencial, como los datos de las credenciales.

`name`

Tipo: cadena

Obligatorio: sí, si se utiliza `environment`

El nombre de la variable de entorno.

`value`

Tipo: cadena

Obligatorio: sí, si se utiliza `environment`

El valor de la variable de entorno.

```
"environment" : [
    { "name" : "string", "value" : "string" },
    { "name" : "string", "value" : "string" }
]
```

`secrets`

Tipo: matriz de objetos

Obligatorio: no

Un objeto que representa el secreto que se expone en el contenedor. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.

`name`

Tipo: cadena

Obligatorio: sí

El valor que se ha de establecer como la variable de entorno en el contenedor.

`valueFrom`

Tipo: cadena

Obligatorio: sí

El secreto para exponer en el contenedor. Los valores admitidos son el nombre de recurso de Amazon (ARN) completo del secreto de AWS Secrets Manager o el ARN completo del parámetro en el almacén de parámetros de AWS Systems Manager.

> **Note**
>
> Si el parámetro del Almacén de parámetros de Systems Manager o el parámetro de Secrets Manager existe en la misma Región de AWS que la tarea que se va a lanzar, se puede utilizar el ARN completo o el nombre del secreto. Si el parámetro existe en una región distinta, el ARN completo debe especificarse.

```
"secrets": [
    {
        "name": "environment_variable_name",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }
]
```

Configuración de red

`disableNetworking`

Este parámetro no es compatible con las tareas que se ejecutan en Fargate.

Tipo: Booleano

Obligatorio: no

Cuando este parámetro es verdadero, la conexión en red está apagada dentro del contenedor.

El valor predeterminado es `false`.

```
"disableNetworking": true|false
```

`links`

Este parámetro no es compatible con las tareas que utilizan el modo de red `awsvpc`.

Tipo: matriz de cadenas

Obligatorio: no

El parámetro `link` permite a los contenedores comunicarse entre sí sin la necesidad de mapeos de puerto. Este parámetro solo se admite si el modo de red de una definición de tarea se establece en `bridge`. El constructo `name:internalName` es análogo a `name:alias` en enlaces de Docker. Se admiten hasta 255 letras (mayúsculas y minúsculas), números, guiones y caracteres de subrayado.

> **Important**
>
> Es posible que los contenedores que se colocan en la misma instancia de contenedor puedan comunicarse entre sí sin necesidad de enlaces ni asignaciones de puerto de host. El aislamiento de red en una instancia de contenedor se controla mediante los grupos de seguridad y la configuración de VPC.

```
"links": ["name:internalName", ...]
```

`hostname`

Tipo: cadena

Requerido: no

El nombre de host que utilizar para el contenedor. Este parámetro se asigna a `Hostname` en el comando create-container de docker y la opción `--hostname` con docker run.

> **Note**
>
> Si utiliza el modo de red `awsvpc`, el parámetro `hostname` no se admite.

```
"hostname": "string"
```

`dnsServers`

Esto no es compatible con las tareas que se ejecutan en Fargate.

Tipo: matriz de cadenas

Obligatorio: no

Una lista de servidores DNS que se presentan al contenedor.

```
"dnsServers": ["string", ...]
```

`extraHosts`

Este parámetro no es compatible con los contenedores que utilizan el modo de red `awsvpc`.

Tipo: matriz de objetos

Obligatorio: no

Una lista de nombres de host y mapeos de direcciones IP que añadir al archivo `/etc/hosts` en el contenedor.

Este parámetro se corresponde con `ExtraHosts` en el comando create-container de docker y la opción `--add-host` se corresponde con docker run.

```
"extraHosts": [
      {
        "hostname": "string",
        "ipAddress": "string"
      }
      ...
    ]
```

`hostname`

Tipo: cadena

Obligatorio: sí, si se utilizan `extraHosts`.

El nombre de host para utilizar en la entrada `/etc/hosts`.

`ipAddress`

Tipo: cadena

Obligatorio: sí, si se utilizan `extraHosts`.

La dirección IP para utilizar en la entrada `/etc/hosts`.

Almacenamiento y registro

`readonlyRootFilesystem`

Tipo: Booleano

Obligatorio: no

Cuando este parámetro es verdadero, al contenedor se le concede acceso de solo lectura a su sistema de archivos raíz. Este parámetro se corresponde con `ReadonlyRootfs` en el comando create-container de docker y la opción `--read-only` con docker run.

> **Note**
>
> Este parámetro no es compatible con contenedores Windows.

El valor predeterminado es `false`.

```
"readonlyRootFilesystem": true|false
```

`mountPoints`

Tipo: matriz de objetos

Obligatorio: no

Puntos de montaje para los volúmenes de datos del contenedor. Este parámetro asigna a `Volumes` en la API create-container de Docker y la opción `--volume` a docker run.

Los contenedores de Windows pueden montar directorios completos en la misma unidad que `$env:ProgramData`. Los contenedores de Windows no pueden montar directorios en una

unidad diferente y los puntos de montaje no se pueden utilizar entre unidades. Debe especificar los puntos de montaje para adjuntar un volumen de Amazon EBS directamente a una tarea de Amazon ECS.

`sourceVolume`

Tipo: cadena

Obligatorio: sí, si se utilizan `mountPoints`.

El nombre del volumen a montar.

`containerPath`

Tipo: cadena

Obligatorio: sí, si se utilizan `mountPoints`.

La ruta del contenedor donde se montará el volumen.

`readOnly`

Tipo: Booleano

Obligatorio: no

Si este valor es `true`, el acceso del contenedor al volumen es de solo lectura. Si este valor es `false`, el contenedor puede escribir en el volumen. El valor predeterminado es `false`.

Para las tareas que se ejecutan en instancias de EC2 con sistema operativo Windows, deje el valor predeterminado `false`.

`volumesFrom`

Tipo: matriz de objetos

Obligatorio: no

Volúmenes de datos que montar desde otro contenedor. Este parámetro se corresponde con `VolumesFrom` en el comando create-container de docker y la opción `--volumes-from` se corresponde con docker run.

`sourceContainer`

Tipo: cadena

Obligatorio: sí, si se utiliza `volumesFrom`

El nombre del volumen contenedor desde el que montar los volúmenes.

`readOnly`

Tipo: Booleano

Obligatorio: no

Si este valor es `true`, el acceso del contenedor al volumen es de solo lectura. Si este valor es `false`, el contenedor puede escribir en el volumen. El valor predeterminado es `false`.

```
"volumesFrom": [
                {
                  "sourceContainer": "string",
                  "readOnly": true|false
                }
              ]
```

`logConfiguration`

Tipo: objeto LogConfiguration

Obligatorio: no

La especificación de configuración de registros para el contenedor.

Para obtener ejemplos de definiciones de tareas que utilizan una configuración de registro, consulte Ejemplo de definiciones de tareas de Amazon ECS.

Este parámetro se corresponde con `LogConfig` en el comando create-container de docker y la opción `--log-driver` se corresponde con docker run. De forma predeterminada, los contenedores utilizan el mismo controlador de registro que el daemon de Docker. No obstante, el contenedor podría utilizar un controlador de registro diferente al daemon de Docker especificando un controlador de registro con este parámetro en la definición de contenedor. Para utilizar un controlador de registro distinto para un contenedor, el sistema de registro se debe configurar correctamente en la instancia de contenedor (o en un servidor de registro distinto para opciones de registro remotas).

Tenga cuenta lo siguiente al especificar una configuración de registros para los contenedores:

- Amazon ECS admite un subconjunto de controladores de registro disponibles para el daemon de Docker. Es posible que haya controladores de registro adicionales en versiones futuras del agente de contenedor de Amazon ECS.
- Este parámetro requiere la versión 1.18 o posterior de la API remota de Docker en la instancia de contenedor.
- Debe instalar cualquier software adicional fuera de la tarea. Por ejemplo, los agregadores de salida de Fluentd o un host remoto que ejecute Logstash adonde se envían los registros de Gelf.

```
"logConfiguration": {
      "logDriver": "awslogs","fluentd","gelf","json-
file","journald","splunk","syslog","awsfirelens",
      "options": {"string": "string"
        ...},
 "secretOptions": [{
  "name": "string",
  "valueFrom": "string"
 }]
}
```

`logDriver`

Tipo: cadena

Valores válidos: `"awslogs","fluentd","gelf","json-file","journald","splunk","syslog","awsfirelens"`

Obligatorio: sí, si se utiliza `logConfiguration`

El controlador de registro que utilizar para el contenedor. De forma predeterminada, los valores válidos mostrados anteriormente son controladores de registro con los que el agente de contenedor de Amazon ECS se puede comunicar.

Los controladores de registro admitidos son `awslogs`, `splunk` y `awsfirelens`.

Para obtener más información acerca del uso del controlador de registro `awslogs` en las definiciones de tareas para enviar los registros de contenedor a CloudWatch Logs, consulte Envío de registros de Amazon ECS a CloudWatch .

Para obtener más información sobre el uso del controlador del registro `awsfirelens`, consulte Envío de registros personalizados.

> **Note**
>
> Si tiene un controlador personalizado que no figura en la lista, puede adaptar el proyecto del agente de contenedor de Amazon ECS que está disponible en GitHub y personalizarlo para que funcione con dicho controlador. Le recomendamos enviar solicitudes de inserción para los cambios que desea que incluyamos. No obstante, actualmente no admitimos la ejecución de copias modificadas de este software.

Este parámetro requiere la versión 1.18 de la API remota de Docker o superior en su instancia de contenedor.

`options`

Tipo: mapa de cadena a cadena

Obligatorio: no

Las opciones de configuración de asignación clave/valor para enviar al controlador de registro.

Las opciones que puede especificar dependen del controlador de registro. Algunas de las opciones que puede especificar cuando utiliza el router `awslogs` para enrutar los registros a Amazon CloudWatch son las siguientes:

`awslogs-create-group`

Obligatorio: no

Especifique si desea que el grupo de registro se cree automáticamente. Si no se especifica esta opción, el valor predeterminado es `false`.

> **Note**
>
> Su política de IAM debe incluir el permiso `logs:CreateLogGroup` antes de intentar utilizar `awslogs-create-group`.

`awslogs-region`

Obligatorio: sí

Especifique la Región de AWS a la que el controlador de registro `awslogs` enviará los registros de Docker. Puede elegir enviar todos los registros desde clústeres de diferentes regiones a una única región de CloudWatch Logs. Esto es para que todos sean visibles en una sola ubicación. De lo contrario, puede separarlos por región para obtener un mayor grado de detalle. Asegúrese de que el grupo de registro especificado exista en la región que especifique con esta opción.

`awslogs-group`

Obligatorio: sí

Asegúrese de especificar un grupo de registro al que el controlador de registros `awslogs` envíe sus flujos de registros.

`awslogs-stream-prefix`

Obligatorio: sí

Utilice esta opción `awslogs-stream-prefix` para asociar un flujo de registro con el prefijo especificado, el nombre del contenedor y el ID de la tarea de Amazon ECS a la que pertenece el contenedor. Si especifica un prefijo con esta opción, el flujo de registro adopta el siguiente formato.

```
prefix-name/container-name/ecs-task-id
```

Si no especifica un prefijo con esta opción, el flujo de registro se nombra según el ID del contenedor que asigna el daemon de Docker en la instancia de contenedor. Dado que es difícil realizar un seguimiento de los registros hasta el contenedor que los envió solo con el ID de contenedor de Docker (que solo está disponible en la instancia de contenedor), le recomendamos que especifique un prefijo con esta opción.

En el caso de los servicios de Amazon ECS, puede utilizar el nombre del servicio como prefijo. De este modo, puede realizar un seguimiento de flujos de registros hasta el servicio al que pertenece el contenedor, el nombre del contenedor que los envió y el ID de la tarea a la que pertenece el contenedor.

Debe especificar un prefijo de flujo para los registros para que aparezcan en el panel de registros al utilizar la consola de Amazon ECS.

`awslogs-datetime-format`

Obligatorio: no

Esta opción define un patrón de inicio de varias líneas en formato `strftime` de Python. Un mensaje de registro consta de una línea que coincide con el patrón y de líneas siguientes que no coinciden con el patrón. La línea coincidente es el delimitador entre los mensajes de registro.

Un ejemplo de caso de uso de este formato es para analizar la salida como un volcado de pila, que de lo contrario podría registrarse en varias entradas. El patrón correcto permite capturarla en una sola entrada.

Para obtener más información, consulte awslogs-datetime-format.

No puede configurar las opciones `awslogs-datetime-format` y `awslogs-multiline-pattern` a la vez.

> **Note**
>
> El registro de varias líneas realiza análisis de las expresiones regulares y correspondencia de todos los mensajes de registro. Esto puede tener un impacto negativo sobre el rendimiento de registro.

`awslogs-multiline-pattern`

Obligatorio: no

Esta opción define un patrón de inicio de varias líneas que utiliza una expresión regular. Un mensaje de registro consta de una línea que coincide con el patrón y de líneas siguientes que no coinciden con el patrón. La línea coincidente es el delimitador entre los mensajes de registro.

Para obtener más información, consulte awslogs-multiline-pattern.

Esta opción se pasa por alto si también se ha configurado `awslogs-datetime-format`.

No puede configurar las opciones `awslogs-datetime-format` y `awslogs-multiline-pattern` a la vez.

> **Note**
>
> El registro de varias líneas realiza análisis de las expresiones regulares y correspondencia de todos los mensajes de registro. Esto puede tener un impacto negativo sobre el rendimiento de registro.

Las siguientes opciones se aplican a todos los controladores de registro compatibles.

`mode`

Obligatorio: no

Valores válidos: `non-blocking` | `blocking`

Esta opción define el modo de entrega de los mensajes de registro del contenedor al controlador de registro especificado con `logDriver`. El modo de entrega que elige afecta la disponibilidad de las aplicaciones cuando se interrumpe el flujo de registros desde el contenedor.

Si utiliza el modo `blocking` y se interrumpe el flujo de registros, se bloquearán las llamadas desde el código del contenedor para escribir en los flujos `stdout` y `stderr`. Como resultado, el hilo del registro de la aplicación se bloqueará. Esto puede provocar que la aplicación deje de responder y que se produzca un error en la comprobación del estado del contenedor.

Si utiliza el modo `non-blocking`, los registros del contenedor se almacenan en un búfer intermedio en memoria configurado con la opción. `max-buffer-size` Esto evita que la aplicación deje de responder cuando los registros no se puedan enviar. Le recomendamos que utilice este modo si quiere garantizar la disponibilidad del servicio y si no hay problema con la pérdida de registros. Para obtener más información, consulte Evitar la pérdida de registros con el modo sin bloqueo en el controlador `awslogs` de registros del contenedor.

Puede establecer un `mode` predeterminado para todos los contenedores de una Región de AWS específica mediante la configuración de la cuenta de `defaultLogDriverMode`. Si no especifica la opción `mode` en `logConfiguration` ni configura los ajustes de la cuenta, Amazon ECS utilizará el modo `non-blocking` de manera predeterminada. Para obtener más información acerca de la configuración de la cuenta, consulte Modo de controlador de registro predeterminado.

Cuando se utiliza el modo `non-blocking`, la opción de registro `max-buffer-size` controla el tamaño del búfer que se utiliza para el almacenamiento intermedio de mensajes. Asegúrese de especificar un tamaño de búfer adecuado en función de su aplicación. La cantidad total de memoria asignada a nivel de tarea debe ser superior a la cantidad de memoria asignada a todos los contenedores, además del búfer de memoria del controlador de registro.

> **Note**
>
> El 25 de junio de 2025, Amazon ECS cambió el modo de controlador de registro predeterminado de `blocking` a `non-blocking` a fin de priorizar la disponibilidad de las tareas sobre el registro. Para seguir utilizando el modo `blocking` después de este cambio, realice una de las siguientes acciones:
>
> - Establezca la opción `mode` en la `logConfiguration` de su definición de contenedor como `blocking`.
> - Establezca el ajuste de cuenta `defaultLogDriverMode` en `blocking`.

`max-buffer-size`

Obligatorio: no

Valor predeterminado: `10m`

Cuando se utiliza el modo `non-blocking`, la opción de registro `max-buffer-size` controla el tamaño del búfer que se utiliza para el almacenamiento intermedio de mensajes. Asegúrese de especificar un tamaño de búfer adecuado en función de su aplicación. Cuando el búfer se llene, no se podrán almacenar más registros. Los registros que no se pueden almacenar se pierden.

Para enrutar los registros mediante el enrutador de registros `splunk`, debe especificar un `splunk-token` y un `splunk-url`.

Cuando utiliza el enrutador de registros `awsfirelens` para enrutar a un destino de Servicio de AWS o AWS Partner Network para el almacenamiento y análisis de registros, puede configurar la opción `log-driver-buffer-limit` hasta el límite del número de eventos almacenados en búfer en la memoria antes de enviarlos al contenedor del enrutador de registros. Puede ayudar a resolver el posible problema de pérdida de registros porque un alto rendimiento podría provocar que se quede sin memoria para el búfer dentro de Docker.

Para obtener más información, consulte the section called “Configuración de los registros para conseguir un alto rendimiento”.

Otras opciones que puede especificar cuando se utiliza `awsfirelens` para enrutar los registros dependen del destino. Al exportar registros a Amazon Data Firehose, puede especificar la Región de AWS con `region` y el nombre del flujo de registros con `delivery_stream`.

Al exportar registros a Amazon Kinesis Data Streams, puede especificar una Región de AWS con `region` y un nombre de flujo de datos con `stream`.

Al exportar registros a Amazon OpenSearch Service, puede especificar opciones como `Name`, `Host` (punto final de OpenSearch Service sin protocolo), `Port`, `Index`, `Type`, `Aws_auth`, `Aws_region`, `Suppress_Type_Name` y `tls`.

Al exportar registros a Amazon S3, puede especificar el bucket mediante la opción `bucket`. También puede especificar `region`, `total_file_size`, `upload_timeout` y `use_put_object` como opciones.

Este parámetro requiere la versión 1.19 de la API remota de Docker o superior en su instancia de contenedor.

`secretOptions`

Tipo: matriz de objetos

Obligatorio: no

Un objeto que representa el secreto que transferir a la configuración de registro. Los secretos utilizados en la configuración de registros pueden incluir un token de autenticación, un certificado o una clave de cifrado. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.

`name`

Tipo: cadena

Obligatorio: sí

El valor que se ha de establecer como la variable de entorno en el contenedor.

`valueFrom`

Tipo: cadena

Obligatorio: sí

El secreto que exponer a la configuración de registro del contenedor.

```
"logConfiguration": {
 "logDriver": "splunk",
 "options": {
  "splunk-url": "https://cloud.splunk.com:8080",
  "splunk-token": "...",
  "tag": "...",
  ...
 },
 "secretOptions": [{
  "name": "splunk-token",
  "valueFrom": "/ecs/logconfig/splunkcred"
 }]
}
```

`firelensConfiguration`

Tipo: objeto FirelensConfiguration

Obligatorio: no

La configuración de FireLens para el contenedor. Esto se utiliza para especificar y configurar un enrutador de registro para registros de contenedores. Para obtener más información, consulte Envío de registros de Amazon ECS a un servicio de AWS o AWS Partner.

```
{
    "firelensConfiguration": {
        "type": "fluentd",
        "options": {
            "KeyName": ""
        }
    }
}
```

`options`

Tipo: mapa de cadena a cadena

Obligatorio: no

Las opciones de asignación de clave/valor que se deben usar al configurar el enrutador de registros. Este campo es opcional y se puede utilizar para especificar un archivo de configuración personalizado o agregar metadatos adicionales, como la tarea, la definición de tareas, el clúster y detalles de la instancia de contenedor al evento de registro. Si se especifica, la sintaxis que se va a utilizar es `"options":{"enable-ecs-log-metadata":"true|false","config-file-type:"s3|file","config-file-value":"arn:aws:s3:::amzn-s3-demo-bucket/fluent.conf|filepath"}`. Para obtener más información, consulte Definición de tareas de Amazon ECS de ejemplo: enrutar registros a FireLens.

`type`

Tipo: cadena

Obligatorio: sí

El router de registros que se va a utilizar. Los valores válidos son `fluentd` o `fluentbit`.

Seguridad

Para obtener más información sobre la seguridad del contenedor, consulte Prácticas recomendadas de seguridad para las tareas y contenedores de Amazon ECS.

`credentialSpecs`

Tipo: matriz de cadenas

Obligatorio: no

Una lista de los ARN de SSM o Amazon S3 en un archivo de especificaciones de credenciales (`CredSpec`) que configura el contenedor para la autenticación de Active Directory. Le recomendamos que utilice este parámetro en lugar de `dockerSecurityOptions`. El número máximo de ARN es 1.

Hay dos formatos para cada ARN.

Especificación de credenciales sin dominio: MyARN

Utiliza `credentialspecdomainless:MyARN` para proporcionar a la `CredSpec` una sección adicional para un secreto en Secrets Manager. Proporciona las credenciales de inicio de sesión al dominio en el secreto.

Cada tarea que se ejecute en cualquier instancia de contenedor puede unirse a diferentes dominios.

Puede utilizar este formato sin unir la instancia de contenedor a un dominio.

Especificación de credenciales: MyARN

Utiliza `credentialspec:MyARN` para proporcionar una `CredSpec` para un solo dominio.

Debe unir la instancia de contenedor al dominio antes de iniciar cualquier tarea que utilice esta definición de tarea.

En ambos formatos, sustituya `MyARN` por el ARN en SSM o Amazon S3.

La `credspec` debe proporcionar un ARN en Secrets Manager para un secreto que contenga el nombre de usuario, la contraseña y el dominio para conectarse. Para mayor seguridad, la instancia no está unida al dominio para la autenticación sin dominio. Las demás aplicaciones de la instancia no pueden utilizar las credenciales sin dominio. Puede utilizar este parámetro para ejecutar tareas en la misma instancia, incluso si las tareas necesitan unirse a dominios diferentes. Para obtener más información, consulte Uso de gMSA para contenedores de Windows y Uso de gMSA para contenedores de Linux.

`user`

Tipo: cadena

Requerido: no

El usuario que se utiliza dentro del contenedor. Este parámetro se corresponde con `User` en el comando create-container de docker y la opción `--user` con docker run.

> **Important**
>
> Al ejecutar tareas que utilizan el modo de red de `host`, no debe ejecutar contenedores con el usuario raíz (UID 0). Como práctica recomendada de seguridad, utilice siempre un usuario que no sea usuario raíz.

Puede especificar el elemento `user` utilizando los siguientes formatos. Si se especifica un UID o GID, debe especificarlo como número entero positivo.

- `user`

- `user:group`
- `uid`
- `uid:gid`
- `user:gid`
- `uid:group`

> **Note**
> Este parámetro no es compatible con contenedores Windows.

```
"user": "string"
```

Límites de recursos

`ulimits`

Tipo: matriz de objetos

Obligatorio: no

Lista de valores `ulimit` a definir para un contenedor. Este valor sobrescribe la configuración predeterminada de la cuota de recursos para el sistema operativo. Este parámetro se corresponde con `Ulimits` en el comando create-container de docker y la opción `--ulimit` se corresponde con docker run.

Las tareas de Amazon ECS alojadas en Fargate utilizan los valores límite de recursos predeterminados que establece el sistema operativo, a excepción del parámetro límite de recursos `nofile`. El límite de recursos `nofile` define una restricción en el número de archivos abiertos que puede utilizar un contenedor. En Fargate, el límite flexible `nofile` predeterminado es `65535` y el límite invariable es `65535`. Puede establecer los valores de ambos límites en un valor máximo de `1048576`. Para obtener más información, consulte Límites de recursos de tareas.

Este parámetro requiere la versión 1.18 de la API remota de Docker o superior en su instancia de contenedor.

> **Note**
>
> Este parámetro no es compatible con contenedores Windows.

```
"ulimits": [
      {
        "name":
 "core"|"cpu"|"data"|"fsize"|"locks"|"memlock"|"msgqueue"|"nice"|"nofile"|"nproc"|"rss"|"rtp
        "softLimit": integer,
        "hardLimit": integer
      }
      ...
    ]
```

`name`

Tipo: cadena

Valores válidos: `"core" | "cpu" | "data" | "fsize" | "locks" | "memlock" | "msgqueue" | "nice" | "nofile" | "nproc" | "rss" | "rtprio" | "rttime" | "sigpending" | "stack"`

Obligatorio: sí, si se utilizan `ulimits`.

El valor `type` de `ulimit`.

`hardLimit`

Tipo: entero

Obligatorio: sí, si se utilizan `ulimits`.

El límite máximo para el tipo de `ulimit`. El valor se puede especificar en bytes, segundos o como un conteo, según el `type` del `ulimit`.

`softLimit`

Tipo: entero

Obligatorio: sí, si se utilizan `ulimits`.

El límite flexible para el tipo de `ulimit`. El valor se puede especificar en bytes, segundos o como un conteo, según el `type` del `ulimit`.

Etiquetas de Docker

`dockerLabels`

Tipo: mapa de cadena a cadena

Obligatorio: no

Un mapa de clave/valor de etiquetas que agregar al contenedor. Este parámetro se corresponde con `Labels` en el comando create-container de docker y la opción `--label` se corresponde con docker run.

Este parámetro requiere la versión 1.18 de la API remota de Docker o superior en su instancia de contenedor.

```
"dockerLabels": {"string": "string"
      ...}
```

Otros parámetros de definición de contenedor

Los siguientes parámetros de definición de contenedor se pueden utilizar al registrar definiciones de tareas en la consola de Amazon ECS mediante la opción Configure via JSON (Configurar mediante JSON). Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

Temas

- Parámetros de Linux
- Dependencia de contenedor
- Tiempos de espera de contenedor
- Controles del sistema
- Interactivo
- Pseudo terminal

Parámetros de Linux

`linuxParameters`

Tipo: objeto LinuxParameters

Obligatorio: no

Opciones específicas de Linux que se aplican al contenedor, como KernelCapabilities.

> **Note**
>
> Este parámetro no es compatible con los contenedores de Windows.

```
"linuxParameters": {
      "capabilities": {
        "add": ["string", ...],
        "drop": ["string", ...]
        }
      }
```

`capabilities`

Tipo: objeto KernelCapabilities

Obligatorio: no

Las capacidades de Linux para el contenedor que se eliminan de la configuración predeterminada proporcionada por Docker. Para obtener más información sobre estas capacidades de Linux, consulte la página del manual de Linux sobre capacidades(7).

`add`

Tipo: matriz de cadenas

Valores válidos: `"SYS_PTRACE"`

Obligatorio: no

Las capacidades de Linux para el contenedor que se deben agregar a la configuración predeterminada proporcionada por Docker. Este parámetro se corresponde con `CapAdd` en el comando create-container de docker y la opción `--cap-add` se corresponde con docker run.

`drop`

Tipo: matriz de cadenas

Valores válidos: `"ALL" | "AUDIT_CONTROL" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE" | "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" | "NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW" | "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" | "SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT" | "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" | "SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"`

Obligatorio: no

Las capacidades de Linux para el contenedor que se deben eliminar de la configuración predeterminada proporcionada por Docker. Este parámetro se corresponde con `CapDrop` en el comando create-container de docker y la opción `--cap-drop` con docker run.

`devices`

Cualquier dispositivo host que exponer en el contenedor. Este parámetro se corresponde con `Devices` en el comando create-container de docker y la opción `--device` con docker run.

> **Note**
>
> El parámetro `devices` no se admite cuando se utiliza el tipo de lanzamiento de Fargate.

Tipo: matriz de objetos Dispositivo

Obligatorio: no

`hostPath`

La ruta del dispositivo de la instancia de contenedor del host.

Tipo: cadena

Obligatorio: sí

`containerPath`

La ruta dentro del contenedor en la cual exponer el dispositivo host.

Tipo: cadena

Requerido: no

`permissions`

Los permisos explícitos que proporcionar al contenedor para el dispositivo. De forma predeterminada, el contenedor tiene permisos para `read`, `write` y `mknod` en el dispositivo.

Tipo: matriz de cadenas

Valores válidos: `read` | `write` | `mknod`

`initProcessEnabled`

Ejecute un proceso `init` dentro del contenedor que reenvíe señales y aproveche procesos. Este parámetro se corresponde con la opción `--init` con docker run.

Este parámetro requiere la versión 1.25 de la API remota de Docker o superior en su instancia de contenedor.

`maxSwap`

Esto no es compatible con las tareas que se ejecutan en Fargate.

La cantidad total de memoria de intercambio (en MiB) que puede utilizar un contenedor. Este parámetro se traduce en la opción `--memory-swap` de docker run donde el valor es la suma de la memoria del contenedor más el valor de `maxSwap`.

Si se especifica un valor `maxSwap` para `0`, el contenedor no utiliza el intercambio. Los valores aceptados son `0` o cualquier entero positivo. Si se omite el parámetro `maxSwap`, el contenedor utiliza la configuración de intercambio de la instancia de contenedor en la que se está ejecutando. Debe establecerse un valor de `maxSwap` para el parámetro `swappiness`.

`sharedMemorySize`

El valor del tamaño (en MiB) del volumen `/dev/shm`. Este parámetro se corresponde con la opción `--shm-size` con docker run.

> **Note**
>
> Si utiliza tareas que emplean el tipo de lanzamiento de Fargate, no se admite el parámetro `sharedMemorySize`.

Tipo: entero

`tmpfs`

La ruta del contenedor, las opciones de montaje y el tamaño máximo (en MiB) del montaje tmpfs. Este parámetro se corresponde con la opción `--tmpfs` con docker run.

> **Note**
> Si utiliza tareas que emplean el tipo de lanzamiento de Fargate, no se admite el parámetro `tmpfs`.

Tipo: matriz de objetos Tmpfs

Obligatorio: no

`containerPath`

La ruta de archivo absoluta en la que se montará el volumen tmpfs.

Tipo: cadena

Obligatorio: sí

`mountOptions`

La lista de opciones de montaje del volumen tmpfs.

Tipo: matriz de cadenas

Obligatorio: no

Valores válidos: `"defaults" | "ro" | "rw" | "suid" | "nosuid" | "dev" | "nodev" | "exec" | "noexec" | "sync" | "async" | "dirsync" | "remount" | "mand" | "nomand" | "atime" | "noatime" | "diratime" | "nodiratime" | "bind" | "rbind" | "unbindable" | "runbindable" | "private" | "rprivate" | "shared" | "rshared" | "slave" | "rslave" | "relatime" | "norelatime" | "strictatime" | "nostrictatime" | "mode" | "uid" | "gid" | "nr_inodes" | "nr_blocks" | "mpol"`

`size`

El tamaño máximo (en MiB) del volumen tmpfs.

Tipo: entero

Obligatorio: sí

Dependencia de contenedor

`dependsOn`

Tipo: matriz de objetos ContainerDependency

Obligatorio: no

Las dependencias definidas para el inicio y apagado del contenedor. Un contenedor puede contener varias dependencias. Cuando una dependencia se define para el inicio del contenedor, se invierte para el apagado del contenedor. Para ver un ejemplo, consulta Dependencia de contenedor.

> **Note**
>
> Si un contenedor no cumple una restricción de dependencia o agota el tiempo de espera antes de cumplir la restricción, Amazon ECS no adelanta los contenedores dependientes a su siguiente estado.

Este parámetro requiere que la tarea o el servicio utilicen la versión de la plataforma `1.3.0` o una posterior (Linux) o `1.0.0` (Windows).

```
"dependsOn": [
    {
        "containerName": "string",
        "condition": "string"
    }
]
```

`containerName`

Tipo: cadena

Obligatorio: sí

El nombre del contenedor que debe cumplir la condición especificada.

`condition`

Tipo: cadena

Obligatorio: sí

La condición de dependencia del contenedor. Están disponibles las siguientes condiciones y su comportamiento:

- `START`: esta condición simula el comportamiento de enlaces y volúmenes en la actualidad. La condición valida que un contenedor dependiente se inicia antes de permitir que se inicien otros contenedores.
- `COMPLETE`: esta condición valida que un contenedor dependiente se ejecute hasta su finalización (se cierre) antes de permitir que otros contenedores se inicien. Esto puede resultar útil para contenedores no esenciales que ejecutan un script y, a continuación, se cierran. Esta condición no se puede establecer en un contenedor esencial.
- `SUCCESS`: esta condición es la misma que `COMPLETE`, pero además requiere que el contenedor se cierre con un estado `zero`. Esta condición no se puede establecer en un contenedor esencial.
- `HEALTHY`: esta condición valida que el contenedor dependiente pase su comprobación de estado de contenedor antes de permitir que otros contenedores se inicien. Esto requiere que el contenedor dependiente tenga configuradas las comprobaciones de estado en la definición de tarea. Esta condición solo se confirma durante el inicio de tarea.

Tiempos de espera de contenedor

`startTimeout`

Tipo: entero

Obligatorio: no

Valores de ejemplo: `120`

Tiempo que hay que esperar (en segundos) antes de desistir en resolver dependencias para un contenedor.

Por ejemplo, se especifican dos contenedores en una definición de tareas donde `containerA` tenga una dependencia en `containerB` que alcance un estado `COMPLETE`, `SUCCESS` o

`HEALTHY`. Si se especifica un valor `startTimeout` para `containerB` y este no alcanza el estado deseado en ese tiempo, `containerA` no se inicia.

> **Note**
>
> Si un contenedor no cumple una restricción de dependencia o agota el tiempo de espera antes de cumplir la restricción, Amazon ECS no adelanta los contenedores dependientes a su siguiente estado.

Este parámetro requiere que la tarea o el servicio utilicen la versión de la plataforma `1.3.0` o una posterior (Linux). El valor máximo es 120 segundos.

`stopTimeout`

Tipo: entero

Obligatorio: no

Valores de ejemplo: `120`

Duración de tiempo (en segundos) que esperar a que el contenedor se cancelen de forma forzada si no sale de forma normal por sí mismo.

Este parámetro requiere que la tarea o el servicio utilicen la versión de la plataforma `1.3.0` o una posterior (Linux). Si no se especifica el parámetro, se utiliza el valor predeterminado de 30 segundos. El valor máximo es 120 segundos.

Controles del sistema

`systemControls`

Tipo: objeto SystemControl

Obligatorio: no

Una lista de parámetros de kernel del espacio de nombres que se van a establecer en el contenedor. Este parámetro se corresponde con `Sysctls` en el comando create-container de docker y la opción `--sysctl` con docker run. Por ejemplo, puede configurar la configuración `net.ipv4.tcp_keepalive_time` para mantener conexiones de mayor duración.

No se recomienda especificar parámetros `systemControls` relacionados con la red para varios contenedores en una única tarea que también utilice el modo de red `awsvpc` u `host`. De ese modo, se obtienen las siguientes desventajas:

- Si establece `systemControls` en cualquier contenedor, se aplicará a todos los contenedores de la tarea. Si establece diferentes parámetros `systemControls` para varios contenedores en una sola tarea, el contenedor que se inicie en último lugar determinará qué `systemControls` se aplicará.

Si configura un espacio de nombres de recursos de IPC para usarlo para los contenedores de la tarea, se aplican las siguientes condiciones a los controles del sistema. Para obtener más información, consulte Modo IPC.

- Para las tareas que usan el modo de IPC `host`, no se admiten los `systemControls` del espacio de nombres IPC.
- Para las tareas que utilizan el modo de IPC `task`, los valores de `systemControls` del espacio de nombres IPC se aplican a todos los contenedores de una tarea.

> **Note**
>
> Este parámetro no es compatible con contenedores Windows.

> **Note**
>
> Este parámetro solo se admite para las tareas que están alojadas en AWS Fargate si utilizan la versión de la plataforma `1.4.0` o una posterior (Linux). Este parámetro no es compatible con contenedores de Windows en Fargate.

```
"systemControls": [
    {
         "namespace":"string",
         "value":"string"
    }
]
```

`namespace`

Tipo: cadena

Requerido: no

El parámetro de kernel del espacio de nombres para el que se va a establecer un `value`.

Valores de espacio de nombres IPC válidos: `"kernel.msgmax" | "kernel.msgmnb" | "kernel.msgmni" | "kernel.sem" | "kernel.shmall" | "kernel.shmmax" | "kernel.shmmni" | "kernel.shm_rmid_forced"` y `Sysctls` que comiencen por `"fs.mqueue.*"`

Valores de espacio de nombre de red válidos: `Sysctls` que comiencen con `"net.*"`. En Fargate solo se aceptan los `Sysctls` con espacios de nombre que existen dentro del contenedor.

Todos estos valores son compatibles con Fargate.

`value`

Tipo: cadena

Requerido: no

El valor del parámetro de kernel del espacio de nombres especificado en `namespace`.

Interactivo

`interactive`

Tipo: Booleano

Obligatorio: no

Si este parámetro es `true`, puede implementar aplicaciones en contenedores que requieran la asignación de `stdin` o un `tty`. Este parámetro se corresponde con `OpenStdin` en el comando create-container de docker y la opción `--interactive` con docker run.

El valor predeterminado es `false`.

Pseudo terminal

`pseudoTerminal`

Tipo: Booleano

Obligatorio: no

Cuando este parámetro es `true`, se asigna un TTY. Este parámetro se corresponde con `Tty` en el comando create-container de docker y la opción `--tty` con docker run.

El valor predeterminado es `false`.

Nombre del acelerador de Elastic Inference

El requisito de recursos del acelerador de Elastic Inference para la definición de tareas.

> **Note**
>
> Amazon Elastic Inference (EI) ya no está disponible para los clientes.

Los siguientes parámetros están permitidos en una definición de tarea:

`deviceName`

Tipo: cadena

Obligatorio: sí

Nombre del dispositivo del acelerador de inferencia elástica. También debe hacerse referencia a `deviceName` en una definición de contenedor. Consulte `Elastic Inference accelerator`.

`deviceType`

Tipo: cadena

Obligatorio: sí

El tipo de acelerador de Elastic Inference que se va a utilizar.

Configuración del proxy

`proxyConfiguration`

Tipo: objeto ProxyConfiguration

Obligatorio: no

Los detalles de configuración del proxy App Mesh.

> **Note**
>
> Este parámetro no es compatible con contenedores Windows.

```
"proxyConfiguration": {
    "type": "APPMESH",
    "containerName": "string",
    "properties": [
        {
           "name": "string",
           "value": "string"
        }
    ]
}
```

`type`

Tipo: cadena

Valores válidos: `APPMESH`

Obligatorio: no

El tipo de proxy. El único valor admitido es `APPMESH`.

`containerName`

Tipo: cadena

Obligatorio: sí

El nombre del contenedor que sirve como proxy de App Mesh.

`properties`

Tipo: matriz de objetos KeyValuePair

Obligatorio: no

El conjunto de parámetros de configuración de red para proporcionar el complemento Container Network Interface (CNI), especificado como pares clave-valor.

- `IgnoredUID`: (obligatorio) el ID de usuario (UID) del contenedor proxy tal y como lo define el parámetro `user` en una definición de contenedor. Se utiliza para garantizar que el proxy pasa por alto su propio tráfico. Si se especifica `IgnoredGID`, este campo puede estar vacío.
- `IgnoredGID`: (obligatorio) el ID de grupo (GID) del contenedor proxy tal y como lo define el parámetro `user` en una definición de contenedor. Se utiliza para garantizar que el proxy pasa por alto su propio tráfico. Si se especifica `IgnoredUID`, este campo puede estar vacío.
- `AppPorts`: (obligatorio) la lista de los puertos que utiliza la aplicación. El tráfico de red hacia estos puertos se reenvía a `ProxyIngressPort` y `ProxyEgressPort`.
- `ProxyIngressPort`: (obligatorio) especifica el puerto al que se dirige el tráfico que ingresa a `AppPorts`.
- `ProxyEgressPort`: (obligatorio) especifica el puerto al que se dirige el tráfico que sale de `AppPorts`.
- `EgressIgnoredPorts`: (obligatorio) el tráfico de salida que se dirige a estos puertos especificados se pasa por alto y no se redirige a `ProxyEgressPort`. Puede ser una lista vacía.
- `EgressIgnoredIPs`: (obligatorio) el tráfico de salida que se dirige a estas direcciones IP especificadas se pasa por alto y no se redirige a `ProxyEgressPort`. Puede ser una lista vacía.

`name`

Tipo: cadena

Requerido: no

El nombre del par clave-valor.

`value`

Tipo: cadena

Requerido: no

El valor del par clave-valor.

Volúmenes

Al registrar una definición de tareas, se puede especificar una lista de los volúmenes que se transferirán al daemon de Docker en una instancia de contenedor, que estará disponible para otros contenedores en la misma instancia de contenedor.

A continuación se indican los tipos de volúmenes de datos que se pueden usar:

- Volúmenes de Amazon EBS: proporciona almacenamiento en bloques rentable, duradero y de alto rendimiento para cargas de trabajo en contenedores con uso intensivo de datos. Puede adjuntar un volumen de Amazon EBS por tarea de Amazon ECS cuando ejecute una tarea independiente o cuando cree o actualice un servicio. Se admiten volúmenes de Amazon EBS para tareas de Linux alojadas en Fargate. Para obtener más información, consulte Uso de volúmenes de Amazon EBS con Amazon ECS.
- Volúmenes de Amazon EFS: proporciona almacenamiento de archivos sencillo, escalable y persistente para usarlo con tareas de Amazon ECS. Con Amazon EFS, la capacidad de almacenamiento es elástica. Aumenta y disminuye automáticamente a medida que se agregan o eliminan archivos. Sus aplicaciones disponen del almacenamiento que necesitan, cuando lo necesitan. Se admiten volúmenes de Amazon EFS para tareas que están alojadas en Fargate. Para obtener más información, consulte Uso de volúmenes de Amazon EFS con Amazon ECS.
- Volúmenes de FSx for Windows File Server: proporciona servidores de archivos de Microsoft Windows completamente administrados. Estos servidores de archivos están respaldados por un sistema de archivos de Windows. Cuando utiliza FSx for Windows File Server junto con Amazon ECS, puede aprovisionar sus tareas de Windows con almacenamiento persistente, distribuido, compartido y estático de archivos. Para obtener más información, consulte Uso de volúmenes de FSx para Windows File Server con Amazon ECS.

  Los contenedores Windows de Fargate no admiten esta opción.
- Montajes de unión: un archivo o directorio de la máquina host que se monta en un contenedor. Se admiten volúmenes de host de montaje vinculado cuando se ejecutan tareas. Para utilizar volúmenes de host de montaje vinculado, especifique `host` y un valor de `sourcePath` opcional en su definición de tarea.

Para obtener más información, consulte Opciones de almacenamiento para las tareas de Amazon ECS.

Los siguientes parámetros están permitidos en una definición de contenedor.

`name`

Tipo: cadena

Requerido: no

El nombre del volumen. Se admiten hasta 255 letras (mayúsculas y minúsculas), números, guiones (`-`) y caracteres de subrayado (`_`). Se hace referencia a este nombre en el parámetro `sourceVolume` del objeto `mountPoints` de la definición de contenedor.

`host`

Obligatorio: no

El parámetro `host` se utiliza para vincular el ciclo de vida del montaje de enlace a la instancia de Amazon EC2 del host, en lugar de a la tarea, y donde se almacena. Si el parámetro `host` está vacío, entonces el daemon de Docker asigna una ruta de host a su volumen de datos, pero no se garantiza que los datos persistan después de que los contenedores asociados dejen de funcionar.

Los contenedores de Windows pueden montar directorios completos en la misma unidad que `$env:ProgramData`.

> **Note**
>
> El parámetro `sourcePath` se admite solo cuando se utilizan las tareas que se alojan en instancias de Amazon EC2.

`sourcePath`

Tipo: cadena

Requerido: no

Cuando utilice el parámetro `host`, especifique una `sourcePath` para declarar la ruta de la instancia de Amazon EC2 del host que se presenta al contenedor. Si este parámetro está vacío, el daemon de Docker asigna una ruta de host. Si el parámetro `host` contiene una ubicación de archivos `sourcePath`, el volumen de datos persiste en la ubicación especificada en la instancia de Amazon EC2 del host hasta que la elimine manualmente. Si el valor

`sourcePath` no existe en la instancia de Amazon EC2 del host, el daemon de Docker lo crea. Si la ubicación existe, el contenido de la carpeta de la ruta de origen se exporta.

`configuredAtLaunch`

Tipo: Booleano

Obligatorio: no

Indica si un volumen se puede configurar durante el lanzamiento. Cuando se establece en `true`, puede configurarlo al ejecutar una tarea independiente o al crear o actualizar un servicio. Cuando se establece en `true`, no podrá proporcionar otra configuración de volumen en la definición de la tarea. Este parámetro se debe establecer en `true` para configurar un volumen de Amazon EBS para adjuntarlo a una tarea. Establecer `configuredAtLaunch` en `true` y aplazar la configuración del volumen hasta la fase de lanzamiento permite crear definiciones de tareas que no se limitan a un tipo de volumen o a una configuración de volumen específica. De este modo, la definición de la tarea se puede reutilizar en distintos entornos de ejecución. Para obtener más información, consulte Volúmenes de Amazon EBS.

`dockerVolumeConfiguration`

Type: objeto de DockerVolumeConfiguration

Obligatorio: no

Este parámetro se especifica cuando se usan volúmenes de Docker. Los volúmenes de Docker se admiten solo cuando se ejecutan tareas en instancias de EC2. Los contenedores de Windows admiten solo el uso del controlador `local`. Para utilizar montajes vinculados, especifique `host` en su lugar.

`scope`

Tipo: cadena

Valores válidos: `task` | `shared`

Obligatorio: no

El ámbito del volumen de Docker, que determina su ciclo de vida. Los volúmenes de Docker con un ámbito de `task` se aprovisionan automáticamente cuando se inicia la tarea y se destruyen cuando la tarea se detiene. Los volúmenes de Docker cuyo ámbito es `shared` se conservan una vez detenida la tarea.

`autoprovision`

Tipo: Booleano

Valor predeterminado: `false`

Obligatorio: no

Si este valor es `true`, el volumen de Docker se crea si aún no existe. Este campo se usa solo si `scope` es `shared`. Si el `scope` es `task`, este parámetro debe omitirse.

`driver`

Tipo: cadena

Requerido: no

El controlador del volumen de Docker que se va a usar. El valor de controlador debe coincidir con el nombre del controlador proporcionado por Docker, ya que se utiliza para la colocación de tareas. Si el controlador se instaló mediante la CLI del complemento de Docker, utilice `docker plugin ls` para recuperar el nombre de controlador de la instancia de contenedor. Si el controlador se instaló con otro método, utilice la detección de complementos de Docker para recuperar el nombre del controlador.

`driverOpts`

Tipo: cadena

Requerido: no

Un mapa de las opciones específicas del controlador de Docker que se deben transferir. Este parámetro se corresponde con `DriverOpts` en la sección Crear un volumen de Docker.

`labels`

Tipo: cadena

Requerido: no

Metadatos personalizados que se añaden al volumen de Docker.

`efsVolumeConfiguration`

Tipo: objeto de EFSVolumeConfiguration

Obligatorio: no

Este parámetro se especifica cuando se usan volúmenes de Amazon EFS.

`fileSystemId`

Tipo: cadena

Obligatorio: sí

El ID del sistema de archivos de Amazon EFS que se va a usar.

`rootDirectory`

Tipo: cadena

Obligatorio: no

Directorio del sistema de archivos de Amazon EFS que se va a montar como directorio raíz dentro del host. Si se omite este parámetro, se utilizará la raíz del volumen de Amazon EFS. Si se especifica `/`, se obtiene el mismo efecto que si se omite este parámetro.

> **Important**
>
> Si se especifica un punto de acceso de EFS en `authorizationConfig`, se debe omitir el parámetro del directorio raíz o establecerlo en `/`, lo que aplicará la ruta establecida en el punto de acceso de EFS.

`transitEncryption`

Tipo: cadena

Valores válidos: `ENABLED` | `DISABLED`

Obligatorio: no

Especifica si se habilita el cifrado para los datos en tránsito de Amazon EFS entre el host de Amazon ECS y el servidor de Amazon EFS. Si se utiliza la autorización de IAM en Amazon EFS, el cifrado en tránsito debe estar habilitado. Si se omite este parámetro, se usa el valor predeterminado de `DISABLED`. Para obtener más información, consulte Cifrado de datos en tránsito en la Guía del usuario de Amazon Elastic File System.

`transitEncryptionPort`

Tipo: entero

Obligatorio: no

El puerto que se utilizará al enviar datos cifrados entre el host de Amazon ECS y el servidor de Amazon EFS. Si no se especifica un puerto de cifrado en tránsito, la tarea utilizará la estrategia de selección de puertos que utiliza el ayudante de montaje de Amazon EFS. Para obtener más información, consulte Ayudante de montaje de EFS en la Guía del usuario de Amazon Elastic File System.

`authorizationConfig`

Tipo: objeto de EFSAuthorizationConfig

Obligatorio: no

Los detalles de configuración de autorización en el sistema de archivos de Amazon EFS.

`accessPointId`

Tipo: cadena

Requerido: no

ID de punto de acceso que se va a utilizar. Si se especifica un punto de acceso, el valor del directorio raíz en `efsVolumeConfiguration` se debe omitir o establecer en `/`, lo que aplicará la ruta establecida en el punto de acceso de EFS. Si se utiliza un punto de acceso, el cifrado de tránsito debe estar habilitado en el `EFSVolumeConfiguration`. Para obtener más información, consulte Trabajo con puntos de acceso de Amazon EFS en la Guía del usuario de Amazon Elastic File System.

`iam`

Tipo: cadena

Valores válidos: `ENABLED` | `DISABLED`

Obligatorio: no

Indica si se debe utilizar el rol de IAM de tarea de Amazon ECS definido en una definición de tareas al montar el sistema de archivos de Amazon EFS. Si está habilitado, el cifrado de tránsito debe estar habilitado en el `EFSVolumeConfiguration`. Si se omite este

parámetro, se usa el valor predeterminado de `DISABLED`. Para obtener más información, consulte Roles de IAM para las tareas.

`FSxWindowsFileServerVolumeConfiguration`

Tipo: objeto de FSXWindowsFileServerVolumeConfiguration

Obligatorio: sí

Este parámetro se indica cuando se utiliza el sistema de archivos Amazon FSx para Windows File Server para el almacenamiento de tareas.

`fileSystemId`

Tipo: cadena

Obligatorio: sí

ID del sistema de archivos FSx for Windows File Server que se va a utilizar.

`rootDirectory`

Tipo: cadena

Obligatorio: sí

Directorio dentro del sistema de archivos de FSx for Windows File Server que se va a montar como directorio raíz dentro del host.

`authorizationConfig`

`credentialsParameter`

Tipo: cadena

Obligatorio: sí

Opciones de credenciales de autorización.

opciones:

- Nombre de recurso de Amazon (ARN) del secreto de AWS Secrets Manager.
- ARN de un parámetro de AWS Systems Manager.

`domain`

Tipo: cadena

Obligatorio: sí

Nombre de dominio completo alojado por un directorio de AWS Directory Service for Microsoft Active Directory(AWS Managed Microsoft AD) o un directorio de Active Directory de EC2 con alojamiento propio.

Etiquetas

Cuando se registra una definición de tareas, se pueden especificar etiquetas de metadatos que se aplican a la definición de tareas. Las etiquetas ayudan a clasificar y organizar la definición de tareas. Cada etiqueta consta de una clave y un valor opcional. Los define a los dos. Para obtener más información, consulte Etiquetado de los recursos de Amazon ECS.

> **Important**
>
> No agregue información de identificación personal ni otra información confidencial en las etiquetas. Las etiquetas son accesibles para muchos servicios de AWS, incluida la facturación. Las etiquetas no se diseñaron para utilizarse con información privada o confidencial.

Los siguientes parámetros se admiten en un objeto de etiqueta.

`key`

Tipo: cadena

Requerido: no

Una parte de un par clave-valor que compone una etiqueta. Un clave es una etiqueta general que actúa como una categoría para valores de etiqueta más específicos.

`value`

Tipo: cadena

Requerido: no

La parte opcional de un par clave-valor que compone una etiqueta. Un valor actúa como un descriptor en una categoría de etiquetas (clave).

Otros parámetros de definición de tarea

Los siguientes parámetros de definición de tareas se pueden utilizar cuando se registran definiciones de tareas en la consola de Amazon ECS mediante la opción Configure via JSON (Configurar mediante JSON). Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

Temas

- Almacenamiento efímero
- Modo IPC
- Modo PID
- Inyección de errores

Almacenamiento efímero

`ephemeralStorage`

Tipo: objeto de EphemeralStorage

Obligatorio: no

Cantidad de almacenamiento efímero (en GB) que se va a asignar para la tarea. Este parámetro se utiliza para expandir la cantidad total de almacenamiento efímero disponible, más allá de la cantidad predeterminada, para las tareas que están alojadas en AWS Fargate. Para obtener más información, consulte the section called “Montajes de enlace”.

> **Note**
>
> Este parámetro solo se admite en la versión `1.4.0` de la plataforma o posterior para Linux, o `1.0.0` o posterior para Windows.

Modo IPC

`ipcMode`

Esto no es compatible con las tareas que se ejecutan en Fargate.

Tipo: cadena

Requerido: no

El espacio de nombres de recurso de IPC que usarán los contenedores de la tarea. Los valores válidos son `host`, `task` o `none`. Si se especifica `host`, todos los contenedores que están dentro de las tareas que tienen especificado el modo de IPC `host` en la misma instancia de contenedor comparten los mismos recursos de IPC con la instancia Amazon EC2 del host. Si se especifica `task`, todos los contenedores que están dentro de la tarea especificada comparten los mismos recursos de IPC. Si se especifica `none`, los recursos de IPC dentro de los contenedores de una tarea son privados y no se comparten con otros contenedores en una tarea o en la instancia de contenedor. Si no se especifica ningún valor, el uso compartido del espacio de nombre de recursos de IPC depende de la configuración del daemon de Docker en la instancia de contenedor.

Si se utiliza el modo de IPC `host`, existe un mayor riesgo de exposición de espacio de nombres de IPC no deseada.

Si configura parámetros del kernel del espacio de nombres mediante `systemControls` para los contenedores de la tarea, se aplica lo siguiente a su espacio de nombres de recursos de IPC.

- Para las tareas que utilizan el modo de IPC `host`, no se admiten los `systemControls` relacionados con el espacio de nombres de IPC.
- Para las tareas que utilizan el modo de IPC `task`, los `systemControls` relacionados con el espacio de nombres de IPC se aplican a todos los contenedores de una tarea.

> **Note**
>
> Este parámetro no es compatible con contenedores Windows o tareas con el tipo de lanzamiento Fargate.

Modo PID

`pidMode`

Tipo: cadena

Valores válidos: `host` | `task`

Obligatorio: no

El espacio de nombres del proceso que usarán los contenedores de la tarea. Los valores válidos son `host` o `task`. En los contenedores de Linux, el único valor válido es `task`. Por ejemplo, la supervisión de los archivos sidecar puede necesitar `pidMode` para acceder a información sobre otros contenedores que se ejecutan en la misma tarea.

Si se especifica `task`, todos los contenedores dentro de la tarea especificada comparten el mismo espacio de nombres del proceso.

Si no se especifica ningún valor, el valor predeterminado es un espacio de nombre privado para cada contenedor.

> **Note**
>
> Este parámetro solo se admite para las tareas que están alojadas en AWS Fargate si utilizan la versión de la plataforma `1.4.0` o una posterior (Linux). Este parámetro no es compatible con contenedores de Windows en Fargate.

Inyección de errores

`enableFaultInjection`

Tipo: Booleano

Valores válidos: `true` | `false`

Obligatorio: no

Si este parámetro se establece en `true`, en la carga útil de una tarea, Amazon ECS y Fargate aceptarán solicitudes de inyección de errores de los contenedores de la tarea. Este parámetro está establecido en `false` de forma predeterminada.

Parámetros en la definición de tareas de Amazon ECS para el tipo de lanzamiento de EC2

Las definiciones de tareas se dividen en partes independientes: la familia de tareas, el rol de tarea de AWS Identity and Access Management (IAM), el modo de red, las definiciones de contenedor,

los volúmenes, las restricciones de ubicación de tareas y los tipos de lanzamiento. En una definición de tareas, son necesarias las definiciones de familia y contenedor. En cambio, el rol de tarea, el modo de red, los volúmenes, las restricciones de ubicación de tareas y el tipo de lanzamiento son opcionales.

Puede utilizar estos parámetros en un archivo JSON para configurar la definición de tarea.

A continuación, se muestran descripciones más detalladas de cada uno de los parámetros de definición de tareas para el tipo de lanzamiento EC2.

Familia

`family`

Tipo: cadena

Obligatorio: sí

Cuando se registra una definición de tarea, se le da una familia, que es similar a un nombre para varias versiones de la definición de tarea, especificado con un número de revisión. A la primera definición de tareas que se registra en una familia particular se le da una revisión de 1 y a cualquier definición de tarea registrada después se le da un número de revisión secuencial.

Tipos de lanzamiento

Cuando se registre una definición de tareas, puede especificar un tipo de lanzamiento con respecto al cual Amazon ECS debe validar la definición de tareas. Si la definición de tareas no se valida con respecto a las compatibilidades especificadas, se devuelve una excepción de cliente. Para obtener más información, consulte Tipos de lanzamiento de Amazon ECS.

El parámetro a continuación está permitido en una definición de tareas.

`requiresCompatibilities`

Tipo: matriz de cadenas

Obligatorio: no

Valores válidos: `EC2`

Los tipos de lanzamiento con respecto a los cuales se validó la definición de tareas. Esto inicia una comprobación para garantizar que todos los parámetros utilizados en la definición de tareas cumplan los requisitos del tipo de lanzamiento.

Rol de la tarea

`taskRoleArn`

Tipo: cadena

Requerido: no

Cuando se registra una definición de tareas, se puede proporcionar un rol de tarea para un rol de IAM que permita a los contenedores del permiso de tareas llamar en su nombre a las API de AWS especificadas en sus políticas asociadas. Para obtener más información, consulte Rol de IAM de tarea de Amazon ECS.

Al lanzar la AMI de Windows Server optimizada para Amazon ECS, los roles de IAM para las tareas de Windows requieren que se haya establecido la opción `-EnableTaskIAMRole`. Los contenedores deben ejecutar también código de configuración para utilizar la característica. Para obtener más información, consulte Configuración adicional de las instancias de Amazon EC2 de Windows.

Rol de ejecución de tareas

`executionRoleArn`

Tipo: cadena

Obligatorio: condicional

El nombre de recurso de Amazon (ARN) del rol de ejecución de tarea que concede permiso al agente de contenedor de Amazon ECS para realizar llamadas a la API de AWS en su nombre.

> **Note**
>
> El rol de IAM de ejecución de tareas es necesario en función de los requisitos de la tarea. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

Modo de red

`networkMode`

Tipo: cadena

Requerido: no

El modo de red de Docker que utilizar para los contenedores en la tarea. Para las tareas de Amazon ECS que están alojadas en instancias de Linux de Amazon EC2, los valores válidos son `none`, `bridge`, `awsvpc` y `host`. Si no se especifica ningún modo de red, el modo de red predeterminado es `bridge`. Para las tareas de Amazon ECS alojadas en instancias de Windows de Amazon EC2, los valores válidos son `default` y `awsvpc`. Si no se especifica ningún modo de red, se utiliza el modo de red `default`.

Si el modo de red se establece en `none`, los contenedores de tarea no tienen conectividad externa y no es posible especificar la asignación de puertos en la definición del contenedor.

Si el modo de red es `bridge`, la tarea usa la red virtual integrada de Docker en Linux, la cual se ejecuta dentro de cada instancia de Amazon EC2 que aloja la tarea. La red virtual integrada en Linux usa el controlador de red `bridge` Docker.

Si el modo de red es `host`, la tarea usa la red del host que omite la red virtual integrada de Docker al asignar los puertos del contenedor directamente a la ENI de la instancia de Amazon EC2 que aloja la tarea. Las asignaciones de puertos dinámicas no se pueden usar en este modo de red. Un contenedor de una definición de tarea que use este modo debe especificar un número de `hostPort` específico. Varias tareas no pueden usar el número de puerto de un host. Por lo tanto, no se pueden ejecutar varias tareas de la misma definición de tarea en una instancia de Amazon EC2 única.

> **Important**
>
> Para mayor seguridad, cuando ejecute tareas utilizando el modo de red `host`, no debe ejecutar contenedores con el usuario raíz (UID 0). Como práctica recomendada de seguridad, utilice siempre un usuario que no sea usuario raíz.

Si el modo de red es `awsvpc`, a la tarea se le asigna una interfaz de red elástica y debe especificar `NetworkConfiguration` al crear un servicio o ejecutar una tarea con la definición

de tarea. Para obtener más información, consulte Opciones de redes de tareas de Amazon ECS para el tipo de lanzamiento de EC2.

Si el modo de red es `default`, la tarea usa la red virtual integrada de Docker en Windows, la cual se ejecuta dentro de cada instancia de Amazon EC2 que aloja la tarea. La red virtual integrada en Windows usa el controlador de red `nat` Docker.

Los modos de red `host` y `awsvpc` ofrecen el rendimiento de redes más alto para contenedores dado que estos utilizan la pila de la red de Amazon EC2. Con los modos de red `host` y `awsvpc`, los puertos de contenedor expuestos se asignan directamente al puerto de host correspondiente (para el modo de red `host`) o al puerto de interfaz de red elástica asociado (para el modo de red `awsvpc`). Por este motivo, no se pueden utilizar asignaciones de puerto de host dinámico.

El modo de red permitido depende del sistema operativo de la instancia de EC2 subyacente. Si es Linux, se puede usar cualquier modo de red. Si es Windows, se pueden usar los modos `default` y `awsvpc`.

Plataforma de tiempo de ejecución

`operatingSystemFamily`

Tipo: cadena

Obligatorio: condicional

Predeterminado: LINUX

Cuando se registra una definición de tarea, especifica la familia de sistemas operativos.

Los valores válidos son `LINUX`, `WINDOWS_SERVER_2025_FULL`, `WINDOWS_SERVER_2025_CORE`, `WINDOWS_SERVER_2022_CORE`, `WINDOWS_SERVER_2022_FULL`, `WINDOWS_SERVER_2019_FULL`, `WINDOWS_SERVER_2019_CORE`, `WINDOWS_SERVER_2016_FULL`, `WINDOWS_SERVER_2004_CORE` y `WINDOWS_SERVER_20H2_CORE`.

Todas las definiciones de tareas que se utilizan en un servicio deben tener el mismo valor para este parámetro.

Cuando una definición de tarea forma parte de un servicio, este valor debe coincidir con el valor `platformFamily` de servicio.

`cpuArchitecture`

Tipo: cadena

Obligatorio: condicional

Predeterminado: X86_64

Cuando se registra una definición de tarea, especifica la arquitectura de CPU. Los valores válidos son `X86_64` y `ARM64`.

Todas las definiciones de tareas que se utilizan en un servicio deben tener el mismo valor para este parámetro.

Si tiene tareas de Linux, puede establecer el valor en `ARM64`. Para obtener más información, consulte the section called “Definiciones de tareas para cargas de trabajo de ARM de 64 bits”.

Tamaño de tarea

Cuando se registra una definición de tareas, puede especificar la cantidad total de CPU y memoria usada para la tarea. Es distinto de los valores `cpu` y `memory` en el nivel de definición de contenedor. Para las tareas que están alojadas en instancias de Amazon EC2, estos campos son opcionales.

> **Note**
>
> Los parámetros de CPU y memoria de nivel de tarea se omiten para los contenedores de Windows. Le recomendamos que especifique recursos de nivel de contenedor para los contenedores de Windows.

`cpu`

Tipo: cadena

Obligatorio: condicional

> **Note**
>
> Este parámetro no es compatible con contenedores Windows.

El límite máximo de unidades de CPU que presentar para la tarea. Puede indicar los valores de la CPU en el archivo JSON como una cadena en unidades de CPU o CPU virtuales (vCPU). Por ejemplo, puede especificar un valor de CPU `1024` en unidades de CPU o `1 vCPU` en vCPU. Cuando se registra la definición de tarea, el valor de una CPU virtual se convierte en un entero que indica las unidades de CPU.

Este campo es opcional. Si el clúster no tiene registradas instancias de contenedor con las unidades de CPU solicitadas disponibles, la tarea no funciona. Los valores admitidos se encuentran entre `0.125` y `192` vCPU.

`memory`

Tipo: cadena

Obligatorio: condicional

> **Note**
>
> Este parámetro no es compatible con contenedores Windows.

El límite máximo de memoria para presentar a la tarea. Puede especificar los valores de memoria en la definición de tareas como una cadena en mebibytes (MiB) o gigabytes (GB). Por ejemplo, puede especificar un valor de memoria `3072` en MiB o `3 GB` en GB. Cuando se registra la definición de tarea, el valor de GB se convierte en un entero que indica el número de MiB.

Este campo es opcional y se puede utilizar cualquier valor. Si se especifica un valor de memoria de nivel de tarea, el valor de memoria de nivel de contenedor es opcional. Si el clúster no tiene registradas instancias de contenedor con la memoria solicitada disponible, la tarea no funciona. Puede maximizar la utilización de los recursos proporcionando a las tareas la mayor cantidad de memoria posible para un tipo de instancia determinado. Para obtener más información, consulte Reserva de la memoria de instancias de contenedor de Linux de Amazon ECS.

Definiciones de contenedores

Al registrar una definición de tarea, debe especificar una lista de definiciones de contenedor que se transfieren al daemon de Docker en una instancia de contenedor. Los siguientes parámetros están permitidos en una definición de contenedor.

Temas

- Parámetros de definición de contenedor estándar
- Parámetros de definición de contenedor avanzados
- Otros parámetros de definición de contenedor

Parámetros de definición de contenedor estándar

Los siguientes parámetros de definición de tarea son obligatorios o utilizados en la mayoría de definiciones de contenedor.

Temas

- Nombre
- Imagen
- Memoria
- Mapeos de puertos
- Credenciales de repositorio privado

Nombre

`name`

Tipo: cadena

Obligatorio: sí

El nombre de un contenedor. Se admiten hasta 255 letras (mayúsculas y minúsculas), números, guiones y caracteres de subrayado. Si está vinculando varios contenedores en una definición de tareas, el `name` de un contenedor se puede introducir en los `links` de otro contenedor. Esto sirve para conectar los contenedores.

Imagen

`image`

Tipo: cadena

Obligatorio: sí

La imagen que se utiliza para iniciar un contenedor. Esta cadena se transfiere directamente al daemon de Docker. De manera predeterminada, las imágenes del registro de Docker Hub están disponibles. También puede especificar otros repositorios con `repository-url/image:tag` o `repository-url/image@digest`. Se permiten hasta 255 letras (mayúsculas y minúsculas), números, guiones, caracteres de subrayado, comas, puntos, barras diagonales y signos numéricos. Este parámetro se corresponde con `Image` en el comando create-container de docker y el parámetro `IMAGE` del comando de docker run.

- Cuando se inicia una tarea nueva, el agente de contenedor de Amazon ECS extrae la última versión de la imagen y la etiqueta especificadas para que las utilice el contenedor. Sin embargo, las actualizaciones realizadas posteriormente en un repositorio de imágenes no se propagan a las tareas en ejecución.
- Cuando no se especifica una etiqueta ni un resumen en la ruta de imagen en la definición de la tarea, el agente de contenedor de Amazon ECS extrae la última versión de la imagen especificada.
- Sin embargo, las actualizaciones realizadas posteriormente en un repositorio de imágenes no se propagan a las tareas en ejecución.
- Se admiten las imágenes de registros privados. Para obtener más información, consulte Uso de imágenes de contenedor que no sean de AWS en Amazon ECS.
- Para especificar imágenes de los repositorios de Amazon ECR, se puede utilizar la convención de nomenclatura completa `registry/repository:tag` o `registry/repository@digest` (por ejemplo, `aws_account_id.dkr.ecr.region.amazonaws.com/my-web-app:latest` o `aws_account_id.dkr.ecr.region.amazonaws.com/my-web-app@sha256:94afd1f2e64d908bc90dbca0035a5b567EXAMPLE`).
- Las imágenes de los repositorios oficiales de Docker Hub utilizan un solo nombre (por ejemplo, `ubuntu` o `mongo`).
- Las imágenes de otros repositorios de Docker Hub se clasifican con un nombre de organización (por ejemplo, `amazon/amazon-ecs-agent`).
- Las imágenes de otros repositorios online se cualifican más con un nombre de dominio (por ejemplo, `quay.io/assemblyline/ubuntu`).

`versionConsistency`

Tipo: cadena

Valores válidos: `enabled|disabled`

Obligatorio: no

Especifica si Amazon ECS convertirá la etiqueta de imagen de contenedores proporcionada en la definición de contenedores en un resumen de imágenes. De manera predeterminada, el comportamiento es `enabled`. Si establece `disabled` como valor de un contenedor, Amazon ECS no convertirá la etiqueta de imagen del contenedor en un resumen y utilizará el URI de imagen original especificado en la definición del contenedor para la implementación. Para obtener más información acerca de la conversión de imagen de contenedor, consulte Resolución de imagen de contenedor.

Memoria

`memory`

Tipo: entero

Obligatorio: no

La cantidad (en MiB) de memoria que se presenta al contenedor. Si su contenedor intenta superar la memoria especificada aquí, el contenedor se cancela. La cantidad total de memoria reservada para todos los contenedores dentro de una tarea debe ser menor que el valor `memory` de la tarea, si se especifica. Este parámetro se corresponde con `Memory` en el comando create-container de docker y la opción `--memory` se corresponde con docker run.

Debe especificar un valor de memoria de nivel de tarea o un valor de memoria de nivel de contenedor. Si especifica un valor de `memory` y un valor de `memoryReservation` en el nivel de contenedor, el valor de `memory` debe ser mayor que el valor de `memoryReservation`. Si especifica `memoryReservation`, el valor se resta de los recursos de memoria disponibles para la instancia de contenedor en la que se coloca el contenedor. De lo contrario, se utiliza el valor de `memory`.

El daemon de Docker 20.10.0 o posterior reserva un mínimo de 6 MiB de memoria para un contenedor. Por lo tanto, no debe especificar menos de 6 MiB de memoria para los contenedores.

El daemon de Docker 19.03.13-ce o anterior reserva un mínimo de 4 MiB de memoria para un contenedor. Por lo tanto, no debe especificar menos de 4 MiB de memoria para los contenedores.

> **Note**
>
> Si intenta maximizar la utilización de los recursos proporcionando a las tareas la mayor cantidad de memoria posible para un tipo de instancia determinado, consulte Reserva de la memoria de instancias de contenedor de Linux de Amazon ECS.

`memoryReservation`

Tipo: entero

Obligatorio: no

El límite flexible (en MiB) de memoria que reservar para el contenedor. Cuando la memoria del sistema está en conflicto, Docker intenta mantener la memoria del contenedor en este límite flexible. Sin embargo, el contenedor puede utilizar más memoria cuando sea necesario. El contenedor puede utilizar hasta el límite invariable especificado con el parámetro de `memory` (si procede) o toda la memoria disponible en la instancia de contenedor, lo que ocurra primero. Este parámetro se corresponde con `MemoryReservation` en el comando create-container de docker y la opción `--memory-reservation` se corresponde con docker run.

Si no se especifica un valor de memoria de nivel de tarea, debe especificar un entero distinto de cero para `memory` o `memoryReservation`, o para ambos, en una definición de contenedor. Si especifica ambos, `memory` debe ser mayor que `memoryReservation`. Si especifica `memoryReservation`, el valor se resta de los recursos de memoria disponibles para la instancia de contenedor en la que se coloca el contenedor. De lo contrario, se utiliza el valor de `memory`.

Por ejemplo, supongamos que el contenedor utiliza normalmente 128 MiB de memoria, pero con ráfagas ocasionales de hasta 256 MiB durante períodos de tiempo breves. Puede establecer una `memoryReservation` de 128 MiB y un límite invariable de `memory` de 300 MiB. Esta configuración permite al contenedor reservar solo 128 MiB de memoria de los recursos restantes en la instancia de contenedor. Al mismo tiempo, esta configuración también permite que el contenedor consuma más recursos de memoria en caso de ser necesario.

> **Note**
>
> Este parámetro no es compatible con los contenedores de Windows.

El daemon de Docker 20.10.0 o posterior reserva un mínimo de 6 MiB de memoria para un contenedor. Por lo tanto, no debe especificar menos de 6 MiB de memoria para los contenedores.

El daemon de Docker 19.03.13-ce o anterior reserva un mínimo de 4 MiB de memoria para un contenedor. Por lo tanto, no debe especificar menos de 4 MiB de memoria para los contenedores.

> **Note**
>
> Si intenta maximizar la utilización de los recursos proporcionando a las tareas la mayor cantidad de memoria posible para un tipo de instancia determinado, consulte Reserva de la memoria de instancias de contenedor de Linux de Amazon ECS.

Mapeos de puertos

`portMappings`

Tipo: matriz de objetos

Obligatorio: no

Las asignaciones de puertos exponen los puertos de red del contenedor al mundo exterior, lo que permite a los clientes acceder a su aplicación. También se utilizan para la comunicación entre contenedores dentro de la misma tarea.

Para las definiciones de tareas que utilizan el modo de red `awsvpc`, solo especifique `containerPort`. El `hostPort` siempre se ignora y el puerto del contenedor se asigna automáticamente a un puerto aleatorio con un número alto del host.

Las asignaciones de puertos en Windows utilizan la dirección de puerto de enlace `NetNAT` en lugar de `localhost`. No existe ningún bucle invertido para el mapeo de puertos en Windows, por lo que no puede acceder a un puerto mapeado del contenedor desde el propio host.

La mayoría de los campos de este parámetro (incluidos `containerPort`, `hostPort`, `protocol`) se corresponden con `PortBindings` en el comando create-container de docker y con la opción `--publish` para docker run. Si el modo de red de una definición de tareas se establece en `host`, los puertos de host deben estar sin definir o deben corresponder al puerto de contenedor en la asignación de puerto.

> **Note**
>
> Después de que una tarea alcanza el estado `RUNNING`, las asignaciones manuales y automáticas de puertos de contenedor y de host se pueden ver en las siguientes ubicaciones:
>
> - Consola: la sección Network Bindings (Conexiones de red) de la descripción de un contenedor para una tarea seleccionada.
> - AWS CLI: la sección `networkBindings` de la salida del comando describe-tasks.
> - API: la respuesta de `DescribeTasks`.
> - Metadatos: el punto de conexión de metadatos de la tarea.

`appProtocol`

Tipo: cadena

Requerido: no

El protocolo de aplicación que se utiliza para la asignación de puertos. Este parámetro solo se aplica a Service Connect. Recomendamos que defina este parámetro para que sea coherente con el protocolo que utiliza la aplicación. Si se establece este parámetro, Amazon ECS agrega la administración de la conexión específica del protocolo al proxy de Service Connect. Si se establece este parámetro, Amazon ECS agrega telemetría específica del protocolo en la consola de Amazon ECS y CloudWatch.

Si no establece un valor para este parámetro, se utilizará TCP. Sin embargo, Amazon ECS no agrega telemetría específica del protocolo para TCP.

Para obtener más información, consulte the section called “ Service Connect”.

Valores de protocolo válidos: `"http" | "http2" | "grpc"`

`containerPort`

Tipo: entero

Obligatorio: sí, si se utilizan `portMappings`.

El número de puerto en el contenedor que está vinculado al puerto de host especificado por el usuario o asignado automáticamente.

Para las tareas que utilizan el modo de red `awsvpc`, debe utilizar `containerPort` para especificar los puertos expuestos.

Suponga que utiliza contenedores en una tarea con el tipo de lanzamiento de EC2 y especifica un puerto de contenedor y no un puerto de host. A continuación, el contenedor recibe automáticamente un puerto de host en el rango de puertos efímeros. Para obtener más información, consulte `hostPort`. Las asignaciones de puerto que se asignan automáticamente de esta manera no se contabilizan en la cuota de 100 puertos reservados de una instancia de contenedor.

`containerPortRange`

Tipo: cadena

Requerido: no

El rango de números de puerto en el contenedor que está vinculado al rango de puertos de host asignado de manera dinámica.

Solo puede configurar este parámetro mediante la API `register-task-definition`. La opción está disponible en el parámetro `portMappings`. Para obtener más información, consulte register-task-definition en la Referencia de AWS Command Line Interface.

Las siguientes reglas se aplican al especificar un `containerPortRange`:

- Debe utilizar el modo de red `bridge` o el modo de red `awsvpc`.
- Este parámetro está disponible tanto para sistemas operativos Windows como Linux.
- La instancia de contenedor debe tener al menos la versión 1.67.0 del agente de contenedor y al menos la versión 1.67.0-1 del paquete `ecs-init`
- Puede especificar 100 rangos de puertos como máximo por cada contenedor.
- No especifica un `hostPortRange`. El valor de `hostPortRange` se establece de la siguiente manera:
  - Para los contenedores de una tarea con el modo de red `awsvpc`, `hostPort` se establece en el mismo valor que `containerPort`. Se trata de una estrategia de asignación estática.
  - Para los contenedores en una tarea con el modo de red `bridge`, el agente de Amazon ECS busca los puertos de host abiertos del rango efímero predeterminado y los pasa a un docker para vincularlos a los puertos del contenedor.
- Los valores válidos de `containerPortRange` se encuentran entre 1 y 65 535.

- Un puerto solo puede incluirse en una asignación de puertos por cada contenedor.
- No puede especificar rangos de puertos superpuestos.
- El primer puerto del rango debe ser menor que el último puerto del rango.
- Docker recomienda desactivar el docker-proxy en el archivo de configuración del daemon de Docker cuando haya una gran cantidad de puertos.

  Para más información, consulte Issue #11185 en GitHub.

  Para obtener información sobre cómo desactivar el docker-proxy en el archivo de configuración del daemon de Docker, consulte Daemon de Docker en la Guía para desarrolladores de Amazon ECS.

Puede llamar a DescribeTasks para ver el valor de `hostPortRange`, es decir, los puertos del host que están vinculados a los puertos del contenedor.

Los rangos de puertos no se incluyen en los eventos de tareas de Amazon ECS que se envían a EventBridge. Para obtener más información, consulte the section called “Automaticzación de las respuestas a los errores de Amazon ECS mediante EventBridge”.

`hostPortRange`

Tipo: string

Requerido: no

El rango de números de puerto del host que se usa con el enlace de red. Docker lo asigna y el agente de Amazon ECS lo entrega.

`hostPort`

Tipo: entero

Obligatorio: no

El número de puerto en la instancia de contenedor que reservar para el contenedor.

Puede especificar un puerto de host no reservado para la asignación de puertos de su contenedor. Esto se conoce como asignación estática de puertos de host. O bien, puede omitir el `hostPort` (o configurarlo en `0`) al especificar un `containerPort`. El contenedor recibe automáticamente un puerto en el rango de puertos efímeros del sistema operativo de la instancia de contenedor y la versión de Docker. Esto se conoce como asignación dinámica de puertos de host.

La versión 1.6.0 de Docker y posteriores para el rango de puertos efímeros predeterminado se puede ver en la instancia en `/proc/sys/net/ipv4/ip_local_port_range`. Si este parámetro de kernel no está disponible, se usa el intervalo de puertos efímeros predeterminado, de `49153-65535`. No intente especificar un puerto de host en el rango de puertos efímeros. Esto se debe a que están reservados para la asignación automática. En general, los puertos bajo `32768` están fuera del rango de puertos efímeros.

Los puertos reservados predeterminados son el `22` para SSH; los puertos de Docker `2375` y `2376` y los puertos `51678-51680` del agente de contenedor de Amazon ECS. Cualquier puerto de host especificado por el usuario con anterioridad para una tarea en ejecución también se reserva mientras la tarea está en ejecución. Cuando se detiene una tarea, se libera el puerto del host. Los puertos reservados actuales se muestran en los `remainingResources` de la salida de describe-container-instances. Es posible que una instancia de contenedor tenga hasta 100 puertos reservados por vez, incluidos los puertos reservados predeterminados. Los puertos asignados automáticamente no se contabilizan para la cuota de 100 puertos reservados.

`name`

Tipo: cadena

Obligatorio: no, es obligatorio para configurar Service Connect y VPC Lattice en un servicio

El nombre que se utiliza para la asignación de puertos. Este parámetro solo se aplica a Service Connect y VPC Lattice. Este parámetro es el nombre que se utiliza en la configuración de Service Connect y VPC Lattice de un servicio.

Para obtener más información, consulte Uso de Service Connect para conectar los servicios de Amazon ECS con nombres abreviados.

En el siguiente ejemplo, se utilizan los dos campos obligatorios de Service Connect y VPC Lattice.

```
"portMappings": [
    {
        "name": string,
        "containerPort": integer
    }
]
```

`protocol`

Tipo: cadena

Requerido: no

El protocolo que se utiliza para la asignación de puertos. Los valores válidos son `tcp` y `udp`. El valor predeterminado es `tcp`.

> **Important**
>
> Solo `tcp` es compatible con Service Connect. Recuerde que `tcp` está implícito si este campo no está configurado.

> **Important**
>
> UDP solo se admite en las instancias de contenedor que se lanzaron con la versión 1.2.0 del agente de contenedor de Amazon ECS (como, por ejemplo, la AMI `amzn-ami-2015.03.c-amazon-ecs-optimized`) o una posterior, o con los agentes de contenedor que se han actualizado a la versión 1.3.0 o una posterior. Para actualizar el agente de contenedor a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS.

Utilice la sintaxis a continuación para especificar un puerto de host.

```
"portMappings": [
    {
        "containerPort": integer,
        "hostPort": integer
    }
    ...
]
```

Use la siguiente sintaxis si desea asignar automáticamente un puerto de host.

```
"portMappings": [
    {
```

```
        "containerPort": integer
    }
    ...
]
```

Credenciales de repositorio privado

`repositoryCredentials`

Tipo: objeto de RepositoryCredentials

Obligatorio: no

Las credenciales del repositorio para la autenticación de registros privados.

Para obtener más información, consulte Uso de imágenes de contenedor que no sean de AWS en Amazon ECS.

`credentialsParameter`

Tipo: cadena

Obligatorio: sí, si se utilizan `repositoryCredentials`.

El nombre de recurso de Amazon (ARN) del secreto que contiene las credenciales del repositorio privado.

Para obtener más información, consulte Uso de imágenes de contenedor que no sean de AWS en Amazon ECS.

> **Note**
>
> Cuando se utiliza la API de Amazon ECS, la AWS CLI o los SDK de AWS, si el secreto existe en la misma región que la tarea que se va a lanzar, se puede utilizar el ARN completo o el nombre del secreto. Cuando se utiliza la AWS Management Console, debe especificar el ARN completo del secreto.

A continuación, se incluye un fragmento de definición de tareas que muestra los parámetros necesarios:

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
        }
    }
]
```

Parámetros de definición de contenedor avanzados

Los siguientes parámetros avanzados de definición de contenedor ofrecen capacidades ampliadas para el comando de docker run que se utiliza para lanzar contenedores en las instancias de contenedor de Amazon ECS.

Temas

- Política de reinicio
- Comprobación de estado
- Entorno
- Configuración de red
- Almacenamiento y registro
- Seguridad
- Límites de recursos
- Etiquetas de Docker

Política de reinicio

`restartPolicy`

La política de reinicio del contenedor y los parámetros de configuración asociados. Al configurar una política de reinicio para un contenedor, Amazon ECS puede reiniciar el contenedor sin necesidad de reemplazar la tarea. Para obtener más información, consulte Reinicio de contenedores individuales en tareas de Amazon ECS con políticas de reinicio de contenedores.

`enabled`

Tipo: Booleano

Obligatorio: sí

Especifica si una política de reinicio está habilitada para el contenedor.

`ignoredExitCodes`

Tipo: matriz de enteros

Obligatorio: no

Una lista de códigos de salida que Amazon ECS ignorará y no intentará reiniciar. Puede especificar un máximo de 50 códigos de salida de contenedor. De forma predeterminada, Amazon ECS no ignora ningún código de salida.

`restartAttemptPeriod`

Tipo: entero

Obligatorio: no

La cantidad de tiempo (en segundos) que debe ejecutarse el contenedor antes de que se pueda intentar un reinicio. Un contenedor solo se puede reiniciar una vez cada `restartAttemptPeriod` segundos. Si un contenedor no puede ejecutarse durante este período de tiempo y se cierra antes, no se reiniciará. Puede especificar un mínimo `restartAttemptPeriod` de 60 segundos y un `restartAttemptPeriod` máximo de 1.800 segundos. De forma predeterminada, un contenedor debe ejecutarse durante 300 segundos antes de poder reiniciarse.

Comprobación de estado

`healthCheck`

El parámetro de comprobación de estado del contenedor y los parámetros de configuración asociados para el contenedor. Para obtener más información, consulte Determine el estado de las tareas de Amazon ECS mediante comprobaciones de estado de los contenedores.

`command`

Una matriz de cadenas que representa el comando que ejecuta el contenedor para determinar si está en buen estado. La matriz de cadenas puede comenzar por `CMD` para ejecutar los

argumentos del comando directamente, o por `CMD-SHELL` para ejecutar el comando con el shell predeterminado del contenedor. Si no se especifica ninguno, se utiliza `CMD`.

Al registrar una definición de tarea en la AWS Management Console, utilice una lista de comandos separados por comas. Estos comandos se convierten en una cadena una vez que se cree la definición de tareas. A continuación, se muestra un ejemplo de entrada de comprobación de estado.

```
CMD-SHELL, curl -f http://localhost/ || exit 1
```

Cuando registre una definición de tarea mediante el panel de JSON de la AWS Management Console, la AWS CLI o las API, incluya la lista de comandos entre corchetes. A continuación, se muestra un ejemplo de entrada de comprobación de estado.

```
[ "CMD-SHELL", "curl -f http://localhost/ || exit 1" ]
```

Un código de salida de 0, sin salida `stderr`, indica una ejecución correcta y cualquier código de salida distinto de cero indica un error.

`interval`

El periodo de tiempo (en segundos) entre cada comprobación de estado. Es posible especificar entre 5 y 300 segundos. El valor predeterminado es de 30 segundos.

`timeout`

El periodo de tiempo (en segundos) que hay que esperar para que una comprobación de estado se realice correctamente antes de que se considere un error. Puede especificar entre 2 y 60 segundos. El valor de predeterminado es de 5 segundos.

`retries`

Es el número de veces que se reintenta una comprobación de estado fallida antes de que se considere que el contenedor está en mal estado. Puede especificar entre 1 y 10 reintentos. El valor predeterminado es tres reintentos.

`startPeriod`

El periodo de gracia opcional dentro del cual se puede proporcionar tiempo a los contenedores para el arranque antes de que una comprobación de estado fallida se cuente para el número máximo de reintentos. Es posible especificar entre 0 y 300 segundos. De forma predeterminada, `startPeriod` está deshabilitado.

Si una comprobación de estado se realiza correctamente dentro del periodo indicado en `startPeriod`, se considera que el estado del contenedor es correcto y los errores posteriores se contabilizan al calcular el número máximo de intentos.

Entorno

`cpu`

Tipo: entero

Obligatorio: no

El número de unidades de `cpu` que el agente de contenedor de Amazon ECS reserva para el contenedor. En Linux, este parámetro se corresponde con `CpuShares` en la sección Crear un contenedor.

> **Note**
>
> Puede determinar el número de unidades de CPU disponibles para cada tipo de instancia de Amazon EC2. Para ello, multiplique el número de vCPU listadas para dicho tipo de instancia en la página de detalles sobre instancias de Amazon EC2 por 1024.

Los contenedores de Linux comparten unidades de CPU no asignadas con otros contenedores en la instancia de contenedor en la misma proporción que la cantidad asignada. Por ejemplo, supongamos que ejecuta una tarea de contenedor único en un tipo de instancia de un solo núcleo con 512 unidades de CPU especificadas para dicho contenedor. Además, esa tarea es la única que se ejecuta en la instancia de contenedor. En este ejemplo, el contenedor puede utilizar la cuota completa de 1024 unidades de CPU en un momento dado. Sin embargo, supongamos que lanzó otra copia de la misma tarea en esa instancia de contenedor. Cada tarea tiene garantizado un mínimo de 512 unidades de CPU cuando sea necesario. Del mismo modo, si el otro contenedor no utiliza la CPU restante, cada contenedor puede aumentar el uso de la CPU. Sin embargo, si ambas tareas estuvieran 100 % activas todo el tiempo, están limitadas a 512 unidades de CPU.

En las instancias de contenedor de Linux, el daemon de Docker en la instancia de contenedor utiliza el valor de CPU para calcular las proporciones de cuota de CPU relativas para los contenedores en ejecución. El valor de cuota de CPU válido mínimo que permite el kernel de Linux es 2 y el valor de cuota de CPU válido máximo que permite el kernel de Linux es 262144.

Sin embargo, el parámetro de CPU no es obligatorio y puede utilizar valores de CPU por debajo de 2 y por encima de 262144 en las definiciones de contenedor. Para valores de CPU por debajo de 2 (incluido el valor nulo) y por encima de 262144, el comportamiento varía en función de la versión de agente de contenedor de Amazon ECS:

- Versiones del agente <= 1.1.0: los valores de CPU nulo y cero se pasan a Docker como 0, que Docker convierte a continuación a 1024 cuotas de CPU. Los valores de CPU de uno se transfieren a Docker como uno, que el kernel de Linux convierte a dos cuotas de CPU.
- Versiones del agente >= 1.2.0: nulo, cero y los valores de CPU de uno se transfieren a Docker como dos cuotas de CPU.
- Versiones del agente >= 1.84.0: los valores de CPU superiores a 256 vCPU se transfieren a Docker como 256, lo que equivale a 262144 cuotas de CPU.

En las instancias de contenedor de Windows, la cuota de CPU se aplica como una cuota absoluta. Los contenedores de Windows solo tienen acceso a la cantidad de CPU especificada que se establece en la definición de tareas. Un valor de CPU nulo o cero se pasa a Docker como`0`. A continuación, Windows interpreta este valor como el 1 % de una CPU.

Para ver otros ejemplos, consulte Cómo administra Amazon ECS los recursos de CPU y memoria.

`gpu`

Tipo: objeto ResourceRequirement

Obligatorio: no

El número de `GPU`s físicas que el agente de contenedor de Amazon ECS reserva para el contenedor. El número de GPU reservadas para todos los contenedores de una tarea no debe superar el número de GPU disponibles en la instancia de contenedor en la que se lanza la tarea. Para obtener más información, consulte Definiciones de tareas de Amazon ECS para cargas de trabajo de GPU.

> **Note**
>
> Este parámetro no es compatible con los contenedores de Windows.

`Elastic Inference accelerator`

Tipo: objeto ResourceRequirement

Obligatorio: no

Para el tipo `InferenceAccelerator`, el `value` coincide con el `deviceName` para un `InferenceAccelerator` especificado en una definición de tareas. Para obtener más información, consulte the section called “Nombre del acelerador de Elastic Inference”.

> **Note**
>
> Este parámetro no es compatible con los contenedores de Windows.

`essential`

Tipo: Booleano

Obligatorio: no

Supongamos que el parámetro `essential` de un contenedor se marca como `true` y dicho contenedor falla o se detiene por algún motivo. A continuación, se detienen todos los demás contenedores que forman parte de la tarea. Si el parámetro `essential` de un contenedor se marca como `false`, su error no afecta al resto de los contenedores en una tarea. Si este parámetro se omite, se supone que un contenedor es esencial.

Todas las tareas deben tener al menos un contenedor esencial. Supongamos que tiene una aplicación compuesta por varios contenedores. Agrupe los contenedores que se utilizan para un fin común en componentes y sepárelos en diversas definiciones de tareas. Para obtener más información, consulte Diseño de la arquitectura de su aplicación para Amazon ECS.

```
"essential": true|false
```

`entryPoint`

> **Important**
>
> Las primeras versiones del agente de contenedor de Amazon ECS no tratan correctamente los parámetros `entryPoint`. Si tiene problemas al utilizar `entryPoint` , actualice el agente de contenedor o introduzca los comandos y argumentos como elementos de matriz de `command` en su lugar.

Tipo: matriz de cadenas

Obligatorio: no

El punto de entrada que se transfiere al contenedor.

```
"entryPoint": ["string", ...]
```

`command`

Tipo: matriz de cadenas

Obligatorio: no

El comando que se transfiere al contenedor. Este parámetro se corresponde con `Cmd` en el comando create-container y el parámetro `COMMAND` para ejecutar el docker. Si hay varios argumentos, asegúrese de que cada uno de ellos sea una cadena distinta en la matriz.

```
"command": ["string", ...]
```

`workingDirectory`

Tipo: cadena

Requerido: no

El directorio de trabajo para ejecutar los comandos dentro del contenedor. Este parámetro se corresponde con `WorkingDir` en la sección Crear un contenedor de la API remota de Docker y con la opción `--workdir` de docker run.

```
"workingDirectory": "string"
```

`environmentFiles`

Tipo: matriz de objetos

Obligatorio: no

Una lista de archivos que contienen las variables de entorno que transferir a un contenedor. Este parámetro se corresponde con la opción `--env-file` del comando de docker run.

Cuando se habilita FIPS, los nombres de los buckets que tienen puntos (.) (por ejemplo, amzn-s3-demo-bucket1.name.example) no son compatibles. Tener puntos (.) en el nombre del bucket impide que la tarea se inicie porque el agente no puede extraer el archivo de variables de entorno de Amazon S3.

Esto no está disponible para los contenedores de Windows.

Puede especificar hasta diez archivos de entorno. El archivo debe tener una extensión de archivo `.env`. Cada línea de un archivo de entorno contiene una variable de entorno con el formato `VARIABLE=VALUE`. Las líneas que comienzan por # se tratan como comentarios y se ignoran.

Si hay variables de entorno individuales especificadas en la definición del contenedor, tienen prioridad sobre las variables que contiene un archivo de entorno. Si se especifican varios archivos de entorno que contienen la misma variable, se procesan en orden descendente. Le recomendamos que utilice nombres de variables únicos. Para obtener más información, consulte Transferencia de una variable de entorno individual a un contenedor de Amazon ECS.

`value`

Tipo: cadena

Obligatorio: sí

Nombre de recurso de Amazon (ARN) del objeto Amazon S3 que contiene el archivo de variable de entorno.

`type`

Tipo: cadena

Obligatorio: sí

Tipo de archivo que se utilizará. El único valor admitido es `s3`.

`environment`

Tipo: matriz de objetos

Obligatorio: no

Las variables de entorno a transferir a un contenedor. Este parámetro se corresponde con `Env` en el comando create-container de docker y la opción `--env` con el comando de docker run.

> **Important**
>
> No es recomendable que utilice variables del entorno en texto sin formato para información confidencial, como los datos de las credenciales.

`name`

Tipo: cadena

Obligatorio: sí, si se utiliza `environment`

El nombre de la variable de entorno.

`value`

Tipo: cadena

Obligatorio: sí, si se utiliza `environment`

El valor de la variable de entorno.

```
"environment" : [
    { "name" : "string", "value" : "string" },
    { "name" : "string", "value" : "string" }
]
```

`secrets`

Tipo: matriz de objetos

Obligatorio: no

Un objeto que representa el secreto que se expone en el contenedor. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.

`name`

Tipo: cadena

Obligatorio: sí

El valor que se ha de establecer como la variable de entorno en el contenedor.

`valueFrom`

Tipo: cadena

Obligatorio: sí

El secreto para exponer en el contenedor. Los valores admitidos son el nombre de recurso de Amazon (ARN) completo del secreto de AWS Secrets Manager o el ARN completo del parámetro en el almacén de parámetros de AWS Systems Manager.

> **Note**
>
> Si el parámetro del Almacén de parámetros de Systems Manager o el parámetro de Secrets Manager existe en la misma Región de AWS que la tarea que se va a lanzar, se puede utilizar el ARN completo o el nombre del secreto. Si el parámetro existe en una región distinta, el ARN completo debe especificarse.

```
"secrets": [
    {
        "name": "environment_variable_name",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }
]
```

Configuración de red

`disableNetworking`

Tipo: Booleano

Obligatorio: no

Cuando este parámetro es verdadero, la conexión en red está apagada dentro del contenedor.

> **Note**
>
> Este parámetro no es compatible con los contenedores o las tareas de Windows que utilizan el modo de red `awsvpc`.

El valor predeterminado es `false`.

```
"disableNetworking": true|false
```

`links`

Tipo: matriz de cadenas

Obligatorio: no

El parámetro `link` permite a los contenedores comunicarse entre sí sin la necesidad de mapeos de puerto. Este parámetro solo se admite si el modo de red de una definición de tarea se establece en `bridge`. El constructo `name:internalName` es análogo a `name:alias` en enlaces de Docker. Se admiten hasta 255 letras (mayúsculas y minúsculas), números, guiones y caracteres de subrayado.

> **Note**
>
> Este parámetro no es compatible con los contenedores o las tareas de Windows que utilizan el modo de red `awsvpc`.

> **Important**
>
> Es posible que los contenedores que se colocan en la misma instancia de contenedor puedan comunicarse entre sí sin necesidad de enlaces ni asignaciones de puerto de host. El aislamiento de red en una instancia de contenedor se controla mediante los grupos de seguridad y la configuración de VPC.

```
"links": ["name:internalName", ...]
```

`hostname`

Tipo: cadena

Requerido: no

El nombre de host que utilizar para el contenedor. Este parámetro se asigna a `Hostname` en el comando create-container de docker y la opción `--hostname` con docker run.

> **Note**
>
> Si utiliza el modo de red `awsvpc`, el parámetro `hostname` no se admite.

```
"hostname": "string"
```

`dnsServers`

Tipo: matriz de cadenas

Obligatorio: no

Una lista de servidores DNS que se presentan al contenedor.

> **Note**
>
> Este parámetro no es compatible con los contenedores o las tareas de Windows que utilizan el modo de red `awsvpc`.

```
"dnsServers": ["string", ...]
```

`dnsSearchDomains`

Tipo: matriz de cadenas

Obligatorio: no

Patrón: ^[a-zA-Z0-9-.]{0,253}[a-zA-Z0-9]$

Una lista de dominios de búsqueda DNS que se presentan al contenedor. Este parámetro se corresponde con `DnsSearch` en el comando create-container de docker y la opción `--dns-search` con docker run.

> **Note**
>
> Este parámetro no es compatible con los contenedores o las tareas de Windows que utilizan el modo de red `awsvpc`.

```
"dnsSearchDomains": ["string", ...]
```

`extraHosts`

Tipo: matriz de objetos

Obligatorio: no

Una lista de nombres de host y mapeos de direcciones IP que añadir al archivo `/etc/hosts` en el contenedor.

Este parámetro se corresponde con `ExtraHosts` en el comando create-container de docker y la opción `--add-host` se corresponde con docker run.

> **Note**
>
> Este parámetro no es compatible con los contenedores o las tareas de Windows que utilizan el modo de red `awsvpc`.

```
"extraHosts": [
      {
        "hostname": "string",
        "ipAddress": "string"
      }
      ...
    ]
```

`hostname`

Tipo: cadena

Obligatorio: sí, si se utilizan `extraHosts`.

El nombre de host para utilizar en la entrada `/etc/hosts`.

`ipAddress`

Tipo: cadena

Obligatorio: sí, si se utilizan `extraHosts`.

La dirección IP para utilizar en la entrada `/etc/hosts`.

Almacenamiento y registro

`readonlyRootFilesystem`

Tipo: Booleano

Obligatorio: no

Cuando este parámetro es verdadero, al contenedor se le concede acceso de solo lectura a su sistema de archivos raíz. Este parámetro se corresponde con `ReadonlyRootfs` en el comando create-container de docker y la opción `--read-only` con docker run.

> **Note**
>
> Este parámetro no es compatible con contenedores Windows.

El valor predeterminado es `false`.

```
"readonlyRootFilesystem": true|false
```

`mountPoints`

Tipo: matriz de objetos

Obligatorio: no

Puntos de montaje para los volúmenes de datos del contenedor. Este parámetro asigna a `Volumes` en la API create-container de Docker y la opción `--volume` a docker run.

Los contenedores de Windows pueden montar directorios completos en la misma unidad que `$env:ProgramData`. Los contenedores de Windows no pueden montar directorios en una unidad diferente y los puntos de montaje no se pueden utilizar entre unidades. Debe especificar los puntos de montaje para adjuntar un volumen de Amazon EBS directamente a una tarea de Amazon ECS.

`sourceVolume`

Tipo: cadena

Obligatorio: sí, si se utilizan `mountPoints`.

El nombre del volumen a montar.

`containerPath`

Tipo: cadena

Obligatorio: sí, si se utilizan `mountPoints`.

La ruta del contenedor donde se montará el volumen.

`readOnly`

Tipo: Booleano

Obligatorio: no

Si este valor es `true`, el acceso del contenedor al volumen es de solo lectura. Si este valor es `false`, el contenedor puede escribir en el volumen. El valor predeterminado es `false`.

Para las tareas que ejecutan el sistema operativo Windows, deje el valor predeterminado `false`.

`volumesFrom`

Tipo: matriz de objetos

Obligatorio: no

Volúmenes de datos que montar desde otro contenedor. Este parámetro se corresponde con `VolumesFrom` en el comando create-container de docker y la opción `--volumes-from` se corresponde con docker run.

`sourceContainer`

Tipo: cadena

Obligatorio: sí, si se utiliza `volumesFrom`

El nombre del volumen contenedor desde el que montar los volúmenes.

`readOnly`

Tipo: Booleano

Obligatorio: no

Si este valor es `true`, el acceso del contenedor al volumen es de solo lectura. Si este valor es `false`, el contenedor puede escribir en el volumen. El valor predeterminado es `false`.

```
"volumesFrom": [
                {
                  "sourceContainer": "string",
                  "readOnly": true|false
                }
              ]
```

`logConfiguration`

Tipo: objeto LogConfiguration

Obligatorio: no

La especificación de configuración de registros para el contenedor.

Para obtener ejemplos de definiciones de tareas que utilizan una configuración de registro, consulte Ejemplo de definiciones de tareas de Amazon ECS.

Este parámetro se corresponde con `LogConfig` en el comando create-container de docker y la opción `--log-driver` se corresponde con docker run. De forma predeterminada, los contenedores utilizan el mismo controlador de registro que el daemon de Docker. No obstante, el contenedor podría utilizar un controlador de registro diferente al daemon de Docker especificando un controlador de registro con este parámetro en la definición de contenedor. Para utilizar un controlador de registro distinto para un contenedor, el sistema de registro se debe configurar correctamente en la instancia de contenedor (o en un servidor de registro distinto para opciones de registro remotas).

Tenga cuenta lo siguiente al especificar una configuración de registros para los contenedores:

- Amazon ECS admite un subconjunto de controladores de registro disponibles para el daemon de Docker. Es posible que haya controladores de registro adicionales en versiones futuras del agente de contenedor de Amazon ECS.
- Este parámetro requiere la versión 1.18 o posterior de la API remota de Docker en la instancia de contenedor.
- El agente de contenedor de Amazon ECS que se ejecuta en una instancia de contenedor debe registrar los controladores de registro disponibles en dicha instancia con la variable de entorno `ECS_AVAILABLE_LOGGING_DRIVERS` antes de que los contenedores situados en

dicha instancia puedan utilizar estas opciones de configuración de registro. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS.

```
"logConfiguration": {
      "logDriver": "awslogs","fluentd","gelf","json-
file","journald","splunk","syslog","awsfirelens",
      "options": {"string": "string"
        ...},
 "secretOptions": [{
  "name": "string",
  "valueFrom": "string"
 }]
}
```

`logDriver`

Tipo: cadena

Valores válidos: `"awslogs","fluentd","gelf","json-file","journald","splunk","syslog","awsfirelens"`

Obligatorio: sí, si se utiliza `logConfiguration`

El controlador de registro que utilizar para el contenedor. De forma predeterminada, los valores válidos mostrados anteriormente son controladores de registro con los que el agente de contenedor de Amazon ECS se puede comunicar.

Los controladores de registro admitidos son `awslogs`, `fluentd`, `gelf`, `json-file`, `journald`, `syslog`, `splunk` y `awsfirelens`.

Para obtener más información acerca del uso del controlador de registro `awslogs` en las definiciones de tareas para enviar los registros de contenedor a CloudWatch Logs, consulte Envío de registros de Amazon ECS a CloudWatch .

Para obtener más información sobre el uso del controlador del registro `awsfirelens`, consulte Envío de registros personalizados.

> **Note**
>
> Si tiene un controlador personalizado que no figura en la lista, puede adaptar el proyecto del agente de contenedor de Amazon ECS que está disponible en GitHub

y personalizarlo para que funcione con dicho controlador. Le recomendamos enviar solicitudes de inserción para los cambios que desea que incluyamos. No obstante, actualmente no admitimos la ejecución de copias modificadas de este software.

Este parámetro requiere la versión 1.18 de la API remota de Docker o superior en su instancia de contenedor.

`options`

Tipo: mapa de cadena a cadena

Obligatorio: no

Las opciones de configuración de asignación clave/valor para enviar al controlador de registro.

Las opciones que puede especificar dependen del controlador de registro. Algunas de las opciones que puede especificar cuando utiliza el router `awslogs` para enrutar los registros a Amazon CloudWatch son las siguientes:

`awslogs-create-group`

Obligatorio: no

Especifique si desea que el grupo de registro se cree automáticamente. Si no se especifica esta opción, el valor predeterminado es `false`.

> **Note**
>
> Su política de IAM debe incluir el permiso `logs:CreateLogGroup` antes de intentar utilizar `awslogs-create-group`.

`awslogs-region`

Obligatorio: sí

Especifique la Región de AWS a la que el controlador de registro `awslogs` enviará los registros de Docker. Puede elegir enviar todos los registros desde clústeres de diferentes regiones a una única región de CloudWatch Logs. Esto es para que todos sean visibles en una sola ubicación. De lo contrario, puede separarlos por región para obtener un mayor

grado de detalle. Asegúrese de que el grupo de registro especificado exista en la región que especifique con esta opción.

`awslogs-group`

Obligatorio: sí

Asegúrese de especificar un grupo de registro al que el controlador de registros `awslogs` envíe sus flujos de registros.

`awslogs-stream-prefix`

Obligatorio: opcional

Utilice esta opción `awslogs-stream-prefix` para asociar un flujo de registro con el prefijo especificado, el nombre del contenedor y el ID de la tarea de Amazon ECS a la que pertenece el contenedor. Si especifica un prefijo con esta opción, el flujo de registro adopta el siguiente formato.

```
prefix-name/container-name/ecs-task-id
```

Si no especifica un prefijo con esta opción, el flujo de registro se nombra según el ID del contenedor que asigna el daemon de Docker en la instancia de contenedor. Dado que es difícil realizar un seguimiento de los registros hasta el contenedor que los envió solo con el ID de contenedor de Docker (que solo está disponible en la instancia de contenedor), le recomendamos que especifique un prefijo con esta opción.

En el caso de los servicios de Amazon ECS, puede utilizar el nombre del servicio como prefijo. De este modo, puede realizar un seguimiento de flujos de registros hasta el servicio al que pertenece el contenedor, el nombre del contenedor que los envió y el ID de la tarea a la que pertenece el contenedor.

Debe especificar un prefijo de flujo para los registros para que aparezcan en el panel de registros al utilizar la consola de Amazon ECS.

`awslogs-datetime-format`

Obligatorio: no

Esta opción define un patrón de inicio de varias líneas en formato `strftime` de Python. Un mensaje de registro consta de una línea que coincide con el patrón y de líneas

siguientes que no coinciden con el patrón. La línea coincidente es el delimitador entre los mensajes de registro.

Un ejemplo de caso de uso de este formato es para analizar la salida como un volcado de pila, que de lo contrario podría registrarse en varias entradas. El patrón correcto permite capturarla en una sola entrada.

Para obtener más información, consulte awslogs-datetime-format.

No puede configurar las opciones `awslogs-datetime-format` y `awslogs-multiline-pattern` a la vez.

> **Note**
>
> El registro de varias líneas realiza análisis de las expresiones regulares y correspondencia de todos los mensajes de registro. Esto puede tener un impacto negativo sobre el rendimiento de registro.

`awslogs-multiline-pattern`

Obligatorio: no

Esta opción define un patrón de inicio de varias líneas que utiliza una expresión regular. Un mensaje de registro consta de una línea que coincide con el patrón y de líneas siguientes que no coinciden con el patrón. La línea coincidente es el delimitador entre los mensajes de registro.

Para obtener más información, consulte awslogs-multiline-pattern.

Esta opción se pasa por alto si también se ha configurado `awslogs-datetime-format`.

No puede configurar las opciones `awslogs-datetime-format` y `awslogs-multiline-pattern` a la vez.

> **Note**
>
> El registro de varias líneas realiza análisis de las expresiones regulares y correspondencia de todos los mensajes de registro. Esto puede tener un impacto negativo sobre el rendimiento de registro.

Las siguientes opciones se aplican a todos los controladores de registro compatibles.

`mode`

Obligatorio: no

Valores válidos: `non-blocking` | `blocking`

Esta opción define el modo de entrega de los mensajes de registro del contenedor al controlador de registro especificado con `logDriver`. El modo de entrega que elige afecta la disponibilidad de las aplicaciones cuando se interrumpe el flujo de registros desde el contenedor.

Si utiliza el modo `blocking` y se interrumpe el flujo de registros, se bloquearán las llamadas desde el código del contenedor para escribir en los flujos `stdout` y `stderr`. Como resultado, el hilo del registro de la aplicación se bloqueará. Esto puede provocar que la aplicación deje de responder y que se produzca un error en la comprobación del estado del contenedor.

Si utiliza el modo `non-blocking`, los registros del contenedor se almacenan en un búfer intermedio en memoria configurado con la opción. `max-buffer-size` Esto evita que la aplicación deje de responder cuando los registros no se puedan enviar. Le recomendamos que utilice este modo si quiere garantizar la disponibilidad del servicio y si no hay problema con la pérdida de registros. Para obtener más información, consulte Evitar la pérdida de registros con el modo sin bloqueo en el controlador `awslogs` de registros del contenedor.

Puede establecer un `mode` predeterminado para todos los contenedores de una Región de AWS específica mediante la configuración de la cuenta de `defaultLogDriverMode`. Si no especifica la opción `mode` en `logConfiguration` ni configura los ajustes de la cuenta, Amazon ECS utilizará el modo `non-blocking` de manera predeterminada. Para obtener más información acerca de la configuración de la cuenta, consulte Modo de controlador de registro predeterminado.

Cuando se utiliza el modo `non-blocking`, la opción de registro `max-buffer-size` controla el tamaño del búfer que se utiliza para el almacenamiento intermedio de mensajes. Asegúrese de especificar un tamaño de búfer adecuado en función de su aplicación. La cantidad total de memoria asignada a nivel de tarea debe ser superior a la cantidad de memoria asignada a todos los contenedores, además del búfer de memoria del controlador de registro.

> **Note**
>
> El 25 de junio de 2025, Amazon ECS cambió el modo de controlador de registro predeterminado de `blocking` a `non-blocking` a fin de priorizar la disponibilidad de las tareas sobre el registro. Para seguir utilizando el modo `blocking` después de este cambio, realice una de las siguientes acciones:
>
> - Establezca la opción `mode` en la `logConfiguration` de su definición de contenedor como `blocking`.
> - Establezca el ajuste de cuenta `defaultLogDriverMode` en `blocking`.

`max-buffer-size`

Obligatorio: no

Valor predeterminado: `10 m`

Cuando se utiliza el modo `non-blocking`, la opción de registro `max-buffer-size` controla el tamaño del búfer que se utiliza para el almacenamiento intermedio de mensajes. Asegúrese de especificar un tamaño de búfer adecuado en función de su aplicación. Cuando el búfer se llene, no se podrán almacenar más registros. Los registros que no se pueden almacenar se pierden.

Para enrutar los registros mediante el enrutador de registros `splunk`, debe especificar un `splunk-token` y un `splunk-url`.

Cuando utiliza el enrutador de registros `awsfirelens` para enrutar a un destino de Servicio de AWS o AWS Partner Network para el almacenamiento y análisis de registros, puede configurar la opción `log-driver-buffer-limit` hasta el límite del número de eventos almacenados en búfer en la memoria antes de enviarlos al contenedor del enrutador de registros. Puede ayudar a resolver el posible problema de pérdida de registros porque un alto rendimiento podría provocar que se quede sin memoria para el búfer dentro de Docker. Para obtener más información, consulte the section called “Configuración de los registros para conseguir un alto rendimiento”.

Otras opciones que puede especificar cuando se utiliza `awsfirelens` para enrutar los registros dependen del destino. Al exportar registros a Amazon Data Firehose, puede especificar la Región de AWS con `region` y el nombre del flujo de registros con `delivery_stream`.

Al exportar registros a Amazon Kinesis Data Streams, puede especificar una Región de AWS con `region` y un nombre de flujo de datos con `stream`.

Al exportar registros a Amazon OpenSearch Service, puede especificar opciones como `Name`, `Host` (punto final de OpenSearch Service sin protocolo), `Port`, `Index`, `Type`, `Aws_auth`, `Aws_region`, `Suppress_Type_Name` y `tls`.

Al exportar registros a Amazon S3, puede especificar el bucket mediante la opción `bucket`. También puede especificar `region`, `total_file_size`, `upload_timeout` y `use_put_object` como opciones.

Este parámetro requiere la versión 1.19 de la API remota de Docker o superior en su instancia de contenedor.

`secretOptions`

Tipo: matriz de objetos

Obligatorio: no

Un objeto que representa el secreto que transferir a la configuración de registro. Los secretos utilizados en la configuración de registros pueden incluir un token de autenticación, un certificado o una clave de cifrado. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.

`name`

Tipo: cadena

Obligatorio: sí

El valor que se ha de establecer como la variable de entorno en el contenedor.

`valueFrom`

Tipo: cadena

Obligatorio: sí

El secreto que exponer a la configuración de registro del contenedor.

```
"logConfiguration": {
 "logDriver": "splunk",
```

```
  "options": {
   "splunk-url": "https://cloud.splunk.com:8080",
   "splunk-token": "...",
   "tag": "...",
   ...
  },
  "secretOptions": [{
   "name": "splunk-token",
   "valueFrom": "/ecs/logconfig/splunkcred"
  }]
 }
```

`firelensConfiguration`

Tipo: objeto FirelensConfiguration

Obligatorio: no

La configuración de FireLens para el contenedor. Esto se utiliza para especificar y configurar un enrutador de registro para registros de contenedores. Para obtener más información, consulte Envío de registros de Amazon ECS a un servicio de AWS o AWS Partner.

```
{
    "firelensConfiguration": {
        "type": "fluentd",
        "options": {
            "KeyName": ""
        }
    }
}
```

`options`

Tipo: mapa de cadena a cadena

Obligatorio: no

Las opciones de asignación de clave/valor que se deben usar al configurar el enrutador de registros. Este campo es opcional y se puede utilizar para especificar un archivo de configuración personalizado o agregar metadatos adicionales, como la tarea, la definición de tareas, el clúster y detalles de la instancia de contenedor al evento de registro. Si se especifica, la sintaxis que se va a utilizar es `"options":{"enable-ecs-log-`

`metadata":"true|false","config-file-type:"s3|file","config-file-value":"arn:aws:s3:::amzn-s3-demo-bucket/fluent.conf|filepath"}`. Para obtener más información, consulte Definición de tareas de Amazon ECS de ejemplo: enrutar registros a FireLens.

`type`

Tipo: cadena

Obligatorio: sí

El router de registros que se va a utilizar. Los valores válidos son `fluentd` o `fluentbit`.

Seguridad

Para obtener más información sobre la seguridad del contenedor, consulte Prácticas recomendadas de seguridad para las tareas y contenedores de Amazon ECS.

`credentialSpecs`

Tipo: matriz de cadenas

Obligatorio: no

Una lista de los ARN de SSM o Amazon S3 en un archivo de especificaciones de credenciales (`CredSpec`) que configura el contenedor para la autenticación de Active Directory. Le recomendamos que utilice este parámetro en lugar de `dockerSecurityOptions`. El número máximo de ARN es 1.

Hay dos formatos para cada ARN.

Especificación de credenciales sin dominio: MyARN

Utiliza `credentialspecdomainless:MyARN` para proporcionar a la `CredSpec` una sección adicional para un secreto en Secrets Manager. Proporciona las credenciales de inicio de sesión al dominio en el secreto.

Cada tarea que se ejecute en cualquier instancia de contenedor puede unirse a diferentes dominios.

Puede utilizar este formato sin unir la instancia de contenedor a un dominio.

Especificación de credenciales: MyARN

Utiliza `credentialspec:MyARN` para proporcionar una `CredSpec` para un solo dominio.

Debe unir la instancia de contenedor al dominio antes de iniciar cualquier tarea que utilice esta definición de tarea.

En ambos formatos, sustituya `MyARN` por el ARN en SSM o Amazon S3.

La `credspec` debe proporcionar un ARN en Secrets Manager para un secreto que contenga el nombre de usuario, la contraseña y el dominio para conectarse. Para mayor seguridad, la instancia no está unida al dominio para la autenticación sin dominio. Las demás aplicaciones de la instancia no pueden utilizar las credenciales sin dominio. Puede utilizar este parámetro para ejecutar tareas en la misma instancia, incluso si las tareas necesitan unirse a dominios diferentes. Para obtener más información, consulte Uso de gMSA para contenedores de Windows y Uso de gMSA para contenedores de Linux.

`privileged`

Tipo: Booleano

Obligatorio: no

Cuando este parámetro es verdadero, al contenedor se le conceden privilegios elevados en la instancia de contenedor de host, similares a los de un usuario `root`. Recomendamos no utilizar contenedores con `privileged`. En la mayoría de los casos, puede especificar los privilegios exactos que necesita mediante los parámetros específicos en lugar de usar `privileged`.

Este parámetro se corresponde con `Privileged` en el comando create-container de docker y la opción `--privileged` con docker run.

> **Note**
>
> Este parámetro no es compatible con contenedores Windows o tareas con el tipo de lanzamiento Fargate.

El valor predeterminado es `false`.

```
"privileged": true|false
```

`user`

Tipo: cadena

Requerido: no

El usuario que se utiliza dentro del contenedor. Este parámetro se corresponde con `User` en el comando create-container de docker y la opción `--user` con docker run.

> **Important**
>
> Al ejecutar tareas que utilizan el modo de red de `host`, no debe ejecutar contenedores con el usuario raíz (UID 0). Como práctica recomendada de seguridad, utilice siempre un usuario que no sea usuario raíz.

Puede especificar el elemento `user` utilizando los siguientes formatos. Si se especifica un UID o GID, debe especificarlo como número entero positivo.

- `user`
- `user:group`
- `uid`
- `uid:gid`
- `user:gid`
- `uid:group`

> **Note**
>
> Este parámetro no es compatible con contenedores Windows.

```
"user": "string"
```

`dockerSecurityOptions`

Tipo: matriz de cadenas

Valores válidos: “no-new-privileges” | “apparmor:PROFILE” | “label:*value*” | “credentialspec:*CredentialSpecFilePath*”

Obligatorio: no

Una lista de cadenas para proporcionar una configuración personalizada para varios sistemas de seguridad.

Para las tareas de Linux, este parámetro se puede utilizar para hacer referencia a etiquetas personalizadas para SELinux y sistemas de seguridad de varios niveles de AppArmor .

Este parámetro se puede utilizar para hacer referencia a un archivo de especificación de credenciales que configure un contenedor para la autenticación de Active Directory. Para obtener más información, consulte Obtenga información sobre cómo utilizar gMSA para contenedores de EC2 para Windows en Amazon ECS. y Uso de gMSA para contenedores de EC2 Linux en Amazon ECS.

Este parámetro se corresponde con `SecurityOpt` en el comando create-container de docker y la opción `--security-opt` se corresponde con docker run.

```
"dockerSecurityOptions": ["string", ...]
```

> **Note**
>
> El agente de contenedor de Amazon ECS que se ejecuta en una instancia de contenedor se debe registrar con las variables de entorno `ECS_SELINUX_CAPABLE=true` o `ECS_APPARMOR_CAPABLE=true` antes de que los contenedores situados en dicha instancia puedan utilizar estas opciones de seguridad. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS.

Límites de recursos

`ulimits`

Tipo: matriz de objetos

Obligatorio: no

Lista de valores `ulimit` a definir para un contenedor. Este valor sobrescribe la configuración predeterminada de la cuota de recursos para el sistema operativo. Este parámetro se corresponde con `Ulimits` en el comando create-container de docker y la opción `--ulimit` se corresponde con docker run.

Este parámetro requiere la versión 1.18 de la API remota de Docker o superior en su instancia de contenedor.

> **Note**
>
> Este parámetro no es compatible con contenedores Windows.

```
"ulimits": [
      {
        "name":
 "core"|"cpu"|"data"|"fsize"|"locks"|"memlock"|"msgqueue"|"nice"|"nofile"|"nproc"|"rss"|"rtp
        "softLimit": integer,
        "hardLimit": integer
      }
      ...
    ]
```

`name`

Tipo: cadena

Valores válidos: `"core" | "cpu" | "data" | "fsize" | "locks" | "memlock" | "msgqueue" | "nice" | "nofile" | "nproc" | "rss" | "rtprio" | "rttime" | "sigpending" | "stack"`

Obligatorio: sí, si se utilizan `ulimits`.

El valor `type` de `ulimit`.

`hardLimit`

Tipo: entero

Obligatorio: sí, si se utilizan `ulimits`.

El límite máximo para el tipo de `ulimit`. El valor se puede especificar en bytes, segundos o como un conteo, según el `type` del `ulimit`.

`softLimit`

Tipo: entero

Obligatorio: sí, si se utilizan `ulimits`.

El límite flexible para el tipo de `ulimit`. El valor se puede especificar en bytes, segundos o como un conteo, según el `type` del `ulimit`.

Etiquetas de Docker

`dockerLabels`

Tipo: mapa de cadena a cadena

Obligatorio: no

Un mapa de clave/valor de etiquetas que agregar al contenedor. Este parámetro se corresponde con `Labels` en el comando create-container de docker y la opción `--label` se corresponde con docker run.

Este parámetro requiere la versión 1.18 de la API remota de Docker o superior en su instancia de contenedor.

```
"dockerLabels": {"string": "string"
      ...}
```

Otros parámetros de definición de contenedor

Los siguientes parámetros de definición de contenedor se pueden utilizar al registrar definiciones de tareas en la consola de Amazon ECS mediante la opción Configure via JSON (Configurar mediante JSON). Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

Temas

- Parámetros de Linux
- Dependencia de contenedor
- Tiempos de espera de contenedor
- Controles del sistema
- Interactivo
- Pseudo terminal

Parámetros de Linux

`linuxParameters`

Tipo: objeto LinuxParameters

Obligatorio: no

Opciones específicas de Linux que se aplican al contenedor, como KernelCapabilities.

> **Note**
>
> Este parámetro no es compatible con los contenedores de Windows.

```
"linuxParameters": {
      "capabilities": {
        "add": ["string", ...],
        "drop": ["string", ...]
        }
      }
```

`capabilities`

Tipo: objeto KernelCapabilities

Obligatorio: no

Las capacidades de Linux para el contenedor que se agregan a la configuración predeterminada proporcionada por Docker o se eliminan de ella. Para obtener más información sobre estas capacidades de Linux, consulte la página del manual de Linux sobre capacidades(7).

`add`

Tipo: matriz de cadenas

Valores válidos: `"ALL" | "AUDIT_CONTROL" | "AUDIT_READ" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE" | "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" |`

"NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW" | "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" | "SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT" | "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" | "SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"

Obligatorio: no

Las capacidades de Linux para el contenedor que se deben añadir a la configuración predeterminada proporcionada por Docker. Este parámetro se corresponde con `CapAdd` en el comando create-container de docker y la opción `--cap-add` se corresponde con docker run.

`add`

Tipo: matriz de cadenas

Valores válidos: `"SYS_PTRACE"`

Obligatorio: no

Las capacidades de Linux para el contenedor que se deben agregar a la configuración predeterminada proporcionada por Docker. Este parámetro se corresponde con `CapAdd` en el comando create-container de docker y la opción `--cap-add` se corresponde con docker run.

`drop`

Tipo: matriz de cadenas

Valores válidos: "ALL" | "AUDIT_CONTROL" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE" | "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" | "NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW" | "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" | "SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT" | "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" | "SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"

Obligatorio: no

Las capacidades de Linux para el contenedor que se deben eliminar de la configuración predeterminada proporcionada por Docker. Este parámetro se corresponde con `CapDrop` en el comando create-container de docker y la opción `--cap-drop` con docker run.

`devices`

Cualquier dispositivo host que exponer en el contenedor. Este parámetro se corresponde con `Devices` en el comando create-container de docker y la opción `--device` con docker run.

Tipo: matriz de objetos Dispositivo

Obligatorio: no

`hostPath`

La ruta del dispositivo de la instancia de contenedor del host.

Tipo: cadena

Obligatorio: sí

`containerPath`

La ruta dentro del contenedor en la cual exponer el dispositivo host.

Tipo: cadena

Requerido: no

`permissions`

Los permisos explícitos que proporcionar al contenedor para el dispositivo. De forma predeterminada, el contenedor tiene permisos para `read`, `write` y `mknod` en el dispositivo.

Tipo: matriz de cadenas

Valores válidos: `read` | `write` | `mknod`

`initProcessEnabled`

Ejecute un proceso `init` dentro del contenedor que reenvíe señales y aproveche procesos. Este parámetro se corresponde con la opción `--init` con docker run.

Este parámetro requiere la versión 1.25 de la API remota de Docker o superior en su instancia de contenedor.

`maxSwap`

La cantidad total de memoria de intercambio (en MiB) que puede utilizar un contenedor. Este parámetro se traduce en la opción `--memory-swap` de docker run donde el valor es la suma de la memoria del contenedor más el valor de `maxSwap`.

Si se especifica un valor `maxSwap` para `0`, el contenedor no utiliza el intercambio. Los valores aceptados son `0` o cualquier entero positivo. Si se omite el parámetro `maxSwap`, el contenedor utiliza la configuración de intercambio de la instancia de contenedor en la que se está ejecutando. Debe establecerse un valor de `maxSwap` para el parámetro `swappiness`.

`sharedMemorySize`

El valor del tamaño (en MiB) del volumen `/dev/shm`. Este parámetro se corresponde con la opción `--shm-size` con docker run.

Tipo: entero

`swappiness`

Puede utilizar este parámetro para ajustar el comportamiento de intercambio de memoria de un contenedor. Un valor `swappiness` de `0` evita que se produzca el intercambio a menos que sea necesario. Un valor `swappiness` de `100` hace que las páginas se intercambien con frecuencia. Los valores aceptados son números enteros comprendidos entre `0` y `100`. Si no especifica un valor, se utiliza el valor predeterminado de `60`. Además, si no se especifica un valor para `maxSwap`, este parámetro se omite. Este parámetro se corresponde con la opción `--memory-swappiness` con docker run.

> **Note**
>
> Si utiliza tareas en Amazon Linux 2023, no se admite el parámetro `swappiness`.

`tmpfs`

La ruta del contenedor, las opciones de montaje y el tamaño máximo (en MiB) del montaje tmpfs. Este parámetro se corresponde con la opción `--tmpfs` con docker run.

Tipo: matriz de objetos Tmpfs

Obligatorio: no

`containerPath`

La ruta de archivo absoluta en la que se montará el volumen tmpfs.

Tipo: cadena

Obligatorio: sí

`mountOptions`

La lista de opciones de montaje del volumen tmpfs.

Tipo: matriz de cadenas

Obligatorio: no

Valores válidos: `"defaults" | "ro" | "rw" | "suid" | "nosuid" | "dev" | "nodev" | "exec" | "noexec" | "sync" | "async" | "dirsync" | "remount" | "mand" | "nomand" | "atime" | "noatime" | "diratime" | "nodiratime" | "bind" | "rbind" | "unbindable" | "runbindable" | "private" | "rprivate" | "shared" | "rshared" | "slave" | "rslave" | "relatime" | "norelatime" | "strictatime" | "nostrictatime" | "mode" | "uid" | "gid" | "nr_inodes" | "nr_blocks" | "mpol"`

`size`

El tamaño máximo (en MiB) del volumen tmpfs.

Tipo: entero

Obligatorio: sí

Dependencia de contenedor

`dependsOn`

Tipo: matriz de objetos ContainerDependency

Obligatorio: no

Las dependencias definidas para el inicio y apagado del contenedor. Un contenedor puede contener varias dependencias. Cuando una dependencia se define para el inicio del contenedor, se invierte para el apagado del contenedor. Para ver un ejemplo, consulta Dependencia de contenedor.

> **Note**
>
> Si un contenedor no cumple una restricción de dependencia o agota el tiempo de espera antes de cumplir la restricción, Amazon ECS no adelanta los contenedores dependientes a su siguiente estado.

Las instancias requieren al menos la versión `1.26.0` del agente de contenedor para activar las dependencias de contenedor. No obstante, recomendamos utilizar la versión del agente de contenedor más reciente. Para obtener información sobre la comprobación de la versión del agente y la actualización a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS. Si utiliza una AMI de Amazon Linux optimizada para Amazon ECS, la instancia necesita al menos la versión `1.26.0-1` del paquete `ecs-init`. Si las instancias de contenedor se lanzan desde la versión `20190301` o posterior, contienen las versiones requeridas del agente de contenedor y `ecs-init`. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.

```
"dependsOn": [
    {
        "containerName": "string",
        "condition": "string"
    }
]
```

`containerName`

Tipo: cadena

Obligatorio: sí

El nombre del contenedor que debe cumplir la condición especificada.

`condition`

Tipo: cadena

Obligatorio: sí

La condición de dependencia del contenedor. Están disponibles las siguientes condiciones y su comportamiento:

- `START`: esta condición simula el comportamiento de enlaces y volúmenes en la actualidad. La condición valida que un contenedor dependiente se inicia antes de permitir que se inicien otros contenedores.
- `COMPLETE`: esta condición valida que un contenedor dependiente se ejecute hasta su finalización (se cierre) antes de permitir que otros contenedores se inicien. Esto puede resultar útil para contenedores no esenciales que ejecutan un script y, a continuación, se cierran. Esta condición no se puede establecer en un contenedor esencial.
- `SUCCESS`: esta condición es la misma que `COMPLETE`, pero además requiere que el contenedor se cierre con un estado `zero`. Esta condición no se puede establecer en un contenedor esencial.
- `HEALTHY`: esta condición valida que el contenedor dependiente pase su comprobación de estado de contenedor antes de permitir que otros contenedores se inicien. Esto requiere que el contenedor dependiente tenga configuradas las comprobaciones de estado en la definición de tarea. Esta condición solo se confirma durante el inicio de tarea.

Tiempos de espera de contenedor

`startTimeout`

Tipo: entero

Obligatorio: no

Valores de ejemplo: `120`

Tiempo que hay que esperar (en segundos) antes de desistir en resolver dependencias para un contenedor.

Por ejemplo, se especifican dos contenedores en una definición de tareas donde `containerA` tenga una dependencia en `containerB` que alcance un estado `COMPLETE`, `SUCCESS` o `HEALTHY`. Si se especifica un valor `startTimeout` para `containerB` y este no alcanza el estado deseado en ese tiempo, `containerA` no se inicia.

> **Note**
>
> Si un contenedor no cumple una restricción de dependencia o agota el tiempo de espera antes de cumplir la restricción, Amazon ECS no adelanta los contenedores dependientes a su siguiente estado.

El valor máximo es 120 segundos.

`stopTimeout`

Tipo: entero

Obligatorio: no

Valores de ejemplo: `120`

Duración de tiempo (en segundos) que esperar a que el contenedor se cancelen de forma forzada si no sale de forma normal por sí mismo.

Si no se especifica el parámetro `stopTimeout`, se utiliza el valor establecido para la variable de configuración del agente de contenedor de Amazon ECS `ECS_CONTAINER_STOP_TIMEOUT`. Si no se establece ni el parámetro `stopTimeout` ni la variable de configuración del agente `ECS_CONTAINER_STOP_TIMEOUT`, se utilizan los valores predeterminados de 30 segundos para los contenedores Linux y de 30 segundos en contenedores de Windows. Las instancias de contenedor requieren al menos la versión 1.26.0 del agente de contenedor para habilitar un valor de tiempo de espera de parada de contenedor. No obstante, recomendamos utilizar la versión del agente de contenedor más reciente. Para obtener información acerca de cómo comprobar la versión del agente y actualizar a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS. Si utiliza la AMI de Amazon Linux optimizada para Amazon ECS, la instancia necesita al menos la versión 1.26.0-1 del paquete `ecs-init`. Si las instancias de contenedor se lanzan desde la versión `20190301` o posterior, contienen las versiones requeridas del agente de contenedor y `ecs-init`. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.

Controles del sistema

`systemControls`

Tipo: objeto SystemControl

Obligatorio: no

Una lista de parámetros de kernel del espacio de nombres que se van a establecer en el contenedor. Este parámetro se corresponde con `Sysctls` en el comando create-container de docker y la opción `--sysctl` con docker run. Por ejemplo, puede configurar la configuración `net.ipv4.tcp_keepalive_time` para mantener conexiones de mayor duración.

No se recomienda especificar parámetros `systemControls` relacionados con la red para varios contenedores en una única tarea que también utilice el modo de red `awsvpc` u `host`. De ese modo, se obtienen las siguientes desventajas:

- En las tareas que utilicen el modo de red `awsvpc`, si establece `systemControls` en cualquier contenedor, esto se aplicará a todos los contenedores de la tarea. Si establece diferentes parámetros `systemControls` para varios contenedores en una sola tarea, el contenedor que se inicie en último lugar determinará qué `systemControls` se aplicará.
- Para las tareas que utilizan el modo de red `host`, no se admiten los `systemControls` del espacio de nombres de la red.

Si configura un espacio de nombres de recursos de IPC para usarlo para los contenedores de la tarea, se aplican las siguientes condiciones a los controles del sistema. Para obtener más información, consulte Modo IPC.

- Para las tareas que usan el modo de IPC `host`, no se admiten los `systemControls` del espacio de nombres IPC.
- Para las tareas que utilizan el modo de IPC `task`, los valores de `systemControls` del espacio de nombres IPC se aplican a todos los contenedores de una tarea.

> **Note**
>
> Este parámetro no es compatible con contenedores Windows.

```
"systemControls": [
    {
         "namespace":"string",
         "value":"string"
    }
]
```

`namespace`

Tipo: cadena

Requerido: no

El parámetro de kernel del espacio de nombres para el que se va a establecer un `value`.

Valores de espacio de nombres IPC válidos: `"kernel.msgmax" | "kernel.msgmnb" | "kernel.msgmni" | "kernel.sem" | "kernel.shmall" | "kernel.shmmax" | "kernel.shmmni" | "kernel.shm_rmid_forced"` y `Sysctls` que comiencen por `"fs.mqueue.*"`

Valores de espacio de nombres de red válidos: `Sysctls` que comience por `"net.*"`

`value`

Tipo: cadena

Requerido: no

El valor del parámetro de kernel del espacio de nombres especificado en `namespace`.

Interactivo

`interactive`

Tipo: Booleano

Obligatorio: no

Si este parámetro es `true`, puede implementar aplicaciones en contenedores que requieran la asignación de `stdin` o un `tty`. Este parámetro se corresponde con `OpenStdin` en el comando create-container de docker y la opción `--interactive` con docker run.

El valor predeterminado es `false`.

Pseudo terminal

`pseudoTerminal`

Tipo: Booleano

Obligatorio: no

Cuando este parámetro es `true`, se asigna un TTY. Este parámetro se corresponde con `Tty` en el comando create-container de docker y la opción `--tty` con docker run.

El valor predeterminado es `false`.

Nombre del acelerador de Elastic Inference

El requisito de recursos del acelerador de Elastic Inference para la definición de tareas.

> **Note**
>
> Amazon Elastic Inference (EI) ya no está disponible para los clientes.

Los siguientes parámetros están permitidos en una definición de tarea:

`deviceName`

Tipo: cadena

Obligatorio: sí

Nombre del dispositivo del acelerador de inferencia elástica. También debe hacerse referencia a `deviceName` en una definición de contenedor. Consulte `Elastic Inference accelerator`.

`deviceType`

Tipo: cadena

Obligatorio: sí

El tipo de acelerador de Elastic Inference que se va a utilizar.

Restricciones para ubicación de tareas

Cuando se registra una definición de tareas, se pueden proporcionar restricciones de ubicación de tareas que personalizan la forma en la que Amazon ECS las coloca.

Puede usar las restricciones para ubicar tareas basadas en zona de disponibilidad, tipo de instancia o atributos personalizados. Para obtener más información, consulte Definición de las instancias de contenedor que utiliza Amazon ECS para las tareas.

Los siguientes parámetros están permitidos en una definición de contenedor:

`expression`

Tipo: cadena

Requerido: no

Una expresión de lenguaje de consulte de clúster que aplicar a la restricción. Para obtener más información, consulte Creación de expresiones para definir instancias de contenedor para las tareas de Amazon ECS.

`type`

Tipo: cadena

Obligatorio: sí

El tipo de restricción. Utilice `memberOf` para restringir la selección a un grupo de candidatos válidos.

Configuración del proxy

`proxyConfiguration`

Tipo: objeto ProxyConfiguration

Obligatorio: no

Los detalles de configuración del proxy App Mesh.

Para las tareas que utilizan el tipo de lanzamiento de EC2, las instancias de contenedor requieren al menos la versión 1.26.0 del agente de contenedor y al menos la versión 1.26.0-1 del paquete `ecs-init` para habilitar una configuración de proxy. Si las instancias de contenedor se lanzan desde la versión de AMI optimizada para Amazon ECS `20190301` o posterior, entonces contienen las versiones requeridas del agente de contenedor y `ecs-init`. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.

> **Note**
>
> Este parámetro no es compatible con contenedores Windows.

```
"proxyConfiguration": {
    "type": "APPMESH",
    "containerName": "string",
```

```
    "properties": [
        {
           "name": "string",
           "value": "string"
        }
    ]
}
```

`type`

Tipo: cadena

Valores válidos: `APPMESH`

Obligatorio: no

El tipo de proxy. El único valor admitido es `APPMESH`.

`containerName`

Tipo: cadena

Obligatorio: sí

El nombre del contenedor que sirve como proxy de App Mesh.

`properties`

Tipo: matriz de objetos KeyValuePair

Obligatorio: no

El conjunto de parámetros de configuración de red para proporcionar el complemento Container Network Interface (CNI), especificado como pares clave-valor.

- `IgnoredUID`: (obligatorio) el ID de usuario (UID) del contenedor proxy tal y como lo define el parámetro `user` en una definición de contenedor. Se utiliza para garantizar que el proxy pasa por alto su propio tráfico. Si se especifica `IgnoredGID`, este campo puede estar vacío.
- `IgnoredGID`: (obligatorio) el ID de grupo (GID) del contenedor proxy tal y como lo define el parámetro `user` en una definición de contenedor. Se utiliza para garantizar que el proxy pasa por alto su propio tráfico. Si se especifica `IgnoredUID`, este campo puede estar vacío.

- `AppPorts`: (obligatorio) la lista de los puertos que utiliza la aplicación. El tráfico de red hacia estos puertos se reenvía a `ProxyIngressPort` y `ProxyEgressPort`.
- `ProxyIngressPort`: (obligatorio) especifica el puerto al que se dirige el tráfico que ingresa a `AppPorts`.
- `ProxyEgressPort`: (obligatorio) especifica el puerto al que se dirige el tráfico que sale de `AppPorts`.
- `EgressIgnoredPorts`: (obligatorio) el tráfico de salida que se dirige a estos puertos especificados se pasa por alto y no se redirige a `ProxyEgressPort`. Puede ser una lista vacía.
- `EgressIgnoredIPs`: (obligatorio) el tráfico de salida que se dirige a estas direcciones IP especificadas se pasa por alto y no se redirige a `ProxyEgressPort`. Puede ser una lista vacía.

`name`

Tipo: cadena

Requerido: no

El nombre del par clave-valor.

`value`

Tipo: cadena

Requerido: no

El valor del par clave-valor.

Volúmenes

Al registrar una definición de tareas, se puede especificar una lista de los volúmenes que se transferirán al daemon de Docker en una instancia de contenedor, que estará disponible para otros contenedores en la misma instancia de contenedor.

A continuación se indican los tipos de volúmenes de datos que se pueden usar:

- Volúmenes de Amazon EBS: proporciona almacenamiento en bloques rentable, duradero y de alto rendimiento para cargas de trabajo en contenedores con uso intensivo de datos. Puede adjuntar

un volumen de Amazon EBS por tarea de Amazon ECS cuando ejecute una tarea independiente o cuando cree o actualice un servicio. Se admiten volúmenes de Amazon EBS para tareas de Linux. Para obtener más información, consulte Uso de volúmenes de Amazon EBS con Amazon ECS.

- Volúmenes de Amazon EFS: proporciona almacenamiento de archivos sencillo, escalable y persistente para usarlo con tareas de Amazon ECS. Con Amazon EFS, la capacidad de almacenamiento es elástica. Aumenta y disminuye automáticamente a medida que se agregan o eliminan archivos. Sus aplicaciones disponen del almacenamiento que necesitan, cuando lo necesitan. Se admiten volúmenes de Amazon EFS. Para obtener más información, consulte Uso de volúmenes de Amazon EFS con Amazon ECS.
- Volúmenes de FSx for Windows File Server: proporciona servidores de archivos de Microsoft Windows completamente administrados. Estos servidores de archivos están respaldados por un sistema de archivos de Windows. Cuando utiliza FSx for Windows File Server junto con Amazon ECS, puede aprovisionar sus tareas de Windows con almacenamiento persistente, distribuido, compartido y estático de archivos. Para obtener más información, consulte Uso de volúmenes de FSx para Windows File Server con Amazon ECS.

  Los contenedores Windows de Fargate no admiten esta opción.
- Volúmenes de Docker: un volumen administrado por Docker que se crea en `/var/lib/docker/volumes` en la instancia de Amazon EC2 del host. Los controladores de volúmenes de Docker (también llamados complementos) se usan para integrar los volúmenes con sistemas de almacenamiento externos como Amazon EBS. Se puede usar el controlador de volumen `local` integrado o un controlador de volumen de terceros. Los volúmenes de Docker se admiten solo cuando se ejecutan tareas en instancias de Amazon EC2. Los contenedores de Windows admiten solo el uso del controlador `local`. Para utilizar volúmenes de Docker, especifique `dockerVolumeConfiguration` en su definición de tarea.
- Montajes de unión: un archivo o directorio de la máquina host que se monta en un contenedor. Se admiten volúmenes de host de montaje vinculado. Para utilizar volúmenes de host de montaje vinculado, especifique `host` y un valor de `sourcePath` opcional en su definición de tarea.

Para obtener más información, consulte Opciones de almacenamiento para las tareas de Amazon ECS.

Los siguientes parámetros están permitidos en una definición de contenedor.

`name`

Tipo: cadena

Requerido: no

El nombre del volumen. Se admiten hasta 255 letras (mayúsculas y minúsculas), números, guiones (`-`) y caracteres de subrayado (`_`). Se hace referencia a este nombre en el parámetro `sourceVolume` del objeto `mountPoints` de la definición de contenedor.

`host`

Obligatorio: no

El parámetro `host` se utiliza para vincular el ciclo de vida del montaje de enlace a la instancia de Amazon EC2 del host, en lugar de a la tarea, y donde se almacena. Si el parámetro `host` está vacío, entonces el daemon de Docker asigna una ruta de host a su volumen de datos, pero no se garantiza que los datos persistan después de que los contenedores asociados dejen de funcionar.

Los contenedores de Windows pueden montar directorios completos en la misma unidad que `$env:ProgramData`.

> **Note**
>
> El parámetro `sourcePath` se admite solo cuando se utilizan las tareas que se alojan en instancias de Amazon EC2.

`sourcePath`

Tipo: cadena

Requerido: no

Cuando utilice el parámetro `host`, especifique una `sourcePath` para declarar la ruta de la instancia de Amazon EC2 del host que se presenta al contenedor. Si este parámetro está vacío, el daemon de Docker asigna una ruta de host. Si el parámetro `host` contiene una ubicación de archivos `sourcePath`, el volumen de datos persiste en la ubicación especificada en la instancia de Amazon EC2 del host hasta que la elimine manualmente. Si el valor `sourcePath` no existe en la instancia de Amazon EC2 del host, el daemon de Docker lo crea. Si la ubicación existe, el contenido de la carpeta de la ruta de origen se exporta.

`configuredAtLaunch`

Tipo: Booleano

Obligatorio: no

Indica si un volumen se puede configurar durante el lanzamiento. Cuando se establece en `true`, puede configurarlo al ejecutar una tarea independiente o al crear o actualizar un servicio. Cuando se establece en `true`, no podrá proporcionar otra configuración de volumen en la definición de la tarea. Este parámetro se debe establecer en `true` para configurar un volumen de Amazon EBS para adjuntarlo a una tarea. Establecer `configuredAtLaunch` en `true` y aplazar la configuración del volumen hasta la fase de lanzamiento permite crear definiciones de tareas que no se limitan a un tipo de volumen o a una configuración de volumen específica. De este modo, la definición de la tarea se puede reutilizar en distintos entornos de ejecución. Para obtener más información, consulte Volúmenes de Amazon EBS.

`dockerVolumeConfiguration`

Type: objeto de DockerVolumeConfiguration

Obligatorio: no

Este parámetro se especifica cuando se usan volúmenes de Docker. Los volúmenes de Docker se admiten solo cuando se ejecutan tareas en instancias de EC2. Los contenedores de Windows admiten solo el uso del controlador `local`. Para utilizar montajes vinculados, especifique `host` en su lugar.

`scope`

Tipo: cadena

Valores válidos: `task` | `shared`

Obligatorio: no

El ámbito del volumen de Docker, que determina su ciclo de vida. Los volúmenes de Docker con un ámbito de `task` se aprovisionan automáticamente cuando se inicia la tarea y se destruyen cuando la tarea se detiene. Los volúmenes de Docker cuyo ámbito es `shared` se conservan una vez detenida la tarea.

`autoprovision`

Tipo: Booleano

Valor predeterminado: `false`

Obligatorio: no

Si este valor es `true`, el volumen de Docker se crea si aún no existe. Este campo se usa solo si `scope` es `shared`. Si el `scope` es `task`, este parámetro debe omitirse.

`driver`

Tipo: cadena

Requerido: no

El controlador del volumen de Docker que se va a usar. El valor de controlador debe coincidir con el nombre del controlador proporcionado por Docker, ya que se utiliza para la colocación de tareas. Si el controlador se instaló mediante la CLI del complemento de Docker, utilice `docker plugin ls` para recuperar el nombre de controlador de la instancia de contenedor. Si el controlador se instaló con otro método, utilice la detección de complementos de Docker para recuperar el nombre del controlador.

`driverOpts`

Tipo: cadena

Requerido: no

Un mapa de las opciones específicas del controlador de Docker que se deben transferir. Este parámetro se corresponde con `DriverOpts` en la sección Crear un volumen de Docker.

`labels`

Tipo: cadena

Requerido: no

Metadatos personalizados que se añaden al volumen de Docker.

`efsVolumeConfiguration`

Tipo: objeto de EFSVolumeConfiguration

Obligatorio: no

Este parámetro se especifica cuando se usan volúmenes de Amazon EFS.

`fileSystemId`

Tipo: cadena

Obligatorio: sí

El ID del sistema de archivos de Amazon EFS que se va a usar.

`rootDirectory`

Tipo: cadena

Obligatorio: no

Directorio del sistema de archivos de Amazon EFS que se va a montar como directorio raíz dentro del host. Si se omite este parámetro, se utilizará la raíz del volumen de Amazon EFS. Si se especifica `/`, se obtiene el mismo efecto que si se omite este parámetro.

> **Important**
>
> Si se especifica un punto de acceso de EFS en `authorizationConfig`, se debe omitir el parámetro del directorio raíz o establecerlo en `/`, lo que aplicará la ruta establecida en el punto de acceso de EFS.

`transitEncryption`

Tipo: cadena

Valores válidos: `ENABLED` | `DISABLED`

Obligatorio: no

Especifica si se habilita el cifrado para los datos en tránsito de Amazon EFS entre el host de Amazon ECS y el servidor de Amazon EFS. Si se utiliza la autorización de IAM en Amazon EFS, el cifrado en tránsito debe estar habilitado. Si se omite este parámetro, se usa el valor predeterminado de `DISABLED`. Para obtener más información, consulte Cifrado de datos en tránsito en la Guía del usuario de Amazon Elastic File System.

`transitEncryptionPort`

Tipo: entero

Obligatorio: no

El puerto que se utilizará al enviar datos cifrados entre el host de Amazon ECS y el servidor de Amazon EFS. Si no se especifica un puerto de cifrado en tránsito, la tarea utilizará la estrategia de selección de puertos que utiliza el ayudante de montaje de Amazon EFS. Para

obtener más información, consulte Ayudante de montaje de EFS en la Guía del usuario de Amazon Elastic File System.

`authorizationConfig`

Tipo: objeto de EFSAuthorizationConfig

Obligatorio: no

Los detalles de configuración de autorización en el sistema de archivos de Amazon EFS.

`accessPointId`

Tipo: cadena

Requerido: no

ID de punto de acceso que se va a utilizar. Si se especifica un punto de acceso, el valor del directorio raíz en `efsVolumeConfiguration` se debe omitir o establecer en `/`, lo que aplicará la ruta establecida en el punto de acceso de EFS. Si se utiliza un punto de acceso, el cifrado de tránsito debe estar habilitado en el `EFSVolumeConfiguration`. Para obtener más información, consulte Trabajo con puntos de acceso de Amazon EFS en la Guía del usuario de Amazon Elastic File System.

`iam`

Tipo: cadena

Valores válidos: `ENABLED` | `DISABLED`

Obligatorio: no

Indica si se debe utilizar el rol de IAM de tarea de Amazon ECS definido en una definición de tareas al montar el sistema de archivos de Amazon EFS. Si está habilitado, el cifrado de tránsito debe estar habilitado en el `EFSVolumeConfiguration`. Si se omite este parámetro, se usa el valor predeterminado de `DISABLED`. Para obtener más información, consulte Roles de IAM para las tareas.

`FSxWindowsFileServerVolumeConfiguration`

Tipo: objeto de FSXWindowsFileServerVolumeConfiguration

Obligatorio: sí

Este parámetro se indica cuando se utiliza el sistema de archivos Amazon FSx para Windows File Server para el almacenamiento de tareas.

`fileSystemId`

Tipo: cadena

Obligatorio: sí

ID del sistema de archivos FSx for Windows File Server que se va a utilizar.

`rootDirectory`

Tipo: cadena

Obligatorio: sí

Directorio dentro del sistema de archivos de FSx for Windows File Server que se va a montar como directorio raíz dentro del host.

`authorizationConfig`

`credentialsParameter`

Tipo: cadena

Obligatorio: sí

Opciones de credenciales de autorización.

opciones:

- Nombre de recurso de Amazon (ARN) del secreto de AWS Secrets Manager.
- ARN de un parámetro de AWS Systems Manager.

`domain`

Tipo: cadena

Obligatorio: sí

Nombre de dominio completo alojado por un directorio de AWS Directory Service for Microsoft Active Directory(AWS Managed Microsoft AD) o un directorio de Active Directory de EC2 con alojamiento propio.

Etiquetas

Cuando se registra una definición de tareas, se pueden especificar etiquetas de metadatos que se aplican a la definición de tareas. Las etiquetas ayudan a clasificar y organizar la definición de

tareas. Cada etiqueta consta de una clave y un valor opcional. Los define a los dos. Para obtener más información, consulte Etiquetado de los recursos de Amazon ECS.

> **Important**
>
> No agregue información de identificación personal ni otra información confidencial en las etiquetas. Las etiquetas son accesibles para muchos servicios de AWS, incluida la facturación. Las etiquetas no se diseñaron para utilizarse con información privada o confidencial.

Los siguientes parámetros se admiten en un objeto de etiqueta.

`key`

Tipo: cadena

Requerido: no

Una parte de un par clave-valor que compone una etiqueta. Un clave es una etiqueta general que actúa como una categoría para valores de etiqueta más específicos.

`value`

Tipo: cadena

Requerido: no

La parte opcional de un par clave-valor que compone una etiqueta. Un valor actúa como un descriptor en una categoría de etiquetas (clave).

Otros parámetros de definición de tarea

Los siguientes parámetros de definición de tareas se pueden utilizar cuando se registran definiciones de tareas en la consola de Amazon ECS mediante la opción Configure via JSON (Configurar mediante JSON). Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

Temas

- Modo IPC

- Modo PID
- Inyección de errores

Modo IPC

`ipcMode`

Tipo: cadena

Requerido: no

El espacio de nombres de recurso de IPC que usarán los contenedores de la tarea. Los valores válidos son `host`, `task` o `none`. Si se especifica `host`, todos los contenedores que están dentro de las tareas que tienen especificado el modo de IPC `host` en la misma instancia de contenedor comparten los mismos recursos de IPC con la instancia Amazon EC2 del host. Si se especifica `task`, todos los contenedores que están dentro de la tarea especificada comparten los mismos recursos de IPC. Si se especifica `none`, los recursos de IPC dentro de los contenedores de una tarea son privados y no se comparten con otros contenedores en una tarea o en la instancia de contenedor. Si no se especifica ningún valor, el uso compartido del espacio de nombre de recursos de IPC depende de la configuración del daemon de Docker en la instancia de contenedor.

Si se utiliza el modo de IPC `host`, existe un mayor riesgo de exposición de espacio de nombres de IPC no deseada.

Si configura parámetros del kernel del espacio de nombres mediante `systemControls` para los contenedores de la tarea, se aplica lo siguiente a su espacio de nombres de recursos de IPC.

- Para las tareas que utilizan el modo de IPC `host`, no se admiten los `systemControls` relacionados con el espacio de nombres de IPC.
- Para las tareas que utilizan el modo de IPC `task`, los `systemControls` relacionados con el espacio de nombres de IPC se aplican a todos los contenedores de una tarea.

Modo PID

`pidMode`

Tipo: cadena

Valores válidos: `host` | `task`

Obligatorio: no

El espacio de nombres del proceso que usarán los contenedores de la tarea. Los valores válidos son `host` o `task`. Por ejemplo, la supervisión de los archivos sidecar puede necesitar `pidMode` para acceder a información sobre otros contenedores que se ejecutan en la misma tarea.

Si se especifica `host`, todos los contenedores dentro de las tareas que tienen especificado el modo de PID `host` en la misma instancia de contenedor comparten el mismo espacio de nombres del proceso con la instancia Amazon EC2 del host.

Si se especifica `task`, todos los contenedores dentro de la tarea especificada comparten el mismo espacio de nombres del proceso.

Si no se especifica ningún valor, el valor predeterminado es un espacio de nombre privado para cada contenedor.

Si se utiliza el modo de PID `host`, existe un mayor riesgo de exposición de espacio de nombres del proceso no deseada.

> **Note**
>
> Este parámetro no es compatible con contenedores Windows.

Inyección de errores

`enableFaultInjection`

Tipo: Booleano

Valores válidos: `true` | `false`

Obligatorio: no

Si este parámetro se establece en `true`, en la carga útil de una tarea, Amazon ECS aceptará solicitudes de inyección de errores de los contenedores de la tarea. Este parámetro está establecido en `false` de forma predeterminada.

Plantilla de definición de tareas de Amazon ECS

Una plantilla de definición de tareas vacía se ve así. Utilice esta plantilla para crear la definición de la tarea, que posteriormente se puede pegar en el área de entrada JSON de la consola o guardar en un archivo y utilizarse con la opción de la AWS CLI `--cli-input-json`. Para obtener más información, consulte [Parámetros en la definición de tareas de Amazon ECS para el tipo de lanzamiento de Fargate](#).

Plantilla de tipo de lanzamiento de Amazon EC2

```
{
  "family": "",
  "taskRoleArn": "",
  "executionRoleArn": "",
  "networkMode": "none",
  "containerDefinitions": [
    {
      "name": "",
      "image": "",
      "repositoryCredentials": {
        "credentialsParameter": ""
      },
      "cpu": 0,
      "memory": 0,
      "memoryReservation": 0,
      "links": [""],
      "portMappings": [
        {
          "containerPort": 0,
          "hostPort": 0,
          "protocol": "tcp"
        }
      ],
      "restartPolicy": {
        "enabled": true,
        "ignoredExitCodes": [0],
        "restartAttemptPeriod": 180
      },
      "essential": true,
      "entryPoint": [""],
      "command": [""],
      "environment": [
```

```
        {
          "name": "",
          "value": ""
        }
      ],
      "environmentFiles": [
        {
          "value": "",
          "type": "s3"
        }
      ],
      "mountPoints": [
        {
          "sourceVolume": "",
          "containerPath": "",
          "readOnly": true
        }
      ],
      "volumesFrom": [
        {
          "sourceContainer": "",
          "readOnly": true
        }
      ],
      "linuxParameters": {
        "capabilities": {
          "add": [""],
          "drop": [""]
        },
        "devices": [
          {
            "hostPath": "",
            "containerPath": "",
            "permissions": ["read"]
          }
        ],
        "initProcessEnabled": true,
        "sharedMemorySize": 0,
        "tmpfs": [
          {
            "containerPath": "",
            "size": 0,
            "mountOptions": [""]
          }
```

```
        ],
        "maxSwap": 0,
        "swappiness": 0
      },
      "secrets": [
        {
          "name": "",
          "valueFrom": ""
        }
      ],
      "dependsOn": [
        {
          "containerName": "",
          "condition": "COMPLETE"
        }
      ],
      "startTimeout": 0,
      "stopTimeout": 0,
      "hostname": "",
      "user": "",
      "workingDirectory": "",
      "disableNetworking": true,
      "privileged": true,
      "readonlyRootFilesystem": true,
      "dnsServers": [""],
      "dnsSearchDomains": [""],
      "extraHosts": [
        {
          "hostname": "",
          "ipAddress": ""
        }
      ],
      "dockerSecurityOptions": [""],
      "interactive": true,
      "pseudoTerminal": true,
      "dockerLabels": {
        "KeyName": ""
      },
      "ulimits": [
        {
          "name": "nofile",
          "softLimit": 0,
          "hardLimit": 0
        }
```

```
      ],
      "logConfiguration": {
        "logDriver": "splunk",
        "options": {
          "KeyName": ""
        },
        "secretOptions": [
          {
            "name": "",
            "valueFrom": ""
          }
        ]
      },
      "healthCheck": {
        "command": [""],
        "interval": 0,
        "timeout": 0,
        "retries": 0,
        "startPeriod": 0
      },
      "systemControls": [
        {
          "namespace": "",
          "value": ""
        }
      ],
      "resourceRequirements": [
        {
          "value": "",
          "type": "InferenceAccelerator"
        }
      ],
      "firelensConfiguration": {
        "type": "fluentbit",
        "options": {
          "KeyName": ""
        }
      }
    }
  ],
  "volumes": [
    {
      "name": "",
      "host": {
```

```
        "sourcePath": ""
      },
      "configuredAtLaunch": true,
      "dockerVolumeConfiguration": {
        "scope": "shared",
        "autoprovision": true,
        "driver": "",
        "driverOpts": {
          "KeyName": ""
        },
        "labels": {
          "KeyName": ""
        }
      },
      "efsVolumeConfiguration": {
        "fileSystemId": "",
        "rootDirectory": "",
        "transitEncryption": "DISABLED",
        "transitEncryptionPort": 0,
        "authorizationConfig": {
          "accessPointId": "",
          "iam": "ENABLED"
        }
      },
      "fsxWindowsFileServerVolumeConfiguration": {
        "fileSystemId": "",
        "rootDirectory": "",
        "authorizationConfig": {
          "credentialsParameter": "",
          "domain": ""
        }
      }
    }
  ],
  "placementConstraints": [
    {
      "type": "memberOf",
      "expression": ""
    }
  ],
  "requiresCompatibilities": ["EC2"],
  "cpu": "",
  "memory": "",
  "tags": [
```

```
    {
      "key": "",
      "value": ""
    }
  ],
  "pidMode": "task",
  "ipcMode": "task",
  "proxyConfiguration": {
    "type": "APPMESH",
    "containerName": "",
    "properties": [
      {
        "name": "",
        "value": ""
      }
    ]
  },
  "inferenceAccelerators": [
    {
      "deviceName": "",
      "deviceType": ""
    }
  ],
  "ephemeralStorage": {
    "sizeInGiB": 0
  },
  "runtimePlatform": {
    "cpuArchitecture": "X86_64",
    "operatingSystemFamily": "WINDOWS_SERVER_20H2_CORE"
  }
}
```

Plantilla de tipo de lanzamiento de Fargate

> **Important**
>
> En el caso de un tipo de lanzamiento de Fargate, debe incluir el parámetro `operatingSystemFamily` con uno de los valores siguientes:
>
> - `LINUX`
> - `WINDOWS_SERVER_2019_FULL`

- WINDOWS_SERVER_2019_CORE
- WINDOWS_SERVER_2022_FULL
- WINDOWS_SERVER_2022_CORE

```
{
    "family": "",
    "runtimePlatform": {"operatingSystemFamily": ""},
    "taskRoleArn": "",
    "executionRoleArn": "",
    "networkMode": "awsvpc",
    "platformFamily": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            "repositoryCredentials": {"credentialsParameter": ""},
            "cpu": 0,
            "memory": 0,
            "memoryReservation": 0,
            "links": [""],
            "portMappings": [
                {
                    "containerPort": 0,
                    "hostPort": 0,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": [""],
            "command": [""],
            "environment": [
                {
                    "name": "",
                    "value": ""
                }
            ],
            "environmentFiles": [
                {
                    "value": "",
                    "type": "s3"
                }
```

```
            ],
            "mountPoints": [
                {
                    "sourceVolume": "",
                    "containerPath": "",
                    "readOnly": true
                }
            ],
            "volumesFrom": [
                {
                    "sourceContainer": "",
                    "readOnly": true
                }
            ],
            "linuxParameters": {
                "capabilities": {
                    "add": [""],
                    "drop": [""]
                },
                "devices": [
                    {
                        "hostPath": "",
                        "containerPath": "",
                        "permissions": ["read"]
                    }
                ],
                "initProcessEnabled": true,
                "sharedMemorySize": 0,
                "tmpfs": [
                    {
                        "containerPath": "",
                        "size": 0,
                        "mountOptions": [""]
                    }
                ],
                "maxSwap": 0,
                "swappiness": 0
            },
            "secrets": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ],
```

```
            "dependsOn": [
                {
                    "containerName": "",
                    "condition": "HEALTHY"
                }
            ],
            "startTimeout": 0,
            "stopTimeout": 0,
            "hostname": "",
            "user": "",
            "workingDirectory": "",
            "disableNetworking": true,
            "privileged": true,
            "readonlyRootFilesystem": true,
            "dnsServers": [""],
            "dnsSearchDomains": [""],
            "extraHosts": [
                {
                    "hostname": "",
                    "ipAddress": ""
                }
            ],
            "dockerSecurityOptions": [""],
            "interactive": true,
            "pseudoTerminal": true,
            "dockerLabels": {"KeyName": ""},
            "ulimits": [
                {
                    "name": "msgqueue",
                    "softLimit": 0,
                    "hardLimit": 0
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {"KeyName": ""},
                "secretOptions": [
                    {
                        "name": "",
                        "valueFrom": ""
                    }
                ]
            },
            "healthCheck": {
```

```
                "command": [""],
                "interval": 0,
                "timeout": 0,
                "retries": 0,
                "startPeriod": 0
            },
            "systemControls": [
                {
                    "namespace": "",
                    "value": ""
                }
            ],
            "resourceRequirements": [
                {
                    "value": "",
                    "type": "GPU"
                }
            ],
            "firelensConfiguration": {
                "type": "fluentd",
                "options": {"KeyName": ""}
            }
        }
    ],
    "volumes": [
        {
            "name": "",
            "host": {"sourcePath": ""},
            "configuredAtLaunch":true,
            "dockerVolumeConfiguration": {
                "scope": "task",
                "autoprovision": true,
                "driver": "",
                "driverOpts": {"KeyName": ""},
                "labels": {"KeyName": ""}
            },
            "efsVolumeConfiguration": {
                "fileSystemId": "",
                "rootDirectory": "",
                "transitEncryption": "ENABLED",
                "transitEncryptionPort": 0,
                "authorizationConfig": {
                    "accessPointId": "",
                    "iam": "ENABLED"
```

```
                }
            }
        }
    ],
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "",
    "memory": "",
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "ephemeralStorage": {"sizeInGiB": 0},
    "pidMode": "task",
    "ipcMode": "none",
    "proxyConfiguration": {
        "type": "APPMESH",
        "containerName": "",
        "properties": [
            {
                "name": "",
                "value": ""
            }
        ]
    },
    "inferenceAccelerators": [
        {
            "deviceName": "",
            "deviceType": ""
        }
    ]
}
```

Puede generar esta plantilla de definición de tareas utilizando el comando de la AWS CLI a continuación.

```
aws ecs register-task-definition --generate-cli-skeleton
```

Ejemplo de definiciones de tareas de Amazon ECS

Puede copiar los ejemplos y fragmentos de código para comenzar a crear sus propias definiciones de tareas.

Puede copiar los ejemplos y, a continuación, pegarlos cuando utilice la opción Configurar mediante JSON en la consola. Asegúrese de personalizar los ejemplos, como usar el ID de su cuenta. Puede incluir los fragmentos en el JSON de definición de tareas. Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola y Parámetros en la definición de tareas de Amazon ECS para el tipo de lanzamiento de Fargate.

Para obtener más ejemplos de definición de tareas, consulte Definiciones de tareas de muestra de AWS en GitHub.

Temas

- Servidor web
- Controlador de registros de splunk
- Controlador de registros de fluentd
- Controlador de registros de gelf
- Cargas de trabajo en instancias externas
- Rol de IAM de definición de tarea e imagen de Amazon ECR
- Punto de entrada con comando
- Dependencia de contenedor
- Volúmenes en definiciones de tareas
- Definiciones de tareas de muestra de Windows

Servidor web

A continuación, se muestra una definición de tarea de ejemplo con el tipo de lanzamiento de Fargate o contenedores Linux que configura un servidor web:

```
{
   "containerDefinitions": [
      {
         "command": [
            "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</
title> <style>body {margin-top: 40px; background-color: #333;} </style> </
```

```
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
 ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && httpd-
foreground\""
         ],
         "entryPoint": [
            "sh",
            "-c"
         ],
         "essential": true,
         "image": "public.ecr.aws/docker/library/httpd:2.4",
         "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
               "awslogs-group" : "/ecs/fargate-task-definition",
               "awslogs-region": "us-east-1",
               "awslogs-stream-prefix": "ecs"
            }
         },
         "name": "sample-fargate-app",
         "portMappings": [
            {
               "containerPort": 80,
               "hostPort": 80,
               "protocol": "tcp"
            }
         ]
      }
   ],
   "cpu": "256",
   "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
   "family": "fargate-task-definition",
   "memory": "512",
   "networkMode": "awsvpc",
   "runtimePlatform": {
        "operatingSystemFamily": "LINUX"
    },
   "requiresCompatibilities": [
       "FARGATE"
    ]
}
```

A continuación, se muestra una definición de tarea de ejemplo con el tipo de lanzamiento de Fargate o contenedores Windows que configura un servidor web:

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

Controlador de registros de **splunk**

En el fragmento siguiente se muestra cómo utilizar el controlador de registros `splunk` en una definición de tarea que envía los registros a un servicio remoto. El parámetro de token Splunk se especifica como una opción secreta, ya que puede tratarse como información confidencial. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.

```
"containerDefinitions": [{
  "logConfiguration": {
   "logDriver": "splunk",
   "options": {
    "splunk-url": "https://cloud.splunk.com:8080",
    "tag": "tag_name",
   },
   "secretOptions": [{
    "name": "splunk-token",
    "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:splunk-token-
KnrBkD"
}],
```

Controlador de registros de **fluentd**

En el fragmento siguiente se muestra cómo utilizar el controlador de registros `fluentd` en una definición de tarea que envía los registros a un servicio remoto. El valor `fluentd-address` se especifica como una opción secreta, ya que puede ser tratado como información confidencial. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.

```
"containerDefinitions": [{
 "logConfiguration": {
  "logDriver": "fluentd",
  "options": {
   "tag": "fluentd demo"
  },
  "secretOptions": [{
   "name": "fluentd-address",
   "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:fluentd-address-
KnrBkD"
  }]
```

```
 },
 "entryPoint": [],
 "portMappings": [{
             "hostPort": 80,
             "protocol": "tcp",
             "containerPort": 80
             },
             {
  "hostPort": 24224,
  "protocol": "tcp",
  "containerPort": 24224
 }]
}],
```

Controlador de registros de **`gelf`**

En el fragmento siguiente se muestra cómo utilizar el controlador de registros `gelf` en una definición de tarea que envía los registros a un host remoto que ejecuta Logstash que toma los registros de Gelf como entrada. Para obtener más información, consulte `logConfiguration`.

```
"containerDefinitions": [{
 "logConfiguration": {
  "logDriver": "gelf",
  "options": {
   "gelf-address": "udp://logstash-service-address:5000",
   "tag": "gelf task demo"
  }
 },
 "entryPoint": [],
 "portMappings": [{
   "hostPort": 5000,
   "protocol": "udp",
   "containerPort": 5000
  },
  {
   "hostPort": 5000,
   "protocol": "tcp",
   "containerPort": 5000
  }
 ]
}],
```

Cargas de trabajo en instancias externas

Cuando registre una definición de tareas de Amazon ECS, utilice el parámetro `requiresCompatibilities` y especifique `EXTERNAL` a fin de validar la compatibilidad de la definición de tareas para su utilización al ejecutar cargas de trabajo de Amazon ECS en las instancias externas. Si utiliza la consola para registrar una definición de tarea, debe utilizar el editor de JSON. Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

> **Important**
>
> Si las tareas requieren un rol de IAM de ejecución de tareas, asegúrese de que esté especificado en la definición de tareas.

Cuando implemente la carga de trabajo, utilice el tipo de lanzamiento `EXTERNAL` al crear el servicio o ejecutar la tarea independiente.

A continuación, se muestra una definición de tareas de ejemplo.

Linux

```
{
 "requiresCompatibilities": [
  "EXTERNAL"
 ],
 "containerDefinitions": [{
  "name": "nginx",
  "image": "public.ecr.aws/nginx/nginx:latest",
  "memory": 256,
  "cpu": 256,
  "essential": true,
  "portMappings": [{
   "containerPort": 80,
   "hostPort": 8080,
   "protocol": "tcp"
  }]
 }],
 "networkMode": "bridge",
 "family": "nginx"
}
```

Windows

```
{
 "requiresCompatibilities": [
  "EXTERNAL"
 ],
 "containerDefinitions": [{
  "name": "windows-container",
  "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-ltsc2019",
  "memory": 256,
  "cpu": 512,
  "essential": true,
  "portMappings": [{
   "containerPort": 80,
   "hostPort": 8080,
   "protocol": "tcp"
  }]
 }],
 "networkMode": "bridge",
 "family": "windows-container"
}
```

Rol de IAM de definición de tarea e imagen de Amazon ECR

El fragmento siguiente utiliza una imagen de Amazon ECR denominada `aws-nodejs-sample` con la etiqueta `v1` del registro `123456789012.dkr.ecr.us-west-2.amazonaws.com`. El contenedor de esta tarea hereda los permisos de IAM del rol `arn:aws:iam::123456789012:role/AmazonECSTaskS3BucketRole`. Para obtener más información, consulte Rol de IAM de tarea de Amazon ECS.

```
{
    "containerDefinitions": [
        {
            "name": "sample-app",
            "image": "123456789012.dkr.ecr.us-west-2.amazonaws.com/aws-nodejs-
sample:v1",
            "memory": 200,
            "cpu": 10,
            "essential": true
        }
    ],
```

```
    "family": "example_task_3",
    "taskRoleArn": "arn:aws:iam::123456789012:role/AmazonECSTaskS3BucketRole"
}
```

Punto de entrada con comando

El fragmento siguiente muestra la sintaxis de un contenedor de Docker que utiliza un punto de entrada y un argumento de comando. Este contenedor realiza ping a `example.com` cuatro veces y, a continuación, se cierra.

```
{
    "containerDefinitions": [
        {
            "memory": 32,
            "essential": true,
            "entryPoint": ["ping"],
            "name": "alpine_ping",
            "readonlyRootFilesystem": true,
            "image": "alpine:3.4",
            "command": [
                "-c",
                "4",
                "example.com"
            ],
            "cpu": 16
        }
    ],
    "family": "example_task_2"
}
```

Dependencia de contenedor

Este fragmento muestra la sintaxis de una definición de tareas con varios contenedores donde se especifica la dependencia de contenedores. En la siguiente definición de tarea, el contenedor `envoy` debe llegar a un estado de funcionamiento correcto, determinado por los parámetros necesarios de comprobación de estado del contenedor, antes de que el contenedor `app` se inicie. Para obtener más información, consulte Dependencia de contenedor.

```
{
  "family": "appmesh-gateway",
  "runtimePlatform": {
```

```
        "operatingSystemFamily": "LINUX"
    },
    "proxyConfiguration":{
        "type": "APPMESH",
        "containerName": "envoy",
        "properties": [
            {
                "name": "IgnoredUID",
                "value": "1337"
            },
            {
                "name": "ProxyIngressPort",
                "value": "15000"
            },
            {
                "name": "ProxyEgressPort",
                "value": "15001"
            },
            {
                "name": "AppPorts",
                "value": "9080"
            },
            {
                "name": "EgressIgnoredIPs",
                "value": "169.254.170.2,169.254.169.254"
            }
        ]
    },
    "containerDefinitions": [
      {
        "name": "app",
        "image": "application_image",
        "portMappings": [
          {
            "containerPort": 9080,
            "hostPort": 9080,
            "protocol": "tcp"
          }
        ],
        "essential": true,
        "dependsOn": [
          {
            "containerName": "envoy",
            "condition": "HEALTHY"
```

```
        }
      ]
    },
    {
      "name": "envoy",
      "image": "840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-
envoy:v1.15.1.0-prod",
      "essential": true,
      "environment": [
        {
          "name": "APPMESH_VIRTUAL_NODE_NAME",
          "value": "mesh/meshName/virtualNode/virtualNodeName"
        },
        {
          "name": "ENVOY_LOG_LEVEL",
          "value": "info"
        }
      ],
      "healthCheck": {
        "command": [
          "CMD-SHELL",
          "echo hello"
        ],
        "interval": 5,
        "timeout": 2,
        "retries": 3
      }
    }
  ],
  "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
  "networkMode": "awsvpc"
}
```

Volúmenes en definiciones de tareas

Utilice lo siguiente para entender cómo especificar los volúmenes en las tareas.

- Para obtener información acerca de cómo configurar un volumen de Amazon EBS, consulte Especificación de la configuración del volumen de Amazon EBS durante la implementación de Amazon ECS.
- Para obtener información acerca de cómo configurar un volumen de Amazon EFS, consulte Configuración de sistemas de archivos de Amazon EFS para Amazon ECS mediante la consola.

- Para obtener información acerca de cómo configurar un volumen de FSx for Windows File Server, consulte Obtenga información sobre cómo configurar FSx para sistemas de archivos de Windows File Server para Amazon ECS..
- Para obtener información acerca de cómo configurar un volumen de Docker, consulte Ejemplos de volúmenes de Docker para Amazon ECS.
- Para obtener información acerca de cómo configurar un montaje vinculado, consulte Ejemplos de montajes vinculados para Amazon ECS.

Definiciones de tareas de muestra de Windows

A continuación, se muestra una definición de tareas de muestra que lo ayudará a familiarizarse con los contenedores de Windows en Amazon ECS.

Example Aplicación de muestra de consola de Amazon ECS para Windows

La siguiente definición de tareas corresponde a la aplicación de muestra de la consola de Amazon ECS que se observa en el asistente de primer uso de Amazon ECS; se ha transferido para que utilice la imagen de contenedor de Windows `microsoft/iis`.

```
{
  "family": "windows-simple-iis",
  "containerDefinitions": [
    {
      "name": "windows_sample_app",
      "image": "mcr.microsoft.com/windows/servercore/iis",
      "cpu": 1024,
      "entryPoint":["powershell", "-Command"],
      "command":["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file -
Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
      "portMappings": [
        {
          "protocol": "tcp",
          "containerPort": 80
        }
      ],
      "memory": 1024,
```

```
      "essential": true
    }
  ],
  "networkMode": "awsvpc",
  "memory": "1024",
  "cpu": "1024"
}
```

Clústeres de Amazon ECS

Un clúster de Amazon ECS es una agrupación lógica de tareas o servicios. Además de las tareas y los servicios, un clúster consta de los siguientes recursos:

- La capacidad de la infraestructura, que puede ser una combinación de las siguientes:
  - Instancias de Amazon EC2 en la nube de AWS
  - Sin servidor (AWS Fargate) en la nube de AWS
  - Máquinas virtuales (VM) o servidores locales en las instalaciones
- La red (VPC y subred) en la que se ejecutan sus tareas y servicios

  Cuando utiliza instancias de Amazon EC2 para la capacidad, la subred puede estar en zonas de disponibilidad, zonas locales, zonas de Wavelength o AWS Outposts.
- Un espacio de nombres opcional

  El espacio de nombres se utiliza para la comunicación de servicio a servicio con Service Connect.
- Una opción de monitoreo

  CloudWatch Container Insights tiene un coste adicional y es un servicio totalmente administrado. Recopila, agrega y resume automáticamente métricas y registros de Amazon ECS.

A continuación, se muestran los conceptos generales sobre los clústeres de Amazon ECS.

- Se crean clústeres para separar los recursos.
- Los clústeres son específicos de la Región de AWS.
- Los clústeres pueden tener alguno de los estados que se indican a continuación.

  ACTIVE

  El clúster está listo para aceptar tareas y, si procede, puede registrar instancias de contenedor con él.

  PROVISIONING

  El clúster tiene proveedores de capacidad asociados y se están creando los recursos necesarios para el proveedor de capacidad.

DEPROVISIONING

El clúster tiene proveedores de capacidad asociados y se están eliminando los recursos necesarios para el proveedor de capacidad.

ERROR

El clúster tiene proveedores de capacidad asociados y no se han podido crear los recursos necesarios para el proveedor de capacidad.

INACTIVE

El clúster se ha eliminado. Es posible que los clústeres con estado `INACTIVE` permanezcan detectables en la cuenta durante un período de tiempo. Este comportamiento está sujeto a cambios en el futuro, por lo que asegúrese de no contar con la permanencia de los clústeres `INACTIVE`.

- Puede utilizar diferentes tipos de instancias para el tipo de lanzamiento de EC2 o proveedores de capacidad de grupos de escalado automático. Una instancia solo se puede registrar en un clúster a la vez.
- Puede restringir el acceso a clústeres mediante la creación de políticas de IAM personalizadas. Para obtener más información, consulte la sección Ejemplos de clústeres de Amazon ECS en Ejemplos de políticas basadas en identidades de Amazon Elastic Container Service.
- Puede utilizar el escalado automático de servicios para escalar las tareas de Fargate. Para obtener más información, consulte Escalado automático de su servicio de Amazon ECS.
- Puede configurar un espacio de nombres de Service Connect predeterminado para un clúster. Después de configurar un espacio de nombres de Service Connect predeterminado, todos los servicios nuevos que se creen en el clúster se pueden agregar como servicios de cliente al espacio de nombres al activar Service Connect. No se necesita configuración adicional. Para obtener más información, consulte Uso de Service Connect para conectar los servicios de Amazon ECS con nombres abreviados.

Proveedores de capacidad

Los proveedores de capacidad de Amazon ECS administran el escalado de la infraestructura para las tareas de los clústeres. Cada clúster puede tener uno o más proveedores de capacidad y, opcionalmente, una estrategia de proveedor de capacidad. Puede asignar una estrategia de proveedor de capacidad predeterminada al clúster. La estrategia del proveedor de capacidad determina cómo se distribuyen las tareas entre los proveedores de capacidad del clúster. Cuando

ejecuta una tarea independiente o crea un servicio, utiliza la estrategia de proveedores de capacidad predeterminada del clúster o una estrategia de proveedores de capacidad que anula la estrategia predeterminada. La estrategia de proveedores de capacidad predeterminada del clúster solo se aplica cuando no especifica un tipo de inicialización o una estrategia de proveedores de capacidad para la tarea o servicio. Si proporciona alguno de estos parámetros, no se utilizará la estrategia predeterminada.

Para Fargate, no necesita crear ni administrar la capacidad. Solo tiene que asociar al clúster cualquiera de los siguientes proveedores de capacidad predefinidos:

- Fargate
- Fargate Spot

Cuando utiliza instancias de Amazon EC2 para su capacidad, utiliza grupos de escalado automático para administrar las instancias de Amazon EC2. El escalado automático ayuda a garantizar que cuente con la cantidad correcta de instancias de Amazon EC2 disponibles para gestionar la carga de la aplicación.

Un clúster puede contener una combinación de tareas alojadas en AWS Fargate, instancias de Amazon EC2 o instancias externas. Las tareas se pueden ejecutar en una infraestructura de Fargate o EC2 como un tipo de lanzamiento o una estrategia de proveedor de capacidad. Si utiliza EC2 como tipo de lanzamiento, Amazon ECS no rastrea ni escala la capacidad de los grupos de Amazon EC2 Auto Scaling. Para obtener más información acerca de los tipos de lanzamiento, consulte Tipos de lanzamiento de Amazon ECS.

Un clúster puede contener una combinación de proveedores de capacidad del grupo de escalado automático y proveedores de capacidad de Fargate. Una estrategia de proveedores de capacidad solo puede incluir proveedores de capacidad de grupos de escalado automático o de Fargate.

Clústeres de Amazon ECS para Fargate

Los proveedores de capacidad de Amazon ECS administran el escalado de la infraestructura para las tareas de los clústeres. Cada clúster puede tener uno o más proveedores de capacidad y, opcionalmente, una estrategia de proveedor de capacidad. La estrategia del proveedor de capacidad determina cómo se distribuyen las tareas entre los proveedores de capacidad del clúster. Cuando ejecuta una tarea individual o crea un servicio, utiliza la estrategia de proveedores de capacidad predeterminada del clúster o una estrategia de proveedores de capacidad que anule la estrategia predeterminada.

Cuando ejecuta las tareas en AWS Fargate, no necesita crear ni administrar la capacidad. Solo tiene que asociar al clúster cualquiera de los siguientes proveedores de capacidad predefinidos:

- Fargate
- Fargate Spot

Los proveedores de capacidad de Amazon ECS en AWS Fargate le permiten utilizar la capacidad de Fargate y de Fargate Spot para las tareas de Amazon ECS.

Con Fargate Spot, puede ejecutar tareas de Amazon ECS tolerantes a interrupciones con un descuento respecto al precio de Fargate. Fargate Spot ejecuta las tareas en la capacidad de cómputo adicional. Cuando AWS necesita recuperar esa capacidad, las tareas se interrumpen previa advertencia con dos minutos de antelación.

Cuando se detienen las tareas que utilizan los proveedores de capacidad de Fargate y Fargate Spot, se envía un evento de cambio de estado de la tarea a Amazon EventBridge. En el motivo de la parada se describe la causa. Para obtener más información, consulte Eventos de cambio de estado de tarea de Amazon ECS.

Un clúster puede contener una combinación de proveedores de capacidad del grupo de escalado automático y Fargate. Sin embargo, una estrategia de proveedores de capacidad solo puede contener proveedores de capacidad de grupos de escalado automático o Fargate, pero no ambos. Para obtener más información, consulte Proveedores de capacidad de grupos de escalado automático.

Cuando utilice proveedores de capacidad, tenga en cuenta lo siguiente:

- Debe asociar un proveedor de capacidad con un clúster para poder asociarlo con la estrategia de proveedores de capacidad.
- Puede especificar un máximo de 20 proveedores de capacidad para una estrategia de proveedores de capacidad.
- No puede actualizar un servicio que utiliza un proveedor de capacidad de grupos de escalado automático para que utilice un proveedor de capacidad de Fargate. En caso de que sea lo contrario, tampoco puede hacerlo.
- En una estrategia de proveedores de capacidad, si no se especifica ningún valor `weight` para un proveedor de capacidad en la consola, entonces se utiliza el valor predeterminado `1`. Si utiliza la API o la AWS CLI, se utiliza el valor predeterminado `0`.

- Cuando se especifican varios proveedores de capacidad dentro de una estrategia de proveedores de capacidad, al menos uno de los proveedores de capacidad deberá tener un valor de peso superior a cero. Los proveedores de capacidad con un peso de cero no se usan para realizar tareas. Si especifica varios proveedores de capacidad en una estrategia en la que todos tienen el mismo peso de 0, se producirá un error en cualquiera de las acciones `RunTask` o `CreateService` que utilicen la estrategia de proveedores de capacidad.
- En una estrategia de proveedores de capacidad, solo un proveedor de capacidad puede tener un valor base definido. Si no se especifica ningún valor base, se utiliza el valor predeterminado 0.
- Un clúster puede contener una combinación de proveedores de capacidad del grupo de escalado automático y proveedores de capacidad de Fargate. Sin embargo, una estrategia de proveedores de capacidad solo puede incluir proveedores de capacidad de grupo de escalado automático o de Fargate, pero no ambos.
- Un clúster puede contener una combinación de servicios y tareas independientes que utilicen proveedores de capacidad y tipos de lanzamiento. Un servicio se puede actualizar para que utilice una estrategia de proveedores de capacidad en lugar de un tipo de lanzamiento. Sin embargo, al hacerlo, debe forzar una nueva implementación.

Avisos de terminación de Fargate Spot

Durante los períodos de demanda extremadamente alta, es posible que la capacidad de Fargate Spot no esté disponible. Esto puede provocar que las tareas de Fargate Spot se retrasen. Cuando sucede esto, los servicios de Amazon ECS vuelven a intentar iniciar las tareas hasta que se disponga de la capacidad necesaria. Fargate no sustituye la capacidad de Spot por la capacidad bajo demanda.

Cuando se detienen tareas que utilizan capacidad de Fargate Spot debido a una interrupción de spot, se envía una advertencia dos minutos antes de la detención de la tarea. La advertencia se envía como un evento de cambio de estado de tarea a Amazon EventBridge y como una señal SIGTERM a la tarea en ejecución. Si utiliza Fargate Spot como parte de un servicio, en esta situación, el programador de servicio recibe la señal de interrupción e intenta lanzar tareas adicionales en Fargate Spot si hay capacidad disponible. Un servicio con una sola tarea se interrumpirá hasta que haya capacidad disponible. Para obtener más información sobre un cierre correcto, consulte Cierres correctos con ECS.

Para asegurarse de que los contenedores realicen una salida correcta antes de que se detenga la tarea, puede configurar lo siguiente:

- Se puede especificar un valor de `stopTimeout` de `120` segundos o menos en la definición del contenedor que la tarea utiliza. El valor de `stopTimeout` predeterminado es de 30 segundos. Puede especificar un valor de `stopTimeout` mayor para disponer de más tiempo desde el momento en que se recibe el evento de cambio de estado de tarea hasta el punto en el que se detiene forzosamente el contenedor. Para obtener más información, consulte Tiempos de espera de contenedor.
- La señal SIGTERM debe recibirse desde dentro del contenedor para realizar cualquier acción de limpieza. Si no se procesa esta señal, la tarea recibirá una señal SIGKILL una vez transcurrido el `stopTimeout` configurado, lo que puede provocar pérdida o corrupción de datos.

A continuación, se ofrece un fragmento de un evento de cambio de estado de tarea. Este fragmento muestra el motivo y el código de detención de una interrupción de Fargate Spot.

```
{
  "version": "0",
  "id": "9bcdac79-b31f-4d3d-9410-fbd727c29fab",
  "detail-type": "ECS Task State Change",
  "source": "aws.ecs",
  "account": "111122223333",
  "resources": [
    "arn:aws:ecs:us-east-1:111122223333:task/b99d40b3-5176-4f71-9a52-9dbd6f1cebef"
  ],
  "detail": {
    "clusterArn": "arn:aws:ecs:us-east-1:111122223333:cluster/default",
    "createdAt": "2016-12-06T16:41:05.702Z",
    "desiredStatus": "STOPPED",
    "lastStatus": "RUNNING",
    "stoppedReason": "Your Spot Task was interrupted.",
    "stopCode": "SpotInterruption",
    "taskArn": "arn:aws:ecs:us-east-1:111122223333:task/
b99d40b3-5176-4f71-9a52-9dbd6fEXAMPLE",
    ...
  }
}
```

El siguiente patrón de eventos se utiliza para crear una regla de EventBridge para los eventos de cambio de estado de tarea de Amazon ECS. Si lo desea, puede especificar un clúster en el campo `detail`. Al hacerlo, recibirá eventos de cambio de estado de la tarea para ese clúster. Para obtener

más información acerca de la creación de una regla de EventBridge, consulte Introducción a Amazon EventBridge en la Guía del usuario de Amazon EventBridge.

```
{
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Task State Change"
    ],
    "detail": {
        "clusterArn": [
            "arn:aws:ecs:us-west-2:111122223333:cluster/default"
        ]
    }
}
```

Creación de un clúster de Amazon ECS para el tipo de lanzamiento de Fargate

Cree un clúster para definir la infraestructura en la que se ejecutan sus tareas y servicios.

Antes de comenzar, asegúrese de haber seguido los pasos que se detallan en Configuración para utilizar Amazon ECS y de asignar el permiso de IAM adecuado. Para obtener más información, consulte the section called “Ejemplos de clústeres de Amazon ECS”. La consola de Amazon ECS crea los recursos que necesita un clúster de Amazon ECS mediante la creación de una pila de AWS CloudFormation.

La consola asocia automáticamente los proveedores de capacidad de Fargate y Fargate Spot al clúster.

Puede modificar las siguientes opciones:

- Agregue un espacio de nombres al clúster.

  Un espacio de nombres permite que los servicios que cree en el clúster puedan conectarse a los demás servicios del espacio de nombres sin configuración adicional. Para obtener más información, consulte Interconexión de los servicios de Amazon ECS.
- Active Información de contenedores con capacidad de observabilidad mejorada, o Información de contenedores.

Información de contenedores de CloudWatch recopila, agrega y resume métricas y registros de las aplicaciones y microservicios en contenedores. Container Insights también proporciona información de diagnóstico, como, por ejemplo, errores de reinicio de contenedores, que usa para aislar problemas y solucionarlos rápidamente. Para obtener más información, consulte the section called "Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada".

El 2 de diciembre de 2024, AWS publicó Información de contenedores con observabilidad mejorada para Amazon ECS. Esta versión admite la observabilidad mejorada de los clústeres de Amazon ECS que utilizan los tipos de lanzamiento de Amazon EC2 y Fargate. Después de configurar Información de contenedores con capacidad de observabilidad mejorada en Amazon ECS, Información de contenedores recopilará automáticamente telemetría de infraestructura detallada desde el nivel de clúster hasta el nivel de contenedor en el entorno, y mostrará los datos en paneles con diversas métricas y dimensiones. A continuación, podrá utilizar estos paneles listos para usar en la consola de Información de contenedores y comprender mejor el estado y el rendimiento de los contenedores, así como mitigar los problemas con mayor rapidez mediante la identificación de anomalías.

Recomendamos que utilice Información de contenedores con capacidad de observabilidad mejorada en lugar de Información de contenedores, ya que proporciona una visibilidad detallada del entorno de contenedores y reduce el tiempo medio de resolución.

- Agregue etiquetas que le ayuden a identificar el clúster.
- Asigne una clave AWS KMS para el almacenamiento administrado. Para obtener información acerca de la creación de una clave, consulte Creación de una clave KMS en la Guía del usuario de AWS Key Management Service.
- Asigne una clave AWS KMS para su almacenamiento efímero de Fargate. Para obtener información acerca de la creación de una clave, consulte Creación de una clave KMS en la Guía del usuario de AWS Key Management Service.

Procedimiento

Para crear un nuevo clúster (consola de Amazon ECS)

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, seleccione la región a utilizar.
3. En el panel de navegación, seleccione Clusters (Clústeres).

4. En la página Clusters (Clústeres), elija Create Cluster (Crear clúster).
5. En Configuraciones del clúster, configure lo siguiente:
   - En Nombre del clúster, escriba un nombre único.

     El nombre puede contener hasta 255 letras (minúsculas y mayúsculas), números y guiones.
   - (Opcional) Para que el espacio de nombre utilizado en Service Connect sea diferente del nombre del clúster, en Espacio de nombre, escriba un nombre único.
6. (Opcional) Utilice Información de contenedores, amplíe Supervisión y, a continuación, seleccione una de las siguientes opciones:
   - Para usar Información de contenedores con observabilidad mejorada, como se recomienda, elija Información de contenedores con observabilidad mejorada.
   - Para usar Información de contenedores, seleccione Información de contenedores.
7. (Opcional) En Cifrado, puede configurar lo siguiente:
   - Cifre sus datos en el almacenamiento efímero de Fargate. En Cifrado, para Almacenamiento efímero de Fargate, introduzca el ARN de la clave AWS KMS que desea utilizar para cifrar los datos del almacenamiento efímero de Fargate.
   - Cifre los datos en el almacenamiento administrado. En Cifrado, para Almacenamiento administrado, introduzca el ARN de la clave AWS KMS que desea utilizar para cifrar los datos del almacenamiento administrado.
8. (Opcional) Para ayudar a identificar el clúster, expanda Tags (Etiquetas) y, a continuación, configure sus etiquetas.

   [Añadir una etiqueta] Elija Add tag (Añadir etiqueta) y haga lo siguiente:
   - En Key (Clave), escriba el nombre de la clave.
   - En Valor, escriba el valor de la clave.

   [Eliminar una etiqueta] Elija Eliminar a la derecha de la clave y el valor de la etiqueta.
9. Seleccione Create (Crear).

Pasos a seguir a continuación

Tras crear el clúster, puede crear definiciones de tareas para sus aplicaciones y, a continuación, ejecutarlas como tareas independientes o como parte de un servicio. Para obtener más información, consulte los siguientes temas:

- Definiciones de tareas de Amazon ECS
- Ejecución de una aplicación como tarea de Amazon ECS
- Creación de una implementación de actualización continua de Amazon ECS

Proveedores de capacidad de Amazon ECS para el tipo de lanzamiento de EC2

Cuando utiliza instancias de Amazon EC2 para su capacidad, utiliza grupos de escalado automático para administrar las instancias de Amazon EC2 registradas en sus clústeres. El escalado automático lo ayuda a garantizar que cuenta con la cantidad correcta de instancias de Amazon EC2 disponibles para gestionar la carga de su aplicación.

Puede utilizar la característica de escalado administrado para que Amazon ECS administre las acciones de reducción o escalado horizontal del grupo de escalado automático o puede administrar las acciones de escalado usted mismo. Para obtener más información, consulte Administración automática de la capacidad de Amazon ECS con el escalado automático de clústeres.

Se recomienda crear un nuevo grupo de escalado automático vacío. Si utiliza un grupo de escalado automático existente, es posible que en el proveedor de capacidad no se registren correctamente las instancias de Amazon EC2 asociadas al grupo que ya se estaban ejecutando y se habían registrado en un clúster de Amazon ECS antes de utilizar el grupo de escalado automático para crear un proveedor de capacidad. Esto puede causar problemas al usar el proveedor de capacidad en una estrategia de proveedores de capacidad. Utilice `DescribeContainerInstances` para confirmar si una instancia de contenedor está asociada a un proveedor de capacidad o no.

> **Note**
>
> Para crear un grupo de Auto Scaling vacío, establezca el recuento deseado en cero. Después de crear el proveedor de capacidad y asociarlo a un clúster, puede escalarlo horizontalmente.

Cuando utiliza la consola de Amazon ECS, el servicio crea una plantilla de lanzamiento de Amazon EC2 y un grupo de escalado automático en su nombre como parte de la pila de AWS CloudFormation. Llevan el prefijo `EC2ContainerService-<ClusterName>`. Puede utilizar el grupo de escalado automático como proveedor de capacidad para ese clúster.

Le recomendamos que utilice el drenaje de instancias administradas para permitir la finalización correcta de las instancias de Amazon EC2 sin interrumpir sus cargas de trabajo. Esta característica está activada de forma predeterminada. Para obtener más información, consulte Detención segura de las cargas de trabajo de Amazon ECS que se ejecutan en instancias de EC2

Tenga en cuenta lo siguiente cuando utilice los proveedores de capacidad de grupos de escalado automático en la consola:

- Un grupo de Auto Scaling debe tener un `MaxSize` mayor que cero para poder realizar un escalado horizontal.
- El grupo de Auto Scaling no puede tener configuración de ponderación de instancias.
- Si el grupo de escalado automático no se puede escalar horizontalmente para incorporar la cantidad de tareas ejecutadas, las tareas no pueden realizar la transición más allá del estado `PROVISIONING`.
- No modifique el recurso de política de escalado asociado a los grupos de escalado automático administrados por los proveedores de capacidad.
- Si el escalado administrado está activado al crear un proveedor de capacidad, el recuento deseado del grupo de escalado automático se puede establecer en `0`. Cuando se activa el escalado administrado, Amazon ECS administra las acciones de reducción horizontal y escalado horizontal del grupo de escalado automático.
- Debe asociar un proveedor de capacidad a un clúster para poder asociarlo a la estrategia de proveedores de capacidad.
- Puede especificar un máximo de 20 proveedores de capacidad para una estrategia de proveedores de capacidad.
- No puede actualizar un servicio que utiliza un proveedor de capacidad de grupos de escalado automático para que utilice un proveedor de capacidad de Fargate. En caso de que sea lo contrario, tampoco puede hacerlo.
- En una estrategia de proveedores de capacidad, si no se especifica ningún valor `weight` para un proveedor de capacidad en la consola, entonces se utiliza el valor predeterminado `1`. Si utiliza la API o la AWS CLI, se utiliza el valor predeterminado `0`.

- Cuando se especifican varios proveedores de capacidad dentro de una estrategia de proveedores de capacidad, al menos uno de los proveedores de capacidad deberá tener un valor de peso superior a cero. Los proveedores de capacidad con un peso de cero no se usan para realizar tareas. Si especifica varios proveedores de capacidad en una estrategia en la que todos tienen el mismo peso de 0, se producirá un error en cualquiera de las acciones `RunTask` o `CreateService` que utilicen la estrategia de proveedores de capacidad.
- En una estrategia de proveedores de capacidad, solo un proveedor de capacidad puede tener un valor base definido. Si no se especifica ningún valor base, se utiliza el valor predeterminado 0.
- Un clúster puede contener una combinación de proveedores de capacidad del grupo de escalado automático y proveedores de capacidad de Fargate. Sin embargo, una estrategia de proveedores de capacidad solo puede incluir proveedores de capacidad de grupo de escalado automático o de Fargate, pero no ambos.
- Un clúster puede contener una combinación de servicios y tareas independientes que utilicen proveedores de capacidad y tipos de lanzamiento. Un servicio se puede actualizar para que utilice una estrategia de proveedores de capacidad en lugar de un tipo de lanzamiento. Sin embargo, al hacerlo, debe forzar una nueva implementación.
- Amazon ECS admite grupos de calentamiento de Amazon EC2 Auto Scaling. Un grupo de calentamiento es un grupo de Amazon EC2 instances (Instancias de Amazon EC2) inicializadas previamente listas para ponerse en servicio. Siempre que su aplicación necesita escalar horizontalmente, Amazon EC2 Auto Scaling utiliza las instancias preinicializadas del grupo de calentamiento en lugar de lanzar instancias en frío. Esto permite ejecutar cualquier proceso de inicialización final antes de que la instancia entre en servicio. Para obtener más información, consulte Configuración de instancias preinicializadas para el grupo de escalado automático de Amazon ECS.

Para obtener más información acerca de cómo crear plantillas de lanzamiento para Amazon EC2 Auto Scaling, consulte Plantillas de lanzamiento de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling. Para obtener más información acerca de cómo crear un grupo de Amazon EC2 Auto Scaling, consulte Grupos de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling.

Consideraciones de seguridad en instancias de contenedor de Amazon EC2 para Amazon ECS

Debe tener en cuenta una única instancia de contenedor y su acceso dentro de su modelo de amenazas. Por ejemplo, es posible que una sola tarea afectada aproveche los permisos de IAM de una tarea no infectada en la misma instancia.

Le recomendamos que utilice el siguiente procedimiento para evitarlo:

- No utilice los privilegios de administrador al ejecutar sus tareas.
- Asigne un rol de tarea con acceso de privilegio mínimo a sus tareas.

  El agente de contenedor crea automáticamente un token con un ID de credencial único que se utiliza para acceder a los recursos de Amazon ECS.
- Para impedir que los contenedores de las tareas que usan el modo de red `awsvpc` obtengan acceso a la información sobre credenciales proporcionada al perfil de instancia de Amazon EC2, pero dejar los permisos proporcionados por el rol de tarea, establezca la variable de configuración del agente `ECS_AWSVPC_BLOCK_IMDS` en verdadero en el archivo de configuración del agente y reinícielo.
- Utilice la supervisión en tiempo de ejecución de Amazon GuardDuty para detectar amenazas para los clústeres y contenedores de su entorno de AWS. La supervisión en tiempo de ejecución utiliza un agente de seguridad de GuardDuty que agrega visibilidad en tiempo de ejecución a las cargas de trabajo individuales de Amazon ECS (por ejemplo, el acceso a los archivos, la ejecución de procesos y las conexiones de red). Para obtener más información, consulte GuardDuty Runtime Monitoring en la Guía del usuario de GuardDuty.

Creación de un clúster de Amazon ECS para el tipo de lanzamiento de Amazon EC2

Cree un clúster para definir la infraestructura en la que se ejecutan sus tareas y servicios.

Antes de comenzar, asegúrese de haber seguido los pasos que se detallan en Configuración para utilizar Amazon ECS y de asignar el permiso de IAM adecuado. Para obtener más información, consulte the section called “Ejemplos de clústeres de Amazon ECS”. La consola de Amazon ECS ofrece una forma sencilla de crear los recursos que necesita un clúster de Amazon ECS mediante la creación de una pila de AWS CloudFormation.

Para simplificar al máximo el proceso de creación del clúster, la consola cuenta con selecciones predeterminadas para muchas de las opciones que describimos a continuación. También hay paneles de ayuda disponibles para la mayoría de las secciones de la consola, que proporcionan más contexto.

Puede registrar instancias de Amazon EC2 durante la creación del clúster o registrar instancias adicionales en el clúster después de crearlo.

Puede modificar las siguientes opciones predeterminadas:

- Cambiar las subredes en las que se lanzan las instancias
- Cambiar los grupos de seguridad que se utilizan para controlar el tráfico a las instancias de contenedor
- Agregue un espacio de nombres al clúster.

  Un espacio de nombres permite que los servicios que cree en el clúster puedan conectarse a los demás servicios del espacio de nombres sin configuración adicional. Para obtener más información, consulte Interconexión de los servicios de Amazon ECS.
- Active Información de contenedores con capacidad de observabilidad mejorada, o Información de contenedores.

  Información de contenedores de CloudWatch recopila, agrega y resume métricas y registros de las aplicaciones y microservicios en contenedores. Container Insights también proporciona información de diagnóstico, como, por ejemplo, errores de reinicio de contenedores, que usa para aislar problemas y solucionarlos rápidamente. Para obtener más información, consulte the section called “Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada”.

  El 2 de diciembre de 2024, AWS publicó Información de contenedores con observabilidad mejorada para Amazon ECS. Esta versión admite la observabilidad mejorada de los clústeres de Amazon ECS que utilizan los tipos de lanzamiento de Amazon EC2 y Fargate. Después de configurar Información de contenedores con capacidad de observabilidad mejorada en Amazon ECS, Información de contenedores recopilará automáticamente telemetría de infraestructura detallada desde el nivel de clúster hasta el nivel de contenedor en el entorno, y mostrará los datos en paneles con diversas métricas y dimensiones. A continuación, podrá utilizar estos paneles listos para usar en la consola de Información de contenedores y comprender mejor el estado y el rendimiento de los contenedores, así como mitigar los problemas con mayor rapidez mediante la identificación de anomalías.

Recomendamos que utilice Información de contenedores con capacidad de observabilidad mejorada en lugar de Información de contenedores, ya que proporciona una visibilidad detallada del entorno de contenedores y reduce el tiempo medio de resolución.

- Asigne una clave AWS KMS para el almacenamiento administrado. Para obtener información acerca de la creación de una clave, consulte Creación de una clave KMS en la Guía del usuario de AWS Key Management Service.
- Asigne una clave AWS KMS para su almacenamiento efímero de Fargate. Para obtener información acerca de la creación de una clave, consulte Creación de una clave KMS en la Guía del usuario de AWS Key Management Service.
- Agregue etiquetas que le ayuden a identificar el clúster.

Opciones de grupo de Auto Scaling

Cuando utiliza Amazon EC2 instances (Instancias de Amazon EC2), debe especificar un grupo de Auto Scaling para administrar la infraestructura en la que se ejecutan sus tareas y servicios.

Cuando elige crear un nuevo grupo de Auto Scaling, se configura automáticamente para el siguiente comportamiento:

- Amazon ECS administra las acciones de reducción y escalado horizontal del grupo de Auto Scaling.
- Amazon ECS impide que las instancias de Amazon EC2 que contiene tareas en un grupo de Auto Scaling se terminen durante una acción de reducción horizontal. Para obtener más información, consulte Protección de instancias en la Guía del usuario de AWS Auto Scaling.

Puede configurar las siguientes propiedades de grupo de Auto Scaling que determinan el tipo y el número de instancias que se van a lanzar para el grupo:

- La AMI optimizada para Amazon ECS.
- El tipo de instancia.
- El par de claves de SSH que demuestra su identidad cuando se conecta a la instancia. Para obtener información acerca cómo crear claves de SSH, consulte Pares de claves de Amazon EC2 e instancias de Linux en la Guía del usuario de Amazon EC2.
- Número mínimo de instancias para lanzar en el grupo de Auto Scaling.
- El número máximo de instancias que se inician para el grupo de Auto Scaling.

Para que el grupo se escale horizontalmente, el máximo debe ser superior a 0.

Amazon ECS crea una plantilla de lanzamiento de Amazon EC2 Auto Scaling y un grupo de Auto Scaling en su nombre como parte de la pila AWS CloudFormation. Los valores especificados para la AMI, los tipos de instancias y el par de claves de SSH forman parte de la plantilla de lanzamiento. Las plantillas tienen el prefijo `EC2ContainerService-<ClusterName>`, por lo que son fáciles de identificar. Los grupos de Auto Scaling llevan el prefijo `<ClusterName>-ECS-Infra-ECSAutoScalingGroup`.

Las instancias lanzadas para el grupo de Auto Scaling utilizan la plantilla de lanzamiento.

Opciones de red

De forma predeterminada, las instancias se lanzan en las subredes predeterminadas de la región. Se utilizan los grupos de seguridad, que controlan el tráfico hacia las instancias de contenedor, actualmente asociados a las subredes. Puede cambiar las subredes y los grupos de seguridad de las instancias.

Puede elegir una subred existente. Puede usar un grupo de seguridad existente o crear uno nuevo. Al crear un nuevo grupo de seguridad, debe especificar al menos una regla de entrada.

Las reglas de entrada determinan qué tráfico puede llegar a las instancias de contenedor e incluyen las siguientes propiedades:

- El protocolo por permitir
- El rango de puertos por permitir
- El tráfico entrante (origen)

Para permitir el tráfico entrante desde una dirección o bloque de CIDR específicos, utilice Personalizado en Origen con el CIDR permitido.

Para permitir el tráfico entrante desde todos los destinos, utilice Anywhere en Origen. Esta opción agrega automáticamente el bloque IPv4 0.0.0.0/0 de CIDR y el bloque IPv6 ::/0 de CIDR.

Para permitir el tráfico entrante desde su equipo local, utilice Grupo de origen en Origen. Esto agrega automáticamente la dirección IP actual de la computadora local como el origen permitido.

Para crear un nuevo clúster (consola de Amazon ECS)

Antes de empezar, asigne el permiso de IAM correspondiente. Para obtener más información, consulte the section called “Ejemplos de clústeres de Amazon ECS”.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, seleccione la región a utilizar.
3. En el panel de navegación, seleccione Clusters (Clústeres).
4. En la página Clusters (Clústeres), elija Create Cluster (Crear clúster).
5. En Configuraciones del clúster, configure lo siguiente:
   - En Nombre del clúster, escriba un nombre único.

     El nombre puede contener hasta 255 letras (minúsculas y mayúsculas), números y guiones.
   - (Opcional) Para que el espacio de nombre utilizado en Service Connect sea diferente del nombre del clúster, en Espacio de nombre, escriba un nombre único.
6. Para agregar instancias de Amazon EC2 al clúster, expanda Infraestructura y, a continuación, seleccione Instancias de Amazon EC2.

   A continuación, configure el grupo de Auto Scaling que actúa como proveedor de capacidad:

   a. Para utilizar un grupo de Auto Scaling existente, desde Auto Scaling group (ASG) (Grupo de Auto Scaling), seleccione el grupo.
   b. Para crear un grupo de Auto Scaling, desde Auto Scaling group (ASG) (Grupo de Auto Scaling), seleccione Create new group (Crear nuevo grupo) y, a continuación, proporcione los siguientes detalles sobre el grupo:
      - En Modelo de aprovisionamiento, seleccione si quiere utilizar instancias bajo demanda o instancias de spot.
      - Si decide utilizar instancias de spot, en Estrategia de asignación, elija qué grupos de capacidad de spot (tipos de instancias y zonas de disponibilidad) se utilizan para las instancias.

        Para la mayoría de las cargas de trabajo, puede elegir Capacidad de precio optimizada.

        Para obtener más información, consulte Estrategias de asignación de instancias de spot en la Guía del usuario de Amazon EC2

- Para la Imagen de máquina de Amazon (AMI) de la instancia de contenedor, elija la AMI optimizada para Amazon ECS para las instancias de grupo de escalado automático.
- Para EC2 instance type (Tipo de instancia EC2), elija el tipo de instancia para sus cargas de trabajo.

  El escalado administrado funciona mejor si el grupo de Auto Scaling utiliza los mismos tipos de instancia o similares.
- En Rol de instancia de EC2, elija un rol de instancia de contenedor existente o puede crear uno nuevo.

  Para obtener más información, consulte Rol de IAM de instancia de contenedor de Amazon ECS.
- Para Capacity (Capacidad), introduzca el número mínimo y el número máximo de instancias que va a lanzar en el grupo de Auto Scaling.
- Para Par de clave de SSH, elija el par que demuestre su identidad cuando se conecta a la instancia.
- Para permitir una imagen y un almacenamiento más grandes, en Tamaño del volumen de EBS raíz, ingrese el valor en GIB.

7. (Opcional) Para cambiar la VPC y las subredes, en Redes para instancias de Amazon EC2, realice cualquiera de las siguientes operaciones:

   - Para eliminar una subred, en Subnets (Subredes), elija Xpara cada subred que desea eliminar.
   - Para cambiar a una VPC distinta de la VPC predeterminada, en VPC, elija una VPC existente y, a continuación, en Subredes, seleccione las subredes.
   - Elija los grupos de seguridad. En Grupo de seguridad, elija una de las siguientes opciones:
     - Para utilizar un grupo de seguridad existente, elija Utilizar un grupo de seguridad existente y, a continuación, elija el grupo de seguridad.
     - Para crear un grupo de seguridad, elija Crear un nuevo grupo de seguridad. A continuación, elija Agregar regla para cada regla de entrada).

       Para obtener información sobre las reglas de entrada, consulte Opciones de red.
   - Para asignar automáticamente direcciones IP públicas a sus instancias de contenedor de Amazon EC2, en Asignación automática de IP pública, elija una de las siguientes opciones:

- Utilizar la configuración de subred: asigne una dirección IP pública a las instancias cuando la subred en la que se lanzan las instancias sea una subred pública.
- Activar: asigna una dirección IP pública a las instancias.

8. (Opcional) Utilice Información de contenedores, amplíe Supervisión y, a continuación, seleccione una de las siguientes opciones:

   - Para usar Información de contenedores con observabilidad mejorada, como se recomienda, elija Información de contenedores con observabilidad mejorada.
   - Para usar Información de contenedores, seleccione Información de contenedores.

9. (Opcional)

   Si utiliza la supervisión de tiempo de ejecución con la opción manual y quiere que GuardDuty supervise este clúster, seleccione Agregar etiqueta y haga lo siguiente:

   - En Clave, ingrese **`guardDutyRuntimeMonitoringManaged`**.
   - En Valor, introduzca **`true`**.

10. (Opcional) Cifre los datos del almacenamiento administrado. En Cifrado, para Almacenamiento administrado, introduzca el ARN de la clave AWS KMS que desea utilizar para cifrar los datos del almacenamiento administrado.

11. (Opcional) Para administrar las etiquetas de clúster, expanda Tags (Etiquetas) y, a continuación, realice una de las siguientes operaciones:

    [Añadir una etiqueta] Elija Add tag (Añadir etiqueta) y haga lo siguiente:

    - En Key (Clave), escriba el nombre de la clave.
    - En Valor, escriba el valor de la clave.

    [Eliminar una etiqueta] Elija Eliminar a la derecha de la clave y el valor de la etiqueta.

12. Seleccione Create (Crear).

Pasos a seguir a continuación

Tras crear el clúster, puede crear definiciones de tareas para sus aplicaciones y, a continuación, ejecutarlas como tareas independientes o como parte de un servicio. Para obtener más información, consulte los siguientes temas:

- Definiciones de tareas de Amazon ECS
- Ejecución de una aplicación como tarea de Amazon ECS
- Creación de una implementación de actualización continua de Amazon ECS

Administración automática de la capacidad de Amazon ECS con el escalado automático de clústeres

Amazon ECS puede administrar el escalado de las instancias de Amazon EC2 registradas en el clúster. A esto se lo conoce como escalado automático de clústeres de Amazon ECS. Activa el escalado administrado al crear el proveedor de capacidad del grupo de escalado automático de Amazon ECS. A continuación, establece un porcentaje objetivo (`targetCapacity`) para la utilización de instancias en este grupo de escalado automático. Amazon ECS crea dos métricas personalizadas de CloudWatch y una política de escalado de seguimiento de destino para el grupo de escalado automático. A continuación, Amazon ECS administra las acciones de escalado y reducción horizontal en función de la utilización de recursos que usan las tareas.

Para cada proveedor de capacidad de grupo de escalado automático asociado a un clúster, Amazon ECS crea y administra los siguientes recursos:

- Una alarma CloudWatch de bajo valor métrico
- Una alarma CloudWatch de alto valor métrico
- Una política de escalado de seguimiento de destino

> **Note**
>
> Amazon ECS crea la política de escalado de seguimiento de destino y la asocia al grupo de escalado automático. Para actualizar la política de escalado de seguimiento de destino, debe actualizar la configuración de escalado administrada por el proveedor de capacidad en lugar de actualizar la política de escalado directamente.

Cuando se desactiva el escalado administrado o se desasocia el proveedor de capacidad de un clúster, Amazon ECS elimina tanto las métricas de CloudWatch como los recursos de política de escalado de seguimiento de destino.

Amazon ECS utiliza las siguientes métricas para determinar qué acciones se deben llevar a cabo:

`CapacityProviderReservation`

El porcentaje de instancias de contenedores que se utilizan para un proveedor de capacidad específico. Amazon ECS genera esta métrica.

Amazon ECS establece el valor `CapacityProviderReservation` en un número entre 0 y 100. Amazon ECS utiliza la siguiente fórmula para representar la proporción de la capacidad que queda en el grupo de escalado automático. A continuación, Amazon ECS publica la métrica en CloudWatch. Para obtener más información sobre cómo se calcula la métrica, consulte Deep Dive on Amazon ECS Cluster Auto Scaling.

```
CapacityProviderReservation = (number of instances needed) / (number of running
 instances) x 100
```

`DesiredCapacity`

La cantidad de capacidad del grupo de escalado automático. Esta métrica no se publica en CloudWatch.

Amazon ECS publica la métrica `CapacityProviderReservation` en CloudWatch en el espacio de nombres `AWS/ECS/ManagedScaling`. La métrica `CapacityProviderReservation` hace que se produzca una de las siguientes acciones:

El valor **`CapacityProviderReservation`** es igual a **`targetCapacity`**

El grupo de escalado automático no necesita escalar ni reducirse horizontalmente. Se alcanzó el porcentaje de utilización objetivo.

El valor **`CapacityProviderReservation`** es superior a **`targetCapacity`**

Hay más tareas que utilizan un porcentaje de capacidad superior a su porcentaje de `targetCapacity`. El aumento del valor de la métrica `CapacityProviderReservation` hace que la alarma de CloudWatch asociada se active. Esta alarma actualiza el valor `DesiredCapacity` del grupo de Auto Scaling. El grupo de Auto Scaling utiliza este valor para lanzar instancias EC2 y, a continuación, registrarlas en el clúster.

Cuando la `targetCapacity` es el valor predeterminado del 100 %, las nuevas tareas permanecen en el estado `PENDING` durante el escalado horizontal porque no hay capacidad disponible en las instancias para ejecutarlas. Una vez que las nuevas instancias se registren en ECS, estas tareas comenzarán en las nuevas instancias.

El valor **`CapacityProviderReservation`** es inferior a **`targetCapacity`**

Hay menos tareas que utilizan un porcentaje de la capacidad inferior al porcentaje de `targetCapacity` y hay, al menos, una instancia que se puede terminar. El aumento del valor de la métrica de `CapacityProviderReservation` hace que la alarma de CloudWatch asociada se active. Esta alarma actualiza el valor `DesiredCapacity` del grupo de Auto Scaling. El grupo de Auto Scaling utiliza este valor para terminar las instancias de contenedor de EC2 y, a continuación, anularlas del registro del clúster.

El grupo de escalado automático utiliza políticas de terminación de grupos para determinar qué instancias termina primero durante los eventos de reducción horizontal. Además, evita las instancias con la configuración de protección de reducción horizontal de instancias activada. El escalado automático de clústeres puede administrar qué instancias tienen la configuración de protección de escalado horizontal si activa la protección de terminación administrada. Para obtener más información sobre la protección de terminación administrada, consulte Control de las instancias que Amazon ECS termina. Para obtener más información sobre cómo los grupos de escalado automático terminan instancias, consulte Controlar qué instancias de escalado automático terminan durante el escalado horizontal en la guía del usuario de Amazon EC2 Auto Scaling.

Cuando se utiliza el escalado automático de clústeres, se debe tener en cuenta lo siguiente:

- No cambie ni administre la capacidad deseada para el grupo de escalado automático asociado a un proveedor de capacidad con una política de escalado que no sea la que Amazon ECS administra.
- Cuando Amazon ECS se escala horizontalmente desde 0 instancias, lanza automáticamente 2 instancias.
- Amazon ECS utiliza el rol de IAM vinculado al servicio `AWSServiceRoleForECS` para los permisos que necesita para llamar a AWS Auto Scaling en su nombre. Para obtener más información, consulte Uso de roles vinculados al servicio para Amazon ECS.
- Cuando se utilizan proveedores de capacidad con grupos de escalado automático, el usuario, grupo o rol que crea el proveedor de capacidad necesita el permiso `autoscaling:CreateOrUpdateTags`. Esto se debe a que Amazon ECS agrega una etiqueta al grupo de Auto Scaling cuando lo asocia al proveedor de capacidad.

> **Important**
>
> Asegúrese de que las herramientas que utilice no eliminen la etiqueta `AmazonECSManaged` del grupo de escalado automático. Si se elimina esta etiqueta, Amazon ECS no podrá administrar el escalado.

- El escalado automático de clústeres no modifica MinimumCapacity ni MaximumCapacity para el grupo. Para que el grupo escale horizontalmente, el valor de MaximumCapacity (Capacidad máxima) debe ser mayor o igual que 0.
- Un proveedor de capacidad solo se puede conectar a un clúster al mismo tiempo cuando el escalado automático (escalado administrado) está activado. Si el proveedor de capacidad ha desactivado el escalado administrado, puede asociarlo a varios clústeres.
- Cuando se desactiva el escalado administrado, el proveedor de capacidad no realiza operaciones de reducción horizontal ni escalado horizontal. Puede utilizar una estrategia de proveedores de capacidad para equilibrar las tareas entre los proveedores de capacidad.
- La estrategia de `binpack` es la más eficiente en términos de capacidad.
- Cuando la capacidad de destino es inferior al 100 %, la estrategia de colocación debe utilizar la estrategia de `binpack` sin que la estrategia de `spread` tenga un orden superior. Esto evita que el proveedor de capacidad se escale horizontalmente hasta que cada tarea tenga una instancia dedicada o se alcance el límite.

Active el escalado automático de clústeres

Puede activar el escalado automático de clústeres con la consola o la AWS CLI.

Cuando el servicio crea un clúster para el tipo de lanzamiento de EC2 con la consola, Amazon ECS crea un grupo de escalado automático en su nombre y establece la capacidad de destino. Para obtener más información, consulte Creación de un clúster de Amazon ECS para el tipo de lanzamiento de Amazon EC2.

También puede crear un grupo de escalado automático y, luego, asignarlo a un clúster. Para obtener más información, consulte Actualización de un proveedor de capacidad de Amazon ECS.

Al utilizar la AWS CLI, después de crear el clúster

1. Antes de crear el proveedor de capacidad, debe crear un grupo de escalado automático. Para obtener más información, consulte Grupos de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling.
2. Utilice `put-cluster-capacity-providers` para modificar el proveedor de capacidad del clúster. Para obtener más información, consulte Activación del escalado automático de clústeres de Amazon ECS.

Optimización del escalado automático de clústeres de Amazon ECS

Los clientes que ejecutan Amazon ECS en Amazon EC2 pueden aprovechar el escalado automático de clústeres para administrar el escalado de los grupos de Amazon EC2 Auto Scaling. Con el escalado automático de clústeres, puede configurar Amazon ECS para escalar automáticamente el grupo de escalado automático y centrarse únicamente en ejecutar las tareas. Amazon ECS garantizará que el grupo de escalado automático escale en forma ascendente y descendente según sea necesario, sin necesidad de ninguna otra intervención. Los proveedores de capacidad de Amazon ECS están acostumbrados a administrar la infraestructura del clúster al garantizar que haya instancias de contenedor suficientes para satisfacer las exigencias de la aplicación. Para obtener información sobre cómo funciona el escalado automático de clústeres, consulte Deep Dive on Amazon ECS Cluster Auto Scaling.

El escalado automático de clústeres se basa en una integración que se respalda en CloudWatch con el grupo de escalado automático para ajustar la capacidad del clúster. Por lo tanto, cuenta con una latencia inherente asociada.

- Publicación de las métricas de CloudWatch
- El tiempo que tardó la métrica `CapacityProviderReservation` en vulnerar las alarmas de CloudWatch (tanto altas como bajas).
- El tiempo que tardó una instancia de Amazon EC2 recién lanzada en calentarse. Puede realizar las siguientes acciones para que el escalado automático de clústeres responda mejor y, de este modo, las implementaciones sean más rápidas:

Tamaños de escalado por pasos del proveedor de capacidad

Los proveedores de capacidad de Amazon ECS aumentarán o reducirán las instancias de contenedor para satisfacer las demandas de la aplicación. La cantidad mínima de instancias que Amazon ECS lanzará se establece en 1 de forma predeterminada. Esto puede agregar tiempo adicional a las implementaciones si se necesitan varias instancias para realizar las tareas

pendientes. Puede aumentar el valor de `minimumScalingStepSize` a través de la API de Amazon ECS para aumentar el número mínimo de instancias que Amazon ECS escala o reduce horizontalmente cada vez. Un valor de `maximumScalingStepSize` demasiado bajo puede limitar el número de instancias de contenedor que se escalan o reducen horizontalmente cada vez, lo que puede ralentizar las implementaciones.

> **Note**
>
> Actualmente, esta configuración solo está disponible a través de las API `CreateCapacityProvider` o `UpdateCapacityProvider`.

Periodo de preparación de instancias

El periodo de preparación de instancias es el periodo de tiempo después del que una instancia de Amazon EC2 recién lanzada puede contribuir a las métricas de CloudWatch para el grupo de escalado automático. Después de que finaliza el periodo de preparación especificado, la instancia se contabiliza para las métricas agregadas del grupo de escalado automático, y el escalado automático de clústeres continúa con su siguiente iteración de cálculos para estimar la cantidad de instancias necesarias.

El valor predeterminado de `instanceWarmupPeriod` es de 300 segundos, que puede configurar con un valor inferior a través de las API `CreateCapacityProvider` o `UpdateCapacityProvider` para lograr un escalado con mayor capacidad de respuesta. Le recomendamos que establezca el valor en más de 60 segundos para evitar aprovisionar en exceso.

Capacidad sobrante

Si su proveedor de capacidad no tiene instancias de contenedor disponibles para colocar tareas, debe aumentar (escalar horizontalmente) la capacidad del clúster mediante el lanzamiento de instancias de Amazon EC2 sobre la marcha y esperar a que se inicien antes de poder lanzar contenedores en ellas. Esto puede reducir considerablemente la frecuencia de lanzamiento de las tareas. Dispone de dos opciones aquí.

En este caso, disponer de capacidad sobrante de Amazon EC2 ya lanzada y lista para ejecutar tareas aumentará la frecuencia efectiva de lanzamiento de tareas. Puede usar la configuración `Target Capacity` para indicar que desea mantener la capacidad sobrante en sus clústeres. Por ejemplo, si se establece el valor de `Target Capacity` en un 80 %, indica que el clúster necesita un 20 % de capacidad sobrante en todo momento. Esta capacidad sobrante puede permitir que

cualquier tarea independiente se inicie inmediatamente, lo que garantiza que el lanzamiento de las tareas no se limite. La desventaja de este enfoque es el posible aumento de los costos derivados de mantener la capacidad sobrante de los clústeres.

Un enfoque alternativo que puede considerar es agregar margen de maniobra a su servicio, no al proveedor de capacidad. Esto significa que, en lugar de reducir la configuración `Target Capacity` para lanzar la capacidad sobrante, puede aumentar la cantidad de réplicas de su servicio al modificar la métrica de escalado de seguimiento de destino o los umbrales de escalado por pasos del escalado automático del servicio. Tenga en cuenta que este enfoque solo será útil para cargas de trabajo con picos de actividad, pero no tendrá ningún efecto cuando implemente nuevos servicios y pase de 0 a N tareas por primera vez. Para obtener más información sobre las políticas de escalado relacionadas, consulte Políticas de escalado de seguimiento de destino o Políticas de escalado por pasos en la Guía para desarrolladores de Amazon Elastic Container Service.

Comportamiento de escalado administrado de Amazon ECS

Cuando tiene proveedores de capacidad de grupos de escalado automático que usan escalado administrado, Amazon ECS calcula la cantidad óptima de instancias que se van a agregar al clúster y utiliza este valor para determinar cuántas instancias se deben solicitar o liberar.

Comportamiento de escalado horizontal administrado

Amazon ECS selecciona un proveedor de capacidad para cada tarea siguiendo la estrategia del proveedor de capacidad desde el servicio, la tarea independiente o el clúster predeterminado. Amazon ECS sigue el resto de estos pasos para un único proveedor de capacidad.

Los proveedores de capacidad ignoran las tareas que no tienen una estrategia de proveedor de capacidad. Una tarea pendiente que no tenga una estrategia de proveedor de capacidad no hará que ningún proveedor de capacidad escale horizontalmente. Las tareas o los servicios no pueden establecer una estrategia de proveedor de capacidad sin establecer un tipo de lanzamiento.

A continuación, se describe el comportamiento de escalado horizontal en más detalle.

- Agrupe todas las tareas de aprovisionamiento para este proveedor de capacidad de tal modo que cada grupo tenga exactamente los mismos requisitos de recursos.
- Cuando se utilizan varios tipos de instancias en un grupo de escalado automático, los tipos de instancias del grupo de escalado automático se ordenan por sus parámetros. Estos parámetros incluyen vCPU, memoria, interfaces de red elásticas (ENI), puertos y GPU. Se seleccionan los tipos de instancias más pequeños y más grandes para cada parámetro. Para obtener más

información sobre cómo elegir un tipo de instancia, consulte Instancias de contenedor de Amazon EC2 para Amazon ECS.

> **Important**
>
> Si un grupo de tareas tiene requisitos de recursos superiores a los del tipo de instancia más pequeño del grupo de escalado automático, ese grupo de tareas no se puede ejecutar con este proveedor de capacidad. El proveedor de capacidad no escala el grupo de escalado automático. Las tareas permanecen en el estado `PROVISIONING`.
> Para evitar que las tareas permanezcan en el estado `PROVISIONING`, le recomendamos que cree grupos de escalado automático y proveedores de capacidad independientes para diferentes requisitos de recursos mínimos. Cuando ejecute tareas o cree servicios, agregue únicamente proveedores de capacidad a la estrategia del proveedor de capacidad que puedan ejecutar la tarea en el tipo de instancia más pequeño del grupo de escalado automático. Para otros parámetros, puede utilizar restricciones de ubicación.

- Para cada grupo de tareas, Amazon ECS calcula el número de instancias requeridas para ejecutar las tareas no colocadas. Este cálculo utiliza una estrategia `binpack`. Esta estrategia tiene en cuenta los requisitos de vCPU, memoria, interfaces de red elásticas (ENI), puertos y GPU de las tareas. También tiene en cuenta la disponibilidad de recursos de las instancias de Amazon EC2. Los valores de los tipos de instancias más grandes se consideran como el recuento máximo de instancias calculado. Los valores del tipo de instancia más pequeño se utilizan como protección. Si el tipo de instancia más pequeño no puede ejecutar al menos una instancia de la tarea, el cálculo considera que la tarea no es compatible. Como resultado, se excluye del cálculo de escalado horizontal. Cuando ninguna tarea es compatible con el tipo de instancia más pequeño, el escalado automático de clústeres se detiene y el valor `CapacityProviderReservation` se mantiene en el valor `targetCapacity`.
- Amazon ECS publica la métrica `CapacityProviderReservation` a CloudWatch con respecto al `minimumScalingStepSize` si se da alguna de las siguientes circunstancias.
  - El recuento máximo de instancias calculado es inferior al tamaño mínimo del paso de escalado.
  - El valor más bajo de `maximumScalingStepSize` o el recuento máximo de instancias calculado.
- Las alarmas de CloudWatch utilizan la métrica `CapacityProviderReservation` para los proveedores de capacidad. Cuando la métrica `CapacityProviderReservation` es mayor que el valor de `targetCapacity`, las alarmas también aumentan la `DesiredCapacity` del grupo de escalado automático. El valor `targetCapacity` es una configuración de proveedor de capacidad

que se envía a la alarma de CloudWatch durante la fase en que se activa el escalado automático del clúster.

El valor de `targetCapacity` predeterminado es el 100 %.

- El grupo de Auto Scaling lanza instancias de EC2 adicionales. Para evitar el sobreaprovisionamiento, el escalado automático se asegura de que la capacidad de la instancia de EC2 iniciada recientemente se estabilice antes de iniciar nuevas instancias. El escalado automático comprueba si todas las instancias existentes han superado el `instanceWarmupPeriod` (ahora menos el tiempo de lanzamiento de la instancia). El escalado horizontal está bloqueado para las instancias que se encuentran dentro del `instanceWarmupPeriod`.

  El número predeterminado de segundos para que se caliente una instancia recién lanzada es de 300.

Para obtener más información, consulte Análisis detallado del escalado automático de clústeres de Amazon ECS.

Consideraciones de escalado horizontal

Tenga en cuenta lo siguiente para el proceso de escalado horizontal:

- Aunque existen varias restricciones de ubicación, le recomendamos que solo use la restricción de ubicación de tareas `distinctInstance`. Esto evita que el proceso de escalado horizontal se detenga porque está utilizando una restricción de ubicación que no es compatible con las instancias de muestra.
- El escalado administrado funciona mejor si el grupo de Auto Scaling utiliza los mismos tipos de instancia o similares.
- Cuando se requiere un proceso de escalado horizontal y no hay instancias de contenedores en ejecución actualmente, Amazon ECS siempre se escala horizontalmente a dos instancias al principio y, a continuación, realiza procesos de escalado o reducción horizontal adicionales. Cualquier escalado horizontal adicional espera al período de preparación de la instancia. En el caso de los procesos de reducción horizontal, Amazon ECS espera 15 minutos después de un proceso de escalado horizontal, antes de iniciar los procesos de reducción horizontal en todo momento.
- El segundo paso de escalado horizontal debe esperar hasta que `instanceWarmupPeriod` venza, lo cual podría afectar el límite de escalado general. Si necesita reducir este tiempo,

asegúrese de que `instanceWarmupPeriod` sea lo suficientemente grande como para que la instancia de EC2 inicie el agente de Amazon ECS (lo que evita el sobreaprovisionamiento).

- El escalado automático de clústeres admite la configuración de lanzamiento, plantillas de lanzamiento y varios tipos de instancias en el grupo de escalado automático del proveedor de capacidad. También puede utilizar la selección del tipo de instancia basada en atributos sin tener varios tipos de instancias.
- Cuando utilice un grupo de Auto Scaling con tipos de instancias bajo demanda e instancias múltiples o instancias de spot, coloque los tipos de instancias más grandes en la lista de prioridades y no especifique ninguna ponderación. En este momento, no se admite la especificación de ponderaciones. Para obtener más información, consulte Grupos de Auto Scaling con varios tipos de instancias en la Guía del usuario de AWS Auto Scaling.
- A continuación, Amazon ECS lanzará el `minimumScalingStepSize` si el recuento máximo de instancias calculado es menor que el tamaño mínimo del paso de escalado o el `maximumScalingStepSize` o el valor del recuento máximo de instancias calculado, de ambas opciones, la que sea menor.
- Si un servicio de Amazon ECS o `run-task` lanza una tarea y las instancias de contenedor del proveedor de capacidad no tienen recursos suficientes para iniciar la tarea, Amazon ECS limita el número de tareas con este estado para cada clúster e impide que cualquier tarea supere este límite. Para obtener más información, consulte Service Quotas.

Comportamiento de reducción horizontal administrada

Amazon ECS supervisa las instancias de contenedor de cada proveedor de capacidad dentro del clúster. Cuando una instancia de contenedor no ejecuta ninguna tarea, la instancia de contenedor se considera vacía y Amazon ECS inicia el proceso de reducción horizontal.

Las alarmas de reducción horizontal de CloudWatch requieren 15 puntos de datos (15 minutos) antes de que comience el proceso de reducción horizontal del grupo de Auto Scaling. Una vez que se inicia el proceso de reducción horizontal hasta que Amazon ECS necesite reducir el número de instancias de contenedores registradas, el grupo de escalado automático establece el valor `DesireCapacity` para que sea superior a una instancia e inferior al 50 % por minuto.

Cuando Amazon ECS solicita un escalado horizontal (cuando `CapacityProviderReservation` es superior a 100) mientras un proceso de reducción horizontal está en curso, el proceso de reducción se detiene y comenzará desde el principio si es necesario.

A continuación, se describe el comportamiento de reducción horizontal con más detalle:

1. Amazon ECS calcula el número de instancias de contenedores que están vacías. Una instancia de contenedor se considera vacía incluso cuando se están ejecutando tareas de daemon.
2. Amazon ECS establece el valor `CapacityProviderReservation` en un número entre 0 y 100 que utiliza la siguiente fórmula para representar la relación entre el tamaño que debe tener el grupo de escalado automático y su tamaño real, expresado en porcentaje. A continuación, Amazon ECS publica la métrica en CloudWatch. Para obtener más información sobre cómo se calcula la métrica, consulte Profundización en el escalado automático de clústeres de Amazon ECS.

   ```
   CapacityProviderReservation = (number of instances needed) / (number of running
    instances) x 100
   ```

3. La métrica `CapacityProviderReservation` genera una alarma de CloudWatch. Esta alarma actualiza el valor `DesiredCapacity` del grupo de Auto Scaling. A continuación, se lleva a cabo una de las siguientes acciones:
   - Si no utiliza la terminación administrada del proveedor de capacidad, el grupo de escalado automático selecciona las instancias de EC2 mediante la política de terminación de grupos de escalado automático y las termina hasta que el número de instancias de EC2 llegue a la `DesiredCapacity`. A continuación, las instancias de contenedor se anulan del registro del clúster.
   - Si todas las instancias de contenedor utilizan la protección contra terminación administrada, Amazon ECS elimina la protección para la reducción horizontal de las instancias de contenedor que están vacías. El grupo de Auto Scaling podrá terminar las instancias de EC2. A continuación, las instancias de contenedor se anulan del registro del clúster.

Control de las instancias que Amazon ECS termina

> **Important**
>
> Debe activar la protección para la reducción horizontal de instancias de Auto Scaling en el grupo de escalado automático para usar la característica de protección de terminación administrada del escalado automático de clústeres.

La protección contra la terminación administrada permite que el escalado automático de clústeres controle qué instancias se terminan. Cuando utiliza la protección contra la terminación administrada, Amazon ECS solo termina las instancias de EC2 que no tienen ninguna tarea de Amazon ECS en

ejecución. Las tareas que ejecuta un servicio que utiliza la estrategia de programación de `DAEMON` se ignoran y una instancia se puede terminar mediante el escalado automático del clúster, incluso cuando la instancia ejecuta estas tareas. Esto se debe a que todas las instancias del clúster ejecutan estas tareas.

Amazon ECS activa primero la opción de protección contra la reducción horizontal para las instancias de EC2 del grupo de escalado automático. A continuación, Amazon ECS coloca las tareas en las instancias. Cuando se detienen todas las tareas que no son daemon en una instancia, Amazon ECS inicia el proceso de reducción horizontal y desactiva la protección de reducción horizontal de la instancia de EC2. El grupo de Auto Scaling puede terminar la instancia.

La protección para la reducción horizontal en instancias de Auto Scaling controla qué instancias de EC2 se pueden terminar en Auto Scaling. Las instancias con la característica de reducción horizontal activada no se pueden finalizar durante el proceso de reducción horizontal. Para obtener más información sobre la protección para la reducción horizontal en instancias de Auto Scaling, consulte Uso de la protección para la reducción horizontal de instancias en la Guía del usuario de Amazon EC2 Auto Scaling.

Puede establecer el porcentaje de `targetCapacity` para disponer de capacidad sobrante. De este modo, las tareas futuras se inician de forma más rápida porque el grupo de escalado automático no tiene que iniciar más instancias. Amazon ECS utiliza el valor de capacidad de destino para administrar la métrica de CloudWatch que crea el servicio. Amazon ECS administra la métrica de CloudWatch. El grupo de escalado automático se considera un estado estable para que no se requiera ninguna acción de escalado. Los valores pueden ser del 0 al 100 %. Por ejemplo, para configurar Amazon ECS para que 10 % de la capacidad se mantenga libre además de la utilizada por las tareas de Amazon ECS, establezca el valor de capacidad de destino en 90 %. Al establecer el valor `targetCapacity` para un proveedor de capacidad, debe tener en cuenta lo siguiente.

- Un valor `targetCapacity` inferior al 100 % representa la cantidad de capacidad libre (instancias de Amazon EC2) que debe estar presente en el clúster. Capacidad libre significa que no hay tareas en ejecución.
- Restricciones de ubicación, tales como zonas de disponibilidad, sin `binpack` adicional, obliga a Amazon ECS a ejecutar finalmente una tarea por instancia, lo que podría no ser el comportamiento deseado.

Debe activar la protección para la reducción horizontal de instancias de Auto Scaling en el grupo de escalado automático para usar la protección contra terminación administrada. Si no activas la protección para la reducción horizontal, activar la protección contra terminación administrada

puede provocar un comportamiento no deseado. Por ejemplo, es posible que algunas instancias se estanquen en estado de vaciado. Para obtener más información, consulte Uso de la protección para la reducción horizontal de instancias en la Guía del usuario de Amazon EC2 Auto Scaling.

Cuando utilice la protección contra terminación con un proveedor de capacidad, no realice ninguna acción manual, como separar la instancia, en el grupo de escalado automático asociado al proveedor de capacidad. Las acciones manuales pueden interrumpir la operación de reducción horizontal del proveedor de capacidad. Si separa una instancia del grupo de escalado automático, también debe anular el registro de la instancia separada del clúster de Amazon ECS.

Actualización de la protección contra la finalización administrada para los proveedores de capacidad de Amazon ECS

Cuando use la protección contra la finalización administrada, debe actualizar la configuración de los proveedores de capacidad existentes.

Consola

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), elija el clúster.
3. En la página del clúster: elija la pestaña Infraestructura.
4. Elija el proveedor de capacidad.
5. Elija Actualizar para modificar la configuración del proveedor de capacidad.
6. En Configuración del grupo de escalado automático, active o desactive Protección contra la finalización administrada para activar o desactivar la característica.
7. Elija Actualizar.

AWS CLI

Puede actualizar la configuración de protección contra la finalización administrada de un proveedor de capacidad mediante el comando `update-capacity-provider`:

Para activar la protección contra la finalización administrada:

```
aws ecs update-capacity-provider \
  --capacity-provider CapacityProviderName \
  --auto-scaling-group-provider
 "managedScaling={status=ENABLED,targetCapacity=70,minimumScalingStepSize=1,maximumScalingStepS
```

Para desactivar la protección contra la finalización administrada:

```
aws ecs update-capacity-provider \
  --capacity-provider CapacityProviderName \
  --auto-scaling-group-provider
 "managedScaling={status=ENABLED,targetCapacity=70,minimumScalingStepSize=1,maximumScalingStepS
```

> **Note**
>
> Es posible que los cambios tarden unos minutos en aplicarse en todo el clúster. Al activar la protección contra la finalización administrada, las instancias que ya estén ejecutando tareas se protegerán de los eventos de reducción horizontal. Al desactivar la protección de finalización administrada, el indicador de protección se eliminará de las instancias durante el siguiente ciclo de administración del proveedor de capacidad de ECS.

Consola para ejecutar tareas

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), elija el clúster.
3. En la página del clúster, elija la pestaña Tareas.
4. Elija la tarea.
5. En Configuración, active o desactive Protección de terminación administrada para activar o desactivar la característica.
6. Elija Configurar la protección de reducción horizontal de tareas.

   Se muestra el cuadro de diálogo Configurar la protección de reducción horizontal de tareas.

   a. En la sección Protección de reducción horizontal de tareas, active la opción Activar.
   b. En Caduca en minutos, introduzca el número de minutos que faltan para que finalice la protección de reducción horizontal de tareas.
   c. Elija Update (Actualizar).

Activación del escalado automático de clústeres de Amazon ECS

Active el escalado automático del clúster para que Amazon ECS administre el escalado de las instancias de Amazon EC2 registradas en su clúster.

Si quiere utilizar la consola para activar el escalado automático de clústeres, consulte Creación de un proveedor de capacidad de Amazon ECS.

Antes de comenzar, cree un grupo de escalado automático y un proveedor de capacidad. Para obtener más información, consulte the section called “Proveedores de capacidad para el tipo de lanzamiento de EC2”.

Para activar el escalado automático de clústeres, asocie el proveedor de capacidad al clúster y, a continuación, active el escalado automático de clústeres.

1. Utilice el comando `put-cluster-capacity-providers` para asociar uno o más proveedores de capacidad con el clúster.

   Para agregar proveedores de capacidad de AWS Fargate, incluya los proveedores de capacidad de `FARGATE` y `FARGATE_SPOT` en la solicitud. Para obtener más información, consulte `put-cluster-capacity-providers` en la Referencia de los comandos de AWS CLI.

   ```
   aws ecs put-cluster-capacity-providers \
     --cluster ClusterName \
     --capacity-providers CapacityProviderName FARGATE FARGATE_SPOT \
     --default-capacity-provider-strategy capacityProvider=CapacityProvider,weight=1
   ```

   Para agregar un grupo de escalado automático para el tipo de inicio de EC2, incluya el nombre del grupo de escalado automático en la solicitud. Para obtener más información, consulte `put-cluster-capacity-providers` en la Referencia de los comandos de AWS CLI.

   ```
   aws ecs put-cluster-capacity-providers \
     --cluster ClusterName \
     --capacity-providers CapacityProviderName \
     --default-capacity-provider-strategy capacityProvider=CapacityProvider,weight=1
   ```

2. Utilice el comando `describe-clusters` para verificar que la asociación se haya realizado correctamente. Para obtener más información, consulte `describe-clusters` en la Referencia de los comandos de AWS CLI.

   ```
   aws ecs describe-clusters \
     --cluster ClusterName \
     --include ATTACHMENTS
   ```

3. Utilice el comando `update-capacity-provider` para activar el escalado automático administrado para el proveedor de capacidad. Para obtener más información, consulte `update-capacity-provider` en la Referencia de los comandos de AWS CLI.

```
aws ecs update-capacity-provider \
  --capacity-providers CapacityProviderName \
  --auto-scaling-group-provider managedScaling=ENABLED
```

Desactivación del escalado automático de clústeres de Amazon ECS

Desactive el escalado automático del clúster cuando necesite un control más granular de las instancias EC2 registradas en su clúster,

Para desactivar el escalado automático de clústeres para un clúster, puede desasociar el proveedor de capacidad con el escalado administrado activado desde el clúster o actualizar el proveedor de capacidad para desactivar el escalado administrado.

Desasociación del proveedor de capacidad

Siga estos pasos para desasociar un proveedor de capacidad de un clúster.

1. Utilice el comando `put-cluster-capacity-providers` para desasociar el proveedor de capacidad de grupo de escalado automático con el clúster. El clúster puede mantener la asociación con los proveedores de capacidad de AWS Fargate. Para obtener más información, consulte `put-cluster-capacity-providers` en la Referencia de los comandos de AWS CLI.

```
aws ecs put-cluster-capacity-providers \
  --cluster ClusterName \
  --capacity-providers FARGATE FARGATE_SPOT \
  --default-capacity-provider-strategy '[]'
```

Utilice el comando `put-cluster-capacity-providers` para desasociar el proveedor de capacidad de grupo de escalado automático con el clúster. Para obtener más información, consulte `put-cluster-capacity-providers` en la Referencia de los comandos de AWS CLI.

```
aws ecs put-cluster-capacity-providers \
  --cluster ClusterName \
```

```
  --capacity-providers [] \
  --default-capacity-provider-strategy '[]'
```

2. Utilice el comando `describe-clusters` para verificar que la disociación se haya realizado correctamente. Para obtener más información, consulte `describe-clusters` en la Referencia de los comandos de AWS CLI.

```
aws ecs describe-clusters \
  --cluster ClusterName \
  --include ATTACHMENTS
```

Desactive el escalado administrado para el proveedor de capacidad

Siga estos pasos para desactivar el escalado administrado para el proveedor de capacidad.

- Utilice el comando `update-capacity-provider` para desactivar el escalado automático administrado para el proveedor de capacidad. Para obtener más información, consulte `update-capacity-provider` en la Referencia de los comandos de AWS CLI.

```
aws ecs update-capacity-provider \
  --capacity-providers CapacityProviderName \
  --auto-scaling-group-provider managedScaling=DISABLED
```

Creación de un proveedor de capacidad de Amazon ECS

Una vez finalizada la creación del clúster, puede crear un nuevo proveedor de capacidad (grupo de escalado automático) para el tipo de lanzamiento EC2. Los proveedores de capacidad ayudan a administrar y escalar la infraestructura para sus aplicaciones.

Antes de crear el proveedor de capacidad, debe crear un grupo de escalado automático. Para obtener más información, consulte Grupos de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling.

Para crear un proveedor de capacidad para el clúster (consola de Amazon ECS)

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. En la página Clusters (Clústeres), elija el clúster.

4. En la página Cluster: ***name*** (Clúster: nombre), elija Infrastructure (Infraestructura) y, a continuación, elija Create (Crear).
5. En la página Create capacity providers (Crear proveedores de capacidad), configure las siguientes opciones.
   a. En Basic details (Detalles básicos), ingrese un nombre de proveedor de capacidad único en Capacity provider name (Nombre de proveedor de capacidad).
   b. En Auto Scaling group (Grupo de escalado automático), para Use an existing Auto Scaling group (Usar un grupo de escalado automático existente), elija el grupo de escalado automático.
   c. (Opcional) Para configurar una política de escalado, en Scaling policies (Políticas de escalado), configure las siguientes opciones.
      - Para que Amazon ECS administre las acciones de reducción y escalado horizontal, seleccione Turn on managed scaling (Activar el escalado administrado).
      - Para evitar que finalice la instancia de EC2 con tareas de Amazon ECS en ejecución, seleccione Turn on scaling protection (Activar la protección de escalado).
      - En Set target capacity (Definir capacidad de destino), ingrese el valor de destino para la métrica de CloudWatch utilizada en la política de escalado de seguimiento de destino administrada por Amazon ECS.
6. Seleccione Crear.

Actualización de un proveedor de capacidad de Amazon ECS

Al utilizar un grupo de escalado automático como proveedor de capacidad, puede modificar la política de escalado del grupo.

Actualización de un proveedor de capacidad del clúster (consola de Amazon ECS)

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. En la página Clusters (Clústeres), elija el clúster.
4. En la página Cluster: ***name*** (Clúster: nombre), elija Infrastructure (Infraestructura) y, a continuación, elija Update (Actualizar).
5. En la página Create capacity providers (Crear proveedores de capacidad), configure las siguientes opciones.

- En Grupo de escalado automático, en Políticas de escalado, configure las siguientes opciones.
  - Para que Amazon ECS administre las acciones de reducción y escalado horizontal, seleccione Turn on managed scaling (Activar el escalado administrado).
  - Para evitar que las instancias de EC2 con tareas de Amazon ECS en ejecución terminen, seleccione Activar la protección de escalado.
  - En Set target capacity (Definir capacidad de destino), ingrese el valor de destino para la métrica de CloudWatch utilizada en la política de escalado de seguimiento de destino administrada por Amazon ECS.

6. Elija Actualizar.

Eliminación de un proveedor de capacidad de Amazon ECS

Si ha terminado de utilizar un proveedor de capacidad de grupos de Auto Scaling, puede eliminarlo. Una vez eliminado el grupo, el proveedor de capacidad del grupo de escalado automático pasa al estado `INACTIVE`. Es posible que los proveedores de capacidad con estado `INACTIVE` permanezcan detectables en la cuenta durante un período de tiempo. Sin embargo, este comportamiento está sujeto a cambios en el futuro, por lo que no debe contar con la permanencia de los proveedores de capacidad `INACTIVE`. Antes de eliminar un proveedor de capacidad del grupo de escalado automático, se debe eliminar el proveedor de capacidad de la estrategia de proveedores de capacidad de todos los servicios. Puede usar la API `UpdateService` o el flujo de trabajo del servicio de actualización de la consola de Amazon ECS para eliminar un proveedor de capacidad de la estrategia de proveedores de capacidad de un servicio. Utilice la opción Forzar una nueva implementación a fin de garantizar que cualquier tarea que utilice la capacidad de la instancia de Amazon EC2 proporcionada por el proveedor de capacidad pase a utilizar la capacidad de los proveedores de capacidad restantes.

Para eliminar un proveedor de capacidad para el clúster (consola de Amazon ECS)

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. En la página Clusters (Clústeres), elija el clúster.
4. En la página Cluster: ***name*** (Clúster: nombre), elija Infrastructure (Infraestructura), el grupo de escalado automático y, a continuación, elija Delete (Eliminar).

5. En el cuadro de confirmación, ingrese eliminar ***nombre del grupo de escalado automático***.
6. Elija Eliminar.

Detención segura de las cargas de trabajo de Amazon ECS que se ejecutan en instancias de EC2

El drenaje de instancias administradas facilita la terminación correcta de instancias de Amazon EC2. Esto permite que sus cargas de trabajo se detengan de forma segura y se reprogramen para convertirlas en instancias que no terminan. El mantenimiento y las actualizaciones de la infraestructura se llevan a cabo sin preocuparse por la interrupción de las cargas de trabajo. Al utilizar el drenaje de instancias administradas, simplifica los flujos de trabajo de administración de la infraestructura que requieren el reemplazo de las instancias de Amazon EC2 y, al mismo tiempo, garantiza la resiliencia y la disponibilidad de sus aplicaciones.

El drenaje de instancias administradas por Amazon ECS funciona con los reemplazos de instancias de grupos de escalado automático. En función de la actualización de las instancias y de su vida útil máxima, los clientes pueden asegurarse de cumplir con las normas de seguridad y sistema operativo más recientes en lo que respecta a su capacidad.

El drenaje de instancias administradas solo se puede utilizar con los proveedores de capacidad de Amazon ECS. Puede activar el drenaje de instancias administradas al crear o actualizar los proveedores de capacidad del grupo de escalado automático mediante la consola de Amazon ECS, la AWS CLI o el SDK.

El drenaje de instancias administradas por Amazon ECS cubre los siguientes eventos.

- Actualización de instancias de grupos de escalado automático: utilice la actualización de instancias para reemplazar de forma continua las instancias de Amazon EC2 del grupo de escalado automático, en lugar de hacerlo manualmente por lotes. Esto es útil cuando necesita reemplazar un gran número de instancias. La actualización de instancias se inicia a través de la consola de Amazon EC2 o la API `StartInstanceRefresh`. Asegúrese de seleccionar `Replace` para la protección de la reducción horizontal cuando llame a `StartInstanceRefresh` si utiliza la protección contra terminación administrada.
- Duración máxima de la instancia: puede definir una vida útil máxima cuando se trata de reemplazar las instancias del grupo de escalado automático. Esto resulta útil para programar las instancias de reemplazo en función de las políticas de seguridad internas o el cumplimiento.

- Reducción horizontal de grupos de escalado automático: en función de las políticas de escalado y las acciones de escalado programadas, el grupo de escalado automático admite el escalado automático de instancias. Al utilizar un grupo de escalado automático como proveedor de capacidad de Amazon ECS, puede reducir horizontalmente las instancias de grupo de escalado automático cuando no se esté ejecutando ninguna tarea en ellas.
- Comprobaciones de estado de grupos de escalado automático: el grupo de escalado automático admite muchas comprobaciones de estado para administrar la terminación de instancias en mal estado.
- Actualizaciones de pila de AWS CloudFormation: puede agregar un atributo `UpdatePolicy` a su pila de AWS CloudFormation para llevar a cabo actualizaciones continuas cuando el grupo cambie.
- Reequilibrio de la capacidad de spot: el grupo de escalado automático intenta reemplazar de forma proactiva las instancias de spot que tienen un mayor riesgo de interrupción en función del aviso de reequilibrio de capacidad de Amazon EC2. El grupo de escalado automático finaliza la instancia anterior cuando se inicia la instancia de reemplazo y está en buen estado. El drenaje de instancias administradas por Amazon ECS drena la instancia de spot del mismo modo que drena una instancia que no es de spot.
- Interrupción de spot: las instancias de spot se finalizan con dos minutos de antelación. El drenaje de instancias administradas por Amazon ECS pone a la instancia en estado de drenaje como respuesta.

Enlaces de ciclo de vida de Amazon EC2 Auto Scaling con el drenaje de instancias administradas

Los enlaces de ciclo de vida de los grupos de escalado automático permiten al cliente crear soluciones que se activan mediante ciertos eventos del ciclo de vida de la instancia, así como llevar a cabo una acción personalizada cuando se produce ese evento determinado. Un grupo de escalado automático permite hasta 50 enlaces. Pueden existir varios enlaces de terminación y se ejecutan en paralelo, y el grupo de escalado automático espera a que todos los enlaces terminen antes de terminar una instancia.

Además de la terminación de enlace administrada por Amazon ECS, también puede configurar sus propios enlaces de terminación del ciclo de vida. Los enlaces de ciclo de vida tienen una `default action` y recomendamos configurar `continue` como valor predeterminado para garantizar que otros enlaces, como el enlace administrado por Amazon ECS, no se vean afectados por ningún error de los enlaces personalizados.

Si ya configuró un enlace de ciclo de vida de terminación de un grupo de escalado automático y también habilitó el drenaje de instancias administradas por Amazon ECS, se ejecutarán ambos

enlaces de ciclo de vida. Sin embargo, los tiempos relativos no están garantizados. Los enlaces de ciclo de vida tienen una configuración de `default action` para especificar la acción que se debe llevar a cabo cuando se agota el tiempo de espera. En caso de errores, le recomendamos el uso de `continue` como resultado predeterminado en su enlace personalizado. Esto garantiza que otros enlaces, especialmente los administrados por Amazon ECS, no se vean afectados por ningún error en su enlace de ciclo de vida personalizado. El resultado alternativo de `abandon` provoca que se omitan los demás ganchos y debe evitarse. Para obtener más información sobre los enlaces de ciclo de vida del grupo de escalado automático, consulte Amazon EC2 Auto Scaling lifecycle hooks en la Guía del usuario de Amazon EC2 Auto Scaling.

Drenaje de instancias administradas y tareas

El drenaje de instancias administradas por Amazon ECS utiliza la característica de drenaje existente que se encuentra en las instancias de contenedor. La característica de drenaje de instancias de contenedor reemplaza y detiene las tareas de réplica que pertenecen a un servicio de Amazon ECS. Una tarea independiente, como una invocada por `RunTask`, que esté en estado `PENDING` o `RUNNING` no se ve afectada. Tiene que esperar a que se completen o detenerlas manualmente. La instancia de contenedor permanece en ese estado `DRAINING` hasta que se detienen todas las tareas o hasta que hayan transcurrido 48 horas. Las tareas de daemon son las últimas en detenerse una vez que se hayan detenido todas las tareas de réplica.

Drenaje de instancias administradas y protección contra terminación administrada

El drenaje de instancias administradas funciona incluso si la terminación administrada está deshabilitada. Para obtener más información sobre la protección contra la terminación administrada, consulte Control de las instancias que Amazon ECS termina.

En la siguiente tabla se resume el comportamiento de las diferentes combinaciones de terminación administrada y drenaje administrado.

Terminación administrada	Drenaje administrado	Resultado
Habilitado	Habilitado	Amazon ECS protege las instancias de Amazon EC2 que ejecutan

Terminación administrada	Drenaje administrado	Resultado
		tareas para que no se terminen debido a eventos de reducción horizontal. Todas las instancias en proceso de terminación, como las que no tienen configurada la protección contra terminación, las que han recibido una interrupción de spot o las que se ven forzadas por una actualización de la instancia, se

Terminación administrada	Drenaje administrado	Resultado
		drenan sin problemas.
Deshabilitado	Habilitado	Amazon ECS no protege las instancias de Amazon EC2 que ejecutan tareas para que no se reduzcan horizonta lmente. Sin embargo, cualquier instancia que se termine se drena sin problemas.

Terminación administrada	Drenaje administrado	Resultado
Habilitado	Deshabilitado	Amazon ECS protege las instancias de Amazon EC2 que ejecutan tareas para que no se terminen debido a eventos de reducción horizontal. Sin embargo, las instancias aún se pueden terminar si se produce una interrupción de spot o se fuerza una actualización de la instancia, o si no se está ejecutando ninguna

Terminación administrada	Drenaje administrado	Resultado
		tarea. Amazon ECS no lleva a cabo correctamente el drenaje de estas instancias e inicia tareas de servicio de reemplazo una vez que se detienen.

Terminación administrada	Drenaje administrado	Resultado
Deshabilitad	Deshabilitado	Las instancias de Amazon EC2 se pueden reducir horizontalmente o terminar en cualquier momento, incluso si ejecutan tareas de Amazon ECS. Amazon ECS iniciará las tareas de servicio de reemplazo una vez que se detengan.

Drenaje de instancias administradas y drenaje de instancias de spot

Con el drenaje de instancias de spot, puede configurar una variable de entorno `ECS_ENABLE_SPOT_INSTANCE_DRAINING` en el agente de Amazon ECS que permita a Amazon ECS colocar una instancia en estado de drenaje en respuesta a la interrupción de spot de dos minutos. El drenaje de instancias administradas por Amazon ECS facilita el cierre correcto de las instancias de Amazon EC2 que se están terminando por muchos motivos, no solo por una interrupción de spot. Por ejemplo, puede utilizar el reequilibrio de capacidad de Amazon EC2

Auto Scaling para reemplazar de forma proactiva la instancia de spot con un riesgo elevado de interrupción, y el drenaje de instancias administradas cierra sin problemas la instancia de spot que se está reemplazando. Cuando utiliza el drenaje de instancias administradas, no necesita habilitar el drenaje de instancias de spot por separado, por lo que `ECS_ENABLE_SPOT_INSTANCE_DRAINING` en los datos de usuario del grupo de escalado automático es redundante. Para obtener más información sobre el drenaje de instancias de spot, consulte Spot Instances.

Funcionamiento del drenaje de instancias administradas con EventBridge

Los eventos de drenaje de instancias administradas por Amazon ECS se publican en Amazon EventBridge, y Amazon ECS crea una regla administrada de EventBridge en el bus predeterminado de su cuenta para admitir el drenaje de instancias administradas. Puede filtrar estos eventos a otros servicios de AWS, como Lambda, Amazon SNS y Amazon SQS, para supervisar y solucionar problemas.

- Amazon EC2 Auto Scaling envía un evento a EventBridge cuando se invoca un enlace de ciclo de vida.
- Los avisos de interrupción de spot se publican en EventBridge.
- Amazon ECS genera mensajes de error que puede recuperar a través de la consola y las API de Amazon ECS.
- EventBridge cuenta con mecanismos de reintento integrados para mitigar los errores temporales.

Configuración de los proveedores de capacidad de Amazon ECS para cerrar las instancias de forma segura

Puede habilitar el drenaje de instancias administradas al crear o actualizar los proveedores de capacidad del grupo de escalado automático mediante la consola de Amazon ECS y la AWS CLI.

> **Note**
>
> El drenaje de instancias administradas está activado de manera predeterminada al crear un proveedor de capacidad.

A continuación, se muestran ejemplos en los que se usa la AWS CLI para crear un proveedor de capacidad con el drenaje de instancias administradas habilitado y para habilitar el drenaje de instancias administradas para el proveedor de capacidad existente de un clúster.

Creación de un proveedor de capacidad con el drenaje de instancias administradas habilitado

Para crear un proveedor de capacidad con el drenaje de instancias administradas habilitado, utilice el comando `create-capacity-provider`. Establezca el parámetro `managedDraining` como `ENABLED`.

```
aws ecs create-capacity-provider \
--name capacity-provider \
--auto-scaling-group-provider '{
  "autoScalingGroupArn": "asg-arn",
  "managedScaling": {
    "status": "ENABLED",
    "targetCapacity": 100,
    "minimumScalingStepSize": 1,
    "maximumScalingStepSize": 1
  },
  "managedDraining": "ENABLED",
  "managedTerminationProtection": "ENABLED",
}'
```

Respuesta:

```
{
    "capacityProvider": {
        "capacityProviderArn": "capacity-provider-arn",
        "name": "capacity-provider",
        "status": "ACTIVE",
        "autoScalingGroupProvider": {
            "autoScalingGroupArn": "asg-arn",
            "managedScaling": {
                "status": "ENABLED",
                "targetCapacity": 100,
                "minimumScalingStepSize": 1,
                "maximumScalingStepSize": 1
            },
            "managedTerminationProtection": "ENABLED"
            "managedDraining": "ENABLED"
        }
    }
}
```

Habilitación del drenaje de instancias administradas para el proveedor de capacidad existente de un clúster

Para habilitar el drenaje de instancias administradas para el proveedor de capacidad existente de un clúster, utilice el comando `update-capacity-provider`. Verá que `managedDraining` actualmente indica `DISABLED` y `updateStatus` indica `UPDATE_IN_PROGRESS`.

```
aws ecs update-capacity-provider \
--name cp-draining \
--auto-scaling-group-provider '{
  "managedDraining": "ENABLED"
}
```

Respuesta:

```
{
    "capacityProvider": {
        "capacityProviderArn": "cp-draining-arn",
        "name": "cp-draining",
        "status": "ACTIVE",
        "autoScalingGroupProvider": {
            "autoScalingGroupArn": "asg-draining-arn",
            "managedScaling": {
                "status": "ENABLED",
                "targetCapacity": 100,
                "minimumScalingStepSize": 1,
                "maximumScalingStepSize": 1,
                "instanceWarmupPeriod": 300
            },
            "managedTerminationProtection": "DISABLED",
            "managedDraining": "DISABLED" // before update
        },
        "updateStatus": "UPDATE_IN_PROGRESS", // in progress and need describe again to
 find out the result
        "tags": [
        ]
    }
}
```

Use el comando `describe-clusters` e incluya `ATTACHMENTS`. El `status` del archivo adjunto de drenaje de instancias administradas es `PRECREATED`, y el `attachmentsStatus` general es `UPDATING`.

```
aws ecs describe-clusters --clusters cluster-name --include ATTACHMENTS
```

Respuesta:

```
{
    "clusters": [
        {
            ...

            "capacityProviders": [
                "cp-draining"
            ],
            "defaultCapacityProviderStrategy": [],
            "attachments": [
                # new precreated managed draining attachment
                {
                    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                    "type": "managed_draining",
                    "status": "PRECREATED",
                    "details": [
                        {
                            "name": "capacityProviderName",
                            "value": "cp-draining"
                        },
                        {
                            "name": "autoScalingLifecycleHookName",
                            "value": "ecs-managed-draining-termination-hook"
                        }
                    ]
                },

                ...

            ],
            "attachmentsStatus": "UPDATING"
        }
    ],
    "failures": []
```

```
}
```

Cuando finalice la actualización, use `describe-capacity-providers` y verá que `managedDraining` ahora está en estado `ENABLED`.

```
aws ecs describe-capacity-providers --capacity-providers cp-draining
```

Respuesta:

```
{
    "capacityProviders": [
        {
            "capacityProviderArn": "cp-draining-arn",
            "name": "cp-draining",
            "status": "ACTIVE",
            "autoScalingGroupProvider": {
                "autoScalingGroupArn": "asg-draning-arn",
                "managedScaling": {
                    "status": "ENABLED",
                    "targetCapacity": 100,
                    "minimumScalingStepSize": 1,
                    "maximumScalingStepSize": 1,
                    "instanceWarmupPeriod": 300
                },
                "managedTerminationProtection": "DISABLED",
                "managedDraining": "ENABLED" // successfully update
            },
            "updateStatus": "UPDATE_COMPLETE",
            "tags": []
        }
    ]
}
```

Solución de problemas de drenaje de instancias administradas por Amazon ECS

Es posible que tenga que solucionar problemas relacionados con el drenaje de instancias administradas. A continuación, se incluye un ejemplo de un problema y de una solución que puede encontrar al usarlo.

Las instancias no se terminan después de superar la vida útil máxima de las instancias cuando se usa el escalado automático.

Si sus instancias no se terminan incluso después de alcanzar y superar la vida útil máxima de las instancias mientras usa un grupo de escalado automático, es posible que se deba a que están protegidas contra la reducción horizontal. Puede desactivar la terminación administrada y permitir que el drenaje administrado se encargue del reciclaje de instancias.

Creación de recursos para el escalado automático de clústeres de Amazon ECS mediante la AWS Management Console

Obtenga información sobre cómo crear los recursos para el escalado automático de clústeres mediante la AWS Management Console. Cuando los recursos requieren un nombre, utilizamos el prefijo `ConsoleTutorial` para asegurarnos de que todos tengan nombres únicos y sean fáciles de localizar.

Temas

- Requisitos previos
- Paso 1: Crear un clúster de Amazon ECS
- Paso 2: Registrar una definición de tareas
- Paso 3: Ejecutar una tarea
- Paso 4: Verificar
- Paso 5: Eliminar

Requisitos previos

En este tutorial se supone que los siguientes requisitos previos se han completado:

- Se han completado los pasos que se indican en Configuración para utilizar Amazon ECS.
- Su usuario de IAM dispone de los permisos requeridos que se especifican en la política de IAM AmazonECS_FullAccess de ejemplo.
- Se crea el rol de IAM de la instancia de contenedor de Amazon ECS. Para obtener más información, consulte Rol de IAM de instancia de contenedor de Amazon ECS.
- Se crea el rol de IAM vinculado al servicio de Amazon ECS. Para obtener más información, consulte Uso de roles vinculados al servicio para Amazon ECS.
- Se crea el rol de IAM vinculado al servicio de Auto Scaling. Para obtener más información, consulte Roles vinculados al servicio para Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

- Tiene una VPC y un grupo de seguridad creados para utilizarlos. Para obtener más información, consulte the section called “Creación de una nube virtual privada”.

Paso 1: Crear un clúster de Amazon ECS

Siga estos pasos para crear un clúster de Amazon ECS.

Amazon ECS crea una plantilla de lanzamiento de Amazon EC2 Auto Scaling y un grupo de Auto Scaling en su nombre como parte de la pila AWS CloudFormation.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, elija Clústeres y, a continuación, elija Crear un clúster.
3. En Configuración de clúster, para Nombre del clúster, ingrese `ConsoleTutorial-cluster`.
4. En Infraestructura, desactive AWS Fargate (sin servidor) y, a continuación, seleccione Instancias de Amazon EC2. A continuación, configure el grupo de escalado automático que actúa como proveedor de capacidad.
   - En Grupo de escalado automático (ASG). Seleccione Crear nuevo ASG y, a continuación, proporcione los siguientes detalles sobre el grupo:
     - En Sistema operativo/arquitectura, elija Amazon Linux 2.
     - Para tipo de instancia EC2, seleccione t3.nano.
     - Para Capacity (Capacidad), introduzca el número mínimo y el número máximo de instancias que va a lanzar en el grupo de Auto Scaling.
5. (Opcional) Para administrar las etiquetas de clúster, expanda Tags (Etiquetas) y, a continuación, realice una de las siguientes operaciones:

   [Añadir una etiqueta] Elija Add tag (Añadir etiqueta) y haga lo siguiente:

   - En Key (Clave), escriba el nombre de la clave.
   - En Valor, escriba el valor de la clave.

   [Eliminar una etiqueta] Elija Eliminar a la derecha de la clave y el valor de la etiqueta.
6. Seleccione Create (Crear).

Paso 2: Registrar una definición de tareas

Antes de poder ejecutar una tarea en su clúster, debe registrar una definición de tareas. Las definiciones de tareas son listas de contenedores agrupadas. El ejemplo siguiente es una definición de tareas sencilla que utiliza una imagen `amazonlinux` de Docker Hub y se limita a permanecer inactiva. Para obtener más información acerca de los parámetros de definición de tareas disponibles, consulte Definiciones de tareas de Amazon ECS.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, elija Task Definitions (Definiciones de tareas).
3. Elija Create new task definition (Crear nueva definición de tarea) y Create new task definition with JSON (Crear nueva definición de tarea con JSON).
4. En el cuadro del Editor JSON, copie y pegue el siguiente contenido.

```
{
    "family": "ConsoleTutorial-taskdef",
    "containerDefinitions": [
        {
            "name": "sleep",
            "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
            "memory": 20,
            "essential": true,
            "command": [
                "sh",
                "-c",
                "sleep infinity"
            ]
        }
    ],
    "requiresCompatibilities": [
        "EC2"
    ]
}
```

5. Seleccione Crear.

Paso 3: Ejecutar una tarea

Después de registrar una definición de tareas para su cuenta, puede ejecutar una tarea en el clúster. En este tutorial, se ejecutan cinco instancias de la definición de tareas `ConsoleTutorial-taskdef` en el clúster `ConsoleTutorial-cluster`.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clústeres, elija ConsoleTutorial-cluster.
3. En Tareas, elija Ejecutar una nueva tarea.
4. En la sección Entorno, en Opciones de cálculo, elija Estrategia de proveedor de capacidad.
5. En Configuración de implementación, en Tipo de aplicación, elija Tarea.
6. Seleccione ConsoleTutorial-taskdef en la lista desplegable Familia.
7. En Tareas deseadas, introduzca 5.
8. Seleccione Crear.

Paso 4: Verificar

En este punto del tutorial, debe tener un clúster con cinco tareas en ejecución y un grupo de escalado automático con un proveedor de capacidad. El proveedor de capacidad tiene el escalado administrado por Amazon ECS habilitado.

Para comprobar que todo funcione correctamente, consulte las métricas de CloudWatch, la configuración del grupo de Auto Scaling y, por último, el recuento de tareas del clúster de Amazon ECS.

Para consultar las métricas de CloudWatch del clúster

1. Abra la consola de CloudWatch en https://console.aws.amazon.com/cloudwatch/.
2. En la barra de navegación de la parte superior de la pantalla, seleccione la región .
3. En el panel de navegación, en Métricas, elija Todas las métricas.
4. En la página Todas las métricas, en la pestaña Examinar, elija `AWS/ECS/ManagedScaling`.
5. Elija CapacityProviderName, ClusterName.
6. Seleccione la casilla de verificación correspondiente al `ConsoleTutorial-cluster` ClusterName.
7. En la pestaña Métricas gráficas, cambie Período a 30 segundos y Estadística a Máximo.

El valor que aparece en el gráfico muestra el valor de capacidad de destino del proveedor de capacidad. Debería comenzar en `100`, que es el porcentaje de capacidad de destino que hemos establecido. Debería observar cómo se escala hasta `200`, lo que desencadenará una alarma para la política de escalado de seguimiento de destino. La alarma desencadenará la reducción horizontal del grupo de Auto Scaling.

Siga estos pasos para consultar los detalles del grupo de Auto Scaling y confirmar que se ha producido la acción de escalado horizontal.

Para comprobar el escalado horizontal del grupo de Auto Scaling

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, seleccione la región .
3. En el panel de navegación, seleccione Auto Scaling y elija Auto Scaling Groups (Grupos de Auto Scaling).
4. Elija el grupo de escalado automático `ConsoleTutorial-cluster` creado en este tutorial. Consulte el valor en Capacidad deseada y consulte las instancias en la pestaña Administración de instancias para confirmar que su grupo se ha escalado horizontalmente a dos instancias.

Siga estos pasos para consultar el clúster de Amazon ECS y confirmar que las instancias de Amazon EC2 se hayan registrado en el clúster y las tareas hayan pasado al estado `RUNNING`.

Para comprobar las instancias del grupo de Auto Scaling

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. En la página Clusters (Clústeres), elija el clúster `ConsoleTutorial-cluster`.
4. En la pestaña Tareas, confirme que observa cinco tareas en el estado `RUNNING`.

Paso 5: Eliminar

Cuando termine este tutorial, debe limpiar los recursos asociados para evitar incurrir en cargos por recursos que no está utilizando. No se admite la eliminación de proveedores de capacidad y definiciones de tareas, pero no hay ningún costo asociado con estos recursos.

Para borrar los recursos del tutorial, realice el siguiente procedimiento:

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. En la página Clústeres, elija ConsoleTutorial-cluster.
4. En la página ConsoleTutorial-Cluster, seleccione la pestaña Tareas y, a continuación, seleccione Detener y Detener todo.
5. En el panel de navegación, seleccione Clusters (Clústeres).
6. En la página Clústeres, elija ConsoleTutorial-cluster.
7. En la parte superior derecha de la página, seleccione Eliminar clúster.
8. En el cuadro de confirmación, ingrese delete ConsoleTutorial-cluster y, a continuación, seleccione Eliminar.
9. Siga estos pasos para eliminar los grupos de Auto Scaling.
    a. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
    b. En la barra de navegación de la parte superior de la pantalla, seleccione la región .
    c. En el panel de navegación, seleccione Auto Scaling y elija Auto Scaling Groups (Grupos de Auto Scaling).
    d. Seleccione el grupo de escalado automático de `ConsoleTutorial-cluster` y, a continuación, elija Acciones.
    e. En el menú Actions (Acciones), elija Delete (Eliminar). En el cuadro de confirmación, ingrese Eliminar y, a continuación, elija Eliminar.

Instancias de contenedor de Amazon EC2 para Amazon ECS

Una instancia de contenedor de Amazon ECS es una instancia de Amazon EC2 que ejecuta el agente de contenedor de Amazon ECS y se ha registrado en un clúster. Cuando se ejecutan tareas con Amazon ECS mediante el tipo de lanzamiento de EC2, el tipo de lanzamiento externo o un proveedor de capacidad de grupo de escalado automático, las tareas se colocan en las instancias de contenedor activas. Usted es responsable de la administración y el mantenimiento de las instancias de contenedor.

Aunque puede crear su propia AMI de instancia de Amazon EC2 que cumpla con las especificaciones básicas necesarias para ejecutar las cargas de trabajo en contenedores en Amazon ECS, los ingenieros de AWS preconfiguran y prueban las AMI optimizadas para Amazon ECS

en Amazon ECS. Es la forma más sencilla para empezar y para conseguir que los contenedores funcionen en AWS rápidamente.

Al crear un clúster mediante la consola, Amazon ECS crea una plantilla de lanzamiento para las instancias con la AMI más reciente asociada al sistema operativo seleccionado.

Cuando se utiliza AWS CloudFormation para crear un clúster, el parámetro de SSM forma parte de la plantilla de lanzamiento de Amazon EC2 para las instancias del grupo de escalado automático. Puede configurar la plantilla para que utilice un parámetro dinámico de Systems Manager a fin de determinar qué AMI optimizada de Amazon ECS debe implementar. Este parámetro garantiza que, cada vez que implemente la pila, se compruebe si hay alguna actualización disponible que deba aplicarse a las instancias de EC2. Para ver un ejemplo de cómo utilizar el parámetro de Systems Manager, consulte Crear un clúster de Amazon ECS con la AMI de Amazon Linux 2023 optimizada para Amazon ECS en la Guía del usuario AWS CloudFormation.

- Recuperación de metadatos de las AMI de Linux optimizadas para Amazon ECS
- Recuperación de metadatos de la AMI de Bottlerocket optimizada para Amazon ECS
- Recuperación de metadatos de las AMI de Windows optimizadas para Amazon ECS

Puede elegir entre los tipos de instancias que sean compatibles con su aplicación. Con instancias más grandes, puede lanzar más tareas al mismo tiempo. Con instancias más pequeñas, puede escalar horizontalmente de forma más detallada para ahorrar costos. No necesita elegir un único tipo de instancia de Amazon EC2 que se adapte a todas las aplicaciones del clúster. En su lugar, puede crear varios grupos de escalado automático, donde cada grupo tenga un tipo de instancia diferente. A continuación, puede crear un proveedor de capacidad de Amazon EC2 para cada uno de estos grupos.

Utilice las siguientes directrices para determinar los tipos de familia de instancias y el tipo de instancias que debe utilizar:

- Elimine los tipos o familias de instancias que no cumplan con los requisitos específicos de su aplicación. Por ejemplo, si tu aplicación requiere una GPU, puede excluir cualquier tipo de instancia que no tenga una GPU.
- Tenga en cuenta los requisitos, como el almacenamiento y el rendimiento de la red.
- Tenga en cuenta la CPU y la memoria. Como regla general, la CPU y la memoria deben ser lo suficientemente grandes como para contener, al menos, una réplica de la tarea que quiere ejecutar.

Spot Instances

La capacidad de spot puede proporcionar importantes ahorros de costos en comparación con las instancias bajo demanda. La capacidad de spot es un exceso de capacidad cuyo precio es considerablemente inferior al de la capacidad reservada o bajo demanda. La capacidad de spot es adecuada para cargas de trabajo de procesamiento por lotes y machine learning, así como para entornos de desarrollo y ensayo. En términos más generales, es adecuada para cualquier carga de trabajo que tolere tiempos de inactividad temporales.

Tenga en cuenta las siguientes consecuencias, ya que es posible que la capacidad de Spot no esté disponible todo el tiempo.

- Durante los períodos de demanda extremadamente alta, es posible que la capacidad de spot no esté disponible. Esto puede provocar que se retrase el lanzamiento de las instancias de spot de Amazon EC2. En estos casos, los servicios de Amazon ECS vuelven a intentar lanzar las tareas y los grupos de Amazon EC2 Auto Scaling también vuelven a intentar lanzar instancias, hasta que se disponga de la capacidad necesaria. Amazon EC2 no sustituye la capacidad de spot por la capacidad bajo demanda.
- Cuando la demanda general de capacidad aumenta, es posible que las instancias y tareas de spot se terminen con solo dos minutos de aviso. Tras enviar la advertencia, las tareas deben iniciar un cierre ordenado, si fuera necesario, antes de que la instancia termine por completo. Esto ayuda a minimizar la posibilidad de errores. Para obtener más información sobre un cierre correcto, consulte Graceful shutdowns with ECS.

Para ayudar a minimizar la escasez de capacidad de spot, tenga en cuenta las siguientes recomendaciones:

- Utilice varias regiones y zonas de disponibilidad: la capacidad de spot varía según la región y la zona de disponibilidad. Puede mejorar la disponibilidad de spot ejecutando sus cargas de trabajo en varias regiones y zonas de disponibilidad. Si es posible, especifique subredes en todas las zonas de disponibilidad de las regiones en las que ejecuta sus tareas e instancias.
- Utilice varios tipos de instancias de Amazon EC2: cuando utiliza políticas de instancias mixtas con Amazon EC2 Auto Scaling, se lanzan varios tipos de instancias en su grupo de escalado automático. Esto garantiza que se pueda tramitar una solicitud de capacidad de spot cuando sea necesario. Para maximizar la fiabilidad y minimizar la complejidad, utilice tipos de instancias con aproximadamente la misma cantidad de CPU y memoria en la política de instancias mixtas. Estas instancias pueden ser de una generación diferente o ser variantes del mismo tipo de instancia

base. Tenga en cuenta que es posible que incluyan características adicionales que no necesite. Un ejemplo de esta lista podría incluir m4.large, m5.large, m5a.large, m5d.large, m5n.large, m5dn.large y m5ad.large. 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.

- Utilice la estrategia de asignación de spot con capacidad optimizada: con Amazon EC2 Spot, puede elegir entre estrategias de asignación optimizadas en función de la capacidad y de los costos. Si elige la estrategia de capacidad optimizada al lanzar una nueva instancia, Amazon EC2 Spot selecciona el tipo de instancia con mayor disponibilidad en la zona de disponibilidad seleccionada. Esto ayuda a reducir la posibilidad de que la instancia termine poco después de su lanzamiento.

Para obtener información sobre cómo configurar los avisos de terminación de spot en instancias de contenedor, consulte los siguientes recursos:

- Configuración de instancias de contenedor de Linux de Amazon ECS para recibir avisos de instancias de spot
- Configuración de instancias de contenedor de Windows de Amazon ECS para recibir avisos de instancias de spot

AMI de Linux optimizadas para Amazon ECS

> **Important**
>
> La AMI de Amazon Linux 2 optimizada para Amazon ECS finaliza su vida útil el 30 de junio de 2026 y refleja la misma fecha de fin de vida del sistema operativo original Amazon Linux 2 (para más información, consulte las preguntas frecuentes de Amazon Linux 2). Recomendamos a los clientes que actualicen las aplicaciones para utilizar Amazon Linux 2023, que incluye soporte a largo plazo hasta 2028. Para obtener información sobre la migración de Amazon Linux 2 a Amazon Linux 2023, consulte Migrating from the Amazon Linux 2 Amazon ECS-optimized AMI to the Amazon Linux 2023 Amazon ECS-optimized AMI.

Amazon ECS proporciona AMI optimizadas para Amazon ECS que están preconfiguradas con estos requisitos y recomendaciones para ejecutar sus cargas de trabajo de contenedor. Le recomendamos que utilice la AMI de Amazon Linux 2023 optimizada para Amazon ECS para las instancias de

Amazon EC2. Si lanza las instancias de contenedor desde la AMI optimizada para Amazon ECS más reciente, se asegurará de que reciba las actualizaciones de seguridad y la versión del agente de contenedor actuales. Para obtener más información acerca de cómo lanzar una instancia, consulte Lanzamiento de una instancia de contenedor de Linux de Amazon ECS.

Al crear un clúster mediante la consola, Amazon ECS crea una plantilla de lanzamiento para las instancias con la AMI más reciente asociada al sistema operativo seleccionado.

Cuando se utiliza AWS CloudFormation para crear un clúster, el parámetro de SSM forma parte de la plantilla de lanzamiento de Amazon EC2 para las instancias del grupo de escalado automático. Puede configurar la plantilla para que utilice un parámetro dinámico de Systems Manager a fin de determinar qué AMI optimizada de Amazon ECS debe implementar. Este parámetro garantiza que, cada vez que implemente la pila, se compruebe si hay alguna actualización disponible que deba aplicarse a las instancias de EC2. Para ver un ejemplo de cómo utilizar el parámetro de Systems Manager, consulte Crear un clúster de Amazon ECS con la AMI de Amazon Linux 2023 optimizada para Amazon ECS en la Guía del usuario AWS CloudFormation.

Si necesita personalizar la AMI optimizada para Amazon ECS, consulte Amazon ECS Optimized AMI Build Recipies en GitHub.

Están disponibles las siguientes variantes de la AMI optimizada para Amazon ECS para sus instancias de Amazon EC2 con el sistema operativo de Amazon Linux 2023.

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
Amazon Linux 2023	AMI de Amazon Linux 2023 optimizada para Amazon ECS	Amazon Linux 2023 es la próxima generación de Amazon Linux de AWS. En la mayoría de los casos, se recomienda para lanzar instancias de Amazon EC2 para las cargas de trabajo de Amazon ECS. Para obtener más informaci	De forma predeterm inada, la AMI de Amazon Linux 2023 optimizada para Amazon ECS se envía con un volumen raíz único de 30 GiB. Puede modificar el tamaño del volumen de raíz de 30 GiB en el momento del lanzamiento

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
		ón, consulte Qué es Amazon Linux 2023 en la Guía del usuario de Amazon Linux 2023.	para aumentar el almacenamiento disponible en su instancia de contenedor. Este almacenamiento se utiliza para el sistema operativo y para imágenes de Docker y metadatos. El sistema de archivos predeterminado para la AMI de Amazon Linux 2023 optimizada para Amazon ECS es `xfs`, y Docker utiliza el controlador de almacenamiento `overlay2`. Para obtener más información, consulte la sección Use the OverlayFS storage driver en la documentación de Docker.

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
Amazon Linux 2023 (arm64)	AMI de Amazon Linux 2023 (arm64) optimizada para Amazon ECS	Basada en Amazon Linux 2023, se recomienda utilizar esta AMI al inicializar las instancias de Amazon EC2, que cuentan con procesadores AWS Graviton/Graviton 2/Graviton 3/Graviton 4 basados en Arm, para las cargas de trabajo de Amazon ECS. Para obtener más información, consulte Specifications for the Amazon EC2 general purpose instances en la Guía de tipos de instancias de Amazon EC2.	De forma predeterminada, la AMI de Amazon Linux 2023 optimizada para Amazon ECS se envía con un volumen raíz único de 30 GiB. Puede modificar el tamaño del volumen de raíz de 30 GiB en el momento del lanzamiento para aumentar el almacenamiento disponible en su instancia de contenedor. Este almacenamiento se utiliza para el sistema operativo y para imágenes de Docker y metadatos. El sistema de archivos predeterminado para la AMI de Amazon Linux 2023 optimizada para Amazon ECS es `xfs`, y Docker utiliza el controlador de almacenamiento `overlay2`. Para obtener más

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
			información, consulte la sección Use the OverlayFS storage driver en la documentación de Docker.

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
Amazon Linux 2023 (Neuron)	AMI de Amazon Linux 2023 optimizada para Amazon ECS	Esta AMI, que se basa en Amazon Linux 2023, es para las instancias Inf1, Trn1 o Inf2 de Amazon EC2. Viene preconfigurada con controladores de AWS Inferentia y AWS Trainium y el tiempo de ejecución de AWS Neuron para Docker, que facilita la ejecución de cargas de trabajo de inferencia de machine learning en Amazon ECS. Para obtener más información, consulte Definiciones de tareas de Amazon ECS para cargas de trabajo de machine learning de AWS Neuron. La AMI de Amazon Linux 2023 (Neuron) optimizada para Amazon ECS no incluye la AWS CLI preinstalada.	De forma predeterminada, la AMI de Amazon Linux 2023 optimizada para Amazon ECS se envía con un volumen raíz único de 30 GiB. Puede modificar el tamaño del volumen de raíz de 30 GiB en el momento del lanzamiento para aumentar el almacenamiento disponible en su instancia de contenedor. Este almacenamiento se utiliza para el sistema operativo y para imágenes de Docker y metadatos. El sistema de archivos predeterminado para la AMI de Amazon Linux 2023 optimizada para Amazon ECS es `xfs`, y Docker utiliza el controlador de almacenamiento `overlay2`. Para obtener más

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
			información, consulte la sección Use the OverlayFS storage driver en la documentación de Docker.

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
GPU de Amazon Linux 2023	AMI de GPU de Amazon Linux 2023 optimizada para Amazon ECS	Se recomienda utilizar esta AMI basada en Amazon Linux 2023 al inicializar las instancias basadas en GPU de Amazon EC2 para las cargas de trabajo de Amazon ECS. Viene preconfigurada con controladores de kernel de NVIDIA y un tiempo de ejecución de GPU de Docker que permite ejecutar cargas de trabajo que aprovechan las GPU de Amazon ECS. Para obtener más información, consulte Definiciones de tareas de Amazon ECS para cargas de trabajo de GPU.	De forma predeterminada, la AMI de Amazon Linux 2023 optimizada para Amazon ECS se envía con un volumen raíz único de 30 GiB. Puede modificar el tamaño del volumen de raíz de 30 GiB en el momento del lanzamiento para aumentar el almacenamiento disponible en su instancia de contenedor. Este almacenamiento se utiliza para el sistema operativo y para imágenes de Docker y metadatos. El sistema de archivos predeterminado para la AMI de Amazon Linux 2023 optimizada para Amazon ECS es `xfs`, y Docker utiliza el controlador de almacenamiento `overlay2`. Para obtener más

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
			información, consulte la sección Use the OverlayFS storage driver en la documentación de Docker.

Las siguientes son las variantes de la AMI optimizada para Amazon ECS que están disponibles para sus instancias de Amazon EC2 con el sistema operativo de Amazon Linux 2.

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
Amazon Linux 2	AMI de Amazon Linux 2 con kernel 5.10 optimizada para Amazon ECS	Se debe utilizar esta AMI basada en Amazon Linux 2 cuando lanza las instancias de Amazon EC2 y desea usar Linux con kernel 5.10 en lugar de kernel 4.14 para las cargas de trabajo de Amazon ECS. La AMI de Amazon Linux 2 con kernel 5.10 optimizad a para Amazon ECS no incluye la AWS CLI preinstalada.	De forma predeterm inada, las AMI basadas en Amazon Linux 2 optimizadas para Amazon ECS (AMI de Amazon Linux 2 optimizada para Amazon ECS, AMI de Amazon Linux 2 (arm64) optimizad a para Amazon ECS y AMI optimizada para GPU de Amazon ECS) se envían con un único volumen raíz de 30 GiB. Puede modificar el tamaño del volumen de raíz de 30 GiB en el momento

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
			del lanzamiento para aumentar el almacenamiento disponible en su instancia de contenedor. Este almacenamiento se utiliza para el sistema operativo y para imágenes de Docker y metadatos. El sistema de archivos predeterminado para la AMI de Amazon Linux 2 optimizada para Amazon ECS `esxfs`, y Docker utiliza el controlador de almacenamiento `overlay2`. Para obtener más información, consulte la sección Use the OverlayFS storage driver en la documentación de Docker.

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
Amazon Linux 2	AMI de Amazon Linux 2 optimizada para Amazon ECS	Se trata de las cargas de trabajo de Amazon ECS. La AMI de Amazon Linux 2 optimizada para Amazon ECS no incluye la AWS CLI preinstalada.	De forma predeterminada, las AMI basadas en Amazon Linux 2 optimizadas para Amazon ECS (AMI de Amazon Linux 2 optimizada para Amazon ECS, AMI de Amazon Linux 2 (arm64) optimizada para Amazon ECS y AMI optimizada para GPU de Amazon ECS) se envían con un único volumen raíz de 30 GiB. Puede modificar el tamaño del volumen de raíz de 30 GiB en el momento del lanzamiento para aumentar el almacenamiento disponible en su instancia de contenedor. Este almacenamiento se utiliza para el sistema operativo y para imágenes de Docker y metadatos. El sistema de archivos predeterm

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
			inado para la AMI de Amazon Linux 2 optimizada para Amazon ECS esxfs, y Docker utiliza el controlador de almacenamiento overlay2. Para obtener más información, consulte la sección Use the OverlayFS storage driver en la documentación de Docker.

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
Amazon Linux 2 (arm64)	AMI de Amazon Linux 2 con kernel 5.10 (arm64) optimizada para Amazon ECS	Basada en Amazon Linux 2, esta AMI se usa para las instancias de Amazon EC2, que cuentan con procesadores AWS Graviton/Graviton 2/Graviton 3/Graviton 4 basados en Arm, y se recomienda usar Linux con kernel 5.10 en lugar de Linux con kernel 4.14 para las cargas de trabajo de Amazon ECS. Para obtener más información, consulte Specifications for Amazon EC2 general purpose instances en la Guía de tipos de instancias de Amazon EC2. La AMI de Amazon Linux 2 (arm64) optimizada para Amazon ECS no incluye la AWS CLI preinstalada.	De forma predeterminada, las AMI basadas en Amazon Linux 2 optimizadas para Amazon ECS (AMI de Amazon Linux 2 optimizada para Amazon ECS, AMI de Amazon Linux 2 (arm64) optimizada para Amazon ECS y AMI optimizada para GPU de Amazon ECS) se envían con un único volumen raíz de 30 GiB. Puede modificar el tamaño del volumen de raíz de 30 GiB en el momento del lanzamiento para aumentar el almacenamiento disponible en su instancia de contenedor. Este almacenamiento se utiliza para el sistema operativo y para imágenes de Docker y metadatos. El sistema de archivos predeterm

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
			inado para la AMI de Amazon Linux 2 optimizada para Amazon ECS esxfs, y Docker utiliza el controlador de almacenamiento overlay2. Para obtener más información, consulte la sección Use the OverlayFS storage driver en la documentación de Docker.

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
Amazon Linux 2 (arm64)	AMI de Amazon Linux 2 (arm64) optimizada para Amazon ECS	Basada en Amazon Linux 2, esta AMI se usa cuando se lanzan las instancias de Amazon EC2, que cuentan con procesadores AWS Graviton/Graviton 2/Graviton 3/Graviton 4 basados en Arm, para las cargas de trabajo de Amazon ECS. La AMI de Amazon Linux 2 (arm64) optimizada para Amazon ECS no incluye la AWS CLI preinstalada.	De forma predeterminada, las AMI basadas en Amazon Linux 2 optimizadas para Amazon ECS (AMI de Amazon Linux 2 optimizada para Amazon ECS, AMI de Amazon Linux 2 (arm64) optimizada para Amazon ECS y AMI optimizada para GPU de Amazon ECS) se envían con un único volumen raíz de 30 GiB. Puede modificar el tamaño del volumen de raíz de 30 GiB en el momento del lanzamiento para aumentar el almacenamiento disponible en su instancia de contenedor. Este almacenamiento se utiliza para el sistema operativo y para imágenes de Docker y metadatos. El sistema de archivos predeterm

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
			inado para la AMI de Amazon Linux 2 optimizada para Amazon ECS esxfs, y Docker utiliza el controlador de almacenamiento overlay2. Para obtener más información, consulte la sección Use the OverlayFS storage driver en la documentación de Docker.

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
Amazon Linux 2 (GPU)	AMI de kernel 5.10 optimizada para GPU de Amazon ECS	Basada en Amazon Linux 2, se recomiend a utilizar esta AMI cuando inicia las instancias basadas en GPU de Amazon EC2 con el kernel 5.10 de Linux para las cargas de trabajo de Amazon ECS. Viene preconfigurada con controladores de kernel de NVIDIA y un tiempo de ejecución de GPU de Docker que permite ejecutar cargas de trabajo que aprovechan las GPU de Amazon ECS. Para obtener más información, consulte Definiciones de tareas de Amazon ECS para cargas de trabajo de GPU.	De forma predeterm inada, las AMI basadas en Amazon Linux 2 optimizadas para Amazon ECS (AMI de Amazon Linux 2 optimizada para Amazon ECS, AMI de Amazon Linux 2 (arm64) optimizad a para Amazon ECS y AMI optimizada para GPU de Amazon ECS) se envían con un único volumen raíz de 30 GiB. Puede modificar el tamaño del volumen de raíz de 30 GiB en el momento del lanzamiento para aumentar el almacenam iento disponible en su instancia de contenedor. Este almacenamiento se utiliza para el sistema operativo y para imágenes de Docker y metadatos. El sistema de archivos predeterm

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
			inado para la AMI de Amazon Linux 2 optimizada para Amazon ECS esxfs, y Docker utiliza el controlador de almacenamiento overlay2. Para obtener más información, consulte la sección Use the OverlayFS storage driver en la documentación de Docker.

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
Amazon Linux 2 (GPU)	AMI optimizada para GPU de Amazon ECS	Basada en Amazon Linux 2, se recomiend a utilizar esta AMI cuando inicia las instancias basadas en GPU de Amazon EC2 con el kernel 4.14 de Linux para las cargas de trabajo de Amazon ECS. Viene preconfigurada con controladores de kernel de NVIDIA y un tiempo de ejecución de GPU de Docker que permite ejecutar cargas de trabajo que aprovechan las GPU de Amazon ECS. Para obtener más información, consulte Definiciones de tareas de Amazon ECS para cargas de trabajo de GPU.	De forma predeterm inada, las AMI basadas en Amazon Linux 2 optimizadas para Amazon ECS (AMI de Amazon Linux 2 optimizada para Amazon ECS, AMI de Amazon Linux 2 (arm64) optimizad a para Amazon ECS y AMI optimizada para GPU de Amazon ECS) se envían con un único volumen raíz de 30 GiB. Puede modificar el tamaño del volumen de raíz de 30 GiB en el momento del lanzamiento para aumentar el almacenam iento disponible en su instancia de contenedor. Este almacenamiento se utiliza para el sistema operativo y para imágenes de Docker y metadatos. El sistema de archivos predeterm

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
			inado para la AMI de Amazon Linux 2 optimizada para Amazon ECS esxfs, y Docker utiliza el controlador de almacenamiento overlay2. Para obtener más información, consulte la sección Use the OverlayFS storage driver en la documentación de Docker.

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
Amazon Linux 2 (Neuron)	AMI de kernel 5.10 de Amazon Linux 2 (Neuron) optimizada para Amazon ECS	Esta AMI basada en Amazon Linux 2 es para las instancias Inf1, Trn1 o Inf2 de Amazon EC2. Viene preconfigurada con controladores de AWS Inferentia con kernel 5.10 de Linux y AWS Trainium y el tiempo de ejecución de AWS Neuron para Docker, que facilita la ejecución de cargas de trabajo de inferenci a de machine learning en Amazon ECS. Para obtener más información, consulte Definiciones de tareas de Amazon ECS para cargas de trabajo de machine learning de AWS Neuron. La AMI de Amazon Linux 2 (Neuron) optimizada para Amazon ECS no incluye la AWS CLI preinstalada.	De forma predeterm inada, las AMI basadas en Amazon Linux 2 optimizadas para Amazon ECS (AMI de Amazon Linux 2 optimizada para Amazon ECS, AMI de Amazon Linux 2 (arm64) optimizad a para Amazon ECS y AMI optimizada para GPU de Amazon ECS) se envían con un único volumen raíz de 30 GiB. Puede modificar el tamaño del volumen de raíz de 30 GiB en el momento del lanzamiento para aumentar el almacenam iento disponible en su instancia de contenedor. Este almacenamiento se utiliza para el sistema operativo y para imágenes de Docker y metadatos. El sistema de archivos predeterm

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
			inado para la AMI de Amazon Linux 2 optimizada para Amazon ECS esxfs, y Docker utiliza el controlador de almacenamiento overlay2. Para obtener más información, consulte la sección Use the OverlayFS storage driver en la documentación de Docker.

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
Amazon Linux 2 (Neuron)	AMI de Amazon Linux 2 (Neuron) optimizada para Amazon ECS	Esta AMI basada en Amazon Linux 2 es para las instancias Inf1, Trn1 o Inf2 de Amazon EC2. Viene preconfigurada con controladores de AWS Inferentia y AWS Trainium y el tiempo de ejecución de AWS Neuron para Docker, que facilita la ejecución de cargas de trabajo de inferencia de machine learning en Amazon ECS. Para obtener más información, consulte Definiciones de tareas de Amazon ECS para cargas de trabajo de machine learning de AWS Neuron. La AMI de Amazon Linux 2 (Neuron) optimizada para Amazon ECS no incluye la AWS CLI preinstalada.	De forma predeterminada, las AMI basadas en Amazon Linux 2 optimizadas para Amazon ECS (AMI de Amazon Linux 2 optimizada para Amazon ECS, AMI de Amazon Linux 2 (arm64) optimizada para Amazon ECS y AMI optimizada para GPU de Amazon ECS) se envían con un único volumen raíz de 30 GiB. Puede modificar el tamaño del volumen de raíz de 30 GiB en el momento del lanzamiento para aumentar el almacenamiento disponible en su instancia de contenedor. Este almacenamiento se utiliza para el sistema operativo y para imágenes de Docker y metadatos. El sistema de archivos predeterm

Sistema operativo	AMI	Descripción	Configuración de almacenamiento
			inado para la AMI de Amazon Linux 2 optimizada para Amazon ECS esxfs, y Docker utiliza el controlador de almacenamiento overlay2. Para obtener más información, consulte la sección Use the OverlayFS storage driver en la documentación de Docker.

Amazon ECS proporciona un registro de cambios para la variante Linux de la AMI optimizada para Amazon ECS en GitHub. Para obtener más información, consulte Changelog (Registro de cambios).

Las variantes de Linux de la AMI optimizada para Amazon ECS utilizan la AMI de Amazon Linux 2 o la AMI de Linux 2023 como base. Puede recuperar el nombre de la AMI para cada variante consultando la API del Almacén de parámetros de Systems Manager. Para obtener más información, consulte Recuperación de metadatos de las AMI de Linux optimizadas para Amazon ECS. También están disponibles las notas de la versión de la AMI de Amazon Linux 2. Para obtener más información, consulte las notas de la versión de Amazon Linux 2. También están disponibles las notas de la versión de Amazon Linux 2023. Para obtener más información, consulte las notas de la versión de Amazon Linux 2023.

En las páginas siguientes se proporciona información adicional acerca de los cambios:

- Notas de la versión de la AMI de origen en GitHub
- Notas de la versión de Docker Engine en la documentación de Docker
- Documentación de controlador NVIDIA en la documentación de NVIDIA
- Registro de cambios del agente de Amazon ECS en GitHub

El código de origen de la aplicación `ecs-init` y los scripts y la configuración para empaquetar el agente ahora forman parte del repositorio de agentes. Para ver versiones anteriores de `ecs-init` y paquetes, consulte el registro de cambios de Amazon ecs-init en GitHub.

Aplicación de actualizaciones de seguridad a la AMI optimizada para Amazon ECS

Las AMI optimizadas para Amazon ECS basadas en Amazon Linux contienen una versión personalizada de cloud-init. Cloud-init es un paquete que se utiliza para arrancar imágenes de Linux en un entorno de computación en la nube y llevar a cabo las acciones deseadas al iniciar una instancia. De manera predeterminada, todas las AMI optimizadas para Amazon ECS basadas en Amazon Linux publicadas antes del 12 de junio de 2024 tienen todas las actualizaciones de seguridad “críticas” e “importantes” aplicadas al lanzar la instancia.

A partir de las versiones del 12 de junio de 2024 de las AMI optimizadas para Amazon ECS basadas en Amazon Linux 2, el comportamiento predeterminado ya no incluirá la actualización de paquetes en el momento del lanzamiento. En su lugar, le recomendamos que actualice a una nueva AMI optimizada para Amazon ECS a medida que haya versiones disponibles. Las AMI optimizadas para Amazon ECS se publican cuando hay actualizaciones de seguridad disponibles o cambios en la AMI base. Esto garantizará que reciba las últimas versiones del paquete y las actualizaciones de seguridad más recientes y que las versiones del paquete sean inmutables durante el inicio de las instancias. Para obtener más información acerca de cómo recuperar las AMI optimizadas para Amazon ECS más recientes, consulte Recuperación de metadatos de las AMI de Linux optimizadas para Amazon ECS.

Recomendamos automatizar el entorno para actualizarlo a una nueva AMI a medida que estén disponibles. Para obtener información sobre las opciones disponibles, consulte Amazon ECS enables easier EC2 capacity management, with managed instance draining.

Para seguir aplicando manualmente las actualizaciones de seguridad “críticas” e “importantes” en una versión de la AMI, puede ejecutar el siguiente comando en su instancia de Amazon EC2.

```
yum update --security
```

Si desea volver a habilitar las actualizaciones de seguridad en el momento del inicio, puede agregar la siguiente línea a la sección `#cloud-config` de datos de usuario de cloud-init al iniciar la instancia de Amazon EC2. Para obtener más información, consulte Uso de cloud-init en Amazon Linux 2 en la Guía del usuario de Amazon Linux.

```
#cloud-config
repo_upgrade: security
```

Recuperación de metadatos de las AMI de Linux optimizadas para Amazon ECS

Puede recuperar mediante programación los metadatos de la AMI optimizada para Amazon ECS. Los metadatos incluyen el nombre de la AMI, la versión del agente de contenedor de Amazon ECS y la versión del tiempo de ejecución de ECS que incluye la versión de Docker.

Al crear un clúster mediante la consola, Amazon ECS crea una plantilla de lanzamiento para las instancias con la AMI más reciente asociada al sistema operativo seleccionado.

Cuando se utiliza AWS CloudFormation para crear un clúster, el parámetro de SSM forma parte de la plantilla de lanzamiento de Amazon EC2 para las instancias del grupo de escalado automático. Puede configurar la plantilla para que utilice un parámetro dinámico de Systems Manager a fin de determinar qué AMI optimizada de Amazon ECS debe implementar. Este parámetro garantiza que, cada vez que implemente la pila, se compruebe si hay alguna actualización disponible que deba aplicarse a las instancias de EC2. Para ver un ejemplo de cómo utilizar el parámetro de Systems Manager, consulte Crear un clúster de Amazon ECS con la AMI de Amazon Linux 2023 optimizada para Amazon ECS en la Guía del usuario AWS CloudFormation.

Para recuperar el ID de la AMI, el nombre de la imagen, el sistema operativo, la versión del agente de contenedor, el nombre de la imagen de origen y la versión del tiempo de ejecución de las AMI optimizada para Amazon ECS mediante programación, consulte la API del Parameter Store de Systems Manager. Para obtener más información acerca de la API del Parameter Store de Systems Manager, consulte GetParameters y GetParametersByPath.

> **Note**
>
> La cuenta administrativa debe tener los siguientes permisos de IAM para recuperar los metadatos de la AMI optimizada para Amazon ECS. Estos permisos se han añadido a la política de IAM `AmazonECS_FullAccess`.
>
> - ssm:GetParameters
> - ssm:GetParameter
> - ssm:GetParametersByPath

Formato de los parámetros de Parameter Store de Systems Manager

A continuación, se muestra el formato del nombre del parámetro para cada variante de AMI optimizada para Amazon ECS.

AMI de Linux optimizadas para Amazon ECS

- Metadatos de AMI de Amazon Linux 2023:

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/<version>
```

- Metadatos de AMI de Amazon Linux 2023 (arm64):

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/arm64/<version>
```

- Metadatos de AMI de Amazon Linux 2023 (Neuron):

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/neuron/<version>
```

- Metadatos de AMI de Amazon Linux 2023 (GPU):

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/gpu/<version>
```

  Metadatos de AMI de Amazon Linux 2:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/<version>
```

- Metadatos de AMI de Amazon Linux 2 con kernel 5.10:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/<version>
```

- Metadatos de AMI de Amazon Linux 2 (arm64):

```
/aws/service/ecs/optimized-ami/amazon-linux-2/arm64/<version>
```

- Metadatos de AMI de Amazon Linux 2 con kernel 5.10 (arm64):

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/arm64/<version>
```

- Metadatos de la AMI de kernel 5.10 optimizada para GPU de Amazon ECS:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/gpu/<version>
```

- Metadatos de AMI de Amazon Linux 2 (GPU):

```
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/<version>
```

- Metadatos de la AMI de kernel 5.10 de Amazon Linux 2 (Neuron) optimizada para Amazon ECS:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/inf/<version>
```

- Metadatos de AMI de Amazon Linux 2 (Neuron):

```
/aws/service/ecs/optimized-ami/amazon-linux-2/inf/<version>
```

El siguiente formato de nombre de parámetro recupera el ID de imagen de la última versión recomendada de la AMI de Amazon Linux 2 optimizada para Amazon ECS mediante el parámetro secundario `image_id`.

```
/aws/service/ecs/optimized-ami/amazon-linux-2/recommended/image_id
```

El siguiente formato de nombre de parámetro recupera los metadatos de una versión específica de la AMI optimizada para Amazon ECS mediante la especificación del nombre de la AMI.

- Metadatos de AMI de Amazon Linux 2 optimizada para Amazon ECS:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/amzn2-ami-ecs-hvm-2.0.20181112-x86_64-
ebs
```

> **Note**
>
> Todas las versiones de AMI Amazon Linux 2 optimizadas para Amazon ECS están disponibles para su recuperación. Solo se pueden recuperar las versiones `amzn-ami-2017.09.l-amazon-ecs-optimized` de AMI (Linux) optimizadas para Amazon ECS y versiones posteriores.

Ejemplos

Los siguientes ejemplos muestran formas en las que pueden recuperar los metadatos de cada variante de AMI optimizada para Amazon ECS.

Recuperación de los metadatos de la última versión recomendada de la AMI optimizada para Amazon ECS

Utilice los siguientes comandos de la AWS CLI para recuperar la última versión recomendada de la AMI optimizada para Amazon ECS mediante la AWS CLI.

AMI de Linux optimizadas para Amazon ECS

- Para las AMI de Amazon Linux 2023 optimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended --region us-east-1
```

- Para las AMI de Amazon Linux 2023 (arm64) optimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
arm64/recommended --region us-east-1
```

- Para las AMI de Amazon Linux 2023 (Neuron) optimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
neuron/recommended --region us-east-1
```

- Para las AMI de Amazon Linux 2023 optimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/gpu/
recommended --region us-east-1
```

- Para las AMI de Amazon Linux 2 con kernel 5.10 optimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/recommended --region us-east-1
```

- Para las AMI de Amazon Linux 2 optimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
recommended --region us-east-1
```

- Para las AMI de Amazon Linux 2 con kernel 5.10 (arm64) optimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/arm64/recommended --region us-east-1
```

- Para las AMI de Amazon Linux 2 (arm64) optimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/arm64/
recommended --region us-east-1
```

- Para las AMI de kernel 5.10 optimizadas para GPU de Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/gpu/recommended --region us-east-1
```

- Para las AMI optimizadas para GPU de Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

- Para las AMI de kernel 5.10 de Amazon Linux 2 (Neuron) optimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/inf/recommended --region us-east-1
```

- Para las AMI de Amazon Linux 2 (Neuron) optimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/inf/
recommended --region us-east-1
```

Recuperación del ID de imagen de la AMI de Amazon Linux 2023 optimizada para Amazon ECS más reciente recomendada

Puede recuperar el ID de imagen del ID de la AMI de Amazon Linux 2023 optimizada para Amazon ECS más reciente recomendada mediante el parámetro secundario `image_id`.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-
linux-2023/recommended/image_id --region us-east-1
```

Para recuperar solo el valor de `image_id`, puede consultar el valor de parámetro específico; por ejemplo:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended/image_id --region us-east-1 --query "Parameters[0].Value"
```

Recuperación de los metadatos de una versión específica de AMI de Amazon Linux 2 optimizada para Amazon ECS

Utilice el siguiente comando de la AWS CLI para recuperar los metadatos de una versión específica de AMI de Amazon Linux optimizada para Amazon ECS mediante la AWS CLI. Sustituya el nombre de la AMI por el nombre de la AMI de Amazon Linux optimizada para Amazon ECS que va a recuperar.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/amzn2-ami-
ecs-hvm-2.0.20200928-x86_64-ebs --region us-east-1
```

Recuperación de los metadatos de la AMI de kernel 5.10 de Amazon Linux 2 optimizada para Amazon ECS mediante la API GetParametersByPath de Systems Manager

Utilice el siguiente comando de la AWS CLI para recuperar los metadatos de la AMI de Amazon Linux 2 optimizada para Amazon ECS mediante la API GetParametersByPath de Systems Manager.

```
aws ssm get-parameters-by-path --path /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/ --region us-east-1
```

Recuperación del ID de imagen de la AMI de kernel 5.10 de Amazon Linux 2 optimizada para Amazon ECS más reciente recomendada

Puede recuperar el ID de imagen del ID de la AMI de kernel 5.10 de Amazon Linux 2 optimizada para Amazon ECS más reciente recomendada mediante el parámetro secundario `image_id`.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/recommended/image_id --region us-east-1
```

Para recuperar solo el valor de `image_id`, puede consultar el valor de parámetro específico; por ejemplo:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
recommended/image_id --region us-east-1 --query "Parameters[0].Value"
```

Utilización de la AMI optimizada para Amazon ECS más reciente recomendada en una plantilla de AWS CloudFormation

Para hacer referencia a la AMI optimizada para Amazon ECS recomendada en una plantilla de AWS CloudFormation, pude hacer referencia al nombre del almacén de parámetros de Systems Manager.

Ejemplo de Linux

```
Parameters:kernel-5.10
  LatestECSOptimizedAMI:
    Description: AMI ID
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/recommended/
image_id
```

Migración de una AMI de Amazon Linux 2 a una AMI de Amazon Linux 2023 optimizada para Amazon ECS

Después de Amazon Linux, Amazon ECS finaliza el soporte estándar para las AMI optimizadas para Amazon ECS de Amazon Linux 2 a partir del 30 de junio de 2026. Después de esta fecha, la versión del agente de Amazon ECS queda anclada y las nuevas AMI optimizadas para Amazon ECS de Amazon Linux 2 solo se publicarán cuando se actualice la AMI de Amazon Linux 2 de origen. El fin del ciclo de vida completo (EOL) se produce el 30 de junio de 2026, tras lo cual no se publicarán más AMI de Amazon Linux 2 optimizadas para Amazon ECS, aunque se actualice la AMI de origen.

Amazon Linux 2023 ofrece un enfoque de seguridad desde el diseño con políticas de seguridad previamente configuradas, SELinux en modo permisivo, modo exclusivo de IMDSv2 habilitado de manera predeterminada, tiempos de arranque optimizados y administración de paquetes mejorada para aumentar la seguridad y el rendimiento.

Existe un alto grado de compatibilidad entre las AMI optimizadas para Amazon ECS de Amazon Linux 2 y Amazon Linux 2023, y la mayoría de los clientes verán cambios mínimos o nulos en las cargas de trabajo entre los dos sistemas operativos.

Para más información, consulte Comparing Amazon Linux 2 and Amazon Linux 2023 en la Guía del usuario de Amazon Linux 2023 y las Preguntas frecuentes sobre Amazon Linux 2023.

Consideraciones sobre compatibilidad

Administración de paquetes y actualizaciones del sistema operativo

A diferencia de las versiones anteriores de Amazon Linux, las AMI de Amazon Linux 2023 optimizadas para Amazon ECS están bloqueadas en una versión específica del repositorio de Amazon Linux. Esto evita que los usuarios actualicen los paquetes de manera inadvertida, lo que podría provocar cambios no deseados o perjudiciales. Para más información, consulte Managing repositories and OS updates in Amazon Linux 2023 en la Guía del usuario de Amazon Linux 2023.

Versiones del kernel de Linux

Las AMI de Amazon Linux 2 se basan en los kerneles de Linux 4.14 y 5.10, mientras que Amazon Linux 2023 utiliza los kerneles 6.1 y 6.12 de Linux. Para más información, consulte Comparing Amazon Linux 2 and Amazon Linux 2023 kernels en la Guía del usuario de Amazon Linux 2023.

Cambios en la disponibilidad de los paquetes

Los siguientes son cambios notables en los paquetes de Amazon Linux 2023:

- Algunos paquetes binarios de origen disponibles en Amazon Linux 2 ya no lo están en Amazon Linux 2023. Para más información, consulte Packages removed from Amazon Linux 2023 en las Notas de la versión de Amazon Linux 2023.
- Cambios en la forma en que Amazon Linux admite las distintas versiones de paquetes. El sistema `amazon-linux-extras` que se utiliza en Amazon Linux 2 no existe en Amazon Linux 2023. Todos los paquetes simplemente están disponibles en el repositorio “principal”.
- Los paquetes adicionales para Enterprise Linux (EPEL) no son compatibles con Amazon Linux 2023. Para más información, consulte EPEL compatibility in Amazon Linux 2023 en la Guía del usuario de Amazon Linux 2023.
- Las aplicaciones de 32 bits no son compatibles con Amazon Linux 2023. Para más información, consulte precated features from Amazon Linux 2 en la Guía del usuario de Amazon Linux 2023.

Cambios en los grupos de control (cgroups)

Un grupo de control (cgroup) es una característica del código kernel de Linux que permite organizar jerárquicamente los procesos y distribuir los recursos del sistema entre ellos. Los grupos de control se utilizan ampliamente para implementar un tiempo de ejecución de contenedores, y mediante `systemd`.

El agente de Amazon ECS, Docker y containerd son compatibles con cgroupv1 y cgroupv2. El agente de Amazon ECS y el tiempo de ejecución del contenedor administran cgroups, por lo que los clientes de Amazon ECS no tienen que hacer ningún cambio en esta actualización de cgroup subyacente.

Para más información sobre cgroupv2, consulte Control groups v2 in Amazon Linux 2023 en la Guía del usuario de Amazon Linux 2023.

Cambios en el servicio de metadatos de instancias (IMDS)

De manera predeterminada, es necesaria la versión 2 del servicio de metadatos de instancias (IMDSv2) para Amazon Linux 2023. IMDSv2 tiene varios beneficios que ayudan a mejorar la postura de seguridad. Utiliza un método de autenticación orientado a la sesión que requiere la creación de un token secreto en una solicitud sencilla de HTTP PUT para iniciar la sesión. El tiempo de validez de un token de sesión puede oscilar entre 1 segundo y 6 horas.

Para más información sobre cómo hacer la transición de IMDSv1 a IMDSv2, consulte Transición al uso del servicio de metadatos de instancias, versión 2 en la Guía del usuario de Amazon EC2.

Si desea utilizar IMDSv1, anule de manera manual la configuración mediante las propiedades de inicio de la opción de metadatos de la instancia.

Cambios en el intercambio de memoria

No se admite el intercambio de memoria por contenedor en Amazon Linux 2023 ni en cgroups v2. Para obtener más información, consulte Administración del espacio de memoria de intercambio de contenedores en Amazon ECS.

Cambios en la validación de FIPS

Amazon Linux 2 cuenta con la certificación FIPS 140-2 y Amazon Linux 2023 con la certificación FIPS 140-3.

Para habilitar el modo FIPS en Amazon Linux 2023, instale los paquetes necesarios en la instancia de Amazon EC2 y siga las instrucciones de configuración que se indican en Enable FIPS Mode on Amazon Linux 2023 en la Guía de usuario de Amazon Linux 2023.

Compatibilidad de instancias aceleradas

Las AMI de Amazon Linux 2023 optimizadas para Amazon ECS admiten los tipos de instancias aceleradas por GPU. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.

Creación de AMI personalizadas

Si bien le recomendamos cambiarse a las AMI optimizadas para Amazon ECS publicadas y compatibles oficialmente para Amazon Linux 2023, puede seguir creando AMI personalizadas optimizadas para Amazon ECS de Amazon Linux 2 mediante los scripts de compilación de código abierto que se utilizan para crear las variantes de Linux de la AMI optimizada para Amazon ECS. Para obtener más información, consulte Script de compilación de la AMI de Linux optimizada para Amazon ECS.

Estrategias de migración

Recomendamos crear e implementar un plan de migración que incluya pruebas exhaustivas de las aplicaciones. En las siguientes secciones se describen las distintas estrategias de migración en función de la forma en que administre la infraestructura de Amazon ECS.

Migración con proveedores de capacidad de Amazon ECS

1. Cree un nuevo proveedor de capacidad con una plantilla de lanzamiento nueva. Debe hacer referencia a un grupo de escalado automático con una plantilla de lanzamiento similar a la actual, pero en lugar de la AMI optimizada para Amazon ECS de Amazon Linux 2, debe especificar una de las variantes de Amazon Linux 2023. Agregue este nuevo proveedor de capacidad al clúster de Amazon ECS existente.
2. Actualice la estrategia de proveedor de capacidad predeterminado del clúster para incluir el proveedor de capacidad existente de Amazon Linux 2 y el nuevo proveedor de capacidad de Amazon Linux 2023. Comience con una ponderación más alta en el proveedor Amazon Linux 2 y una ponderación más baja en el proveedor Amazon Linux 2023 (por ejemplo, Amazon Linux 2: ponderación, 80; Amazon Linux 2023: ponderación, 20). Esto hace que Amazon ECS comience a aprovisionar instancias de Amazon Linux 2023 a medida que se programan nuevas tareas. Compruebe que las instancias se registren correctamente y que las tareas se puedan ejecutar correctamente en las nuevas instancias.
3. Ajuste gradualmente las ponderaciones de los proveedores de capacidad en la estrategia predeterminada del clúster, aumentando la ponderación del proveedor de Amazon Linux 2023 y disminuyendo la ponderación del proveedor de Amazon Linux 2 con el tiempo (por ejemplo, 60/40, después 40/60 y, por último, 20/80). También puede actualizar las estrategias individuales de los proveedores de capacidad de servicios para priorizar las instancias de Amazon Linux 2023. Supervise la ubicación de las tareas para asegurarse de que se ejecutan correctamente en las instancias de Amazon Linux 2023.

4. De manera opcional, vacíe las instancias de contenedor de Amazon Linux 2 para acelerar la migración de tareas. Si tiene suficiente capacidad de reemplazo de Amazon Linux 2023, puede vaciar manualmente las instancias de contenedor de Amazon Linux 2 a través de la consola de Amazon ECS o la AWS CLI para acelerar la transición de las tareas de Amazon Linux 2 hacia Amazon Linux 2023. Una vez completada la migración, elimine el proveedor de capacidad de Amazon Linux 2 del clúster y elimine el grupo de escalado automático asociado.

Migración con un grupo de Amazon EC2 Auto Scaling

1. Cree un nuevo grupo de Amazon EC2 Auto Scaling con una plantilla de lanzamiento nueva. Debe ser similar a la plantilla de lanzamiento actual, pero en lugar de la AMI optimizada para Amazon ECS de Amazon Linux 2, debe especificar una de las variantes de Amazon Linux 2023. Este nuevo grupo de escalado automático puede lanzar instancias al clúster actual.
2. Escale verticalmente el grupo de escalado automático para que las instancias de Amazon Linux 2023 se registren en el clúster. Compruebe que las instancias se registren correctamente y que las tareas se puedan ejecutar correctamente en las nuevas instancias.
3. Una vez que se haya comprobado que las tareas funcionan en Amazon Linux 2023, escale verticalmente el grupo de escalado automático de Amazon Linux 2023 y, a la vez, reduzca gradualmente el grupo de escalado automático de Amazon Linux 2, hasta que haya sustituido por completo todas las instancias de Amazon Linux 2.
4. Si tiene suficiente capacidad de reemplazo de Amazon Linux 2023, puede que desee vaciar explícitamente las instancias de contenedor para acelerar la transición de las tareas de Amazon Linux 2 hacia Amazon Linux 2023. Para obtener más información, consulte Drenaje de instancias de contenedor de Amazon ECS.

Migración con instancias administradas manualmente

1. Lance manualmente (o ajuste los scripts que lanzan) las instancias de Amazon EC2 mediante la AMI de Amazon Linux 2023 optimizada para Amazon ECS en lugar de Amazon Linux 2. Asegúrese de que estas instancias utilizan los mismos grupos de seguridad, subredes, roles de IAM y configuración de clúster que las instancias de Amazon Linux 2 existentes. Las instancias deben registrarse automáticamente en el clúster de Amazon ECS existente en el momento del lanzamiento.
2. Compruebe que las nuevas instancias de Amazon Linux 2023 se registren correctamente en el clúster de Amazon ECS y que se encuentren en el estado `ACTIVE`. Compruebe que las tareas se puedan programar y ejecutar correctamente en estas nuevas instancias, ya sea que espere

a que se asignen las tareas de forma natural o que las detenga o las inicie manualmente para activar la reprogramación.

3. Sustituya gradualmente las instancias de Amazon Linux 2; para ello, lance las instancias de Amazon Linux 2023 adicionales según sea necesario y, a continuación, vacíe y finalice manualmente las instancias de Amazon Linux 2 una por una. Puede vaciar las instancias a través de la consola de Amazon ECS; para ello, establezca la instancia en el estado `DRAINING`, lo que dejará de asignarle nuevas tareas y permitirá que las tareas existentes finalicen o se reprogramen en otro lugar.

Script de compilación de la AMI de Linux optimizada para Amazon ECS

Amazon ECS ha establecido en código abierto los scripts de compilación que se utilizan para crear las variantes de Linux de la AMI optimizada para Amazon ECS. Estos scripts de compilación están ahora disponibles en GitHub. Para obtener más información, consulte amazon-ecs-ami en GitHub.

Si necesita personalizar la AMI optimizada para Amazon ECS, consulte Amazon ECS Optimized AMI Build Recipies en GitHub.

El repositorio de scripts de compilación incluye una plantilla HashiCorp packer y crea scripts para generar cada una de las variantes de Linux de las AMI optimizadas para Amazon ECS. Estos scripts son el origen de confianza para las compilaciones de la AMI optimizada para Amazon ECS, de modo que pueda seguir el repositorio de GitHub para monitorear los cambios en nuestras AMI. Por ejemplo, quizás desee su propia AMI para utilizar la misma versión de Docker que el equipo de Amazon ECS utiliza para la AMI oficial.

Para obtener más información, consulte el repositorio de AMI de Amazon ECS en aws/amazon-ecs-ami en GitHub.

Para crear una AMI de Linux optimizada para Amazon ECS

1. Clonar el repositorio `aws/amazon-ecs-ami` GitHub.

```
git clone https://github.com/aws/amazon-ecs-ami.git
```

2. Agregue una variable de entorno para la región AWS que se utilizará al crear la AMI. Sustituya el valor `us-west-2` con la región que se va a utilizar.

```
export REGION=us-west-2
```

3. Se proporciona un archivo Makefile para crear la AMI. Desde el directorio raíz del repositorio clonado, utilice uno de los siguientes comandos, correspondiente a la variante Linux de la AMI optimizada de Amazon ECS que desea crear.

   - AMI de Amazon Linux 2 optimizada para Amazon ECS

     ```
     make al2
     ```

   - AMI de Amazon Linux 2 (arm64) optimizada para Amazon ECS

     ```
     make al2arm
     ```

   - AMI optimizada para GPU de Amazon ECS

     ```
     make al2gpu
     ```

   - AMI de Amazon Linux 2 (Neuron) optimizada para Amazon ECS

     ```
     make al2inf
     ```

   - AMI de Amazon Linux 2023 optimizada para Amazon ECS

     ```
     make al2023
     ```

   - AMI de Amazon Linux 2023 (arm64) optimizada para Amazon ECS

     ```
     make al2023arm
     ```

   - AMI de Amazon Linux 2023 (Neuron) optimizada por Amazon ECS

     ```
     make al2023neu
     ```

AMI Bottlerocket optimizadas para Amazon ECS

Bottlerocket es un sistema operativo de código abierto basado en Linux creado específicamente por AWS para ejecutar contenedores en máquinas virtuales o hosts bare metal. La AMI de Bottlerocket optimizada para Amazon ECS es segura y solo incluye la cantidad mínima de paquetes necesarios para ejecutar contenedores. Esto mejora el uso de los recursos, reduce la superficie expuesta a ataques contra la seguridad y ayuda a reducir los gastos administrativos. La AMI de Bottlerocket

también está integrada con Amazon ECS para ayudar a reducir la sobrecarga operativa que implica la actualización de las instancias de contenedores en un clúster.

Bottlerocket se diferencia de Amazon Linux en los siguientes aspectos:

- Bottlerocket no incluye un administrador de paquetes y su software solo se puede ejecutar como contenedores. Las actualizaciones de Bottlerocket se aplican y se pueden revertir en un solo paso, lo que reduce la probabilidad de que se produzcan errores de actualización.
- El mecanismo principal para administrar los hosts de Bottlerocket es mediante un programador de contenedores. A diferencia de Amazon Linux, el inicio de sesión en instancias de Bottlerocket individuales está pensado para ser una operación poco frecuente únicamente con fines avanzados de depuración y solución de problemas.

Para obtener más información acerca de Bottlerocket, consulte la documentación y las versiones en GitHub.

Existen variantes de la AMI de Bottlerocket optimizada para Amazon ECS para los kernel 6.1 y 5.10.

Las siguientes variantes utilizan el kernel 6.1:

- `aws-ecs-2`
- `aws-ecs-2-nvidia`

Las siguientes variantes utilizan kernel 5.10:

- `aws-ecs-1`
- `aws-ecs-1-nvidia`

  Para obtener más información acerca de la variante `aws-ecs-1-nvidia`, consulte Anuncio de la compatibilidad con GPU NVIDIA para Bottlerocket en Amazon ECS.

Consideraciones

Tenga en cuenta lo siguiente al utilizar la AMI de Bottlerocket con Amazon ECS.

- Bottlerocket admite instancias de Amazon EC2 con procesadores `x86_64` y `arm64`. No se recomienda utilizar la AMI Bottlerocket con instancias de Amazon EC2 con un chip Inferentia.

- Las imágenes de Bottlerocket no incluyen un servidor SSH ni un shell. Sin embargo, puede utilizar herramientas de administración fuera de banda para obtener acceso de administrador SSH y realizar tareas de arranque. Para obtener más información, consulte estas secciones en bottlerocket README.md en GitHub:
  - Exploration (Exploración)
  - Contenedor de administrador
- De forma predeterminada, Bottlerocket tiene un contenedor de control que está habilitado. Este contenedor ejecuta el agente de AWS Systems Manager que puede utilizar para ejecutar comandos o iniciar sesiones de shell en instancias Bottlerocket de Amazon EC2. Para obtener más información, consulte Configuración del administrador de sesiones en la Guía del usuario de AWS Systems Manager.

- Bottlerocket está optimizado para cargas de trabajo de contenedores y se centra en la seguridad. Bottlerocket no incluye un administrador de paquetes y es inmutable. Para obtener más información sobre las características y directrices de seguridad, consulte Características de seguridad y Directrices sobre seguridad en GitHub.
- Se admite el modo de red `awsvpc` con la versión de AMI de Bottlerocket `1.1.0` o una posterior.
- App Mesh en una definición de tarea es compatible con la versión AMI `1.15.0` de Bottlerocket o una posterior.
- El parámetro de definición de tareas `initProcessEnabled` es compatible con la versión `1.19.0` de la AMI de Bottlerocket o una posterior.
- Las AMI de Bottlerocket tampoco admiten los siguientes servicios y características:
  - ECS Anywhere
  - Service Connect
  - Amazon EFS en modo cifrado
  - Amazon EFS en modo de red `awsvpc`
  - Los volúmenes de Amazon EBS no se pueden montar
  - Acelerador de Elastic Inference

Recuperación de metadatos de la AMI de Bottlerocket optimizada para Amazon ECS

Puede recuperar el ID de Imagen de máquina de Amazon (AMI) de las AMI optimizadas para Amazon ECS al consultar la API de Parameter Store de AWS Systems Manager. Al utilizar este parámetro, no necesita buscar de manera manual los ID de la AMI optimizada para Amazon ECS.

Para obtener más información acerca de la API de Systems Manager Parameter Store, consulte GetParameter. El usuario que utiliza debe tener el permiso de IAM `ssm:GetParameter` para recuperar los metadatos de la AMI optimizada para Amazon ECS.

Variante de AMI de Bottlerocket **`aws-ecs-2`**

Puede recuperar la última variante estable de la AMI de `aws-ecs-2` Bottlerocket con Región de AWS y arquitectura mediante la AWS CLI o la AWS Management Console.

- AWS CLI: puede recuperar el ID de imagen de la última AMI de Bottlerocket optimizada para Amazon ECS recomendada con el siguiente comando de la AWS CLI mediante el parámetro secundario `image_id`. Sustituya *`region`* con el código de región para el que desee el ID de la AMI. Para obtener información sobre las Regiones de AWS compatibles, consulte Finding an AMI en GitHub. Para recuperar una versión que no sea la más reciente, sustituya `latest` con el número de versión.
  - Para 64 bits (`x86_64`) de arquitectura:

    ```
    aws ssm get-parameter --region us-east-2 --name "/aws/service/bottlerocket/aws-ecs-2/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Para 64 bits Arm (`arm64`) de arquitectura:

    ```
    aws ssm get-parameter --region us-east-2 --name "/aws/service/bottlerocket/aws-ecs-2/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- AWS Management Console: puede consultar el ID de la AMI optimizada para Amazon ECS recomendada mediante una URL en la AWS Management Console. La URL abre la consola de Amazon EC2 Systems Manager con el valor del ID del parámetro. En la siguiente URL, sustituya *`region`* con el código de región para el que desee el ID de la AMI. Para obtener información sobre las Regiones de AWS compatibles, consulte Finding an AMI en GitHub.
  - Para 64 bits (`x86_64`) de arquitectura:

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/aws-ecs-2/x86_64/latest/image_id/description?region=region#
    ```

  - Para 64 bits Arm (`arm64`) de arquitectura:

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/aws-ecs-2/arm64/latest/image_id/description?region=region#
    ```

Variante de AMI de Bottlerocket **`aws-ecs-2-nvidia`**

Puede recuperar la última variante estable de la AMI de `aws-ecs-2-nvdia` Bottlerocket por región y arquitectura con la AWS CLI o la AWS Management Console.

- AWS CLI: puede recuperar el ID de imagen de la última AMI de Bottlerocket optimizada para Amazon ECS recomendada con el siguiente comando de la AWS CLI mediante el parámetro secundario `image_id`. Sustituya *`region`* con el código de región para el que desee el ID de la AMI. Para obtener información sobre las Regiones de AWS compatibles, consulte Finding an AMI en GitHub. Para recuperar una versión que no sea la más reciente, sustituya `latest` con el número de versión.
  - Para 64 bits (`x86_64`) de arquitectura:

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-2-nvidia/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Para 64 bits Arm (`arm64`) de arquitectura:

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-2-nvidia/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- AWS Management Console: puede consultar el ID de la AMI optimizada para Amazon ECS recomendada mediante una URL en la AWS Management Console. La URL abre la consola de Amazon EC2 Systems Manager con el valor del ID del parámetro. En la siguiente URL, sustituya *`region`* con el código de región para el que desee el ID de la AMI. Para obtener información sobre las Regiones de AWS compatibles, consulte Finding an AMI en GitHub.
  - Para 64 bits (`x86_64`) de arquitectura:

    ```
    https://regionconsole.aws.amazon.com/systems-manager/parameters/aws/service/
    bottlerocket/aws-ecs-2-nvidia/x86_64/latest/image_id/description?region=region#
    ```

  - Para 64 bits Arm (`arm64`) de arquitectura:

    ```
    https://regionconsole.aws.amazon.com/systems-manager/parameters/aws/service/
    bottlerocket/aws-ecs-2-nvidia/arm64/latest/image_id/description?region=region#
    ```

Variante de AMI de Bottlerocket **`aws-ecs-1`**

Puede recuperar la última variante estable de la AMI de `aws-ecs-1` Bottlerocket con Región de AWS y arquitectura mediante la AWS CLI o la AWS Management Console.

- AWS CLI: puede recuperar el ID de imagen de la última AMI de Bottlerocket optimizada para Amazon ECS recomendada con el siguiente comando de la AWS CLI mediante el parámetro secundario `image_id`. Sustituya *`region`* con el código de región para el que desee el ID de la AMI. Para obtener información sobre las Regiones de AWS compatibles, consulte Finding an AMI en GitHub. Para recuperar una versión que no sea la más reciente, sustituya `latest` con el número de versión.
  - Para 64 bits (`x86_64`) de arquitectura:

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Para 64 bits Arm (`arm64`) de arquitectura:

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- AWS Management Console: puede consultar el ID de la AMI optimizada para Amazon ECS recomendada mediante una URL en la AWS Management Console. La URL abre la consola de Amazon EC2 Systems Manager con el valor del ID del parámetro. En la siguiente URL, sustituya *`region`* con el código de región para el que desee el ID de la AMI. Para obtener información sobre las Regiones de AWS compatibles, consulte Finding an AMI en GitHub.
  - Para 64 bits (`x86_64`) de arquitectura:

    ```
    https://region.console.aws.amazon.com/systems-manager/parameters/aws/service/
    bottlerocket/aws-ecs-1/x86_64/latest/image_id/description
    ```

  - Para 64 bits Arm (`arm64`) de arquitectura:

    ```
    https://region.console.aws.amazon.com/systems-manager/parameters/aws/service/
    bottlerocket/aws-ecs-1/arm64/latest/image_id/description
    ```

Variante de AMI de Bottlerocket **`aws-ecs-1-nvidia`**

Puede recuperar la última variante estable de la AMI de `aws-ecs-1-nvdia` Bottlerocket por región y arquitectura con la AWS CLI o la AWS Management Console.

- AWS CLI: puede recuperar el ID de imagen de la última AMI de Bottlerocket optimizada para Amazon ECS recomendada con el siguiente comando de la AWS CLI mediante el parámetro secundario `image_id`. Sustituya *`region`* con el código de región para el que desee el ID de la AMI. Para obtener información sobre las Regiones de AWS compatibles, consulte Finding an AMI en GitHub. Para recuperar una versión que no sea la más reciente, sustituya `latest` con el número de versión.
  - Para 64 bits (`x86_64`) de arquitectura:

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1-nvidia/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Para 64 bits Arm (`arm64`) de arquitectura:

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1-nvidia/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- AWS Management Console: puede consultar el ID de la AMI optimizada para Amazon ECS recomendada mediante una URL en la AWS Management Console. La URL abre la consola de Amazon EC2 Systems Manager con el valor del ID del parámetro. En la siguiente URL, sustituya *`region`* con el código de región para el que desee el ID de la AMI. Para obtener información sobre las Regiones de AWS compatibles, consulte Finding an AMI en GitHub.
  - Para 64 bits (`x86_64`) de arquitectura:

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/
    aws-ecs-1-nvidia/x86_64/latest/image_id/description?region=region#
    ```

  - Para 64 bits Arm (`arm64`) de arquitectura:

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/
    aws-ecs-1-nvidia/arm64/latest/image_id/description?region=region#
    ```

Pasos a seguir a continuación

Para obtener un tutorial detallado sobre cómo empezar a utilizar el sistema operativo Bottlerocket en Amazon ECS, consulte Using a Bottlerocket AMI with Amazon ECS en GitHub y Getting started withBottlerocket and Amazon ECS en el blog de AWS.

Para obtener más información acerca de cómo iniciar una instancia de Bottlerocket, consulte Lanzamiento de una instancia de Bottlerocket para Amazon ECS.

Lanzamiento de una instancia de Bottlerocket para Amazon ECS

Puede iniciar una instancia de Bottlerocket para poder ejecutar sus cargas de trabajo de contenedor.

Puede utilizar la AWS CLI para iniciar la instancia de Bottlerocket.

1. Cree un archivo denominado `userdata.toml`. Este archivo se utiliza para los datos de usuario de la instancia. Sustituya *`cluster-name`* por el nombre de su clúster.

```
[settings.ecs]
cluster = "cluster-name"
```

2. Utilice uno de los comandos que se incluyen en the section called “Recuperación de metadatos de la AMI de Bottlerocket optimizada para Amazon ECS” para obtener el ID de la AMI de Bottlerocket. Utilice esto en el siguiente paso.
3. Ejecute el siguiente comando para lanzar una instancia de Bottlerocket. Recuerde reemplazar los siguientes parámetros:
   - Sustituya la *`subred`* por el ID de la subred pública o privada en la que se lanzará la instancia.
   - Sustituya *`bottlerocket_ami`* por el ID de la AMI del paso anterior.
   - Sustituya *`t3.large`* por el tipo de instancia que desee usar.
   - Sustituya *`región`* por su código de región.

```
aws ec2 run-instances --key-name ecs-bottlerocket-example \
   --subnet-id subnet \
   --image-id bottlerocket_ami \
   --instance-type t3.large \
   --region region \
   --tag-specifications
 'ResourceType=instance,Tags=[{Key=bottlerocket,Value=example}]' \
```

```
    --user-data file://userdata.toml \
    --iam-instance-profile Name=ecsInstanceRole
```

4. Ejecute el siguiente comando para comprobar que la instancia de contenedor está registrada en el clúster. Al ejecutar este comando, recuerde reemplazar los siguientes parámetros:

   - Sustituya *clúster* por el nombre del clúster.
   - Sustituya *región* por el código de región.

```
aws ecs list-container-instances --cluster cluster-name --region region
```

Para obtener una explicación detallada sobre cómo empezar a utilizar el sistema operativo Bottlerocket en Amazon ECS, consulte Utilización de una AMI de Bottlerocket con Amazon ECS en GitHub e Introducción a Bottlerocket y Amazon ECS en el blog de AWS.

Administración de instancias de contenedor de Linux de Amazon ECS

Cuando utiliza instancias de EC2 para las cargas de trabajo de Amazon ECS, es responsable del mantenimiento de instancias.

Procedimientos de administración

- Lanzamiento de una instancia de contenedor de Linux de Amazon ECS
- Arranque de instancias de contenedor de Linux de Amazon ECS para la transferencia de datos
- Configuración de instancias de contenedor de Linux de Amazon ECS para recibir avisos de instancias de spot
- Ejecución de un script al lanzar una instancia de contenedor de Linux de Amazon ECS
- Aumento de las interfaces de red de instancias de contenedor de Linux de Amazon ECS
- Reserva de la memoria de instancias de contenedor de Linux de Amazon ECS
- Administración remota de instancias de contenedor de Amazon ECS mediante AWS Systems Manager
- Uso de un proxy HTTP para instancias de contenedor de Linux de Amazon ECS
- Configuración de instancias preinicializadas para el grupo de escalado automático de Amazon ECS
- Actualización del agente de contenedor de Amazon ECS

Cada versión del agente de contenedor de Amazon ECS admite un conjunto de características diferente y proporciona correcciones de errores de versiones anteriores. Cuando sea posible, siempre recomendamos utilizar la versión más reciente del agente de contenedor de Amazon ECS. Para actualizar el agente de contenedor a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS.

Para ver las características y mejoras incluidas en cada versión del agente, consulte https://github.com/aws/amazon-ecs-agent/releases.

> **Important**
>
> La versión mínima de Docker para obtener métricas fiables es la versión de Docker `v20.10.13` y versiones posteriores, que se incluyen en la AMI `20220607` optimizada para Amazon ECS y versiones posteriores.
> Los agentes de Amazon ECS versión `1.20.0` y posteriores ya no admiten versiones de Docker anteriores a la `18.01.0`.

Lanzamiento de una instancia de contenedor de Linux de Amazon ECS

Puede crear instancias de contenedor de Amazon ECS mediante la consola de Amazon EC2.

Puede lanzar una instancia con varios métodos, incluidos la consola de Amazon EC2, AWS CLI y SDK. Para obtener información sobre los demás métodos para lanzar una instancia, consulte Iniciar la instancia en la Guía del usuario de Amazon EC2.

Para obtener más información acerca del asistente de inicialización, consulte Lance una instancia con el nuevo asistente de inicialización de instancias en la Guía del usuario de Amazon EC2.

Antes de comenzar, complete los pasos de Configuración para utilizar Amazon ECS.

Puede utilizar el nuevo asistente de Amazon EC2 para lanzar una instancia. El asistente de inicialización de instancias especifica todos los parámetros de inicialización necesarios para iniciar una instancia.

Parámetros de configuración de instancias

- Procedimiento
- Nombre y etiquetas
- Imágenes de aplicaciones y sistema operativo (Imagen de máquina de Amazon)

- Tipo de instancia
- Par de claves (inicio de sesión)
- Configuración de red
- Configurar almacenamiento
- Detalles avanzados

Procedimiento

Antes de comenzar, complete los pasos de Configuración para utilizar Amazon ECS.

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, se muestra la región de AWS actual (por ejemplo, Este de EE. UU. [Ohio]). Seleccione una región en la que se va a iniciar la instancia.
3. En el panel de la consola de Amazon EC2, elija Iniciar instancia.

Nombre y etiquetas

El nombre de la instancia es una etiqueta, donde la clave es Name (Nombre) y el valor es el nombre que especifique. Puede etiquetar la instancia, los volúmenes y los gráficos elásticos. Para las instancias de spot, solo puede etiquetar la solicitud de instancia de spot.

Especificar un nombre de instancia y etiquetas adicionales es opcional.

- En Name (Nombre), ingrese un nombre descriptivo para la instancia. Si no especifica un nombre, la instancia se puede identificar mediante su ID, que se genera automáticamente al iniciar la instancia.
- Para agregar otras etiquetas, elija Add additional tag (Agregar etiqueta adicional). Elija Add tag (Agregar etiqueta) y, a continuación, ingrese una clave y un valor, y seleccione el tipo de recurso que desea etiquetar. Elija Add tag (Agregar etiqueta) para cada etiqueta adicional.

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

Una Imagen de máquina de Amazon (AMI) proporciona la información necesaria para crear una instancia. Por ejemplo, una AMI puede contener el software necesario para funcionar como servidor web, como Apache, y su sitio web.

Utilice la barra de búsqueda para buscar una AMI optimizada para Amazon ECS adecuada publicada por AWS.

1. Escriba uno de los siguientes términos en la barra de búsqueda.
   - **`ami-ecs`**
   - El valor de una AMI optimizada para Amazon ECS.

     Para obtener las AMI optimizadas para Amazon ECS más recientes y sus valores, consulte Versiones de AMI de Linux optimizadas para Amazon ECS.
2. Pulse Intro.
3. En la página Choose an Amazon Machine Image (AMI) (Elija una imagen de máquina de Amazon [AMI]), seleccione la categoría AWSMarketplace AMIs (AMI de Marketplace).
4. En el panel Refine results (Limitar resultados) situado a la izquierda, seleccione Amazon Web Services como publicador.
5. Elija Select (Seleccionar) en la fila de la AMI que desea utilizar.

   De manera alternativa, elija Cancel (Cancelar) (en la parte superior derecha) para volver al asistente de instancias de lanzamiento sin elegir una AMI. Se seleccionará una AMI predeterminada. Asegúrese de que la AMI cumpla con los requisitos descritos en las AMI de Linux optimizadas para Amazon ECS.

Tipo de instancia

El tipo de instancia define la configuración de hardware y el tamaño de la instancia. Los tipos de instancia más grandes tienen una CPU y memoria superiores. Para obtener más información, consulte Tipos de instancia.

- En Instance Type (Tipo de instancia), seleccione el tipo de instancia de la instancia.

  El tipo de instancia que seleccione determina los recursos disponibles para ejecutar sus tareas.

Par de claves (inicio de sesión)

En Key pair name (Nombre de par de claves) seleccione un par de claves existente o seleccione Create new key pair (Crear nuevo par de claves) para crear uno nuevo.

> **Important**
>
> Si elige la opción Proceed without key pair (Not recommended) (Continuar sin un par de claves [No recomendado]), no podrá conectarse a la instancia a menos que elija una AMI que esté configurada para ofrecer a los usuarios otra forma de iniciar sesión.

Configuración de red

Establezca la configuración de red, según sea necesario.

- Networking platform (Plataforma de redes): elija Virtual Private Cloud (VPC) (Nube privada virtual [VPC]) y, a continuación, especifique la subred en la sección Network interfaces (Interfaces de red).
- VPC: seleccione una VPC existente en la que desea crear el grupo de seguridad.
- Subnet (Subred): puede lanzar una instancia en una subred asociada con una zona de disponibilidad, zona local, zona Wavelength u Outpost.

  Para iniciar la instancia en una zona de disponibilidad, seleccione la subred en la que desea iniciar la instancia. Para crear una subred, elija Crear nueva subred para ir a la consola de Amazon VPC. Cuando haya terminado, vuelva al asistente de lanzamiento de instancias y elija el ícono Refresh (Actualizar) para cargar la subred en la lista.

  Para iniciar la instancia en una zona local, seleccione una subred que haya creado en la zona local.

  Para iniciar una instancia en un Outpost, seleccione una subred en una VPC que haya asociado a un Outpost.
- Auto-assign Public IP (Asignar automáticamente IP pública): si desea que se pueda acceder a la instancia desde Internet, compruebe que el campo Auto-assign Public IP (Asignar automáticamente IP pública) esté configurado como Enable (Habilitar). De lo contrario, configure este campo como Disable (Deshabilitar).

  > **Note**
  >
  > Las instancias de contenedor deben obtener acceso para comunicarse con el punto de conexión del servicio de Amazon ECS. Esto puede ser a través de un punto de conexión

de VPC de la interfaz o a través de las instancias de contenedor con direcciones IP públicas.
Para obtener más información acerca de los puntos de conexión de VPC, consulte Puntos de enlace de la VPC de interfaz de Amazon ECS (AWS PrivateLink).
Si no tiene configurado un punto de conexión de VPC de la interfaz y las instancias de contenedor no tienen direcciones IP públicas, deberán utilizar traducción de direcciones de red (NAT) para proporcionar este acceso. Para obtener más información, consulte Puertas de enlace NAT en la Guía del usuario de Amazon VPC y Uso de un proxy HTTP para instancias de contenedor de Linux de Amazon ECS en esta guía.

- Firewall (security groups) Firewall (grupos de seguridad): utilice un grupo de seguridad para definir reglas de firewall para la instancia de contenedor. Estas reglas especifican qué tráfico procedente de la red se entregará en la instancia de contenedor. El resto del tráfico se ignora.
  - Para seleccionar un grupo de seguridad existente, elija Select an existing security group (Seleccionar un grupo de seguridad existente) y seleccione el grupo de seguridad que creó en Configuración para utilizar Amazon ECS.

Configurar almacenamiento

La AMI seleccionada incluye uno o más volúmenes de almacenamiento, incluido el volumen de dispositivo raíz. Se pueden especificar volúmenes adicionales para adjuntar a la instancia.

Se puede utilizar la vista Simple (Simple).

- Storage type (Tipo de almacenamiento): configure el almacenamiento de la instancia de contenedor.

  Si utiliza la AMI de Amazon Linux 2 optimizada para Amazon ECS, la instancia tiene un único volumen de 30 GiB configurado, que se comparte entre el sistema operativo y Docker.

  Si utiliza la AMI optimizada para Amazon ECS, la instancia tiene configurados dos volúmenes. El volumen raíz lo utiliza el sistema operativo y el segundo volumen de Amazon EBS (asociado a `/dev/xvdcz`) lo utiliza Docker.

  Si lo desea, puede aumentar o reducir el tamaño de volumen para su instancia de acuerdo con las necesidades de su aplicación.

Detalles avanzados

En Detalles avanzados, expanda la sección para ver los campos y especifique cualquier parámetro adicional para la instancia.

- Purchasing option (Opción de compra): elija Request Spot instances (Solicitar instancias de spot) para solicitar una instancia de spot. También debe establecer el resto de los campos relacionados con las instancias de Spot. Para obtener más información, consulte Spot Instance Requests (Solicitudes de instancias de Spot).

  > **Note**
  >
  > Si utiliza instancias de Spot y ve un mensaje que indica `Not available`, es posible que deba elegir un tipo de instancia diferente.

  .

- En IAM instance profile (Perfil de instancia de IAM), seleccione el rol de IAM de la instancia de contenedor. Suele llamarse `ecsInstanceRole`.

  > **Important**
  >
  > Si no lanza la instancia de contenedor con los permisos de IAM correspondientes, el agente de Amazon ECS no puede conectarse al clúster. Para obtener más información, consulte Rol de IAM de instancia de contenedor de Amazon ECS.

- Datos de usuario: configure la instancia de contenedor de Amazon ECS con los datos de usuario, por ejemplo, las variables de entorno del agente de Configuración del agente de contenedor de Amazon ECS. Los scripts de datos de usuario de Amazon EC2 se ejecutan solo una vez, cuando la instancia se lanza por primera vez. A continuación, se muestran ejemplos comunes del uso de los datos del usuario:
  - De forma predeterminada, su instancia de contenedor se abre en su clúster predeterminado. Para abrirlo en un clúster no predeterminado, seleccione la lista Advanced Details.
    A continuación, pegue el siguiente script en el campo User data, reemplazando *`your_cluster_name`* con el nombre de su clúster.

    ```
    #!/bin/bash
    echo ECS_CLUSTER=your_cluster_name >> /etc/ecs/ecs.config
    ```

- Si tiene un archivo `ecs.config` en Amazon S3 y ha habilitado el acceso de solo lectura de Amazon S3 en el rol de instancia de contenedor, elija la lista Advanced Details (Detalles avanzados). A continuación, pegue el siguiente script en el campo User data (Datos de usuario), sustituyendo *`your_bucket_name`* por el nombre del bucket para instalar la AWS CLI y escriba su archivo de configuración en el momento del lanzamiento.

> **Note**
>
> Para obtener más información acerca de esta configuración, consulte Almacenamiento de la configuración de instancia de contenedor de Amazon ECS en Amazon S3.

```
#!/bin/bash
yum install -y aws-cli
aws s3 cp s3://your_bucket_name/ecs.config /etc/ecs/ecs.config
```

- Especifique las etiquetas para su instancia de contenedor mediante el parámetro de configuración `ECS_CONTAINER_INSTANCE_TAGS`. De esto modo, se crean etiquetas que solo están asociadas a Amazon ECS; no se pueden enumerar mediante la API de Amazon EC2.

> **Important**
>
> Si inicia las instancias de contenedor mediante un grupo de Amazon EC2 Auto Scaling, debe utilizar el parámetro de configuración del agente ECS_CONTAINER_INSTANCE_TAGS para agregar etiquetas. Esto se debe a la forma en que se agregan las etiquetas a las instancias de Amazon EC2 que se lanzan mediante grupos de Auto Scaling.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=your_cluster_name
ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
EOF
```

- Especifique las etiquetas para la instancia de contenedor y, a continuación, utilice el parámetro de configuración `ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM` para propagarlas de Amazon EC2 a Amazon ECS

A continuación se muestra un ejemplo de un script de datos de usuario que propaga las etiquetas asociadas a una instancia de contenedor, y registra la instancia de contenedor con un clúster denominado `your_cluster_name`:

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=your_cluster_name
ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM=ec2_instance
EOF
```

Para obtener más información, consulte Arranque de instancias de contenedor de Linux de Amazon ECS para la transferencia de datos.

Arranque de instancias de contenedor de Linux de Amazon ECS para la transferencia de datos

Cuando se lanza una instancia de Amazon EC2, puede transferir los datos de usuario a la instancia de EC2. Los datos se pueden utilizar para llevar a cabo tareas de configuración automatizadas comunes e incluso ejecutar scripts cuando la instancia arranca. En Amazon ECS, los casos de uso más comunes para los datos de usuario consisten en transferir la información de configuración al daemon de Docker y al agente de contenedor de Amazon ECS.

Puede transferir varios tipos de datos de usuario a Amazon EC2, incluidos cloud boothooks, scripts de shell y directivas `cloud-init`. Para obtener más información acerca de estos u otros tipos de formato, consulte la documentación de Cloud-Init.

Para transferir los datos de usuario al utilizar el asistente de lanzamiento de Amazon EC2, consulte Lanzamiento de una instancia de contenedor de Linux de Amazon ECS.

Puede configurar la instancia de contenedor para transferir los datos en la configuración del agente de contenedor o en la configuración del daemon de Docker.

Agente de contenedor de Amazon ECS

Las variantes de Linux de la AMI optimizada para Amazon ECS buscan datos de configuración del agente en el archivo `/etc/ecs/ecs.config` cuando se inicia el agente de contenedor. Puede especificar estos datos de configuración durante el lanzamiento con datos de usuario de Amazon EC2. Para obtener más información acerca de las variables de configuración del agente de contenedor de Amazon ECS disponibles, consulte Configuración del agente de contenedor de Amazon ECS.

Para establecer solo una variable de configuración del agente como, por ejemplo, el nombre del clúster, utilice echo para copiar la variable en el archivo de configuración:

```
#!/bin/bash
echo "ECS_CLUSTER=MyCluster" >> /etc/ecs/ecs.config
```

Si tiene varias variables que escribir en `/etc/ecs/ecs.config`, utilice el formato `heredoc` siguiente. Este formato escribe todo entre las líneas que comienzan por cat y `EOF` en el archivo de configuración.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_ENGINE_AUTH_TYPE=docker
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"username":"my_name","password":"my_password","email":"email@example.com"}}
ECS_LOGLEVEL=debug
ECS_WARM_POOLS_CHECK=true
EOF
```

Para configurar los atributos de instancia personalizados, defina la variable de entorno `ECS_INSTANCE_ATTRIBUTES`.

```
#!/bin/bash
cat <<'EOF' >> ecs.config
ECS_INSTANCE_ATTRIBUTES={"envtype":"prod"}
EOF
```

Daemon de Docker

Puede especificar la información de configuración del daemon de Docker con los datos de usuario de Amazon EC2. Para obtener más información sobre las opciones de configuración, consulte la documentación del daemon de Docker.

> **Note**
>
> AWS no admite configuraciones personalizadas de Docker, ya que a veces pueden entrar en conflicto con futuros cambios o características de Amazon ECS sin previo aviso.

En el ejemplo siguiente, las opciones personalizadas se agregan al archivo de configuración del daemon de Docker, que es `/etc/docker/daemon.json`, y luego se especifica en los datos del usuario cuando se lanza la instancia.

```
#!/bin/bash
cat <<EOF >/etc/docker/daemon.json
{"debug": true}
EOF
systemctl restart docker --no-block
```

En el ejemplo siguiente, las opciones personalizadas se agregan al archivo de configuración del daemon de Docker, que es `/etc/docker/daemon.json`, y luego se especifica en los datos del usuario cuando se lanza la instancia. En este ejemplo se muestra cómo activar o desactivar el docker-proxy en el archivo de configuración de daemon de Docker.

```
#!/bin/bash
cat <<EOF >/etc/docker/daemon.json
{"userland-proxy": false}
EOF
systemctl restart docker --no-block
```

Configuración de instancias de contenedor de Linux de Amazon ECS para recibir avisos de instancias de spot

Amazon EC2 termina, detiene o hiberna la instancia de spot cuando el precio de spot supera el precio máximo de su solicitud o cuando ya no hay más capacidad. Amazon EC2 envía un aviso de interrupción de dos minutos de la instancia de spot para la terminación y la detención de acciones. No proporciona el aviso de dos minutos para la acción de hibernación. Si el drenaje de instancias de spot de Amazon ECS está activado en la instancia, Amazon ECS recibe el aviso de interrupción de la instancia de spot y coloca la instancia en el estado `DRAINING`.

> **Important**
>
> Amazon ECS no recibe ningún aviso de Amazon EC2 cuando Auto Scaling Capacity Rebalancing elimina las instancias. Para obtener más información, consulte Reequilibrio de la capacidad de Amazon EC2 Auto Scaling.

Cuando se establece una instancia de contenedor en `DRAINING`, Amazon ECS evita que se programen nuevas tareas para su ubicación en la instancia de contenedor. Las tareas de servicio en la instancia de contenedor que se está vaciando que están en el estado `PENDING` se paran de inmediato. Si hay instancias de contenedor en el clúster que están disponibles, las tareas de servicio de sustitución se inician en ellas.

El drenaje de instancias de spot está desactivado de forma predeterminada.

Puede activar el drenaje de instancias de spot al lanzar una instancia. Agregue el script siguiente en el campo Datos de usuario. Reemplace *`MyCluster`* por el nombre del clúster en el que se va a registrar la instancia de contenedor.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_ENABLE_SPOT_INSTANCE_DRAINING=true
EOF
```

Para obtener más información, consulte Lanzamiento de una instancia de contenedor de Linux de Amazon ECS.

Para activar el vaciado de instancias de spot para una instancia de contenedor existente

1. Conéctese a la instancia de spot a través de SSH.
2. Edite el archivo `/etc/ecs/ecs.config` y añada lo siguiente:

   ```
   ECS_ENABLE_SPOT_INSTANCE_DRAINING=true
   ```

3. Reinicie el servicio `ecs`.

   - Para la AMI de Amazon Linux 2 optimizada para Amazon ECS:

     ```
     sudo systemctl restart ecs
     ```

4. (Opcional) Puede verificar que el agente se está ejecutando y ver información acerca de la nueva instancia de contenedor consultando la operación de la API de introspección del agente. Para obtener más información, consulte the section called “Introspección de contenedor”.

   ```
   curl http://localhost:51678/v1/metadata
   ```

Ejecución de un script al lanzar una instancia de contenedor de Linux de Amazon ECS

Es posible que tenga que ejecutar un contenedor específico en cada instancia de contenedor para tratar problemas de seguridad o de operaciones tales como la supervisión, seguridad, métricas, detección de servicios o registro.

Para ello, puede configurar sus instancias de contenedor para llamar al comando docker run con el script de datos de usuario durante el lanzamiento o en algún sistema de inicio como Upstart o systemd. Aunque este método funciona, tiene algunas desventajas ya que Amazon ECS no conoce el contenedor y no puede monitorear la CPU, la memoria, los puertos ni ningún otro recurso utilizado. A fin de garantizar que Amazon ECS pueda contabilizar correctamente todos los recursos de tareas, cree una definición de tareas para que el contenedor las ejecute en las instancias de contenedor. A continuación, utilice Amazon ECS para ubicar la tarea en el momento del lanzamiento con los datos de usuario de Amazon EC2.

En el siguiente procedimiento, el script de datos de usuario de Amazon EC2 utiliza la API de introspección de Amazon ECS para identificar la instancia de contenedor. A continuación, utiliza la AWS CLI y el comando start-task para ejecutar una tarea especificada en sí mismo durante el inicio.

Para iniciar una tarea en el momento del lanzamiento de una instancia de contenedor

1. Modifique el rol de IAM `ecsInstanceRole` para añadir permisos para la operación `StartTask` de la API. Para obtener más información, consulte Actualización de los permisos de un rol en la Guía del usuario de AWS Identity and Access Management.
2. Lance una o más instancias de contenedor mediante la AMI de Amazon Linux 2 optimizada para Amazon ECS. Lance nuevas instancias de contenedor y utilice el siguiente script de ejemplo en Datos de usuario de EC2. Reemplace *`your_cluster_name`* por el clúster de la instancia de contenedor en el que desea registrarse y *`my_task_def`* por la definición de tarea que desea ejecutar en la instancia en el momento del lanzamiento.

   Para obtener más información, consulte Lanzamiento de una instancia de contenedor de Linux de Amazon ECS.

   > **Note**
   >
   > El contenido multiparte de MIME a continuación utiliza un script de shell para establecer valores de configuración e instalar paquetes. También utiliza un trabajo systemd

para iniciar la tarea después de la ejecución del servicio ecs y una vez que la API de introspección está disponible.

```
Content-Type: multipart/mixed; boundary="==BOUNDARY==" 
MIME-Version: 1.0

--==BOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
# Specify the cluster that the container instance should register into
cluster=your_cluster_name

# Write the cluster configuration variable to the ecs.config file
# (add any other configuration variables here also)
echo ECS_CLUSTER=$cluster >> /etc/ecs/ecs.config

START_TASK_SCRIPT_FILE="/etc/ecs/ecs-start-task.sh"
cat <<- 'EOF' > ${START_TASK_SCRIPT_FILE}
 exec 2>>/var/log/ecs/ecs-start-task.log
 set -x

 # Install prerequisite tools
 yum install -y jq aws-cli

 # Wait for the ECS service to be responsive
 until curl -s http://localhost:51678/v1/metadata
 do
  sleep 1
 done

 # Grab the container instance ARN and AWS Region from instance metadata
 instance_arn=$(curl -s http://localhost:51678/v1/metadata | jq -r '.
 | .ContainerInstanceArn' | awk -F/ '{print $NF}' )
 cluster=$(curl -s http://localhost:51678/v1/metadata | jq -r '. | .Cluster' | awk
 -F/ '{print $NF}' )
 region=$(curl -s http://localhost:51678/v1/metadata | jq -r '.
 | .ContainerInstanceArn' | awk -F: '{print $4}')

 # Specify the task definition to run at launch
 task_definition=my_task_def
```

```
 # Run the AWS CLI start-task command to start your task on this container instance
 aws ecs start-task --cluster $cluster --task-definition $task_definition --
container-instances $instance_arn --started-by $instance_arn --region $region
EOF

# Write systemd unit file
UNIT="ecs-start-task.service"
cat <<- EOF > /etc/systemd/system/${UNIT}
      [Unit]
      Description=ECS Start Task
      Requires=ecs.service
      After=ecs.service

      [Service]
      Restart=on-failure
      RestartSec=30
      ExecStart=/usr/bin/bash ${START_TASK_SCRIPT_FILE}

      [Install]
      WantedBy=default.target
EOF

# Enable our ecs.service dependent service with `--no-block` to prevent systemd
 deadlock
# See https://github.com/aws/amazon-ecs-agent/issues/1707
systemctl enable --now --no-block "${UNIT}"
--==BOUNDARY==--
```

3. Compruebe que sus instancias de contenedor se lancen en el clúster correcto y que sus tareas se hayan iniciado.

   a. Abra la consola en https://console.aws.amazon.com/ecs/v2.

   b. En la barra de navegación, seleccione la región en la que se encuentra el clúster.

   c. En el panel de navegación, seleccione Clusters y seleccione el clúster que aloja sus instancias de contenedor.

   d. En la página Clúster, elija Tareas y, a continuación, elija las tareas.

      Cada instancia de contenedor que lanzó debe tener la tarea ejecutándose en ella.

Si no ve las tareas, puede iniciar sesión en sus instancias de contenedor con SSH y comprobar la información de depuración del archivo `/var/log/ecs/ecs-start-task.log`.

Aumento de las interfaces de red de instancias de contenedor de Linux de Amazon ECS

> **Note**
>
> Esta característica no está disponible en Fargate.

Cada tarea que usa el modo de red `awsvpc` recibe su propia interfaz de red elástica (ENI), que se asocia a la instancia de contenedor que la aloja. Existe un límite predeterminado en cuanto al número de interfaces de red que pueden asociarse a una instancia de Amazon EC2, y la interfaz de red principal cuenta como una. Por ejemplo, de forma predeterminada una instancia de `c5.large` puede tener asociadas hasta tres ENI. La interfaz de red principal para la instancia cuenta como una, por lo que puede asociar a la instancia dos ENI adicionales. Dado que cada tarea que utiliza el modo de red `awsvpc` requiere una ENI, normalmente solo puede ejecutar dos de esas tareas en este tipo de instancia.

Amazon ECS es compatible con el inicio de instancias de contenedor con mayor densidad de ENI al usar tipos de instancias de Amazon EC2 compatibles. Cuando se utilizan estos tipos de instancias y se activa la configuración de cuenta `awsvpcTrunking`, aparecen ENI adicionales disponibles en las instancias de contenedor recién lanzadas. Esta configuración le permite colocar más tareas en cada instancia de contenedor. Para utilizar la consola con el objeto de activar la característica, consulte Modificación de la configuración de la cuenta de Amazon ECS. Para utilizar la AWS CLI con el objeto de activar la característica, consulte Administración de la configuración de la cuenta de Amazon ECS mediante la AWS CLI.

Por ejemplo, una instancia `c5.large` con `awsvpcTrunking` tiene un límite de ENI aumentado de doce. La instancia de contenedor tendrá la interfaz de red principal, y Amazon ECS crea y asocia una interfaz de red “troncal” a la instancia de contenedor. Por lo tanto, esta configuración le permite lanzar diez tareas en la instancia de contenedor, en lugar de las dos tareas actuales.

La interfaz de red troncal está completamente administrada por Amazon ECS y se elimina cuando se termina o se anula el registro de su instancia de contenedor en el clúster. Para obtener más

información, consulte Opciones de redes de tareas de Amazon ECS para el tipo de lanzamiento de EC2.

Consideraciones

Tenga en cuenta lo siguiente al utilizar la característica de enlace troncal de ENI.

- Solo admiten aumento de los límites de ENI las variantes Linux de la AMI optimizada para Amazon ECS u otras variantes de Amazon Linux con la versión `1.28.1` o una posterior del agente de contenedor, y la versión `1.28.1-2` o una posterior del paquete ecs-init. Si utiliza la variante de Linux más reciente de la AMI optimizada para Amazon ECS, deberá cumplir estos requisitos. Los contenedores de Windows no son en este momento compatibles.
- Solo las nuevas instancias de Amazon EC2 iniciadas después de habilitar `awsvpcTrunking` reciben el aumento de límites de ENI y la interfaz de red troncal. Las instancias lanzadas anteriormente no reciben estas características, independientemente de las acciones realizadas.
- Las instancias de Amazon EC2 deben tener desactivadas las solicitudes DNS IPv4 basadas en recursos. Para desactivar esta opción, anule la opción Habilitar solicitudes DNS IPV4 (registro A) basadas en recursos al crear una nueva instancia mediante la consola de Amazon EC2. Para deshabilitar esta opción mediante el AWS CLI, utilice el siguiente comando.

```
aws ec2 modify-private-dns-name-options --instance-id i-xxxxxxx --no-enable-resource-
name-dns-a-record --no-dry-run
```

- No se admiten instancias de Amazon EC2 en subredes compartidas. No se registrarán en un clúster si se utilizan.
- Las tareas deben utilizar el modo de red `awsvpc` y el tipo de lanzamiento de EC2. Las tareas que utilizan el tipo de lanzamiento de Fargate siempre han recibido una ENI exclusiva, independientemente de cuántas se inicien, por lo que esta característica no es necesaria.
- Las tareas se deben lanzar en la misma Amazon VPC que la instancia de contenedor. Las tareas no se iniciarán con un error de atributo si no están dentro de la misma VPC.
- Al lanzar una nueva instancia de contenedor, la instancia pasa a un estado `REGISTERING` mientras la interfaz de red elástica troncal se aprovisiona para la instancia. Si el registro da error, la instancia pasa a un estado `REGISTRATION_FAILED`. Para solucionar un error de registro, describa la instancia de contenedor para visualizar el campo `statusReason` que describe el motivo del error. A continuación, la instancia de contenedor se puede terminar o anular manualmente su registro. Una vez que se haya anulado el registro de la instancia de contenedor o terminado correctamente, Amazon ECS eliminará la ENI troncal.

> **Note**
>
> Amazon ECS emite eventos de cambio de estado de instancia de contenedor que se pueden monitorear para las instancias que pasan a un estado `REGISTRATION_FAILED`. Para obtener más información, consulte Eventos de cambio de estado de instancia de contenedor de Amazon ECS.

- Una vez terminada la instancia de contenedor, la instancia pasa a un estado `DEREGISTERING` mientras se desaprovisiona la interfaz de red elástica troncal. La instancia después pasa a un estado `INACTIVE`.
- Si se detiene una instancia de contenedor en una subred pública con el aumento de los límites de ENI y, a continuación, se reinicia, la instancia pierde su dirección IP pública y el agente de contenedor pierde su conexión.
- Cuando se habilita `awsvpcTrunking`, las instancias de contenedor reciben un ENI adicional que usa el grupo de seguridad predeterminado de la VPC y es administrado por Amazon ECS.

  Una VPC predeterminada incluye una subred pública en cada zona de disponibilidad, una puerta de enlace de Internet y la configuración para habilitar la resolución DNS. La subred es una subred pública, ya que la tabla de enrutamiento principal envía a la puerta de enlace de Internet el tráfico de la subred que está destinado a Internet. Puede convertir una subred predeterminada en una subred privada eliminando la ruta del destino 0.0.0.0/0 al puerto de enlace a Internet. Sin embargo, si hace esto, ninguna instancia de contenedor que se esté ejecutando en esa subred podrá obtener acceso a Internet. Puede agregar o eliminar reglas del grupo de seguridad para controlar el tráfico que entra y sale de sus subredes. Para obtener más información, consulte Reglas del grupo de seguridad en la Guía del usuario de Amazon Virtual Private Cloud.

Requisitos previos

Antes de iniciar una instancia de contenedor con aumento de límites de ENI, deben completarse los siguientes requisitos previos.

- Se debe crear el rol vinculado al servicio para Amazon ECS. El rol vinculado al servicio de Amazon ECS proporciona a Amazon ECS los permisos para realizar llamadas a otros servicios de AWS en su nombre. Este rol se crea automáticamente al crear un clúster, o bien al crear o actualizar un servicio en la AWS Management Console. Para obtener más información, consulte Uso de roles

vinculados al servicio para Amazon ECS. También puede crear el rol vinculado a un servicio con el siguiente comando de la AWS CLI:

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- El rol de IAM de su cuenta o instancia de contenedor debe inscribirse en la configuración de la cuenta `awsvpcTrunking`. Le recomendamos crear dos roles de instancia de contenedor (`ecsInstanceRole`). A continuación, puede habilitar la configuración de la cuenta de `awsvpcTrunking` para un rol y usar ese rol para las tareas que requieren el enlace troncal de ENI. Para obtener más información sobre el rol de instancia de contenedor, consulte Rol de IAM de instancia de contenedor de Amazon ECS.

Una vez que se cumplan los requisitos previos, puede iniciar una nueva instancia de contenedor con uno de los tipos de instancia de Amazon EC2 compatibles, y la instancia tendrá el aumento de límites de ENI. Para ver una lista de los tipos de instancia admitidos, consulte Instancias admitidas para un aumento de las interfaces de red de contenedores de Amazon ECS. La instancia de contenedor debe tener la versión `1.28.1` o posterior del agente de contenedor y la versión `1.28.1-2` o posterior del paquete ecs-init. Si utiliza la variante de Linux más reciente de la AMI optimizada para Amazon ECS, deberá cumplir estos requisitos. Para obtener más información, consulte Lanzamiento de una instancia de contenedor de Linux de Amazon ECS.

> **Important**
>
> Las instancias de Amazon EC2 deben tener desactivadas las solicitudes DNS IPv4 basadas en recursos. Para deshabilitar esta opción, asegúrese de que Habilitar solicitudes DNS IPV4 (registro A) basadas en recursos se anula la selección de al crear una nueva instancia mediante la consola de Amazon EC2. Para deshabilitar esta opción mediante el AWS CLI, utilice el siguiente comando.
>
> ```
> aws ec2 modify-private-dns-name-options --instance-id i-xxxxxxx --no-enable-
> resource-name-dns-a-record --no-dry-run
> ```

Para ver las instancias de contenedor con aumento de los límites de ENI con la AWS CLI

Cada instancia de contenedor tiene una interfaz de red predeterminada, lo que se denomina interfaz de red troncal. Para utilizar el siguiente comando a fin de ver una lista de las instancias de

contenedor con un aumento de los límites de ENI, consulte el atributo `ecs.awsvpc-trunk-id`, que indica que tiene una interfaz de red troncal.

- list-attributes (AWS CLI)

```
aws ecs list-attributes \
      --target-type container-instance \
      --attribute-name ecs.awsvpc-trunk-id \
      --cluster cluster_name \
      --region us-east-1
```

- Get-ECSAttributeList (AWS Tools for Windows PowerShell)

```
Get-ECSAttributeList -TargetType container-instance -AttributeName ecs.awsvpc-trunk-
id -Region us-east-1
```

Instancias admitidas para un aumento de las interfaces de red de contenedores de Amazon ECS

A continuación, se muestran los tipos de instancias de Amazon EC2 que se admiten y la cantidad de tareas que utilizan el modo de red `awsvpc` que se pueden lanzar en cada tipo de instancias antes y después de optar por incluir la configuración de cuenta `awsvpcTrunking`.

> **Important**
>
> Aunque se admiten otros tipos de instancia en la misma familia de instancias, no se admiten los tipos de instancia `a1.metal`, `c5.metal`, `c5a.8xlarge`, `c5ad.8xlarge`, `c5d.metal`, `m5.metal`, `p3dn.24xlarge`, `r5.metal`, `r5.8xlarge` y `r5d.metal`.
> Las familias de instancias `c5n`, `d3`, `d3en`, `g3`, `g3s`, `g4dn`, `i3`, `i3en`, `inf1`, `m5dn`, `m5n`, `m5zn`, `mac1`, `r5b`, `r5n`, `r5dn`, `u-12tb1`, `u-6tb1`, `u-9tb1` y `z1d` no son compatibles.

Temas

- Fin general
- Optimizada para computación
- Optimizada para memoria
- Optimizada para almacenamiento
- Computación acelerada

- Computación de alto rendimiento

Fin general

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
a1.medium	1	10
a1.large	2	10
a1.xlarge	3	20
a1.2xlarge	3	40
a1.4xlarge	7	60
m5.large	2	10
m5.xlarge	3	20
m5.2xlarge	3	40
m5.4xlarge	7	60
m5.8xlarge	7	60
m5.12xlarge	7	60
m5.16xlarge	14	120
m5.24xlarge	14	120
m5a.large	2	10
m5a.xlarge	3	20
m5a.2xlarge	3	40
m5a.4xlarge	7	60

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
m5a.8xlarge	7	60
m5a.12xlarge	7	60
m5a.16xlarge	14	120
m5a.24xlarge	14	120
m5ad.large	2	10
m5ad.xlarge	3	20
m5ad.2xlarge	3	40
m5ad.4xlarge	7	60
m5ad.8xlarge	7	60
m5ad.12xlarge	7	60
m5ad.16xlarge	14	120
m5ad.24xlarge	14	120
m5d.large	2	10
m5d.xlarge	3	20
m5d.2xlarge	3	40
m5d.4xlarge	7	60
m5d.8xlarge	7	60
m5d.12xlarge	7	60
m5d.16xlarge	14	120
m5d.24xlarge	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
m5d.metal	14	120
m6a.large	2	10
m6a.xlarge	3	20
m6a.2xlarge	3	40
m6a.4xlarge	7	60
m6a.8xlarge	7	90
m6a.12xlarge	7	120
m6a.16xlarge	14	120
m6a.24xlarge	14	120
m6a.32xlarge	14	120
m6a.48xlarge	14	120
m6a.metal	14	120
m6g.medium	1	4
m6g.large	2	10
m6g.xlarge	3	20
m6g.2xlarge	3	40
m6g.4xlarge	7	60
m6g.8xlarge	7	60
m6g.12xlarge	7	60
m6g.16xlarge	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
m6g.metal	14	120
m6gd.medium	1	4
m6gd.large	2	10
m6gd.xlarge	3	20
m6gd.2xlarge	3	40
m6gd.4xlarge	7	60
m6gd.8xlarge	7	60
m6gd.12xlarge	7	60
m6gd.16xlarge	14	120
m6gd.metal	14	120
m6i.large	2	10
m6i.xlarge	3	20
m6i.2xlarge	3	40
m6i.4xlarge	7	60
m6i.8xlarge	7	90
m6i.12xlarge	7	120
m6i.16xlarge	14	120
m6i.24xlarge	14	120
m6i.32xlarge	14	120
m6i.metal	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
m6id.large	2	10
m6id.xlarge	3	20
m6id.2xlarge	3	40
m6id.4xlarge	7	60
m6id.8xlarge	7	90
m6id.12xlarge	7	120
m6id.16xlarge	14	120
m6id.24xlarge	14	120
m6id.32xlarge	14	120
m6id.metal	14	120
m6idn.large	2	10
m6idn.xlarge	3	20
m6idn.2xlarge	3	40
m6idn.4xlarge	7	60
m6idn.8xlarge	7	90
m6idn.12xlarge	7	120
m6idn.16xlarge	14	120
m6idn.24xlarge	14	120
m6idn.32xlarge	15	120
m6idn.metal	15	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
m6in.large	2	10
m6in.xlarge	3	20
m6in.2xlarge	3	40
m6in.4xlarge	7	60
m6in.8xlarge	7	90
m6in.12xlarge	7	120
m6in.16xlarge	14	120
m6in.24xlarge	14	120
m6in.32xlarge	15	120
m6in.metal	15	120
m7a.medium	1	4
m7a.large	2	10
m7a.xlarge	3	20
m7a.2xlarge	3	40
m7a.4xlarge	7	60
m7a.8xlarge	7	90
m7a.12xlarge	7	120
m7a.16xlarge	14	120
m7a.24xlarge	14	120
m7a.32xlarge	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
m7a.48xlarge	14	120
m7a.metal-48xl	14	120
m7g.medium	1	4
m7g.large	2	10
m7g.xlarge	3	20
m7g.2xlarge	3	40
m7g.4xlarge	7	60
m7g.8xlarge	7	60
m7g.12xlarge	7	60
m7g.16xlarge	14	120
m7g.metal	14	120
m7gd.medium	1	4
m7gd.large	2	10
m7gd.xlarge	3	20
m7gd.2xlarge	3	40
m7gd.4xlarge	7	60
m7gd.8xlarge	7	60
m7gd.12xlarge	7	60
m7gd.16xlarge	14	120
m7gd.metal	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
m7i.large	2	10
m7i.xlarge	3	20
m7i.2xlarge	3	40
m7i.4xlarge	7	60
m7i.8xlarge	7	90
m7i.12xlarge	7	120
m7i.16xlarge	14	120
m7i.24xlarge	14	120
m7i.48xlarge	14	120
m7i.metal-24xl	14	120
m7i.metal-48xl	14	120
m7i-flex.large	2	4
m7i-flex.xlarge	3	10
m7i-flex.2xlarge	3	20
m7i-flex.4xlarge	7	40
m7i-flex.8xlarge	7	60
m7i-flex.12xlarge	7	120
m7i-flex.16xlarge	14	120
m8g.medium	1	4
m8g.large	2	10

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
m8g.xlarge	3	20
m8g.2xlarge	3	40
m8g.4xlarge	7	60
m8g.8xlarge	7	60
m8g.12xlarge	7	60
m8g.16xlarge	14	120
m8g.24xlarge	14	120
m8g.48xlarge	14	120
m8g.metal-24xl	14	120
m8g.metal-48xl	14	120
m8gd.medium	1	4
m8gd.large	2	10
m8gd.xlarge	3	20
m8gd.2xlarge	3	40
m8gd.4xlarge	7	60
m8gd.8xlarge	7	60
m8gd.12xlarge	7	60
m8gd.16xlarge	14	120
m8gd.24xlarge	14	120
m8gd.48xlarge	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
m8gd.metal-24xl	14	120
m8gd.metal-48xl	14	120
mac2.metal	7	12
mac2-m1ultra.metal	7	12
mac2-m2.metal	7	12
mac2-m2pro.metal	7	12

Optimizada para computación

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
c5.large	2	10
c5.xlarge	3	20
c5.2xlarge	3	40
c5.4xlarge	7	60
c5.9xlarge	7	60
c5.12xlarge	7	60
c5.18xlarge	14	120
c5.24xlarge	14	120
c5a.large	2	10
c5a.xlarge	3	20

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
c5a.2xlarge	3	40
c5a.4xlarge	7	60
c5a.12xlarge	7	60
c5a.16xlarge	14	120
c5a.24xlarge	14	120
c5ad.large	2	10
c5ad.xlarge	3	20
c5ad.2xlarge	3	40
c5ad.4xlarge	7	60
c5ad.12xlarge	7	60
c5ad.16xlarge	14	120
c5ad.24xlarge	14	120
c5d.large	2	10
c5d.xlarge	3	20
c5d.2xlarge	3	40
c5d.4xlarge	7	60
c5d.9xlarge	7	60
c5d.12xlarge	7	60
c5d.18xlarge	14	120
c5d.24xlarge	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
c6a.large	2	10
c6a.xlarge	3	20
c6a.2xlarge	3	40
c6a.4xlarge	7	60
c6a.8xlarge	7	90
c6a.12xlarge	7	120
c6a.16xlarge	14	120
c6a.24xlarge	14	120
c6a.32xlarge	14	120
c6a.48xlarge	14	120
c6a.metal	14	120
c6g.medium	1	4
c6g.large	2	10
c6g.xlarge	3	20
c6g.2xlarge	3	40
c6g.4xlarge	7	60
c6g.8xlarge	7	60
c6g.12xlarge	7	60
c6g.16xlarge	14	120
c6g.metal	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
c6gd.medium	1	4
c6gd.large	2	10
c6gd.xlarge	3	20
c6gd.2xlarge	3	40
c6gd.4xlarge	7	60
c6gd.8xlarge	7	60
c6gd.12xlarge	7	60
c6gd.16xlarge	14	120
c6gd.metal	14	120
c6gn.medium	1	4
c6gn.large	2	10
c6gn.xlarge	3	20
c6gn.2xlarge	3	40
c6gn.4xlarge	7	60
c6gn.8xlarge	7	60
c6gn.12xlarge	7	60
c6gn.16xlarge	14	120
c6i.large	2	10
c6i.xlarge	3	20
c6i.2xlarge	3	40

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
c6i.4xlarge	7	60
c6i.8xlarge	7	90
c6i.12xlarge	7	120
c6i.16xlarge	14	120
c6i.24xlarge	14	120
c6i.32xlarge	14	120
c6i.metal	14	120
c6id.large	2	10
c6id.xlarge	3	20
c6id.2xlarge	3	40
c6id.4xlarge	7	60
c6id.8xlarge	7	90
c6id.12xlarge	7	120
c6id.16xlarge	14	120
c6id.24xlarge	14	120
c6id.32xlarge	14	120
c6id.metal	14	120
c6in.large	2	10
c6in.xlarge	3	20
c6in.2xlarge	3	40

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
c6in.4xlarge	7	60
c6in.8xlarge	7	90
c6in.12xlarge	7	120
c6in.16xlarge	14	120
c6in.24xlarge	14	120
c6in.32xlarge	15	120
c6in.metal	15	120
c7a.medium	1	4
c7a.large	2	10
c7a.xlarge	3	20
c7a.2xlarge	3	40
c7a.4xlarge	7	60
c7a.8xlarge	7	90
c7a.12xlarge	7	120
c7a.16xlarge	14	120
c7a.24xlarge	14	120
c7a.32xlarge	14	120
c7a.48xlarge	14	120
c7a.metal-48xl	14	120
c7g.medium	1	4

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
c7g.large	2	10
c7g.xlarge	3	20
c7g.2xlarge	3	40
c7g.4xlarge	7	60
c7g.8xlarge	7	60
c7g.12xlarge	7	60
c7g.16xlarge	14	120
c7g.metal	14	120
c7gd.medium	1	4
c7gd.large	2	10
c7gd.xlarge	3	20
c7gd.2xlarge	3	40
c7gd.4xlarge	7	60
c7gd.8xlarge	7	60
c7gd.12xlarge	7	60
c7gd.16xlarge	14	120
c7gd.metal	14	120
c7gn.medium	1	4
c7gn.large	2	10
c7gn.xlarge	3	20

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
c7gn.2xlarge	3	40
c7gn.4xlarge	7	60
c7gn.8xlarge	7	60
c7gn.12xlarge	7	60
c7gn.16xlarge	14	120
c7gn.metal	14	120
c7i.large	2	10
c7i.xlarge	3	20
c7i.2xlarge	3	40
c7i.4xlarge	7	60
c7i.8xlarge	7	90
c7i.12xlarge	7	120
c7i.16xlarge	14	120
c7i.24xlarge	14	120
c7i.48xlarge	14	120
c7i.metal-24xl	14	120
c7i.metal-48xl	14	120
c7i-flex.large	2	4
c7i-flex.xlarge	3	10
c7i-flex.2xlarge	3	20

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
c7i-flex.4xlarge	7	40
c7i-flex.8xlarge	7	60
c7i-flex.12xlarge	7	120
c7i-flex.16xlarge	14	120
c8g.medium	1	4
c8g.large	2	10
c8g.xlarge	3	20
c8g.2xlarge	3	40
c8g.4xlarge	7	60
c8g.8xlarge	7	60
c8g.12xlarge	7	60
c8g.16xlarge	14	120
c8g.24xlarge	14	120
c8g.48xlarge	14	120
c8g.metal-24xl	14	120
c8g.metal-48xl	14	120
c8gd.medium	1	4
c8gd.large	2	10
c8gd.xlarge	3	20
c8gd.2xlarge	3	40

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
c8gd.4xlarge	7	60
c8gd.8xlarge	7	60
c8gd.12xlarge	7	60
c8gd.16xlarge	14	120
c8gd.24xlarge	14	120
c8gd.48xlarge	14	120
c8gd.metal-24xl	14	120
c8gd.metal-48xl	14	120
c8gn.medium	1	4
c8gn.large	2	10
c8gn.xlarge	3	20
c8gn.2xlarge	3	40
c8gn.4xlarge	7	60
c8gn.8xlarge	9	60
c8gn.12xlarge	11	60
c8gn.16xlarge	15	120
c8gn.24xlarge	23	120
c8gn.48xlarge	23	120
c8gn.metal-24xl	23	120
c8gn.metal-48xl	23	120

Optimizada para memoria

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
r5.large	2	10
r5.xlarge	3	20
r5.2xlarge	3	40
r5.4xlarge	7	60
r5.12xlarge	7	60
r5.16xlarge	14	120
r5.24xlarge	14	120
r5a.large	2	10
r5a.xlarge	3	20
r5a.2xlarge	3	40
r5a.4xlarge	7	60
r5a.8xlarge	7	60
r5a.12xlarge	7	60
r5a.16xlarge	14	120
r5a.24xlarge	14	120
r5ad.large	2	10
r5ad.xlarge	3	20
r5ad.2xlarge	3	40
r5ad.4xlarge	7	60

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
r5ad.8xlarge	7	60
r5ad.12xlarge	7	60
r5ad.16xlarge	14	120
r5ad.24xlarge	14	120
r5b.16xlarge	14	120
r5d.large	2	10
r5d.xlarge	3	20
r5d.2xlarge	3	40
r5d.4xlarge	7	60
r5d.8xlarge	7	60
r5d.12xlarge	7	60
r5d.16xlarge	14	120
r5d.24xlarge	14	120
r5dn.16xlarge	14	120
r6a.large	2	10
r6a.xlarge	3	20
r6a.2xlarge	3	40
r6a.4xlarge	7	60
r6a.8xlarge	7	90
r6a.12xlarge	7	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
r6a.16xlarge	14	120
r6a.24xlarge	14	120
r6a.32xlarge	14	120
r6a.48xlarge	14	120
r6a.metal	14	120
r6g.medium	1	4
r6g.large	2	10
r6g.xlarge	3	20
r6g.2xlarge	3	40
r6g.4xlarge	7	60
r6g.8xlarge	7	60
r6g.12xlarge	7	60
r6g.16xlarge	14	120
r6g.metal	14	120
r6gd.medium	1	4
r6gd.large	2	10
r6gd.xlarge	3	20
r6gd.2xlarge	3	40
r6gd.4xlarge	7	60
r6gd.8xlarge	7	60

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
r6gd.12xlarge	7	60
r6gd.16xlarge	14	120
r6gd.metal	14	120
r6i.large	2	10
r6i.xlarge	3	20
r6i.2xlarge	3	40
r6i.4xlarge	7	60
r6i.8xlarge	7	90
r6i.12xlarge	7	120
r6i.16xlarge	14	120
r6i.24xlarge	14	120
r6i.32xlarge	14	120
r6i.metal	14	120
r6idn.large	2	10
r6idn.xlarge	3	20
r6idn.2xlarge	3	40
r6idn.4xlarge	7	60
r6idn.8xlarge	7	90
r6idn.12xlarge	7	120
r6idn.16xlarge	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
r6idn.24xlarge	14	120
r6idn.32xlarge	15	120
r6idn.metal	15	120
r6in.large	2	10
r6in.xlarge	3	20
r6in.2xlarge	3	40
r6in.4xlarge	7	60
r6in.8xlarge	7	90
r6in.12xlarge	7	120
r6in.16xlarge	14	120
r6in.24xlarge	14	120
r6in.32xlarge	15	120
r6in.metal	15	120
r6id.large	2	10
r6id.xlarge	3	20
r6id.2xlarge	3	40
r6id.4xlarge	7	60
r6id.8xlarge	7	90
r6id.12xlarge	7	120
r6id.16xlarge	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
r6id.24xlarge	14	120
r6id.32xlarge	14	120
r6id.metal	14	120
r7a.medium	1	4
r7a.large	2	10
r7a.xlarge	3	20
r7a.2xlarge	3	40
r7a.4xlarge	7	60
r7a.8xlarge	7	90
r7a.12xlarge	7	120
r7a.16xlarge	14	120
r7a.24xlarge	14	120
r7a.32xlarge	14	120
r7a.48xlarge	14	120
r7a.metal-48xl	14	120
r7g.medium	1	4
r7g.large	2	10
r7g.xlarge	3	20
r7g.2xlarge	3	40
r7g.4xlarge	7	60

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
r7g.8xlarge	7	60
r7g.12xlarge	7	60
r7g.16xlarge	14	120
r7g.metal	14	120
r7gd.medium	1	4
r7gd.large	2	10
r7gd.xlarge	3	20
r7gd.2xlarge	3	40
r7gd.4xlarge	7	60
r7gd.8xlarge	7	60
r7gd.12xlarge	7	60
r7gd.16xlarge	14	120
r7gd.metal	14	120
r7i.large	2	10
r7i.xlarge	3	20
r7i.2xlarge	3	40
r7i.4xlarge	7	60
r7i.8xlarge	7	90
r7i.12xlarge	7	120
r7i.16xlarge	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
r7i.24xlarge	14	120
r7i.48xlarge	14	120
r7i.metal-24xl	14	120
r7i.metal-48xl	14	120
r7iz.large	2	10
r7iz.xlarge	3	20
r7iz.2xlarge	3	40
r7iz.4xlarge	7	60
r7iz.8xlarge	7	90
r7iz.12xlarge	7	120
r7iz.16xlarge	14	120
r7iz.32xlarge	14	120
r7iz.metal-16xl	14	120
r7iz.metal-32xl	14	120
r8g.medium	1	4
r8g.large	2	10
r8g.xlarge	3	20
r8g.2xlarge	3	40
r8g.4xlarge	7	60
r8g.8xlarge	7	60

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
r8g.12xlarge	7	60
r8g.16xlarge	14	120
r8g.24xlarge	14	120
r8g.48xlarge	14	120
r8g.metal-24xl	14	120
r8g.metal-48xl	14	120
r8gd.medium	1	4
r8gd.large	2	10
r8gd.xlarge	3	20
r8gd.2xlarge	3	40
r8gd.4xlarge	7	60
r8gd.8xlarge	7	60
r8gd.12xlarge	7	60
r8gd.16xlarge	14	120
r8gd.24xlarge	14	120
r8gd.48xlarge	14	120
r8gd.metal-24xl	14	120
r8gd.metal-48xl	14	120
u-3tb1.56xlarge	7	12
u-6tb1.56xlarge	14	12

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
u-18tb1.112xlarge	14	12
u-18tb1.metal	14	12
u-24tb1.112xlarge	14	12
u-24tb1.metal	14	12
u7i-6tb.112xlarge	14	120
u7i-8tb.112xlarge	14	120
u7i-12tb.224xlarge	14	120
u7in-16tb.224xlarge	15	120
u7in-24tb.224xlarge	15	120
u7in-32tb.224xlarge	15	120
u7inh-32tb.480xlarge	15	120
x2gd.medium	1	10
x2gd.large	2	10
x2gd.xlarge	3	20
x2gd.2xlarge	3	40
x2gd.4xlarge	7	60
x2gd.8xlarge	7	60
x2gd.12xlarge	7	60
x2gd.16xlarge	14	120
x2gd.metal	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
x2idn.16xlarge	14	120
x2idn.24xlarge	14	120
x2idn.32xlarge	14	120
x2idn.metal	14	120
x2iedn.xlarge	3	13
x2iedn.2xlarge	3	29
x2iedn.4xlarge	7	60
x2iedn.8xlarge	7	120
x2iedn.16xlarge	14	120
x2iedn.24xlarge	14	120
x2iedn.32xlarge	14	120
x2iedn.metal	14	120
x2iezn.2xlarge	3	64
x2iezn.4xlarge	7	120
x2iezn.6xlarge	7	120
x2iezn.8xlarge	7	120
x2iezn.12xlarge	14	120
x2iezn.metal	14	120
x8g.medium	1	4
x8g.large	2	10

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
x8g.xlarge	3	20
x8g.2xlarge	3	40
x8g.4xlarge	7	60
x8g.8xlarge	7	60
x8g.12xlarge	7	60
x8g.16xlarge	14	120
x8g.24xlarge	14	120
x8g.48xlarge	14	120
x8g.metal-24xl	14	120
x8g.metal-48xl	14	120

Optimizada para almacenamiento

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
i4g.large	2	10
i4g.xlarge	3	20
i4g.2xlarge	3	40
i4g.4xlarge	7	60
i4g.8xlarge	7	60
i4g.16xlarge	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
i4i.xlarge	3	8
i4i.2xlarge	3	28
i4i.4xlarge	7	58
i4i.8xlarge	7	118
i4i.12xlarge	7	118
i4i.16xlarge	14	248
i4i.24xlarge	14	118
i4i.32xlarge	14	498
i4i.metal	14	498
i7i.large	2	10
i7i.xlarge	3	20
i7i.2xlarge	3	40
i7i.4xlarge	7	60
i7i.8xlarge	7	90
i7i.12xlarge	7	90
i7i.16xlarge	14	120
i7i.24xlarge	14	120
i7i.48xlarge	14	120
i7i.metal-24xl	14	120
i7i.metal-48xl	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
i7ie.large	2	20
i7ie.xlarge	3	29
i7ie.2xlarge	3	29
i7ie.3xlarge	3	29
i7ie.6xlarge	7	60
i7ie.12xlarge	7	60
i7ie.18xlarge	14	120
i7ie.24xlarge	14	120
i7ie.48xlarge	14	120
i7ie.metal-24xl	14	120
i7ie.metal-48xl	14	120
i8g.large	2	10
i8g.xlarge	3	20
i8g.2xlarge	3	40
i8g.4xlarge	7	60
i8g.8xlarge	7	60
i8g.12xlarge	7	60
i8g.16xlarge	14	120
i8g.24xlarge	14	120
i8g.48xlarge	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
i8g.metal-24xl	14	120
im4gn.large	2	10
im4gn.xlarge	3	20
im4gn.2xlarge	3	40
im4gn.4xlarge	7	60
im4gn.8xlarge	7	60
im4gn.16xlarge	14	120
is4gen.medium	1	4
is4gen.large	2	10
is4gen.xlarge	3	20
is4gen.2xlarge	3	40
is4gen.4xlarge	7	60
is4gen.8xlarge	7	60

Computación acelerada

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
dl1.24xlarge	59	120
dl2q.24xlarge	14	120
f2.6xlarge	7	90

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
f2.12xlarge	7	120
f2.48xlarge	14	120
g4ad.xlarge	1	12
g4ad.2xlarge	1	12
g4ad.4xlarge	2	12
g4ad.8xlarge	3	12
g4ad.16xlarge	7	12
g5.xlarge	3	6
g5.2xlarge	3	19
g5.4xlarge	7	40
g5.8xlarge	7	90
g5.12xlarge	14	120
g5.16xlarge	7	120
g5.24xlarge	14	120
g5.48xlarge	6	120
g5g.xlarge	3	20
g5g.2xlarge	3	40
g5g.4xlarge	7	60
g5g.8xlarge	7	60
g5g.16xlarge	14	120

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
g5g.metal	14	120
g6.xlarge	3	20
g6.2xlarge	3	40
g6.4xlarge	7	60
g6.8xlarge	7	90
g6.12xlarge	7	120
g6.16xlarge	14	120
g6.24xlarge	14	120
g6.48xlarge	14	120
g6e.xlarge	3	20
g6e.2xlarge	3	40
g6e.4xlarge	7	60
g6g.8xlarge	7	90
g6e.12xlarge	9	120
g6e.16xlarge	14	120
g6e.24xlarge	19	120
g6e.48xlarge	39	120
g6f.large	1	10
g6f.xlarge	3	20
g6f.2xlarge	3	40

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
g6f.4xlarge	7	60
gr6.4xlarge	7	60
gr6.8xlarge	7	90
gr6f.4xlarge	7	60
inf2.xlarge	3	20
inf2.8xlarge	7	90
inf2.24xlarge	14	120
inf2.48xlarge	14	120
p4d.24xlarge	59	120
p4de.24xlarge	59	120
p5.48xlarge	63	242
p5e.48xlarge	63	242
p5en.48xlarge	63	242
p6-b200.48xlarge	31	242
p6e-gb200.36xlarge	38	120
trn1.2xlarge	3	19
trn1.32xlarge	39	120
trn1n.32xlarge	79	242
trn2.48xlarge	31	242
trn2u.48xlarge	31	242

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
vt1.3xlarge	3	40
vt1.6xlarge	7	60
vt1.24xlarge	14	120

Computación de alto rendimiento

Tipo de instancia	Límite de tareas sin enlace troncal de ENI	Límite de tareas con enlace troncal de ENI
hpc6a.48xlarge	1	120
hpc6id.32xlarge	1	120
hpc7g.4xlarge	3	120
hpc7g.8xlarge	3	120
hpc7g.16xlarge	3	120

Reserva de la memoria de instancias de contenedor de Linux de Amazon ECS

Cuando el agente de contenedor de Amazon ECS registra una instancia de contenedor en un clúster, el agente debe determinar la cantidad de memoria disponible que puede reservar la instancia de contenedor para las tareas. Debido a la sobrecarga de memoria de la plataforma y a la memoria ocupada por el kernel del sistema, este número difiere de la cantidad de memoria instalada que se anuncia para las instancias de Amazon EC2. Por ejemplo, una instancia `m4.large` tiene 8 GiB de memoria instalada. Sin embargo, esto no siempre significa que haya exactamente 8192 MiB de memoria disponible para las tareas cuando se registra la instancia de contenedor.

El agente de contenedor de Amazon ECS proporciona una variable de configuración con el nombre `ECS_RESERVED_MEMORY`, que se puede utilizar para quitar un número concreto de MiB de memoria del grupo asignado a las tareas. Este es un mecanismo eficaz que permite reservar memoria para los procesos críticos del sistema.

Si se ocupa toda la memoria de una instancia de contenedor con las tareas, es posible que las tareas compitan con los procesos críticos del sistema por la memoria y posiblemente inicien un error del sistema.

Por ejemplo, si se especifica `ECS_RESERVED_MEMORY=256` en el archivo de configuración del agente, el agente registrará la memoria total menos 256 MiB de esa instancia y las tareas de ECS no podrán asignar 256 MiB de memoria. Para obtener más información sobre las variables de configuración del agente y cómo definirlas, consulte Configuración del agente de contenedor de Amazon ECS y Arranque de instancias de contenedor de Linux de Amazon ECS para la transferencia de datos.

Si especifica 8192 MiB para la tarea y ninguna de las instancias de contenedor tiene 8192 MiB o más de memoria disponible para satisfacer este requisito, la tarea no se puede ubicar en el clúster. Si utiliza un entorno informático administrado, AWS Batch debe iniciar un tipo de instancia de mayor tamaño para poder acomodar la solicitud.

También debe reservar algo de memoria para el agente de contenedor de Amazon ECS y otros procesos críticos del sistema en las instancias de contenedor, de modo que los contenedores de tareas no compitan por la misma memoria y pueda iniciarse un error del sistema.

El agente de contenedor de Amazon ECS utiliza la función `ReadMemInfo()` de Docker para consultar la memoria total disponible al sistema operativo. Tanto Linux como Windows cuentan con utilidades de línea de comandos para determinar la memoria total.

Example - Determinar la memoria total en Linux

El comando free devuelve la memoria total reconocida por el sistema operativo.

```
$ free -b
```

Ejemplo de la salida de una instancia `m4.large` que ejecuta la AMI de Amazon Linux optimizada para Amazon ECS.

```
             total       used       free     shared    buffers     cached
Mem:    8373026816  348180480 8024846336      90112   25534464  205418496
-/+ buffers/cache:  117227520 8255799296
```

Esta instancia tiene 8 373 026 816 bytes de memoria total, lo que se traduce en 7 985 MiB disponibles para tareas.

Example - Determinar la memoria total en Windows

El comando wmic devuelve la memoria total reconocida por el sistema operativo.

```
C:\> wmic ComputerSystem get TotalPhysicalMemory
```

Ejemplo de la salida de una instancia `m4.large` que ejecuta la AMI de Windows Server optimizada para Amazon ECS.

```
TotalPhysicalMemory
8589524992
```

Esta instancia tiene 8 589 524 992 bytes de memoria total, lo que se traduce en 8 191 MiB disponibles para tareas.

Visualización de la memoria de instancias de contenedor

Puede consultar la cantidad de memoria que registra una instancia de contenedor a través de la consola de Amazon ECS (o mediante la operación de la API DescribeContainerInstances). Si, para intentar maximizar el uso de recursos, proporciona la mayor cantidad de memoria posible a las tareas para un determinado tipo de instancia, puede observar la memoria disponible para esa instancia de contenedor y, a continuación, asignar a las tareas esa cantidad de memoria.

Visualización de la memoria de la instancia de contenedor

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clústeres y elija el clúster que aloja su instancia de contenedor.
3. Elija Infraestructura y, a continuación, en Instancias de contenedor, elija una instancia de contenedor.
4. En la sección Recursos, se muestra la memoria registrada y la memoria disponible para la instancia de contenedor.

   El valor del campo Registrada corresponde a la memoria que la instancia de contenedor registró en Amazon ECS cuando se inició por primera vez, mientras que el valor del campo Disponible corresponde a la memoria que aún no se ha asignado a ninguna tarea.

Administración remota de instancias de contenedor de Amazon ECS mediante AWS Systems Manager

Puede utilizar la función Run Command de AWS Systems Manager (Systems Manager) para administrar la configuración de las instancias de contenedor de Amazon ECS de forma segura y remota. Run Command proporciona una manera sencilla de realizar tareas administrativas comunes sin necesidad de iniciar sesión localmente en la instancia. Puede administrar cambios de configuración en los clústeres ejecutando comandos simultáneamente en varias instancia de contenedor. Ejecutar comando notifica el estado y los resultados de cada comando.

Aquí tiene algunos ejemplos de los tipos de tareas que puede llevar a cabo con Ejecutar comando:

- Instalar o desinstalar paquetes.
- Realizar actualizaciones de seguridad.
- Limpiar imágenes de Docker.
- Parar o comenzar servicios.
- Ver recursos del sistema.
- Ver archivos de registro.
- Realizar operaciones de archivo.

Para obtener más información acerca de Run Command, consulte AWS Systems Manager Run Command en la Guía del usuario de AWS Systems Manager.

A continuación, se indican los son requisitos previos para usar Systems Manager con Amazon ECS.

1. Debe conceder permisos al rol de instancia de contenedor (ecsInstanceRole) para acceder a las API de Systems Manager. Para ello, asigne AmazonSSMManagedInstanceCore al rol `ecsInstanceRole`. Para obtener información acerca de cómo asociar una política a un rol, consulte Actualización de los permisos de un rol en la Guía del usuario de AWS Identity and Access Management.
2. Compruebe que SSM Agent esté instalado en las instancias del contenedor. Para obtener más información, consulte Instalación y desinstalación manual de SSM Agent en instancias EC2 para Linux.

Después de asociar políticas administradas de Systems Manager al `ecsInstanceRole` y de verificar que el agente de AWS Systems Manager (SSM Agent) esté instalado en las instancias de

contenedor, puede comenzar a utilizar Run Command para enviar comandos a las instancias de contenedor. Para obtener información acerca de la ejecución de comandos y scripts del shell en las instancias y la visualización del resultado obtenido, consulte Ejecución de comandos mediante Run Command de Systems Manager y Explicaciones sobre Run Command en la Guía del usuario de AWS Systems Manager.

Un caso de uso común es actualizar el software de la instancia de contenedor con Run Command. Puede seguir los procedimientos de la Guía del usuario de AWS Systems Manager con los siguientes parámetros.

Parámetro	Valor
Documento de comandos	`AWS-RunShellScript`
Comando	`$ yum update -y`
Instancias de destino	Sus instancias de contenedor

Uso de un proxy HTTP para instancias de contenedor de Linux de Amazon ECS

Puede configurar las instancias de contenedor de Amazon ECS para que utilicen un proxy HTTP tanto para el agente de contenedor de Amazon ECS como para el daemon de Docker. Esto resulta útil si las instancias de contenedor no tienen acceso de red externo a través de una gateway de Internet de Amazon VPC, una gateway NAT o una instancia.

Para configurar la instancia de contenedor de Linux de Amazon ECS de modo que utilice un proxy HTTP, establezca las siguientes variables en los archivos correspondiente en el momento del lanzamiento (con datos de usuario de Amazon EC2). También puede editar manualmente el archivo de configuración y, a continuación, reiniciar el agente.

`/etc/ecs/ecs.config` (AMI de Amazon Linux y Amazon Linux 2)

```
HTTP_PROXY=10.0.0.131:3128
```

Establezca este valor en el nombre de host (o dirección IP) y en el número de puerto de un proxy HTTP que se utilizará para que el agente de Amazon ECS se conecte a Internet. Por ejemplo, las instancias de contenedor podrían no tener acceso de red externo a través de una gateway de Internet de Amazon VPC, una gateway NAT o una instancia.

`NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock`

Establezca este valor en `169.254.169.254,169.254.170.2,/var/run/docker.sock` para filtrar los metadatos de las instancias EC2, los roles de IAM para tareas y el tráfico del daemon de Docker procedente del proxy.

`/etc/systemd/system/ecs.service.d/http-proxy.conf` (Amazon Linux 2 solamente)

```
Environment="HTTP_PROXY=10.0.0.131:3128/"
```

Establezca este valor en el nombre de host (o dirección IP) y en el número de puerto de un proxy HTTP que desee utilizar para que `ecs-init` se conecte a Internet. Por ejemplo, las instancias de contenedor podrían no tener acceso de red externo a través de una gateway de Internet de Amazon VPC, una gateway NAT o una instancia.

```
Environment="NO_PROXY=169.254.169.254,169.254.170.2,/var/run/
docker.sock"
```

Establezca este valor en `169.254.169.254,169.254.170.2,/var/run/docker.sock` para filtrar los metadatos de las instancias EC2, los roles de IAM para tareas y el tráfico del daemon de Docker procedente del proxy.

`/etc/init/ecs.override` (AMI de Amazon Linux solamente)

```
env HTTP_PROXY=10.0.0.131:3128
```

Establezca este valor en el nombre de host (o dirección IP) y en el número de puerto de un proxy HTTP que desee utilizar para que `ecs-init` se conecte a Internet. Por ejemplo, las instancias de contenedor podrían no tener acceso de red externo a través de una gateway de Internet de Amazon VPC, una gateway NAT o una instancia.

```
env NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock
```

Establezca este valor en `169.254.169.254,169.254.170.2,/var/run/docker.sock` para filtrar los metadatos de las instancias EC2, los roles de IAM para tareas y el tráfico del daemon de Docker procedente del proxy.

`/etc/systemd/system/docker.service.d/http-proxy.conf` (Amazon Linux 2 solamente)

```
Environment="HTTP_PROXY=http://10.0.0.131:3128"
```

Establezca este valor en el nombre de host (o dirección IP) y en el número de puerto de un proxy HTTP que utilizar para que el daemon de Docker se conecte a Internet. Por ejemplo, las instancias de contenedor podrían no tener acceso de red externo a través de una gateway de Internet de Amazon VPC, una gateway NAT o una instancia.

```
Environment="NO_PROXY=169.254.169.254,169.254.170.2"
```

Establezca este valor en `169.254.169.254,169.254.170.2` para filtrar metadatos de instancia EC2 desde el proxy.

`/etc/sysconfig/docker` (solo AMI de Amazon Linux y Amazon Linux 2)

```
export HTTP_PROXY=http://10.0.0.131:3128
```

Establezca este valor en el nombre de host (o dirección IP) y en el número de puerto de un proxy HTTP que utilizar para que el daemon de Docker se conecte a Internet. Por ejemplo, las instancias de contenedor podrían no tener acceso de red externo a través de una gateway de Internet de Amazon VPC, una gateway NAT o una instancia.

```
export NO_PROXY=169.254.169.254,169.254.170.2
```

Establezca este valor en `169.254.169.254,169.254.170.2` para filtrar metadatos de instancia EC2 desde el proxy.

Establecer estas variables de entorno en los archivos anteriores solo afecta al agente de contenedor de Amazon ECS, a `ecs-init` y al daemon de Docker. No configuran ningún otro servicio (como yum) para que utilice el proxy.

Para obtener información sobre cómo configurar el proxy, consulte How do I set up an HTTP proxy for Docker and the Amazon ECS container agent in Amazon Linux 2 or AL2023.

Configuración de instancias preinicializadas para el grupo de escalado automático de Amazon ECS

Amazon ECS admite grupos de calentamiento de Amazon EC2 Auto Scaling. Un grupo de calentamiento es un grupo de Amazon EC2 instances (Instancias de Amazon EC2) inicializadas previamente listas para ponerse en servicio. Siempre que su aplicación necesita escalar horizontalmente, Amazon EC2 Auto Scaling utiliza las instancias preinicializadas del grupo de calentamiento en lugar de lanzar instancias en frío, permite ejecutar cualquier proceso de inicialización final y, a continuación, pone la instancia en servicio.

Para obtener más información sobre grupos de calentamiento y cómo agregar un grupo de calentamiento a un grupo de Auto Scaling, consulte Grupos de calentamiento para Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

Cuando se crea o actualiza un grupo de calentamiento para un grupo de escalado automático para Amazon ECS, no se puede configurar la opción que devuelve las instancias al grupo de

calentamiento al reducir horizontalmente (`ReuseOnScaleIn`). Para obtener más información, consulte put-warm-pool en la Referencia de AWS Command Line Interface.

Para utilizar los grupos de calentamiento con su clúster de Amazon ECS, establezca la variable de configuración del agente `ECS_WARM_POOLS_CHECK` en `true` en el campo User data (Datos de usuario) de la plantilla de lanzamiento del grupo de Amazon EC2 Auto Scaling.

A continuación, mostramos un ejemplo de cómo se puede especificar la variable de configuración en el campo User data (Datos de usuario) de una plantilla de lanzamiento de Amazon EC2. Reemplace *`MyCluster`* por el nombre del clúster.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_WARM_POOLS_CHECK=true
EOF
```

Esta variable `ECS_WARM_POOLS_CHECK` solo se admite en versiones de agente `1.59.0` y posterior. Para obtener más información sobre la variable, consulte Configuración del agente de contenedor de Amazon ECS.

Actualización del agente de contenedor de Amazon ECS

Ocasionalmente, es posible que tenga que actualizar el agente de contenedor de Amazon ECS para obtener correcciones de errores y nuevas características. La actualización del agente de contenedor de Amazon ECS no interrumpe las tareas ni los servicios en ejecución en la instancia de contenedor. El proceso de actualización del agente difiere en función de si la instancia de contenedor se lanzó con una AMI optimizada para Amazon ECS u otro sistema operativo.

> **Note**
>
> Las actualizaciones del agente no se aplican a instancias de contenedor de Windows. Le recomendamos que lance nuevas instancias de contenedor para actualizar la versión del agente en sus clústeres Windows.

Comprobación de la versión del agente de contenedor de Amazon ECS

Puede comprobar la versión del agente de contenedor que se está ejecutando en sus instancias de contenedor para ver si necesita actualizarlo. La vista de la instancia de contenedor en la consola de

Amazon ECS proporciona la versión del agente. Utilice el siguiente procedimiento para comprobar la versión del agente.

Amazon ECS console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, elija la región en la que se encuentra registrada la instancia externa.
3. En el panel de navegación, elija Clusters (Clústeres) y seleccione el clúster que aloja la instancia externa.
4. En la página de Cluster : ***name*** (Clúster; nombre), elija la pestaña Infrastructure (Infraestructura).
5. En Container instances (Instancias de contenedor), tenga en cuenta la columna Agent version (Versión de agente) para sus instancias de contenedor. Si la instancia de contenedor no contiene la versión más reciente del agente de contenedor, la consola genera un mensaje de alerta y marca la versión del agente obsoleta.

   Si la versión de su agente está desactualizada, puede actualizar el agente de contenedor con los siguientes procedimientos:

   - Si la instancia de contenedor está ejecutando una AMI optimizada para Amazon ECS, consulte Actualización del agente de contenedor de Amazon ECS en una AMI optimizada para Amazon ECS.
   - Si la instancia de contenedor no está ejecutando una AMI optimizada para Amazon ECS, consulte Actualización manual del agente de contenedor de Amazon ECS (para AMI no optimizadas para Amazon ECS).

   > **Important**
   >
   > Para actualizar la versión del agente de Amazon ECS de versiones anteriores a la v1.0.0 en la AMI optimizada para Amazon ECS, le recomendamos que termine la instancia de contenedor actual y lance una instancia nueva con la versión de la AMI más reciente. Cualquier instancia de contenedor que utilice una versión de vista previa se debe retirar y sustituir por la AMI más reciente. Para obtener más información, consulte Lanzamiento de una instancia de contenedor de Linux de Amazon ECS.

Amazon ECS container agent introspection API

También puede utilizar la API de introspección del agente de contenedor de Amazon ECS para comprobar la versión del agente desde la propia instancia de contenedor. Para obtener más información, consulte Introspección de contenedor de Amazon ECS.

Para comprobar si el agente de contenedor de Amazon ECS ejecuta la versión más reciente a través de la API de introspección

1. Inicie sesión en su instancia de contenedor mediante SSH.
2. Consulte la API de introspección.

```
[ec2-user ~]$ curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
```

> **Note**
>
> La API de introspección agregó información de `Version` en la versión v1.0.0 del agente de contenedor de Amazon ECS. Si no tiene `Version` a la hora de consultar la API de introspección o la API de introspección no está presente en el agente en absoluto, entonces la versión que ejecuta es v0.0.3 o anterior. Debería actualizar la versión.

Actualización del agente de contenedor de Amazon ECS en una AMI optimizada para Amazon ECS

Si está utilizando la AMI optimizada para Amazon ECS, dispone de varias opciones para obtener la versión más reciente del agente de contenedor de Amazon ECS (se muestran por orden de recomendación):

- Termine las instancias de contenedor actuales y lance la versión más reciente de la AMI de Amazon Linux 2 optimizada para Amazon ECS (ya sea manualmente o actualizando la configuración de lanzamiento de Auto Scaling con la AMI más reciente). Esto proporciona una instancia de contenedor nueva con las versiones probadas y validadas más recientes de Amazon Linux, Docker, `ecs-init` y el agente de contenedor de Amazon ECS. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.
- Conecte a la instancia con SSH y actualice el paquete `ecs-init` (y sus dependencias) a la versión más reciente. Esta operación ofrece las versiones probadas y validadas más recientes de Docker y `ecs-init` que están disponibles en los repositorios de Amazon Linux, así como la

versión más reciente del agente de contenedor de Amazon ECS. Para obtener más información, consulte Para actualizar el paquete `ecs-init` en la AMI optimizada para Amazon ECS.

- Actualice el agente de contenedor con la operación `UpdateContainerAgent` de la API, ya sea a través de la consola, con la AWS CLI o con los SDK de AWS. Para obtener más información, consulte Actualización del agente de contenedor de Amazon ECS mediante la operación de la API `UpdateContainerAgent`.

> **Note**
>
> Las actualizaciones del agente no se aplican a instancias de contenedor de Windows. Le recomendamos que lance nuevas instancias de contenedor para actualizar la versión del agente en sus clústeres Windows.

Para actualizar el paquete **`ecs-init`** en la AMI optimizada para Amazon ECS

1. Inicie sesión en su instancia de contenedor mediante SSH.
2. Actualice el paquete `ecs-init` con el siguiente comando.

```
sudo yum update -y ecs-init
```

> **Note**
>
> El paquete `ecs-init` y el agente de contenedor de Amazon ECS se actualizan de forma inmediata. Sin embargo, las versiones más recientes de Docker no se cargan hasta que se reinicia el daemon de Docker. Para efectuar el reinicio, puede reiniciar la instancia o ejecutar los siguientes comandos en su instancia:
>
> - AMI de Amazon Linux 2 optimizada para Amazon ECS:
>
> ```
> sudo systemctl restart docker
> ```
>
> - AMI de Amazon Linux optimizada para Amazon ECS:
>
> ```
> sudo service docker restart && sudo start ecs
> ```

Actualización del agente de contenedor de Amazon ECS mediante la operación de la API **`UpdateContainerAgent`**

> **Important**
>
> La API `UpdateContainerAgent` solo se admite en variantes de Linux de la AMI optimizada para Amazon ECS, a excepción de la AMI de Amazon Linux 2 (arm64) optimizada para Amazon ECS. Para instancias de contenedor que utilizan la AMI de Amazon Linux 2 (arm64) optimizada para Amazon ECS, actualice el paquete `ecs-init` para actualizar el agente. Para instancias de contenedor que están ejecutando otros sistemas operativos, consulte Actualización manual del agente de contenedor de Amazon ECS (para AMI no optimizadas para Amazon ECS). Si utiliza instancias de contenedor de Windows, le recomendamos que lance nuevas instancias de contenedor para actualizar la versión del agente en los clústeres Windows.

El proceso de la API `UpdateContainerAgent` comienza cuando solicita una actualización del agente, ya sea a través de la consola o con la AWS CLI o los SDK de AWS. Amazon ECS compara la versión actual del agente con la versión del agente más reciente disponible y si es posible una actualización. Si no es posible una actualización, por ejemplo, si el agente ya está ejecutando la versión más reciente, se devuelve `NoUpdateAvailableException`.

Las fases en el proceso de actualización mostradas más arriba son las siguientes:

`PENDING`

Hay una actualización de agente disponible y el proceso de actualización se ha iniciado.

`STAGING`

El agente ha comenzado a descargar la actualización del agente. Si el agente no puede descargar la actualización o si el contenido de la actualización es incorrecto o está dañada, entonces el agente envía una notificación del error y la actualización pasa al estado `FAILED`.

`STAGED`

La descarga del agente se ha completado y se ha verificado el contenido del agente.

`UPDATING`

El servicio `ecs-init` se reinicia y recoge la nueva versión del agente. Si, por alguna razón, el agente no puede reiniciarse, la actualización pasa al estado `FAILED`; de lo contrario, el agente indica a Amazon ECS que la actualización está completa.

> **Note**
>
> Las actualizaciones del agente no se aplican a instancias de contenedor de Windows. Le recomendamos que lance nuevas instancias de contenedor para actualizar la versión del agente en sus clústeres Windows.

Para actualizar el agente de contenedor de Amazon ECS en una AMI optimizada para Amazon ECS desde la consola

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, elija la región en la que se encuentra registrada la instancia externa.
3. En el panel de navegación, elija Clusters y seleccione el clúster.
4. En la página de Cluster : ***name*** (Clúster; nombre), elija la pestaña Infrastructure (Infraestructura).
5. En Instancias de contenedor, seleccione las instancias que desea actualizar y, a continuación, elija Acciones, Actualización del agente.

Actualización manual del agente de contenedor de Amazon ECS (para AMI no optimizadas para Amazon ECS)

Ocasionalmente, es posible que tenga que actualizar el agente de contenedor de Amazon ECS para obtener correcciones de errores y nuevas características. La actualización del agente de contenedor de Amazon ECS no interrumpe las tareas ni los servicios en ejecución en la instancia de contenedor.

> **Note**
>
> Las actualizaciones del agente no se aplican a instancias de contenedor de Windows. Le recomendamos que lance nuevas instancias de contenedor para actualizar la versión del agente en sus clústeres Windows.

1. Inicie sesión en su instancia de contenedor mediante SSH.
2. Compruebe si su agente utiliza la variable de entorno `ECS_DATADIR` para guardar su estado.

```
ubuntu:~$ docker inspect ecs-agent | grep ECS_DATADIR
```

Salida:

```
"ECS_DATADIR=/data",
```

> **Important**
>
> Si el comando anterior no devuelve la variable de entorno `ECS_DATADIR`, debe detener las tareas en ejecución en esta instancia de contenedor antes de actualizar el agente. Los agentes más recientes con la variable de entorno `ECS_DATADIR` guardan su estado y usted puede actualizarlos mientras que las tareas se ejecuten sin problemas.

3. Detenga el agente de contenedor de Amazon ECS.

```
ubuntu:~$ docker stop ecs-agent
```

4. Elimine el contenedor de agente.

```
ubuntu:~$ docker rm ecs-agent
```

5. Asegúrese de que el directorio `/etc/ecs` y el archivo de configuración del agente de contenedor de Amazon ECS existan en `/etc/ecs/ecs.config`.

```
ubuntu:~$ sudo mkdir -p /etc/ecs && sudo touch /etc/ecs/ecs.config
```

6. Edite el archivo `/etc/ecs/ecs.config` y asegúrese de que contenga al menos las siguientes declaraciones de variables. Si no desea que su instancias de contenedor se registre en el clúster predeterminado, especifique el nombre del clúster como el valor para `ECS_CLUSTER`.

```
ECS_DATADIR=/data
ECS_ENABLE_TASK_IAM_ROLE=true
ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true
ECS_LOGFILE=/log/ecs-agent.log
ECS_AVAILABLE_LOGGING_DRIVERS=["json-file","awslogs"]
ECS_LOGLEVEL=info
```

```
ECS_CLUSTER=default
```

Para obtener más información acerca de estas y otras opciones de tiempo de ejecución de agente, consulte Configuración del agente de contenedor de Amazon ECS.

> **Note**
>
> Si lo desea, puede almacenar las variables de entorno del agente en Amazon S3 (se pueden descargar en las instancias de contenedor en el momento del lanzamiento utilizando datos de usuario de Amazon EC2). Se recomienda su uso para información confidencial como las credenciales de autenticación para repositorios privados. Para obtener más información, consulte Almacenamiento de la configuración de instancia de contenedor de Amazon ECS en Amazon S3 y Uso de imágenes de contenedor que no sean de AWS en Amazon ECS.

7. Extraiga la imagen más reciente del agente de contenedor de Amazon ECS de Amazon Elastic Container Registry Public.

```
ubuntu:~$ docker pull public.ecr.aws/ecs/amazon-ecs-agent:latest
```

Salida:

```
Pulling repository amazon/amazon-ecs-agent
a5a56a5e13dc: Download complete
511136ea3c5a: Download complete
9950b5d678a1: Download complete
c48ddcf21b63: Download complete
Status: Image is up to date for amazon/amazon-ecs-agent:latest
```

8. Ejecute el agente de contenedor de Amazon ECS más reciente en la instancia de contenedor.

> **Note**
>
> Utilice las políticas de reinicio de Docker o un administrador de procesos (como upstart o systemd) para tratar al agente de contenedor como un servicio o un daemon y asegurarse de que se reinicie después de finalizar su ejecución. Para eso, la AMI optimizada para Amazon ECS utiliza el RPM `ecs-init`, y puede consultar el código fuente para este RPM en GitHub.

En el siguiente ejemplo, el comando de ejecución del agente está dividido en líneas separadas para mostrar cada opción. Para obtener más información acerca de estas y otras opciones de tiempo de ejecución de agente, consulte Configuración del agente de contenedor de Amazon ECS.

> **Important**
>
> Los sistemas operativos con SELinux habilitado requieren la opción `--privileged` en el comando docker run. Además, para las instancias de contenedor con SELinux habilitado, recomendamos añadir la opción `:Z` a los montajes de volúmenes `/log` y `/data`. No obstante, los montajes de hosts para estos volúmenes deben existir antes de que ejecute el comando; de lo contrario, recibirá un error `no such file or directory`. Realice la siguiente acción si tiene dificultades para ejecutar el agente de Amazon ECS en una instancia de contenedor con SELinux habilitado:
>
> - Cree los puntos de montaje de volumen del host en su instancia de contenedor.
>
> ```
> ubuntu:~$ sudo mkdir -p /var/log/ecs /var/lib/ecs/data
> ```
>
> - Añada la opción `--privileged` al siguiente comando docker run.
> - Añada la opción `:Z` a los montajes del volumen de contenedor `/log` y `/data` (por ejemplo, `--volume=/var/log/ecs/:/log:Z`) para el siguiente comando docker run.

```
ubuntu:~$ sudo docker run --name ecs-agent \
--detach=true \
--restart=on-failure:10 \
--volume=/var/run:/var/run \
--volume=/var/log/ecs/:/log \
--volume=/var/lib/ecs/data:/data \
--volume=/etc/ecs:/etc/ecs \
--volume=/etc/ecs:/etc/ecs/pki \
--net=host \
--env-file=/etc/ecs/ecs.config \
amazon/amazon-ecs-agent:latest
```

> **Note**
>
> Si recibe el mensaje `Error response from daemon: Cannot start container`, puede eliminar el contenedor con errores con el comando sudo docker rm ecs-agent e intentar volver a ejecutar el agente.

AMI de Windows optimizadas para Amazon ECS

Las AMI optimizadas para Amazon ECS están preconfiguradas con los componentes necesarios para ejecutar cargas de trabajo de Amazon ECS. Aunque puede crear su propia AMI de instancia de contenedor que cumpla con las especificaciones básicas necesarias para ejecutar las cargas de trabajo en contenedores en Amazon ECS, la preconfiguración y la prueba de las AMI optimizadas para Amazon ECS la realizan los ingenieros de AWS en Amazon ECS. Es la forma más sencilla para empezar y para conseguir que los contenedores funcionen en AWS rápidamente.

Los metadatos de la AMI optimizada para Amazon ECS de cada variante, incluidos el nombre de la AMI, la versión del agente de contenedor de Amazon ECS y la versión del tiempo de ejecución de Amazon ECS que incluye la versión de Docker, se pueden recuperar mediante programación. Para obtener más información, consulte the section called “Recuperación de metadatos de las AMI de Windows optimizadas para Amazon ECS”.

> **Important**
>
> Todas las variantes de AMI optimizadas para ECS producidas después de agosto de 2022 migrarán de Docker EE (Mirantis) a Docker CE (proyecto Moby).
> Para asegurarse de que los clientes disponen de las actualizaciones de seguridad más recientes de forma predeterminada, Amazon ECS mantiene al menos las últimas tres AMI de Windows optimizada para Amazon ECS. Después de lanzar nuevas AMI de Windows optimizadas para Amazon ECS, Amazon ECS convierte en privadas las AMI de Windows optimizadas para Amazon ECS más antiguas. Para informarnos que necesita obtener acceso a una AMI privada, envíe un ticket al equipo de Cloud Support.

Variantes de AMI optimizadas para Amazon ECS

Las siguientes variantes de Windows Server de la AMI optimizada para Amazon ECS están disponibles para las instancias de Amazon EC2.

> **Important**
>
> Todas las variantes de AMI optimizadas para ECS producidas después de agosto migrarán de Docker EE (Mirantis) a Docker CE (proyecto Moby).

- AMI de Windows Server 2025 Full optimizada para Amazon ECS
- AMI de Windows Server 2025 Core optimizada para Amazon ECS
- AMI de Windows Server 2022 Full optimizada para Amazon ECS
- AMI de Windows Server 2022 Core optimizada para Amazon ECS
- AMI de Windows Server 2019 Full optimizada para Amazon ECS
- AMI de Windows Server 2019 Core optimizada para Amazon ECS
- AMI de Windows Server 2016 Full optimizada para Amazon ECS

> **Important**
>
> Windows Server 2016 no es compatible con la última versión de Docker, por ejemplo, la 25.x.x. Por lo tanto, las AMI completas de Windows Server 2016 no recibirán parches de seguridad o de errores en el entorno en tiempo de ejecución de Docker. Le recomendamos que cambie a una de las siguientes plataformas de Windows:
>
> - Windows Server 2022 Full
> - Windows Server 2022 Core
> - Windows Server 2019 Full
> - Windows Server 2019 Core

El 9 de agosto de 2022, la AMI de Windows Server 20H2 Core optimizada para Amazon ECS llegó a su fecha de fin de soporte. No se lanzarán nuevas versiones de esta AMI. Para obtener más información, vea Información de la versión de Windows Server.

Windows Server 2025, Windows Server 2022, Windows Server 2019 y Windows Server 2016 son versiones de canal de servicio a largo plazo (LTSC). Windows Server 20H2 es una versión de canal semestral (SAC). Para obtener más información, vea Información de la versión de Windows Server.

Consideraciones

Estas son algunas cuestiones que debería saber acerca de los contenedores de Amazon EC2 Windows y Amazon ECS.

- Los contenedores de Windows no se pueden ejecutar en instancias de contenedor Linux y viceversa. Para lograr una mejor ubicación de tareas de Windows y de Linux, debería mantener las instancias de contenedor de Windows y de Linux en clústeres independientes y colocar solo las tareas de Windows en contenedores de Windows. Puede asegurarse de que las definiciones de tareas de Windows solo se coloquen en instancias de Windows estableciendo la siguiente restricción de colocación: `memberOf(ecs.os-type=='windows')`.
- Los contenedores de Windows son compatibles con las tareas que utilizan el tipo de lanzamiento de EC2 y Fargare.
- Los contenedores y las instancias de contenedor de Windows no pueden admitir todos los parámetros de definición de tareas disponibles para contenedores e instancias de contenedor de Linux. Algunos parámetros directamente no se admiten, mientras que otros se comportan de modo distinto en Windows y en Linux. Para obtener más información, consulte Diferencias en la definición de tareas de Amazon ECS para instancias de EC2 que ejecutan Windows.
- En cuanto a la característica de roles de IAM para las tareas, debe configurar las instancias de contenedor de Windows para habilitarla durante el lanzamiento. Los contenedores deben ejecutar algún código de PowerShell proporcionado cuando utilicen la característica. Para obtener más información, consulte Configuración adicional de las instancias de Amazon EC2 de Windows.
- La característica de roles de IAM para las tareas utiliza un proxy de credenciales que proporcionar credenciales a los contenedores. Este proxy de credenciales ocupa el puerto 80 de la instancia de contenedor, es decir que si utiliza los roles de IAM para las tareas, el puerto 80 no está disponible para tareas. En el caso de los contenedores de servicio web, puede utilizar un Application Load Balancer y un mapeo de puertos dinámico para proporcionar conexiones HTTP estándar en el puerto 80 a los contenedores. Para obtener más información, consulte Uso del equilibrador de carga para distribuir el tráfico de servicio de Amazon ECS.
- Las imágenes de Docker de Windows Server son grandes (9 GiB). Por lo tanto, las instancias de contenedor de Windows requieren más espacio de almacenamiento que las instancias de contenedor de Linux.
- Para ejecutar un contenedor de Windows en un Windows Server, la versión del sistema operativo de imagen base del contenedor debe coincidir con la del host. Para obtener más información, consulte Compatibilidad de versiones de contenedores Windows en el sitio web de documentación de Microsoft. Si el clúster ejecuta varias versiones de Windows, puede

asegurarse de que la tarea se coloque en una instancia de EC2 que se ejecute en la misma versión mediante la restricción de ubicación: `memberOf(attribute:ecs.os-family == WINDOWS_SERVER_<OS_Release>_<FULL or CORE>)`. Para obtener más información, consulte the section called "Recuperación de metadatos de las AMI de Windows optimizadas para Amazon ECS".

Recuperación de metadatos de las AMI de Windows optimizadas para Amazon ECS

Para recuperar el ID de la AMI, el nombre de la imagen, el sistema operativo, la versión del agente de contenedor y la versión del tiempo de ejecución de las AMI optimizada para Amazon ECS mediante programación, consulte la API del Parameter Store de Systems Manager. Para obtener más información acerca de la API del Parameter Store de Systems Manager, consulte GetParameters y GetParametersByPath.

> **Note**
>
> La cuenta administrativa debe tener los siguientes permisos de IAM para recuperar los metadatos de la AMI optimizada para Amazon ECS. Estos permisos se han añadido a la política de IAM `AmazonECS_FullAccess`.
>
> - ssm:GetParameters
> - ssm:GetParameter
> - ssm:GetParametersByPath

Formato de los parámetros de Parameter Store de Systems Manager

> **Note**
>
> Los siguientes parámetros de la API de Parameter Store de Systems Manager están obsoletos y no deben utilizarse para recuperar las AMI de Windows más recientes:
>
> - `/aws/service/ecs/optimized-ami/windows_server/2016/english/full/recommended/image_id`
> - `/aws/service/ecs/optimized-ami/windows_server/2019/english/full/recommended/image_id`

A continuación, se muestra el formato del nombre del parámetro para cada variante de AMI optimizada para Amazon ECS.

- Metadatos de la AMI de Windows Server 2025 Full:

```
/aws/service/ami-windows-latest/Windows_Server-2025-English-Full-ECS_Optimized
```

- Metadatos de la AMI de Windows Server 2025 Core:

```
/aws/service/ami-windows-latest/Windows_Server-2025-English-Core-ECS_Optimized
```

- Metadatos de la AMI de Windows Server 2022 Full:

```
/aws/service/ami-windows-latest/Windows_Server-2022-English-Full-ECS_Optimized
```

- Metadatos de la AMI de Windows Server 2022 Core:

```
/aws/service/ami-windows-latest/Windows_Server-2022-English-Core-ECS_Optimized
```

- Metadatos de la AMI de Windows Server 2019 Full:

```
/aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized
```

- Metadatos de la AMI de Windows Server 2019 Core:

```
/aws/service/ami-windows-latest/Windows_Server-2019-English-Core-ECS_Optimized
```

- Metadatos de la AMI de Windows Server 2016 Full:

```
/aws/service/ami-windows-latest/Windows_Server-2016-English-Full-ECS_Optimized
```

El siguiente formato de nombre de parámetro recupera los metadatos de la versión estable más reciente de la AMI completa 2019 de Windows Server.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-
English-Full-ECS_Optimized
```

A continuación se muestra un ejemplo del objeto JSON que se devuelve para el valor del parámetro.

```
{
```

```
    "Parameters": [
        {
            "Name": "/aws/service/ami-windows-latest/Windows_Server-2019-English-Full-
ECS_Optimized",
            "Type": "String",
            "Value": "{\"image_name\":\"Windows_Server-2019-English-Full-
ECS_Optimized-2023.06.13\",\"image_id\":\"ami-0debc1fb48e4aee16\",\"ecs_runtime_version
\":\"Docker (CE) version 20.10.21\",\"ecs_agent_version\":\"1.72.0\"}",
            "Version": 58,
            "LastModifiedDate": "2023-06-22T19:37:37.841000-04:00",
            "ARN": "arn:aws:ssm:us-east-1::parameter/aws/service/ami-windows-latest/
Windows_Server-2019-English-Full-ECS_Optimized",
            "DataType": "text"
        }
    ],
    "InvalidParameters": []
}
```

Cada uno de los campos de la salida anterior están disponibles para consultarse como parámetros secundarios. Para crear la ruta de parámetros correspondiente a un parámetro secundario, agregue el nombre del parámetro secundario a la ruta de la AMI seleccionada. Están disponibles los siguientes parámetros secundarios:

- `schema_version`
- `image_id`
- `image_name`
- `os`
- `ecs_agent_version`
- `ecs_runtime_version`

Ejemplos

Los siguientes ejemplos muestran formas en las que pueden recuperar los metadatos de cada variante de AMI optimizada para Amazon ECS.

Recuperación de los metadatos de la AMI optimizada para Amazon ECS estable más reciente

Utilice los siguientes comandos de la AWS CLI para recuperar la AMI optimizada para Amazon ECS estable más reciente mediante la AWS CLI.

- Para la AMI de Windows Server 2025 Full optimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-
English-Full-ECS_Optimized --region us-east-1
```

- Para la AMI de Windows Server 2025 Core optimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-
English-Core-ECS_Optimized --region us-east-1
```

- Para la AMI de Windows Server 2022 Full optimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-
English-Full-ECS_Optimized --region us-east-1
```

- Para la AMI de Windows Server 2022 Core optimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-
English-Core-ECS_Optimized --region us-east-1
```

- Para la AMI de Windows Server 2019 Full optimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-
English-Full-ECS_Optimized --region us-east-1
```

- Para la AMI de Windows Server 2019 Core optimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-
English-Core-ECS_Optimized --region us-east-1
```

- Para la AMI de Windows Server 2016 Full optimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2016-
English-Full-ECS_Optimized --region us-east-1
```

Utilización de la AMI optimizada para Amazon ECS más reciente recomendada en una plantilla de AWS CloudFormation

Para hacer referencia a la AMI optimizada para Amazon ECS recomendada en una plantilla de AWS CloudFormation, pude hacer referencia al nombre del almacén de parámetros de Systems Manager.

```
Parameters:
  LatestECSOptimizedAMI:
    Description: AMI ID
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ami-windows-latest/Windows_Server-2019-English-Full-
ECS_Optimized/image_id
```

Versiones de las AMI de Windows optimizadas para Amazon ECS

Vea las versiones anteriores y la actual de las AMI optimizadas para Amazon ECS y sus respectivas versiones del agente de contenedor de Amazon ECS, de Docker y del paquete `ecs-init`.

Los metadatos de la AMI optimizada para Amazon ECS, incluido el ID de la AMI, de cada variante se pueden recuperar mediante programación. Para obtener más información, consulte the section called "Recuperación de metadatos de las AMI de Windows optimizadas para Amazon ECS".

En las siguientes pestañas, se muestra una lista de versiones de AMI de Windows optimizadas para Amazon ECS. Para obtener más información sobre cómo hacer referencia al parámetro de Parameter Store de Systems Manager en una plantilla de AWS CloudFormation, consulte Utilización de la AMI optimizada para Amazon ECS más reciente recomendada en una plantilla de AWS CloudFormation.

> **Important**
>
> Para asegurarse de que los clientes disponen de las actualizaciones de seguridad más recientes de forma predeterminada, Amazon ECS mantiene al menos las últimas tres AMI de Windows optimizada para Amazon ECS. Después de lanzar nuevas AMI de Windows optimizadas para Amazon ECS, Amazon ECS convierte en privadas las AMI de Windows optimizadas para Amazon ECS más antiguas. Para informarnos que necesita obtener acceso a una AMI privada, envíe un ticket al equipo de Cloud Support.
> Windows Server 2016 no es compatible con la última versión de Docker, por ejemplo, la 25.x.x. Por lo tanto, las AMI completas de Windows Server 2016 no recibirán parches de seguridad o de errores en el entorno en tiempo de ejecución de Docker. Le recomendamos que cambie a una de las siguientes plataformas de Windows:
>
> - Windows Server 2022 Full
> - Windows Server 2022 Core
> - Windows Server 2019 Full

- Windows Server 2019 Core

Windows Server 2025 Full AMI versions

En la tabla siguiente, se enumeran las versiones anteriores y la actual de la AMI de Windows Server 2025 Full optimizada para Amazon ECS y sus respectivas versiones del agente de contenedor de Amazon ECS y Docker.

AMI de Windows Server 2025 Full optimizada para Amazon ECS	Versión del agente de contenedor de Amazon ECS	Versión de Docker	Visibilidad
Windows_Server-2025-English-Full-ECS_Optimized-2025.06.13	`1.94.0`	`25.0.6 (Docker CE)`	Público

Utilice el siguiente comando de la AWS CLI para recuperar la AMI de Windows Server 2025 Full optimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-English-Full-ECS_Optimized
```

Windows Server 2025 Core AMI versions

En la tabla siguiente, se enumeran las versiones anteriores y la actual de la AMI de Windows Server 2025 Core optimizada para Amazon ECS y sus respectivas versiones del agente de contenedor de Amazon ECS y Docker.

AMI de Windows Server 2025 Core optimizada para Amazon ECS	Versión del agente de contenedor de Amazon ECS	Versión de Docker	Visibilidad
Windows_Server-2025-English-Core-ECS_Optimized-2025.06.13	`1.94.0`	`25.0.6 (Docker CE)`	Público

Utilice el siguiente comando de la AWS CLI para recuperar la AMI de Windows Server 2025 Core optimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-English-Core-ECS_Optimized
```

Windows Server 2022 Full AMI versions

En la tabla siguiente, se enumeran las versiones anteriores y la actual de la AMI de Windows Server 2022 Full optimizada para Amazon ECS y sus respectivas versiones del agente de contenedor de Amazon ECS y Docker.

AMI de Windows Server 2022 Full optimizada para Amazon ECS	Versión del agente de contenedor de Amazon ECS	Versión de Docker	Visibilidad
Windows_Server-2022-English-Full-ECS_Optimized-2025.06.13	`1.94.0`	`25.0.6 (Docker CE)`	Público
Windows_Server-2022-English-Full-ECS_Optimized-2025.05.17	`1.93.1`	`25.0.6 (Docker CE)`	Público

AMI de Windows Server 2022 Full optimizada para Amazon ECS	Versión del agente de contenedor de Amazon ECS	Versión de Docker	Visibilidad
Windows_Server-2022-English-Full-ECS_Optimized-2025.4.14	`1.91.2`	`25.0.6 (Docker CE)`	Público
Windows_Server-2022-English-Full-ECS_Optimized-2025.3.17	`1.91.0`	`25.0.6 (Docker CE)`	Público
Windows_Server-2022-English-Full-ECS_Optimized-2025.2.14	`1.90.0`	`25.0.6 (Docker CE)`	Público

Utilice el siguiente comando de la AWS CLI para recuperar la AMI de Windows Server 2022 Full optimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Full-ECS_Optimized
```

Windows Server 2022 Core AMI versions

En la tabla siguiente, se enumeran las versiones anteriores y la actual de la AMI de Windows Server 2022 Core optimizada para Amazon ECS y sus respectivas versiones del agente de contenedor de Amazon ECS y Docker.

AMI de Windows Server 2022 Core optimizada para Amazon ECS	Versión del agente de contenedor de Amazon ECS	Versión de Docker	Visibilidad
Windows_Server-2022-English-Core-ECS_Optimized-2025.06.13	`1.94.0`	`25.0.6 (Docker CE)`	Público
Windows_Server-2022-English-Core-ECS_Optimized-2025.05.17	`1.93.1`	`25.0.6 (Docker CE)`	Público
Windows_Server-2022-English-Core-ECS_Optimized-2025.4.14	`1.91.2`	`25.0.6 (Docker CE)`	Público
Windows_Server-2022-English-Core-ECS_Optimized-2025.3.17	`1.91.0`	`25.0.6 (Docker CE)`	Público
Windows_Server-2022-English-Core-ECS_Optimized-2025.2.14	`1.90.0`	`25.0.6 (Docker CE)`	Público

Utilice el siguiente comando de la AWS CLI para recuperar la AMI de Windows Server 2022 Full optimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Core-ECS_Optimized
```

Windows Server 2019 Full AMI versions

En la tabla siguiente, se enumeran las versiones anteriores y la actual de la AMI de Windows Server 2019 Full optimizada para Amazon ECS y sus respectivas versiones del agente de contenedor de Amazon ECS y Docker.

AMI de Windows Server 2019 Full optimizada para Amazon ECS	Versión del agente de contenedor de Amazon ECS	Versión de Docker	Visibilidad
Windows_Server-2019-English-Full-ECS_Optimized-2025.06.13	`1.94.0`	`25.0.6 (Docker CE)`	Público
Windows_Server-2019-English-Full-ECS_Optimized-2025.05.17	`1.93.1`	`25.0.6 (Docker CE)`	Público
Windows_Server-2019-English-Full-ECS_Optimized-2025.4.14	`1.91.2`	`25.0.6 (Docker CE)`	Público
Windows_Server-2019-English-Full-ECS_Optimized-2025.3.17	`1.91.0`	`25.0.6 (Docker CE)`	Público
Windows_Server-2019-English-Full-ECS_Optimized-2025.2.14	`1.90.0`	`25.0.6 (Docker CE)`	Público

Utilice el siguiente comando de la AWS CLI para recuperar la AMI de Windows Server 2019 Full optimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized
```

Windows Server 2019 Core AMI versions

En la tabla siguiente, se enumeran las versiones anteriores y la actual de la AMI de Windows Server 2019 Core optimizada para Amazon ECS y sus respectivas versiones del agente de contenedor de Amazon ECS y Docker.

AMI de Windows Server 2019 Core optimizada para Amazon ECS	Versión del agente de contenedor de Amazon ECS	Versión de Docker	Visibilidad
Windows_Server-2019-English-Core-ECS_Optimized-2025.06.13	`1.94.0`	`25.0.6 (Docker CE)`	Público
Windows_Server-2019-English-Core-ECS_Optimized-2025.05.17	`1.93.1`	`25.0.6 (Docker CE)`	Público
Windows_Server-2019-English-Core-ECS_Optimized-2025.4.14	`1.91.2`	`25.0.6 (Docker CE)`	Público
Windows_Server-2019-English-Core-ECS_Optimized-2025.3.17	`1.91.0`	`25.0.6 (Docker CE)`	Público
Windows_Server-2019-English-Core-ECS_Optimized-2025.2.14	`1.90.0`	`25.0.6 (Docker CE)`	Público

Utilice el siguiente comando de la AWS CLI para recuperar la AMI de Windows Server 2019 Full optimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Core-ECS_Optimized
```

Windows Server 2016 Full AMI versions

> **Important**
>
> Windows Server 2016 no es compatible con la última versión de Docker, por ejemplo, la 25.x.x. Por lo tanto, las AMI completas de Windows Server 2016 no recibirán parches de seguridad o de errores en el entorno en tiempo de ejecución de Docker. Le recomendamos que cambie a una de las siguientes plataformas de Windows:
>
> - Windows Server 2022 Full
> - Windows Server 2022 Core
> - Windows Server 2019 Full
> - Windows Server 2019 Core

En la tabla siguiente, se enumeran las versiones anteriores y la actual de la AMI de Windows Server 2016 Full optimizada para Amazon ECS y sus respectivas versiones del agente de contenedor de Amazon ECS y Docker.

AMI de Windows Server 2016 Full optimizada para Amazon ECS	Versión del agente de contenedor de Amazon ECS	Versión de Docker	Visibilidad
Windows_Server-2016-English-Full-ECS_Optimized-2025.06.13	`1.94.0`	`20.10.23 (Docker CE)`	Público
Windows_Server-2016-English-Full-ECS	`1.93.1`	`20.10.23 (Docker CE)`	Público

AMI de Windows Server 2016 Full optimizada para Amazon ECS	Versión del agente de contenedor de Amazon ECS	Versión de Docker	Visibilidad
_Optimized-2025.05.17			
Windows_Server-2016-English-Full-ECS_Optimized-2025.4.14	`1.91.2`	`20.10.23 (Docker CE)`	Público
Windows_Server-2016-English-Full-ECS_Optimized-2025.3.17	`1.91.0`	`20.10.23 (Docker CE)`	Público
Windows_Server-2016-English-Full-ECS_Optimized-2025.2.14	`1.90.0`	`20.10.23 (Docker CE)`	Público

Utilice la siguiente AWS CLI para la AMI de Windows Server 2016 Full optimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2016-English-Full-ECS_Optimized
```

Creación una AMI de Windows optimizada para Amazon ECS propia

Utilice el Generador de imágenes de EC2 para crear su propia AMI de Windows optimizada para Amazon ECS personalizada. Así, se simplifica la utilización de una AMI de Windows con su propia licencia en Amazon ECS. Amazon ECS proporciona un componente administrado de Image Builder que proporciona la configuración del sistema necesaria para ejecutar las instancias de Windows en las que se van a alojar los contenedores. Cada componente administrado por Amazon ECS incluye un agente de contenedor y una versión de Docker específicos. Puede personalizar la imagen para

que utilice el componente administrado por Amazon ECS más reciente o, si se necesita un agente de contenedor o una versión de Docker anterior, puede especificar un componente diferente.

Para obtener una explicación completa sobre el uso de EC2 Image Builder, consulte Introducción a EC2 Image Builder en la Guía del usuario de EC2 Image Builder.

Al crear una AMI de Windows optimizada para Amazon ECS propia mediante EC2 Image Builder, se crea una receta de imagen. La receta de imagen debe cumplir los siguientes requisitos:

- La imagen de origen debe basarse en Windows Server 2019 Core, Windows Server 2019 Full, Windows Server 2022 Core o Windows Server 2022 Full. No se admiten otros sistemas operativos de Windows y es posible que no sean compatibles con el componente.
- Cuando se especifica la opción Crear componente, se requiere el componente `ecs-optimized-ami-windows`. Se recomienda el componente `update-windows`, lo que garantiza que la imagen contenga las actualizaciones de seguridad más recientes.

  Para especificar otra versión de componente diferente, expanda el menú Opciones de control de versiones y especifique la versión de componente que desea utilizar. Para obtener más información, consulte Enumeración de versiones del componente `ecs-optimized-ami-windows`.

Enumeración de versiones del componente **`ecs-optimized-ami-windows`**

Al crear una receta de EC2 Image Builder y especificar el componente `ecs-optimized-ami-windows`, puede utilizar la opción predeterminada o especificar una versión específica del componente. Para determinar qué versiones del componente están disponibles, junto con las versiones del agente de contenedor de Amazon ECS y de Docker contenidas en el componente, puede utilizar laAWS Management Console.

Para enumerar las versiones disponibles del componente **`ecs-optimized-ami-windows`**

1. Abra la consola de EC2 Image Builder enhttps://console.aws.amazon.com/imagebuilder/.
2. En la barra de navegación, seleccione la región en la que está creando la imagen.
3. En el panel de navegación, elija Components (Componentes) en el menú Saved configurations (Configuraciones guardadas).
4. En la página Components (Componentes), escriba `ecs-optimized-ami-windows` en la barra de búsqueda, despliegue el menú de calificación y seleccione Quick start (Amazon-managed) (Inicio rápido [administrado por Amazon]).

5. Utilice la columna Description (Descripción) para determinar la versión del componente junto con la del agente contenedor de Amazon ECS y la de Docker que requiere su imagen.

Administración de instancias de contenedor de Windows de Amazon ECS

Cuando utiliza instancias de EC2 para las cargas de trabajo de Amazon ECS, es responsable del mantenimiento de instancias.

Las actualizaciones del agente no se aplican a instancias de contenedor de Windows. Le recomendamos que lance nuevas instancias de contenedor para actualizar la versión del agente en sus clústeres Windows.

Procedimientos de administración

- Lanzamiento de una instancia de contenedor de Windows de Amazon ECS
- Arranque de instancias de contenedor de Windows de Amazon ECS para la transferencia de datos
- Uso de un proxy HTTP para instancias de contenedor de Windows de Amazon ECS
- Configuración de instancias de contenedor de Windows de Amazon ECS para recibir avisos de instancias de spot

Lanzamiento de una instancia de contenedor de Windows de Amazon ECS

Las instancias de contenedor de Amazon ECS se crean mediante la consola de Amazon EC2. Antes de comenzar, asegúrese de que ha realizado los pasos que se detallan en Configuración para utilizar Amazon ECS.

Para obtener más información acerca del asistente de inicialización, consulte Lance una instancia con el nuevo asistente de inicialización de instancias en la Guía del usuario de Amazon EC2.

Puede utilizar el nuevo asistente de Amazon EC2 para lanzar una instancia. Puede utilizar la siguiente lista para los parámetros y dejar los parámetros no listados como predeterminados. Las siguientes instrucciones lo guiarán a través de cada grupo de parámetros.

Procedimiento

Antes de comenzar, complete los pasos de Configuración para utilizar Amazon ECS.

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, se muestra la región de AWS actual (por ejemplo, Este de EE. UU. [Ohio]). Seleccione una región en la que se va a iniciar la instancia. Esta elección es importante porque algunos recursos de Amazon EC2 pueden compartirse entre varias regiones, mientras que otros no.
3. En el panel de la consola de Amazon EC2, elija Iniciar instancia.

Nombre y etiquetas

El nombre de la instancia es una etiqueta, donde la clave es Name (Nombre) y el valor es el nombre que especifique. Puede etiquetar la instancia, los volúmenes y los gráficos elásticos. Para las instancias de spot, solo puede etiquetar la solicitud de instancia de spot.

Especificar un nombre de instancia y etiquetas adicionales es opcional.

- En Name (Nombre), ingrese un nombre descriptivo para la instancia. Si no especifica un nombre, la instancia se puede identificar mediante su ID, que se genera automáticamente al iniciar la instancia.
- Para agregar otras etiquetas, elija Add additional tag (Agregar etiqueta adicional). Elija Add tag (Agregar etiqueta) y, a continuación, ingrese una clave y un valor, y seleccione el tipo de recurso que desea etiquetar. Elija Add tag (Agregar etiqueta) para cada etiqueta adicional.

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

Una Imagen de máquina de Amazon (AMI) proporciona la información necesaria para crear una instancia. Por ejemplo, una AMI puede contener el software necesario para funcionar como servidor web, como Apache, y su sitio web.

Para obtener las AMI optimizadas para Amazon ECS más recientes y sus valores, consulte Versiones de AMI de Windows optimizadas para Amazon ECS.

Utilice la barra de búsqueda para buscar una AMI optimizada para Amazon ECS adecuada publicada por AWS.

1. En función de sus requisitos, ingrese una de las AMI siguientes en la barra de búsqueda y pulse Enter (Intro).
   - Windows_Server-2022-English-Full-ECS_Optimized
   - Windows_Server-2022-English-Core-ECS_Optimized
   - Windows_Server-2019-English-Full-ECS_Optimized

- Windows_Server-2019-English-Core-ECS_Optimized
- Windows_Server-2016-English-Full-ECS_Optimized

2. En la página Choose an Amazon Machine Image (AMI) (Elija una imagen de máquina de Amazon [AMI]), seleccione la categoría Community AMIs (AMI de la comunidad).
3. En la lista que aparece, seleccione una AMI verificada por Microsoft con la fecha de publicación más reciente y haga clic en Select (Seleccionar).

Tipo de instancia

El tipo de instancia define la configuración de hardware y el tamaño de la instancia. Los tipos de instancia más grandes tienen una CPU y memoria superiores. Para obtener más información, consulte Tipos de instancia.

- En Instance Type (Tipo de instancia), seleccione el tipo de instancia de la instancia.

  El tipo de instancia que seleccione determina los recursos disponibles para ejecutar sus tareas.

Par de claves (inicio de sesión)

En Key pair name (Nombre de par de claves) seleccione un par de claves existente o seleccione Create new key pair (Crear nuevo par de claves) para crear uno nuevo.

> **Important**
>
> Si elige la opción Proceed without key pair (Not recommended) (Continuar sin un par de claves [No recomendado]), no podrá conectarse a la instancia a menos que elija una AMI que esté configurada para ofrecer a los usuarios otra forma de iniciar sesión.

Configuración de red

Establezca la configuración de red, según sea necesario.

- Networking platform (Plataforma de redes): elija Virtual Private Cloud (VPC) (Nube privada virtual [VPC]) y, a continuación, especifique la subred en la sección Network interfaces (Interfaces de red).
- VPC: seleccione una VPC existente en la que desea crear el grupo de seguridad.

- Subnet (Subred): puede lanzar una instancia en una subred asociada con una zona de disponibilidad, zona local, zona Wavelength u Outpost.

  Para iniciar la instancia en una zona de disponibilidad, seleccione la subred en la que desea iniciar la instancia. Para crear una subred, elija Crear nueva subred para ir a la consola de Amazon VPC. Cuando haya terminado, vuelva al asistente de lanzamiento de instancias y elija el ícono Refresh (Actualizar) para cargar la subred en la lista.

  Para iniciar la instancia en una zona local, seleccione una subred que haya creado en la zona local.

  Para iniciar una instancia en un Outpost, seleccione una subred en una VPC que haya asociado a un Outpost.
- Auto-assign Public IP (Asignar automáticamente IP pública): si desea que se pueda acceder a la instancia desde Internet, compruebe que el campo Auto-assign Public IP (Asignar automáticamente IP pública) esté configurado como Enable (Habilitar). De lo contrario, configure este campo como Disable (Deshabilitar).

  > **Note**
  >
  > Las instancias de contenedor deben obtener acceso para comunicarse con el punto de conexión del servicio de Amazon ECS. Esto puede ser a través de un punto de conexión de VPC de la interfaz o a través de las instancias de contenedor con direcciones IP públicas.
  > Para obtener más información acerca de los puntos de conexión de VPC, consulte Puntos de enlace de la VPC de interfaz de Amazon ECS (AWS PrivateLink).
  > Si no tiene configurado un punto de conexión de VPC de la interfaz y las instancias de contenedor no tienen direcciones IP públicas, deberán utilizar traducción de direcciones de red (NAT) para proporcionar este acceso. Para obtener más información, consulte Puertas de enlace NAT en la Guía del usuario de Amazon VPC y Uso de un proxy HTTP para instancias de contenedor de Linux de Amazon ECS en esta guía.
- Firewall (security groups) Firewall (grupos de seguridad): utilice un grupo de seguridad para definir reglas de firewall para la instancia de contenedor. Estas reglas especifican qué tráfico procedente de la red se entregará en la instancia de contenedor. El resto del tráfico se ignora.
  - Para seleccionar un grupo de seguridad existente, elija Select an existing security group (Seleccionar un grupo de seguridad existente) y seleccione el grupo de seguridad que creó en Configuración para utilizar Amazon ECS.

Configurar almacenamiento

La AMI seleccionada incluye uno o más volúmenes de almacenamiento, incluido el volumen de dispositivo raíz. Se pueden especificar volúmenes adicionales para adjuntar a la instancia.

Se puede utilizar la vista Simple (Simple).

- Storage type (Tipo de almacenamiento): configure el almacenamiento de la instancia de contenedor.

  Si utiliza la AMI de Amazon Linux optimizada para Amazon ECS, la instancia tiene configurados dos volúmenes. El volumen raíz lo utiliza el sistema operativo y el segundo volumen de Amazon EBS (asociado a `/dev/xvdcz`) lo utiliza Docker.

  Si lo desea, puede aumentar o reducir el tamaño de volumen para su instancia de acuerdo con las necesidades de su aplicación.

Detalles avanzados

En Detalles avanzados, expanda la sección para ver los campos y especifique cualquier parámetro adicional para la instancia.

- Purchasing option (Opción de compra): elija Request Spot instances (Solicitar instancias de spot) para solicitar una instancia de spot. También debe establecer el resto de los campos relacionados con las instancias de Spot. Para obtener más información, consulte Spot Instance Requests (Solicitudes de instancias de Spot).

  > **Note**
  >
  > Si utiliza instancias de Spot y ve un mensaje que indica `Not available`, es posible que deba elegir un tipo de instancia diferente.

  .

- En IAM instance profile (Perfil de instancia de IAM), seleccione el rol de IAM de la instancia de contenedor. Suele llamarse `ecsInstanceRole`.

> **Important**
>
> Si no lanza la instancia de contenedor con los permisos de IAM correspondientes, el agente de Amazon ECS no puede conectarse al clúster. Para obtener más información, consulte Rol de IAM de instancia de contenedor de Amazon ECS.

- (Opcional) User data (Datos de usuario): configure la instancia de contenedor de Amazon ECS con los datos de usuario, por ejemplo, las variables de entorno del agente de Configuración del agente de contenedor de Amazon ECS. Los scripts de datos de usuario de Amazon EC2 se ejecutan solo una vez, cuando la instancia se lanza por primera vez. A continuación, se muestran ejemplos comunes del uso de los datos del usuario:
  - De forma predeterminada, su instancia de contenedor se abre en su clúster predeterminado. Para abrirlo en un clúster no predeterminado, seleccione la lista Advanced Details.
    A continuación, pegue el siguiente script en el campo User data, reemplazando *`your_cluster_name`* con el nombre de su clúster.

    El rol `EnableTaskIAMRole` activa la característica de roles de IAM de tareas para las tareas.

    Además, las siguientes opciones están disponibles cuando se utiliza el modo de red `awsvpc`.
    - `EnableTaskENI`: este indicador activa las redes de tareas y se requiere cuando se utiliza el modo de red `awsvpc`.
    - `AwsvpcBlockIMDS`: este indicador opcional bloquea el acceso a IMDS para los contenedores de tareas que se ejecutan en el modo de red `awsvpc`.
    - `AwsvpcAdditionalLocalRoutes`: este indicador opcional le permite tener rutas adicionales en el espacio de nombres de la tarea.

      Sustituya `ip-address` por la dirección IP para las rutas adicionales, por ejemplo, 172.31.42.23/32.

```
<powershell>
Import-Module ECSTools
Initialize-ECSAgent -Cluster your_cluster_name -EnableTaskIAMRole -EnableTaskENI -
AwsvpcBlockIMDS -AwsvpcAdditionalLocalRoutes
'["ip-address"]'
</powershell>
```

Arranque de instancias de contenedor de Windows de Amazon ECS para la transferencia de datos

Cuando se lanza una instancia de Amazon EC2, puede transferir los datos de usuario a la instancia de EC2. Los datos se pueden utilizar para llevar a cabo tareas de configuración automatizadas comunes e incluso ejecutar scripts cuando la instancia arranca. En Amazon ECS, los casos de uso más comunes para los datos de usuario consisten en transferir la información de configuración al daemon de Docker y al agente de contenedor de Amazon ECS.

Puede transferir varios tipos de datos de usuario a Amazon EC2, incluidos cloud boothooks, scripts de shell y directivas `cloud-init`. Para obtener más información acerca de estos u otros tipos de formato, consulte la documentación de Cloud-Init.

Puede transferir estos datos de usuario cuando utilice el asistente de lanzamiento de Amazon EC2. Para obtener más información, consulte Lanzamiento de una instancia de contenedor de Linux de Amazon ECS.

Datos de usuario de Windows predeterminados

Este script de datos de usuario de ejemplo muestra los datos de usuario predeterminados que reciben las instancias de contenedor de Windows si se utiliza la consola. El script a continuación hace lo siguiente:

- Establece el nombre del clúster con el nombre que ha ingresado.
- Establece los roles de IAM para las tareas.
- Establece `json-file` y `awslogs` como los controladores de registro disponibles.

Además, las siguientes opciones están disponibles cuando se utiliza el modo de red `awsvpc`.

- `EnableTaskENI`: este indicador activa las redes de tareas y se requiere cuando se utiliza el modo de red `awsvpc`.
- `AwsvpcBlockIMDS`: este indicador opcional bloquea el acceso a IMDS para los contenedores de tareas que se ejecutan en el modo de red `awsvpc`.
- `AwsvpcAdditionalLocalRoutes`: este indicador opcional le permite disponer de rutas adicionales.

  Sustituya `ip-address` por la dirección IP para las rutas adicionales, por ejemplo, 172.31.42.23/32.

Puede utilizar este script para sus propias instancias de contenedor (siempre que se lancen desde la AMI de Windows Server optimizada para Amazon ECS).

Sustituya la línea `-Cluster cluster-name` para especificar el nombre de su propio clúster.

```
<powershell>
Initialize-ECSAgent -Cluster cluster-name -EnableTaskIAMRole -LoggingDrivers '["json-
file","awslogs"]' -EnableTaskENI -AwsvpcBlockIMDS -AwsvpcAdditionalLocalRoutes
'["ip-address"]'
</powershell>
```

Para las tareas de Windows configuradas para utilizar el controlador de registros `awslogs`, debe también establecer la variable de entorno `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE` en la instancia del contenedor. Utilice la siguiente sintaxis.

Sustituya la línea `-Cluster cluster-name` para especificar el nombre de su propio clúster.

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster cluster-name -EnableTaskIAMRole -LoggingDrivers '["json-
file","awslogs"]'
</powershell>
```

Datos de usuario de la instalación del agente de Windows

Este script de datos de usuario de ejemplo instala el agente de contenedor de Amazon ECS en una instancia lanzada mediante una AMI Windows_Server-2016-English-Full-Containers. Se ha adaptado a partir de las instrucciones de instalación del agente que figuran en la página README del repositorio de GitHub del agente de contenedor de Amazon ECS.

> **Note**
>
> Este script se comparte para fines ilustrativos. Resulta mucho más sencillo comenzar a utilizar los contenedores de Windows mediante la AMI de Windows Server optimizada para Amazon ECS. Para obtener más información, consulte Creación de un clúster de Amazon ECS para el tipo de lanzamiento de Fargate.

Para obtener información sobre cómo instalar el agente de Amazon ECS en Windows Server 2022 Full, consulte Issue 3753 en GitHub.

Puede utilizar este script para sus propias instancias de contenedor (siempre que se lancen con una versión de la AMI Windows_Server-2016-English-Full-Containers). Asegúrese de sustituir la línea *windows* para especificar su propio nombre de clúster (si no está utilizando un clúster denominado windows).

```
<powershell>
# Set up directories the agent uses
New-Item -Type directory -Path ${env:ProgramFiles}\Amazon\ECS -Force
New-Item -Type directory -Path ${env:ProgramData}\Amazon\ECS -Force
New-Item -Type directory -Path ${env:ProgramData}\Amazon\ECS\data -Force
# Set up configuration
$ecsExeDir = "${env:ProgramFiles}\Amazon\ECS"
[Environment]::SetEnvironmentVariable("ECS_CLUSTER", "windows", "Machine")
[Environment]::SetEnvironmentVariable("ECS_LOGFILE", "${env:ProgramData}\Amazon\ECS\log
\ecs-agent.log", "Machine")
[Environment]::SetEnvironmentVariable("ECS_DATADIR", "${env:ProgramData}\Amazon\ECS
\data", "Machine")
# Download the agent
$agentVersion = "latest"
$agentZipUri = "https://s3.amazonaws.com/amazon-ecs-agent/ecs-agent-windows-
$agentVersion.zip"
$zipFile = "${env:TEMP}\ecs-agent.zip"
Invoke-RestMethod -OutFile $zipFile -Uri $agentZipUri
# Put the executables in the executable directory.
Expand-Archive -Path $zipFile -DestinationPath $ecsExeDir -Force
Set-Location ${ecsExeDir}
# Set $EnableTaskIAMRoles to $true to enable task IAM roles
# Note that enabling IAM roles will make port 80 unavailable for tasks.
[bool]$EnableTaskIAMRoles = $false
if (${EnableTaskIAMRoles}) {
  $HostSetupScript = Invoke-WebRequest https://raw.githubusercontent.com/aws/amazon-
ecs-agent/master/misc/windows-deploy/hostsetup.ps1
  Invoke-Expression $($HostSetupScript.Content)
}
# Install the agent service
New-Service -Name "AmazonECS" `
        -BinaryPathName "$ecsExeDir\amazon-ecs-agent.exe -windows-service" `
        -DisplayName "Amazon ECS" `
        -Description "Amazon ECS service runs the Amazon ECS agent" `
        -DependsOn Docker `
```

```
        -StartupType Manual
sc.exe failure AmazonECS reset=300 actions=restart/5000/restart/30000/restart/60000
sc.exe failureflag AmazonECS 1
Start-Service AmazonECS
</powershell>
```

Uso de un proxy HTTP para instancias de contenedor de Windows de Amazon ECS

Puede configurar las instancias de contenedor de Amazon ECS para que utilicen un proxy HTTP tanto para el agente de contenedor de Amazon ECS como para el daemon de Docker. Esto resulta útil si las instancias de contenedor no tienen acceso de red externo a través de una gateway de Internet de Amazon VPC, una gateway NAT o una instancia.

Para configurar la instancia de contenedor de Windows de Amazon ECS de modo que utilice un proxy HTTP, establezca las siguientes variables en el momento del lanzamiento (con datos de usuario de Amazon EC2).

```
[Environment]::SetEnvironmentVariable("HTTP_PROXY",
"http://proxy.mydomain:port", "Machine")
```

Establezca `HTTP_PROXY` en el nombre de host (o la dirección IP) y en el número de puerto de un proxy HTTP que se utilizará para que el agente de Amazon ECS se conecte a Internet. Por ejemplo, las instancias de contenedor podrían no tener acceso de red externo a través de una gateway de Internet de Amazon VPC, una gateway NAT o una instancia.

```
[Environment]::SetEnvironmentVariable("NO_PROXY",
"169.254.169.254,169.254.170.2,\\.\pipe\docker_engine", "Machine")
```

Establezca `NO_PROXY` en `169.254.169.254,169.254.170.2,\\.\pipe\docker_engine` para filtrar los metadatos de la instancia EC2, los roles de IAM para tareas y el tráfico del daemon de Docker procedente del proxy.

Example Script de datos de usuario de proxy HTTP de Windows

El siguiente ejemplo de script de PowerShell de datos de usuario configura al agente de contenedor de Amazon ECS y al daemon de Docker para que utilicen el proxy HTTP que se usted especifique. También puede especificar un clúster en el que se registre la propia instancia de contenedor.

Para utilizar este script al lanzar una instancia de contenedor, siga los pasos especificados en the section called “Lanzamiento de una instancia de contenedor”. Simplemente copie y pegue el script

de PowerShell mostrado a continuación en el campo User data (Datos de usuario) (asegúrese de sustituir los valores de ejemplo en color rojo por su propia información de proxy y de clúster).

> **Note**
>
> Se requiere la opción `-EnableTaskIAMRole` para habilitar los roles de IAM para tareas. Para obtener más información, consulte Configuración adicional de las instancias de Amazon EC2 de Windows.

```
<powershell>
Import-Module ECSTools

$proxy = "http://proxy.mydomain:port"
[Environment]::SetEnvironmentVariable("HTTP_PROXY", $proxy, "Machine")
[Environment]::SetEnvironmentVariable("NO_PROXY", "169.254.169.254,169.254.170.2,\\.
\pipe\docker_engine", "Machine")

Restart-Service Docker
Initialize-ECSAgent -Cluster MyCluster -EnableTaskIAMRole
</powershell>
```

Configuración de instancias de contenedor de Windows de Amazon ECS para recibir avisos de instancias de spot

Amazon EC2 termina, detiene o hiberna la instancia de spot cuando el precio de spot supera el precio máximo de su solicitud o cuando ya no hay más capacidad. Amazon EC2 envía un aviso de interrupción de la instancia de spot, que otorga a la instancia una advertencia dos minutos antes de que se interrumpa. Si el vaciado de instancias de spot de Amazon ECS está habilitado en la instancia, ECS recibe el aviso de interrupción de la instancia de spot y coloca la instancia en el estado `DRAINING`.

> **Important**
>
> Amazon ECS monitorea los avisos de interrupción de instancias de spot que tienen las acciones de instancia `terminate` y `stop`. Si especificó el comportamiento de interrupción de la instancia `hibernate` al solicitar las instancias o la flota de spot, el vaciado de instancias de spot de Amazon ECS no es compatible con esas instancias.

Cuando se establece una instancia de contenedor en `DRAINING`, Amazon ECS evita que se programen nuevas tareas para su ubicación en la instancia de contenedor. Las tareas de servicio en la instancia de contenedor que se está vaciando que están en el estado `PENDING` se paran de inmediato. Si hay instancias de contenedor en el clúster que están disponibles, las tareas de servicio de sustitución se inician en ellas.

Puede activar el drenaje de instancias de spot al lanzar una instancia. Debe configurar el parámetro `ECS_ENABLE_SPOT_INSTANCE_DRAINING` antes de iniciar el agente de contenedor. Reemplace *`my-cluster`* por el nombre de su clúster.

```
[Environment]::SetEnvironmentVariable("ECS_ENABLE_SPOT_INSTANCE_DRAINING", "true",
 "Machine")

# Initialize the agent
Initialize-ECSAgent -Cluster my-cluster
```

Para obtener más información, consulte the section called “Lanzamiento de una instancia de contenedor”.

Clústeres de Amazon ECS para el tipo de lanzamiento externo

Amazon ECS Anywhere admite el registro de una instancia externa, por ejemplo, un servidor ubicado en las instalaciones o una máquina virtual (VM), en el clúster de Amazon ECS. Las instancias externas se han optimizado para que puedan ejecutar aplicaciones que generen tráfico o datos del proceso salientes. Si la aplicación requiere tráfico entrante, la falta de compatibilidad con Elastic Load Balancing hace que la ejecución de estas cargas de trabajo sea menos eficiente. Amazon ECS ha agregado un nuevo tipo de lanzamiento `EXTERNAL` que se puede utilizar para crear servicios o ejecutar tareas en las instancias externas.

Sistemas operativos y arquitecturas de sistemas compatibles

A continuación, se muestra la lista de sistemas operativos y arquitecturas de sistema compatibles.

- Amazon Linux 2
- Amazon Linux 2023
- CentOS Stream 9

- RHEL 7, RHEL 8, RHEL 9: ni los repositorios de paquetes abiertos de Docker ni RHEL admiten la instalación de Docker de forma nativa en RHEL. Debe asegurarse de que Docker esté instalado antes de ejecutar el script de instalación que se describe en este documento.
- Fedora 32, Fedora 33, Fedora 40
- openSUSE Tumbleweed
- Ubuntu 18, Ubuntu 20, Ubuntu 22, Ubuntu 24
- Debian 10

> **Important**
>
> El soporte a largo plazo de Debian 9 (soporte LTS) finalizó el 30 de junio de 2022 y ya no es admitido por Amazon ECS Anywhere.

- Debian 11
- Debian 12
- SUSE Enterprise Server 15
- Se admiten las arquitecturas de CPU `x86_64` y `ARM64`.
- Se admiten las siguientes versiones de los sistemas operativos de Windows:
  - Windows Server 2022
  - Windows Server 2019
  - Windows Server 2016
  - Windows Server 20H2

Consideraciones

Antes de comenzar a utilizar instancias externas, tenga en cuenta lo siguiente.

- Puede registrar una instancia externa en un clúster a la vez. Para obtener instrucciones sobre cómo registrar una instancia externa con un clúster diferente, consulte Anulación del registro de una instancia externa de Amazon ECS.
- Sus instancias externas requieren un rol de IAM que les permita comunicarse con las API de AWS. Para obtener más información, consulte Rol de IAM de Amazon ECS Anywhere.
- Las instancias externas no deben tener una cadena de credenciales de instancia preconfigurada definida localmente, ya que interferiría con el script de registro.

- Para enviar registros de contenedor a CloudWatch Logs, asegúrese de crear y especificar un rol de IAM de ejecución de tareas en la definición de tareas.
- Cuando una instancia externa se registra en un clúster, el atributo `ecs.capability.external` se asocia a la instancia. Este atributo identifica la instancia como una instancia externa. Puede agregar atributos personalizados a las instancias externas para utilizarlos como delimitación de ubicación de tareas. Para obtener más información, consulte Custom attributes (Atributos personalizados).
- Puede agregar etiquetas de recursos a la instancia externa. Para obtener más información, consulte Instancias de contenedor externas.
- ECS Exec se admite en instancias externas. Para obtener más información, consulte Supervisión de los contenedores de Amazon ECS con ECS Exec.
- A continuación, se incluyen consideraciones adicionales específicas de las redes con las instancias externas. Para obtener más información, consulte Red .
  - No se admite el equilibrio de carga del servicio.
  - No se admite la detección de servicios.
  - Las tareas que se ejecutan en instancias externas deben utilizar los modos de red `bridge`, `host`, o `none`. No se admite el modo de red `awsvpc`.
  - Existen dominios de servicio de Amazon ECS en cada región de AWS. Se debe permitir que estos dominios de servicio envíen tráfico a las instancias externas.
  - El SSM Agent que se instaló en la instancia externa mantiene las credenciales de IAM que se rotan cada 30 minutos mediante una huella digital de hardware. Si la instancia externa pierde la conexión con AWS, SSM Agent actualiza automáticamente las credenciales después de que se restablece la conexión. Para obtener más información, consulte Validación de servidores ubicados en las instalaciones y máquinas virtuales mediante una huella digital de hardware en la Guía del usuario de AWS Systems Manager.
- No se admite la API `UpdateContainerAgent`. Para obtener instrucciones sobre cómo actualizar SSM Agent o el agente de Amazon ECS en las instancias externas, consulte Actualización del agente de AWS Systems Manager y del agente de contenedor de Amazon ECS en una instancia externa.
- No se admiten los proveedores de capacidad de Amazon ECS. Para crear un servicio o ejecutar una tarea independiente en las instancias externas, utilice el tipo de lanzamiento `EXTERNAL`.
- No se admite SELinux.
- No se admite la utilización de volúmenes de Amazon EFS ni la especificación de un `EFSVolumeConfiguration`.

- No se admite la integración con App Mesh.
- Si utiliza la consola para crear una definición de tareas de instancia externa, debe crear la definición de tareas con el editor JSON de la consola.
- Cuando ejecuta ECS Anywhere en Windows, debe utilizar su propia licencia de Windows en la infraestructura en las instalaciones.
- Cuando utilice una AMI no optimizada para Amazon ECS, ejecute los siguientes comandos en la instancia de contenedor externa para configurar reglas que utilicen los roles de IAM en las tareas. Para obtener más información, consulte Configuración adicional de las instancias externas.

```
$ sysctl -w net.ipv4.conf.all.route_localnet=1
$ iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
$ iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Red

Las instancias externas de Amazon ECS se han optimizado para que puedan ejecutar aplicaciones que generen tráfico o datos del proceso salientes. Si la aplicación requiere tráfico entrante, como un servicio web, la falta de compatibilidad con Elastic Load Balancing hace que la ejecución de estas cargas de trabajo sea menos eficiente porque no se admite la ubicación de estas cargas de trabajo detrás de un balanceador de carga.

A continuación, se incluyen consideraciones adicionales específicas de las redes con las instancias externas.

- No se admite el equilibrio de carga del servicio.
- No se admite la detección de servicios.
- Las tareas de Linux que se ejecutan en instancias externas deben utilizar los modos de red `bridge`, `host`, o `none`. No se admite el modo de red `awsvpc`.

  Para obtener más información sobre cada modo de red, consulte Opciones de red de tareas de Amazon ECS para el tipo de lanzamiento de EC2.
- Las tareas de Windows que se ejecutan en instancias externas deben utilizar el modo de red `default`.
- Existen dominios de servicio de Amazon ECS en cada región y se les debe permitir enviar tráfico a las instancias externas.

- El SSM Agent que se instaló en la instancia externa mantiene las credenciales de IAM que se rotan cada 30 minutos mediante una huella digital de hardware. Si la instancia externa pierde la conexión con AWS, SSM Agent actualiza automáticamente las credenciales después de que se restablece la conexión. Para obtener más información, consulte Validación de servidores ubicados en las instalaciones y máquinas virtuales mediante una huella digital de hardware en la Guía del usuario de AWS Systems Manager.

Los siguientes dominios se utilizan para la comunicación entre el servicio de Amazon ECS y el agente de Amazon ECS instalado en la instancia externa. Asegúrese de que se permita el tráfico y de que la resolución DNS funcione. En cada punto de enlace, la *región* representa el identificador de región de una región de AWS compatible con Amazon ECS, como `us-east-2` para la región EE. UU. Este (Ohio). Deben permitirse los puntos de enlace de todas las regiones que utilice. Para los puntos de enlace `ecs-a` y `ecs-t`, debe incluir un asterisco (por ejemplo, `ecs-a-*`).

- `ecs-a-*.`*`region`*`.amazonaws.com`: este punto de enlace se utiliza cuando se administran tareas.
- `ecs-t-*.`*`region`*`.amazonaws.com`: este punto de enlace se utiliza para administrar las métricas de tareas y de contenedor.
- `ecs.`*`region`*`.amazonaws.com`: es el punto de enlace de servicio para Amazon ECS.
- `ssm.`*`region`*`.amazonaws.com` : se trata del punto de conexión para AWS Systems Manager.
- `ec2messages.`*`region`*`.amazonaws.com`: este es el punto de conexión AWS Systems Manager utiliza para comunicarse entre el agente de Systems Manager y el servicio de Systems Manager en la nube.
- `ssmmessages.`*`region`*`.amazonaws.com`: este punto de conexión necesario para crear y eliminar los canales de sesión con el servicio Session Manager en la nube.
- Si las tareas requieren comunicación con cualquier otro servicio de AWS, asegúrese de que esos puntos de enlace de servicio se permitan. Las aplicaciones de ejemplo incluyen la utilización de Amazon ECR para extraer imágenes de contenedor o de CloudWatch para CloudWatch Logs. Para obtener más información, consulte Puntos de enlace de servicio en la Referencia general de AWS.

Amazon FSx for Windows File Server con ECS Anywhere

Para utilizar Amazon FSx for Windows File Server con las instancias externas de Amazon ECS debe establecer una conexión entre el centro de datos en las instalaciones y la Nube de AWS. Para

obtener más información acerca de las opciones para conectar su red a VPC, consulte Opciones de conectividad de Amazon Virtual Private Cloud.

gMSA con ECS Anywhere

Se admiten los siguientes casos de uso para ECS Anywhere.

- El Active Directory se encuentra en la Nube de AWS: para esta configuración, cree una conexión de entre la red en las instalaciones y la Nube de AWS mediante una conexión de AWS Direct Connect. Para obtener información acerca de cómo crear la conexión, consulteOpciones de conectividad de Amazon Virtual Private Cloud. Cree un Active Directory en la Nube de AWS. Para obtener información acerca de cómo empezar a usar AWS Directory Service, consulte Configuración de AWS Directory Service en la Guía de administración de AWS Directory Service. A continuación, puede unir las instancias externas al dominio mediante la conexión de AWS Direct Connect. Para obtener información acerca de cómo trabajar con gMSA con Amazon ECS, consulte the section called “Obtenga información sobre cómo utilizar gMSA para contenedores de EC2 para Windows”.
- El Active Directory se encuentra en el centro de datos en las instalaciones. - Para esta configuración, debe unir las instancias externas a Active Directory en las instalaciones. A continuación, se utilizan las credenciales disponibles localmente cuando ejecuta las tareas de Amazon ECS.

Creación de un clúster de Amazon ECS para el tipo de lanzamiento externo

Cree un clúster para definir la infraestructura en la que se ejecutan sus tareas y servicios.

Antes de comenzar, asegúrese de haber seguido los pasos que se detallan en Configuración para utilizar Amazon ECS y de asignar el permiso de IAM adecuado. Para obtener más información, consulte the section called “Ejemplos de clústeres de Amazon ECS”. La consola de Amazon ECS ofrece una forma sencilla de crear los recursos que necesita un clúster de Amazon ECS mediante la creación de una pila de AWS CloudFormation.

Para simplificar al máximo el proceso de creación del clúster, la consola cuenta con selecciones predeterminadas para muchas de las opciones que describimos a continuación. También hay paneles de ayuda disponibles para la mayoría de las secciones de la consola, que proporcionan más contexto.

Puede modificar las siguientes opciones:

- Agregue un espacio de nombres al clúster.

  Un espacio de nombres permite que los servicios que cree en el clúster puedan conectarse a los demás servicios del espacio de nombres sin configuración adicional. Para obtener más información, consulte Interconexión de los servicios de Amazon ECS.

- Configurar el clúster para instancias externas
- Active Información de contenedores con capacidad de observabilidad mejorada, o Información de contenedores.

  Información de contenedores de CloudWatch recopila, agrega y resume métricas y registros de las aplicaciones y microservicios en contenedores. Container Insights también proporciona información de diagnóstico, como, por ejemplo, errores de reinicio de contenedores, que usa para aislar problemas y solucionarlos rápidamente. Para obtener más información, consulte the section called “Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada”.

  El 2 de diciembre de 2024, AWS publicó Información de contenedores con observabilidad mejorada para Amazon ECS. Esta versión admite la observabilidad mejorada de los clústeres de Amazon ECS que utilizan los tipos de lanzamiento de Amazon EC2 y Fargate. Después de configurar Información de contenedores con capacidad de observabilidad mejorada en Amazon ECS, Información de contenedores recopilará automáticamente telemetría de infraestructura detallada desde el nivel de clúster hasta el nivel de contenedor en el entorno, y mostrará los datos en paneles con diversas métricas y dimensiones. A continuación, podrá utilizar estos paneles listos para usar en la consola de Información de contenedores y comprender mejor el estado y el rendimiento de los contenedores, así como mitigar los problemas con mayor rapidez mediante la identificación de anomalías.

  Recomendamos que utilice Información de contenedores con capacidad de observabilidad mejorada en lugar de Información de contenedores, ya que proporciona una visibilidad detallada del entorno de contenedores y reduce el tiempo medio de resolución.

- Asigne una clave AWS KMS para el almacenamiento administrado. Para obtener información acerca de la creación de una clave, consulte Creación de una clave KMS en la Guía del usuario de AWS Key Management Service.
- Agregue etiquetas que le ayuden a identificar el clúster.

Para crear un nuevo clúster (consola de Amazon ECS)

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, seleccione la región a utilizar.
3. En el panel de navegación, seleccione Clusters (Clústeres).
4. En la página Clusters (Clústeres), elija Create Cluster (Crear clúster).
5. En Configuraciones del clúster, configure lo siguiente:
   - En Nombre del clúster, escriba un nombre único.

     El nombre puede contener hasta 255 letras (minúsculas y mayúsculas), números y guiones.
   - (Opcional) Para que el espacio de nombre utilizado en Service Connect sea diferente del nombre del clúster, en Espacio de nombre, escriba un nombre único.
6. (Opcional) Utilice Información de contenedores, amplíe Supervisión y, a continuación, seleccione una de las siguientes opciones:
   - Para usar Información de contenedores con observabilidad mejorada, como se recomienda, elija Información de contenedores con observabilidad mejorada.
   - Para usar Información de contenedores, seleccione Información de contenedores.
7. (Opcional) Cifre los datos en el almacenamiento administrado. En Cifrado, para Almacenamiento administrado, introduzca el ARN de la clave AWS KMS que desea utilizar para cifrar los datos del almacenamiento administrado.
8. (Opcional) Para ayudar a identificar el clúster, expanda Tags (Etiquetas) y, a continuación, configure sus etiquetas.

   [Añadir una etiqueta] Elija Add tag (Añadir etiqueta) y haga lo siguiente:
   - En Key (Clave), escriba el nombre de la clave.
   - En Valor, escriba el valor de la clave.
9. Seleccione Crear.

Pasos a seguir a continuación

Debe registrar las instancias en el clúster. Para obtener más información, consulte Registro de una instancia externa en un clúster de Amazon ECS.

Cree una definición de tarea para el tipo de lanzamiento externo. Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola

Ejecute sus aplicaciones como tareas independientes o como parte de un servicio. Para obtener más información, consulte los siguientes temas:

- Ejecución de una aplicación como tarea de Amazon ECS
- Creación de una implementación de actualización continua de Amazon ECS

Registro de una instancia externa en un clúster de Amazon ECS

Para cada instancia externa que registre en un clúster de Amazon ECS, debe tener instalados SSM Agent, el agente contenedor de Amazon ECS y Docker. Para registrar la instancia externa en un clúster de Amazon ECS, primero debe registrarse como instancia administrada AWS Systems Manager. El script de instalación se puede crear con unos pocos clics desde la consola de Amazon ECS. El script de instalación incluye una clave de activación de Systems Manager y comandos para instalar cada uno de los agentes requeridos y Docker. El script de instalación se debe ejecutar en el servidor ubicado en las instalaciones o en la máquina virtual para completar los pasos de instalación y registro.

> **Note**
>
> Antes de registrar la instancia externa Linux en el clúster, cree el archivo `/etc/ecs/ecs.config` en la instancia externa y agregue los parámetros de configuración del agente de contenedor que desee. No se puede hacer después de registrar la instancia externa en un clúster. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS.

AWS Management Console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, seleccione la región a utilizar.
3. En el panel de navegación, seleccione Clusters (Clústeres).
4. En la página Clusters (Clústeres), elija el clúster en el que desea registrar la instancia externa.

5. En la página de Cluster : ***name*** (Clúster; nombre), elija la pestaña Infrastructure (Infraestructura).
6. En la página Register external instances (Registrar instancias externas), complete los pasos siguientes.
   a. En Activation key duration (in days) (Duración de la clave de activación [en días]), ingrese el número de días durante los que la clave de activación permanece activa. Una vez transcurrido el número de días ingresado, la clave ya no funciona al registrar una instancia externa.
   b. En Number of instances (Número de instancias), ingrese el número de instancias externas que desea registrar en el clúster con la clave de activación.
   c. En Instance role (Rol de instancia), elija el rol de IAM que desea asociar a las instancias externas. Si aún no se ha creado un rol, elija Create new role (Crear nuevo rol) para que Amazon ECS cree un rol en su nombre. Para obtener más información acerca de los permisos de IAM que se requieren para las instancias externas, consulte Rol de IAM de Amazon ECS Anywhere.
   d. Copie el comando de registro. Este comando se debe ejecutar en cada instancia externa que desee registrar en el clúster.

      > **Important**
      > La parte Bash del script debe ejecutarse como raíz. Si el comando no se ejecuta como raíz, se genera un error.

   e. Seleccione Cerrar.

AWS CLI for Linux operating systems

1. Cree un par de activación de Systems Manager. Esto se utiliza para la activación de instancias administradas de Systems Manager. El resultado incluye un `ActivationId` y un `ActivationCode`. Los utilizará en un paso posterior. Asegúrese de especificar el rol de IAM de ECS Anywhere que creó. Para obtener más información, consulte Rol de IAM de Amazon ECS Anywhere.

   ```
   aws ssm create-activation --iam-role ecsAnywhereRole | tee ssm-activation.json
   ```

2. Descargue el script de instalación en el servidor ubicado en las instalaciones o en la máquina virtual (VM).

```
curl --proto "https" -o "/tmp/ecs-anywhere-install.sh" "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh"
```

3. (Opcional) En el servidor ubicado en las instalaciones o en la máquina virtual (VM), siga estos pasos para comprobar el script de instalación mediante el archivo SIGNATURE de script.

   a. Descargue e instale GnuPG. Para obtener más información sobre GNUpg, consulte el sitio web de GnuPG. Para sistemas Linux, instale gpg utilizando el administrador de paquetes de su versión de Linux.

   b. Recupere la clave pública PGP de Amazon ECS.

```
gpg --keyserver hkp://keys.gnupg.net:80 --recv BCE9D9A42D51784F
```

   c. Descargue la firma del script de instalación. La firma es una firma PGP separada en formato ASCII que se almacena en un archivo con la extensión `.asc`.

```
curl --proto "https" -o "/tmp/ecs-anywhere-install.sh.asc" "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh.asc"
```

   d. Verifique el archivo del script de instalación mediante la clave.

```
gpg --verify /tmp/ecs-anywhere-install.sh.asc /tmp/ecs-anywhere-install.sh
```

   El resultado esperado es el siguiente.

```
gpg: Signature made Tue 25 May 2021 07:16:29 PM UTC
gpg:                using RSA key 50DECCC4710E61AF
gpg: Good signature from "Amazon ECS <ecs-security@amazon.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the
 owner.
Primary key fingerprint: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
     Subkey fingerprint: D64B B6F9 0CF3 77E9 B5FB  346F 50DE CCC4 710E 61AF
```

4. Ejecute el script de instalación en el servidor ubicado en las instalaciones o en la máquina virtual (VM). Especifique el nombre del clúster, la región y el ID de activación de Systems Manager, y el código de activación del primer paso.

```
sudo bash /tmp/ecs-anywhere-install.sh \
    --region $REGION \
    --cluster $CLUSTER_NAME \
    --activation-id $ACTIVATION_ID \
    --activation-code $ACTIVATION_CODE
```

Para un servidor en las instalaciones o una máquina virtual (VM) que tenga el controlador NVIDIA instalado para las cargas de trabajo de la GPU, debe agregar el indicador `--enable-gpu` del script de instalación. Cuando se especifica este indicador, el script de instalación verifica que el controlador NVIDIA se está ejecutando y, a continuación, agrega las variables de configuración necesarias para ejecutar las tareas de Amazon ECS. Para obtener más información acerca de cómo ejecutar cargas de trabajo de GPU y especificar los requisitos de GPU en una definición de tarea, consulte Especificación de GPU en una definición de tareas de Amazon ECS.

```
sudo bash /tmp/ecs-anywhere-install.sh \
    --region $REGION \
    --cluster $CLUSTER_NAME \
    --activation-id $ACTIVATION_ID \
    --activation-code $ACTIVATION_CODE \
    --enable-gpu
```

Siga estos pasos para registrar una instancia externa existente en otro clúster.

Para registrar una instancia externa existente en un clúster diferente

1. Detenga el agente de contenedor de Amazon ECS.

```
sudo systemctl stop ecs.service
```

2. Edite el archivo `/etc/ecs/ecs.config` y, en la línea `ECS_CLUSTER`, asegúrese de que el nombre del clúster coincida con el nombre del clúster con el que se va a registrar la instancia externa.
3. Elimine los datos existentes del agente de Amazon ECS.

```
sudo rm /var/lib/ecs/data/agent.db
```

4. Inicie el agente de contenedor de Amazon ECS.

```
sudo systemctl start ecs.service
```

AWS CLI for Windows operating systems

1. Cree un par de activación de Systems Manager. Esto se utiliza para la activación de instancias administradas de Systems Manager. El resultado incluye un `ActivationId` y un `ActivationCode`. Los utilizará en un paso posterior. Asegúrese de especificar el rol de IAM de ECS Anywhere que creó. Para obtener más información, consulte Rol de IAM de Amazon ECS Anywhere.

```
aws ssm create-activation --iam-role ecsAnywhereRole | tee ssm-activation.json
```

2. Descargue el script de instalación en el servidor ubicado en las instalaciones o en la máquina virtual (VM).

```
Invoke-RestMethod -URI "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-
install.ps1" -OutFile “ecs-anywhere-install.ps1”
```

3. (Opcional) Amazon firma el script de Powershell y, por lo tanto, Windows realiza automáticamente la validación del certificado en el mismo. No es necesario realizar ninguna validación manual.

   Para verificar manualmente el certificado, haga clic con el botón derecho en el archivo, vaya a las propiedades y utilice la pestaña Firmas digitales para obtener más detalles.

   Esta opción solo está disponible cuando el host tiene el certificado en el almacén de certificados.

   La verificación debería devolver información similar a la siguiente:

```
# Verification (PowerShell)
Get-AuthenticodeSignature -FilePath .\ecs-anywhere-install.ps1

SignerCertificate                         Status      Path
-----------------                         ------      ----
```

```
EXAMPLECERTIFICATE                        Valid       ecs-anywhere-install.ps1

...

Subject              : CN="Amazon Web Services, Inc.",...

----
```

4. Ejecute el script de instalación en el servidor ubicado en las instalaciones o en la máquina virtual (VM). Especifique el nombre del clúster, la región y el ID de activación de Systems Manager, y el código de activación del primer paso.

```
.\ecs-anywhere-install.ps1 -Region $Region -Cluster $Cluster -
ActivationID $ActivationID -ActivationCode $ActivationCode
```

5. Verifique que el agente de contenedor de Amazon ECS se esté ejecutando.

```
Get-Service AmazonECS

Status   Name               DisplayName
------   ----               -----------
Running  AmazonECS          Amazon ECS
```

Siga estos pasos para registrar una instancia externa existente en otro clúster.

Para registrar una instancia externa existente en un clúster diferente

1. Detenga el agente de contenedor de Amazon ECS.

```
Stop-Service AmazonECS
```

2. Modifique el parámetro `ECS_CLUSTER` de modo que el nombre del clúster coincida con el nombre del clúster con el que se va a registrar la instancia externa.

```
[Environment]::SetEnvironmentVariable("ECS_CLUSTER", $ECSCluster,
 [System.EnvironmentVariableTarget]::Machine)
```

3. Elimine los datos existentes del agente de Amazon ECS.

```
Remove-Item -Recurse -Force $env:ProgramData\Amazon\ECS\data\*
```

4. Inicie el agente de contenedor de Amazon ECS.

```
Start-Service AmazonECS
```

La AWS CLI se puede utilizar para crear una activación de Systems Manager antes de ejecutar el script de instalación a fin de completar el proceso de registro de instancias externas.

Anulación del registro de una instancia externa de Amazon ECS

Le recomendamos que, una vez que termine de utilizar una instancia, anule el registro de la instancia tanto en Amazon ECS como en AWS Systems Manager. Una vez anulado el registro, la instancia externa ya no puede aceptar nuevas tareas.

Si tiene tareas en ejecución en la instancia de contenedor cuando se anula el registro, estas tareas siguen en ejecución hasta que se detengan por otros medios. Sin embargo, Amazon ECS deja de monitorearlas y de considerarlas. Si estas tareas de la instancia externa forman parte de un servicio de Amazon ECS, el programador de servicio inicia otra copia de esa tarea, en una instancia de contenedor distinta, de ser posible.

Después de anular el registro de la instancia, limpie los recursos de AWS de la instancia. A continuación, puede registrarla en un clúster nuevo.

Procedimiento

AWS Management Console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, elija la región en la que se encuentra registrada la instancia externa.
3. En el panel de navegación, elija Clusters (Clústeres) y seleccione el clúster que aloja la instancia externa.
4. En la página de Cluster : ***name*** (Clúster; nombre), elija la pestaña Infrastructure (Infraestructura).
5. En Container instances (Instancias de contenedor), seleccione el ID de instancia externa cuyo registro desea cancelar. Se lo redirigirá a la página de detalles de la instancia de contenedor.
6. En la página Container Instance : ***id***, seleccione Deregister.

7. Revise el mensaje de anulación del registro. Seleccione Deregister from AWS Systems Manager (Anular registro de Systems Manager) para anular el registro de la instancia externa como instancia administrada de Systems Manager. Elija Anular registro.

> **Note**
>
> Puede anular el registro de la instancia externa como instancia administrada de Systems Manager desde la consola de Systems Manager. Para obtener instrucciones, consulte Anulación del registro de nodos administrados en un entorno híbrido y multinube en la Guía del usuario de AWS Systems Manager.

8. Después de anular el registro de la instancia, limpie los recursos de AWS del servidor en las instalaciones o de la VM.

Sistema operativo	Pasos	

Linux	a. Detenga el agente contenedor de Amazon ECS y los servicios de SSM Agent en la instancia.
	`sudo systemctl stop ecs amazon-ssm-agent`
	b. Elimine los paquetes de Amazon ECS y Systems Manager.
	Para CentOS 7, CentOS 8 y RHEL 7
	`sudo yum remove -y amazon-ecs-init amazon-ssm-agent`	

Sistema operativo	Pasos	

	Para SUSE Enterprise Server 15
	`sudo zypper remove -y amazon-ecs-init amazon-ssm-agent`
	Para Debian y Ubuntu
	`sudo apt remove -y amazon-ecs-init amazon-ssm-agent`
	c. Elimine los directorios sobrantes.
	`sudo rm -rf /var/lib/ecs /etc/ecs /var/lib/amazon/ssm /var/log/ecs /var/log/amazon/ssm`	

Sistema operativo	Pasos	

Windows	a. Detenga el agente contenedor de Amazon ECS y los servicios de SSM Agent en la instancia.
	`Stop-Service AmazonECS`
	`Stop-Service AmazonSSMAgent`
	b. Eliminar el paquete de Amazon ECS.
	`.\ecs-anywhere-install.ps1 -Uninstall`	

AWS CLI

1. Necesita el ID de la instancia y el ARN de la instancia de contenedor para anular el registro de la instancia de contenedor. Si no tiene estos valores, ejecute los siguientes comandos

   Ejecute el siguiente comando para obtener el ID de la instancia.

   Utilice el ID de la instancia (`instanceID`) para obtener el ARN (`containerInstanceARN`) de la instancia de contenedor.

   ```
   instanceId=$(aws ssm describe-instance-information --region "{{ region }}" |
    jq ".InstanceInformationList[] |select(.IPAddress==\"{{ IPv4 Address }}\")
    | .InstanceId" | tr -d'"'
   ```

   Ejecute los siguientes comandos.

Utilice el `containerInstanceArn` como parámetro en el comando para anular el registro de la instancia (`deregister-container-instance`).

```
instances=$(aws ecs list-container-instances --cluster "{{ cluster }}" --region
 "{{ region }}" | jq -c '.containerInstanceArns')
containerInstanceArn=$(aws ecs describe-container-instances --cluster
 "{{ cluster }}" --region "{{ region }}" --container-instances $instances
 | jq ".containerInstances[] | select(.ec2InstanceId==\"{{ instanceId }}\")
 | .containerInstanceArn" | tr -d '"')
```

2. Ejecute el siguiente comando para vaciar la instancia.

```
aws ecs update-container-instances-state --cluster "{{ cluster }}" --region
 "{{ region }}" --container-instances "{{ containerInstanceArn }}" --status
 DRAINING
```

3. Cuando la instancia de contenedor termine de vaciarse, ejecute el siguiente comando para anular el registro de la instancia.

```
aws ecs deregister-container-instance --cluster "{{ cluster }}" --region
 "{{ region }}" --container-instance "{{ containerInstanceArn }}"
```

4. Ejecute el siguiente comando para eliminar las instancias de contenedor desde SSM.

```
aws ssm deregister-managed-instance --region "{{ region }}" --instance-id
 "{{ instanceId }}"
```

5. Después de anular el registro de la instancia, limpie los recursos de AWS del servidor en las instalaciones o de la VM.

Sistema operativo	Pasos	
Linux	a. Detenga el agente contenedor de Amazon ECS y los servicios de SSM Agent en la instancia.	

Sistema operativo	Pasos	

	`sudo systemctl stop ecs amazon-ssm-agent`
	b. Elimine los paquetes de Amazon ECS y Systems Manager.
	`sudo (yum/apt/zypper) remove amazon-ecs-init amazon-ssm-agent`
	c. Elimine los directorios sobrantes.
	`sudo rm -rf /var/lib/ecs /etc/ecs /var/lib/amazon/ssm /var/log/ecs /var/log/amazon/ssm`	

Sistema operativo	Pasos	

Windows	a. Detenga el agente contenedor de Amazon ECS y los servicios de SSM Agent en la instancia.
	`Stop-Service AmazonECS`
	`Stop-Service AmazonSSMAgent`
	b. Eliminar el paquete de Amazon ECS.
	`.\ecs-anywhere-install.ps1 -Uninstall`	

Actualización del agente de AWS Systems Manager y del agente de contenedor de Amazon ECS en una instancia externa

El servidor ubicado en las instalaciones o la VM deben ejecutar el agente de AWS Systems Manager (SSM Agent) y el agente de contenedor de Amazon ECS al ejecutar cargas de trabajo de Amazon ECS. AWSlanza nuevas versiones de estos agentes cuando se agregan o actualizan funciones. Si las instancias externas utilizan una versión anterior de cualquiera de los agentes, puede actualizarlas mediante estos procedimientos.

Actualización de SSM Agent en una instancia externa

AWS Systems Manager recomienda automatizar el proceso de actualización de SSM Agent en las instancias. Proporcionan varios métodos para automatizar las actualizaciones. Para obtener más información, consulte Automatización de actualizaciones para SSM Agent en la Guía del usuario de AWS Systems Manager.

Actualización del agente de Amazon ECS en una instancia externa

En las instancias externas, el agente de contenedor de Amazon ECS se actualiza mediante la actualización del paquete `ecs-init`. La actualización del agente de Amazon ECS no interrumpe las tareas o servicios en ejecución. Amazon ECS proporciona el paquete `ecs-init` y el archivo SIGNATURE en un bucket de Amazon S3 en cada región. A partir de la versión `1.52.1-1` de `ecs-init`, Amazon ECS proporciona paquetes `ecs-init` independientes para su utilización en función del sistema operativo y la arquitectura del sistema que utilice la instancia externa.

Utilice la siguiente tabla para determinar qué paquete `ecs-init` debe descargar en función del sistema operativo y la arquitectura del sistema que utiliza su instancia externa.

> **Note**
>
> Para determinar qué sistema operativo y arquitectura del sistema utiliza la instancia externa, utilice los siguientes comandos.
>
> ```
> cat /etc/os-release
> uname -m
> ```

Sistemas operativos (arquitectura)	Paquete ecs-init
CentOS 7 (x86_64) CentOS 8 (x86_64) CentOS Stream 9 (x86_64) SUSE Enterprise Server 15 (x86_64) RHEL 7 (x86_64) RHEL 8 (x86_64)	`amazon-ecs-init-latest.x86_64.rpm`
CentOS 7 (aarch64) CentOS 8 (aarch64) CentOS Stream 9 (aarch64)	`amazon-ecs-init-latest.aarch64.rpm`

Sistemas operativos (arquitectura)	Paquete ecs-init
RHEL 7 (aarch64)	
Debian 9 (x86_64) Debian 10 (x86_64) Debian 11 (x86_64) Debian 12 (x86_64) Ubuntu 18 (x86_64) Ubuntu 20 (x86_64) Ubuntu 22 (x86_64) Ubuntu 24 (x86_64)	`amazon-ecs-init-latest.amd64.deb`
Debian 9 (aarch64) Debian 10 (aarch64) Debian 11 (aarch64) Debian 12 (aarch64) Ubuntu 18 (aarch64) Ubuntu 20 (aarch64) Ubuntu 22 (aarch64) Ubuntu 24 (aarch64)	`amazon-ecs-init-latest.arm64.deb`

Siga estos pasos para actualizar el agente de Amazon ECS.

Para actualizar el agente de Amazon ECS

1. Confirme la versión del agente de Amazon ECS que está ejecutando.

```
curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
```

2. Descargue el paquete `ecs-init` correspondiente a su sistema operativo y arquitectura del sistema. Amazon ECS proporciona el archivo del paquete `ecs-init` en un bucket de Amazon S3 en cada región. Asegúrese de sustituir el identificador *<region>* del comando por el nombre de la región (por ejemplo, `us-west-2`) geográficamente más cercana.

   amazon-ecs-init-latest.x86_64.rpm

   ```
   curl -o amazon-ecs-init.rpm https://s3.<region>.amazonaws.com/amazon-ecs-
   agent-<region>/amazon-ecs-init-latest.x86_64.rpm
   ```

   amazon-ecs-init-latest.aarch64.rpm

   ```
   curl -o amazon-ecs-init.rpm https://s3.<region>.amazonaws.com/amazon-ecs-
   agent-<region>/amazon-ecs-init-latest.aarch64.rpm
   ```

   amazon-ecs-init-latest.amd64.deb

   ```
   curl -o amazon-ecs-init.deb https://s3.<region>.amazonaws.com/amazon-ecs-
   agent-<region>/amazon-ecs-init-latest.amd64.deb
   ```

   amazon-ecs-init-latest.arm64.deb

   ```
   curl -o amazon-ecs-init.deb https://s3.<region>.amazonaws.com/amazon-ecs-
   agent-<region>/amazon-ecs-init-latest.arm64.deb
   ```

3. (Opcional) Verifique la validez del archivo del paquete `ecs-init` mediante la firma PGP.

   a. Descargue e instale GnuPG. Para obtener más información sobre GNUpg, consulte el sitio web de GnuPG. Para sistemas Linux, instale gpg utilizando el administrador de paquetes de su versión de Linux.

   b. Recupere la clave pública PGP de Amazon ECS.

      ```
      gpg --keyserver hkp://keys.gnupg.net:80 --recv BCE9D9A42D51784F
      ```

   c. Descargue la firma del paquete `ecs-init`. La firma es una firma PGP separada en formato ASCII que se almacena en un archivo con la extensión `.asc`. Amazon ECS proporciona el

archivo SIGNATURE en un bucket de Amazon S3 en cada región. Asegúrese de sustituir el identificador `<region>` del comando por el nombre de la región (por ejemplo, `us-west-2`) geográficamente más cercana.

amazon-ecs-init-latest.x86_64.rpm

```
curl -o amazon-ecs-init.rpm.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.x86_64.rpm.asc
```

amazon-ecs-init-latest.aarch64.rpm

```
curl -o amazon-ecs-init.rpm.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.aarch64.rpm.asc
```

amazon-ecs-init-latest.amd64.deb

```
curl -o amazon-ecs-init.deb.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.amd64.deb.asc
```

amazon-ecs-init-latest.arm64.deb

```
curl -o amazon-ecs-init.deb.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.arm64.deb.asc
```

d. Verifique el archivo del paquete `ecs-init` mediante la clave.

Para los paquetes **`rpm`**

```
gpg --verify amazon-ecs-init.rpm.asc ./amazon-ecs-init.rpm
```

Para los paquetes **`deb`**

```
gpg --verify amazon-ecs-init.deb.asc ./amazon-ecs-init.deb
```

El resultado esperado es el siguiente.

```
gpg: Signature made Fri 14 May 2021 09:31:36 PM UTC
gpg:                using RSA key 50DECCC4710E61AF
gpg: Good signature from "Amazon ECS <ecs-security@amazon.com>" [unknown]
```

```
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
     Subkey fingerprint: D64B B6F9 0CF3 77E9 B5FB  346F 50DE CCC4 710E 61AF
```

4. Instale el paquete `ecs-init`.

   Para el paquete **`rpm`** en CentOS 7, CentOS 8 y RHEL 7

   ```
   sudo yum install -y ./amazon-ecs-init.rpm
   ```

   Para el paquete **`rpm`** en SUSE Enterprise Server 15

   ```
   sudo zypper install -y --allow-unsigned-rpm ./amazon-ecs-init.rpm
   ```

   Para el paquete **`deb`**

   ```
   sudo dpkg -i ./amazon-ecs-init.deb
   ```

5. Reinicie el servicio `ecs`.

   ```
   sudo systemctl restart ecs
   ```

6. Verifique que se haya actualizado la versión del agente de Amazon ECS.

   ```
   curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
   ```

Actualización de un clúster de Amazon ECS

Puede modificar las siguientes propiedaes de clúster:

- Establecer un proveedor de capacidad predeterminado

  Cada clúster puede tener uno o más proveedores de capacidad y, opcionalmente, una estrategia de proveedor de capacidad. La estrategia del proveedor de capacidad determina cómo se distribuyen las tareas entre los proveedores de capacidad del clúster. Cuando ejecuta una tarea individual o crea un servicio, utiliza la estrategia de proveedores de capacidad predeterminada del clúster o una estrategia de proveedores de capacidad que anule la estrategia predeterminada.
- Active Container Insights.

Información de contenedores de CloudWatch recopila, agrega y resume métricas y registros de las aplicaciones y microservicios en contenedores. Container Insights también proporciona información de diagnóstico, como, por ejemplo, errores de reinicio de contenedores, que usa para aislar problemas y solucionarlos rápidamente. Para obtener más información, consulte the section called "Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada".

- Agregue etiquetas que le ayuden a identificar los clúster.

Procedimiento

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. En la página Clusters (Clústeres), elija el clúster.
4. En la página Clúster: ***nombre***, elija Actualizar clúster.
5. Para configurar el proveedor de capacidad predeterminado, en Estrategia predeterminada del proveedor de capacidad, elija Agregar más.

   a. En Proveedor de capacidad, elija el proveedor de capacidad.

   b. (Opcional) En Base, introduzca el número mínimo de tareas que se ejecutan en el proveedor de capacidad.

      Solo puede establecer un valor base para un proveedor de capacidad.

   c. (Opcional) En peso, introduzca el porcentaje relativo del número total de tareas lanzadas que utiliza el proveedor de capacidad especificado.

   d. (Opcional) Repita los pasos para cualquier proveedor de capacidad adicional.
6. Para activar o desactivar Container Insights, expanda Supervisión y, a continuación, active Utilizar Container Insights.
7. Para ayudar a identificar el clúster, expanda Etiquetas y, a continuación, configure sus etiquetas.

   [Añadir una etiqueta] Elija Add tag (Añadir etiqueta) y haga lo siguiente:

   - En Key (Clave), escriba el nombre de la clave.
   - En Valor, escriba el valor de la clave.

   [Eliminar una etiqueta] Elija Eliminar a la derecha de la clave y el valor de la etiqueta.

8. Elija Actualizar.

Eliminación de un clúster de Amazon ECS

Si ya ha terminado de usar un clúster, puede eliminarlo. Después de eliminar el clúster, pasa al estado `INACTIVE`. Es posible que los clústeres con estado `INACTIVE` permanezcan detectables en la cuenta durante un período de tiempo. Sin embargo, este comportamiento está sujeto a cambios en el futuro, por lo que no debe contar con la permanencia de los clústeres `INACTIVE`.

Antes de eliminar un clúster, debe realizar las siguientes operaciones:

- Elimine todos los servicios del clúster. Para obtener más información, consulte the section called “Eliminación de un servicio”.
- Detenga todas las tareas en ejecución actualmente. Para obtener más información, consulte the section called “Detención de una tarea”.
- Anule el registro de todas las instancias de contenedor registradas del clúster. Para obtener más información, consulte the section called “Anulación del registro de una instancia de contenedor”.
- Elimine el espacio de nombres de . Para obtener más información, consulte Eliminación de espacios de nombres en la AWS Cloud MapGuía para desarrolladores de .

Procedimiento

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, seleccione la región a utilizar.
3. En el panel de navegación, seleccione Clusters (Clústeres).
4. En la página Clusters, seleccione el clúster que desea eliminar.
5. En la parte superior derecha de la página, elija Delete Cluster (Eliminar clúster).

   Se muestra un mensaje cuando no ha eliminado todo el recurso asociado al clúster.
6. En el cuadro de confirmación, ingrese delete ***`cluster name`*** (eliminar nombre de clúster).

Anulación del registro de una instancia de contenedor de Amazon ECS

> **Important**
>
> Este tema aplica solo a instancias de contenedor creadas en Amazon EC2. Para obtener más información acerca de la anulación de registros de instancias externas, consulte Anulación del registro de una instancia externa de Amazon ECS.

Cuando ya no necesite una instancia de contenedor respaldada por Amazon EC2, debe anular el registro en el clúster. Tras la cancelación del registro, la instancia de contenedor ya no puede aceptar nuevas tareas.

Si tiene tareas en ejecución en la instancia de contenedor cuando cancela el registro, estas tareas siguen en ejecución hasta que termine la instancia o hasta que las tareas se detengan por otros medios. Sin embargo, estas tareas son huérfanas, es decir, Amazon ECS ya no las monitorea ni las considera). Si una tarea huérfana de la instancia de contenedor forma parte de un servicio de Amazon ECS, el programador de servicio inicia otra copia de esa tarea, en una instancia de contenedor distinta, de ser posible. Se anula el registro de los contenedores de las tareas de servicio huérfanas que estén registradas en un grupo de destino del equilibrador de carga de aplicación. Comienzan el vaciado de conexión de acuerdo con la configuración en el balanceador de carga o grupo de destino. Si una tarea huérfana utiliza el modo de red `awsvpc`, se eliminan sus interfaces de red elásticas.

Si pretende utilizar la instancia de contenedor para algún otro fin después de la cancelación del registro, debe detener todas las tareas que se ejecutan en la instancia de contenedor antes de la cancelación de registro. Esto hace que las tareas sin propietario dejen de consumir recursos.

Al anular el registro de una instancia de contenedor, tenga en cuenta lo siguiente.

- Ya que cada instancia de contenedor tiene una información de estado única, no se debe cancelar el registro de un clúster y volver a registrarla en otro. Para reubicar los recursos de la instancia de contenedor, le recomendamos que termine las instancias de contenedor en un clúster y lance nuevas instancias de contenedor en el clúster nuevo. Para obtener más información, consulte Finalizar una instancia en la Guía del usuario de Amazon EC2 y Lanzamiento de una instancia de contenedor de Linux de Amazon ECS.

- Si la instancia de contenedor está administrada por un grupo de Auto Scaling o una pila de AWS CloudFormation, termine la instancia mediante la actualización del grupo de Auto Scaling o la pila de AWS CloudFormation. De lo contrario, el grupo de Auto Scaling o AWS CloudFormation crearán una nueva instancia después de que usted la termine.
- Si termina una instancia de contenedor en ejecución con un agente de contenedor de Amazon ECS conectado, el agente anula automáticamente el registro de la instancia en el clúster. No se cancela automáticamente el registro de las instancias de contenedor paradas o las instancias con agentes desconectados cuando se terminan.
- La anulación de registro de una instancia de contenedor elimina la instancia de un clúster, pero no termina la instancia de Amazon EC2. Si no va a utilizar más la instancia, asegúrese de terminarla a fin de detener la facturación. Para obtener más información, consulte Terminar una instancia en la Guía del usuario de Amazon EC2.

Procedimiento

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, elija la región en la que se encuentra registrada la instancia externa.
3. En el panel de navegación, elija Clusters (Clústeres) y seleccione el clúster que aloja la instancia.
4. En la página de Cluster : ***name*** (Clúster; nombre), elija la pestaña Infrastructure (Infraestructura).
5. En Container instances (Instancias de contenedor), seleccione el ID de instancia para cancelar el registro. Se lo redirigirá a la página de detalles de la instancia de contenedor.
6. En la página Container Instance : ***id***, seleccione Deregister.
7. En la pantalla de confirmación, elija Anular el registro.
8. Si no va a utilizar más la instancia de contenedor, termine la instancia de Amazon EC2 subyacente. Para obtener más información, consulte Finalizar una instancia en la Guía del usuario de Amazon EC2.

Drenaje de instancias de contenedor de Amazon ECS

Es posible que, en ocasiones, se deba eliminar una instancia de contenedor de un clúster; por ejemplo, para realizar actualizaciones del sistema o reducir verticalmente la capacidad del clúster. Amazon ECS ofrece la capacidad de pasar de una instancia de contenedor a un estado `DRAINING`. Esto se denomina vaciado de instancias de contenedor. Cuando se establece una instancia de

contenedor en `DRAINING`, Amazon ECS evita que se programen nuevas tareas para su ubicación en la instancia de contenedor.

Comportamiento de drenaje de los servicios

Se detiene de inmediato cualquier tarea que forme parte de un servicio que se encuentre en estado `PENDING`. Si la instancia de contenedor tienen capacidad disponible en el clúster, el programador de servicios iniciará las tareas de sustitución. Si la instancia de contenedor no dispone de capacidad suficiente, se enviará un mensaje de evento de servicio indicando el problema.

Las tareas que forman parte de un servicio en la instancia de contenedor y se encuentran en estado `RUNNING` pasan al estado `STOPPED`. El programador de servicios intenta sustituir las tareas de acuerdo con los parámetros `minimumHealthyPercent` y `maximumPercent` de configuración y el tipo de implementación del servicio. Para obtener más información, consulte Servicios de Amazon ECS y Parámetros de definición de servicio de Amazon ECS.

- Si `minimumHealthyPercent` está por debajo del 100%, el programador puede hacer caso omiso de `desiredCount` temporalmente durante la sustitución de tareas. Por ejemplo, `desiredCount` son cuatro tareas, un mínimo del 50% permite al programador detener dos tareas existentes antes de iniciar dos nuevas tareas. Si el mínimo es el 100%, el programador de servicio no puede eliminar las tareas existentes hasta que las tareas de sustitución se consideren en buen estado. Si hay tareas para servicios que no utilizan un balanceador de carga en el estado `RUNNING`, se consideran en buen estado. Las tareas para servicios que utilizan un balanceador de carga se consideran en buen estado si están en estado `RUNNING` y el balanceador de carga notifica que la instancia de contenedor en la que están alojados tiene buen estado.

  > **Important**
  >
  > Si usa instancias de spot y `minimumHealthyPercent` es superior o igual al 100 %, el servicio no tendrá tiempo suficiente para reemplazar la tarea antes de que finalice la instancia de spot.

- El parámetro `maximumPercent` representa un límite superior en la cantidad de tareas en ejecución durante la sustitución de tareas, que permite definir el tamaño del lote de sustitución. Por ejemplo, si `desiredCount` de cuatro tareas, un máximo de 200% comienza cuatro tareas nuevas antes de parar las cuatro tareas que se van a vaciar (siempre que los recursos del clúster requeridos para hacer esto estén disponibles). Si el mínimo es 100%, entonces las tareas de sustitución no pueden comenzar hasta que se hayan parado las tareas de vaciado.

> **Important**
>
> Si tanto `minimumHealthyPercent` como `maximumPercent` son el 100 %, entonces el servicio no puede eliminar las tareas existentes y tampoco puede iniciar tareas de reemplazo. Esto impide el drenaje correcto de las instancias de contenedores e impide realizar nuevas implementaciones.

Comportamiento de drenaje para tareas independientes

Las tarea independiente en estado `PENDING` o `RUNNING` no se verán afectadas; debe esperar a que se detengan por su cuenta o detenerlas manualmente. La instancia de contenedor permanecerá en el estado `DRAINING`.

Una instancia de contenedor ha completado el vaciado cuando todas las tareas que se ejecutan en la instancia pasan al estado `STOPPED`. La instancia de contenedor permanece en estado `DRAINING` hasta que se vuelva a activar o se elimine. Puede verificar el estado de las tareas de la instancia de contenedor mediante la operación ListTasks, a través del parámetro `containerInstance`, para obtener una lista de tareas de la instancia, y realizar a continuación la operación DescribeTasks, a través del nombre de recurso de Amazon (ARN) o el ID de cada tarea, para verificar el estado de la tarea.

Cuando esté listo para que la instancia de contenedor comience a alojar nuevamente tareas, cambie el estado de la instancia de contenedor de `DRAINING` a `ACTIVE`. El programador de servicios de Amazon ECS volverá a considerar la instancia de contenedor para la ubicación de tareas.

Procedimiento

Para establecer una instancia de contenedor en vaciado desde la nueva AWS Management Console, siga estos pasos.

También puede utilizar la acción de API UpdateContainerInstancesState o el comando update-container-instances-state para cambiar el estado de una instancia de contenedor a `DRAINING`.

AWS Management Console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).

3. En la página Clusters (Clústeres), elija un clúster que aloja sus instancias.
4. En la página de Cluster : ***name*** (Clúster; nombre), elija la pestaña Infrastructure (Infraestructura). En Container instances (Instancias de contenedor) seleccione la casilla de verificación junto a cada instancia de contenedor que desee vaciar.
5. Elija Acciones, Vaciar.

Instancias de contenedor de EC2 de Amazon ECS

El agente de Amazon ECS es un proceso que se ejecuta en todas las instancias de contenedor que estén registradas en el clúster. Facilita la comunicación entre sus instancias de contenedor y Amazon ECS.

> **Note**
>
> En las instancias de contenedores de Linux, el contenedor de agente monta directorios de nivel superior como `/lib`, `/lib64` y `/proc`. Esto es necesario para las características y funcionalidades de ECS, como los volúmenes de Amazon EBS, el modo de red `awsvpc`, Amazon ECS Service Connect y FireLens para Amazon ECS.

Cada versión del agente de contenedor de Amazon ECS admite un conjunto de características diferente y proporciona correcciones de errores de versiones anteriores. Cuando sea posible, siempre recomendamos utilizar la versión más reciente del agente de contenedor de Amazon ECS. Para actualizar el agente de contenedor a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS.

El agente de contenedor de Amazon ECS contiene la imagen `amazon-ecs-pause`. Amazon ECS utiliza esta imagen para las tareas que utilizan el modo de red `awsvpc`.

Para ver las características y mejoras incluidas en cada versión del agente, consulte https://github.com/aws/amazon-ecs-agent/releases.

> **Important**
>
> La versión mínima de Docker para obtener métricas fiables es la versión de Docker `v20.10.13` y versiones posteriores, que se incluyen en la AMI `20220607` optimizada para Amazon ECS y versiones posteriores.

Los agentes de Amazon ECS versión `1.20.0` y posteriores ya no admiten versiones de Docker anteriores a la `18.01.0`.

Ciclo de vida

Cuando el agente de contenedor de Amazon ECS registra una instancia de Amazon EC2 en el clúster, la instancia de Amazon EC2 notifica su estado como `ACTIVE` y su estado de conexión del agente como `TRUE`. Esta instancia de contenedor puede aceptar solicitudes de ejecución de tareas.

Si detiene (no termina) una instancia de contenedor, el estado permanece `ACTIVE`, pero el estado de conexión del agente cambia a `FALSE` en pocos minutos. Cualquier tarea que se estuviera ejecutando en la instancia de contenedor se para. Si vuelve a comenzar la instancia de contenedor, el agente de contenedor vuelve a conectarse al servicio de Amazon ECS, y usted podrá volver a ejecutar tareas en la instancia.

> **Important**
>
> Si detiene e inicia una instancia de contenedor o reinicia esa instancia, algunas versiones más antiguas del agente de contenedor de Amazon ECS vuelven a registrar la instancia sin anular el registro del ID de la instancia de contenedor original. En este caso, Amazon ECS muestra más instancias de contenedor en el clúster de las que tiene en realidad. (Si dispone de ID de instancias de contenedor duplicados para el mismo ID de instancia de Amazon EC2, puede anular con seguridad el registro de los duplicados que se muestran como `ACTIVE` con un estado de conexión de agente `FALSE`). Este problema se corrige en la versión actual del agente de contenedor de Amazon ECS. Para obtener más información sobre cómo actualizar a la versión actual, consulte Actualización del agente de contenedor de Amazon ECS.

Si cambia el estado de una instancia de contenedor a `DRAINING`, las nuevas tareas no se colocan en la instancia de contenedor. Cualquier tarea de servicio que se ejecute en la instancia de contenedor se elimina, si es posible, para que pueda llevar a cabo las actualizaciones del sistema. Para obtener más información, consulte Drenaje de instancias de contenedor de Amazon ECS.

Si cancela el registro o termina una instancia de contenedor, el estado de la instancia de contenedor cambia a `INACTIVE` de inmediato y la instancia de contenedor ya no se notifica cuando se enumeran las instancias de contenedor. No obstante, puede seguir describiendo la instancia de

contenedor para una hora tras la terminación. Después de una hora, la descripción de la instancia ya no está disponible.

> **Important**
>
> Puede drenar las instancias manualmente o crear un enlace del ciclo de vida del grupo de Auto Scaling para establecer el estado de la instancia en `DRAINING`. Para obtener más información sobre los enlaces de ciclo de vida de Auto Scaling, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling.

Compatibilidad con Docker

Amazon ECS es compatible con las dos últimas versiones principales de Docker publicadas en Amazon Linux. Actualmente, esto incluye Docker 20.10.x y Docker 25.x.

La versión mínima de Docker necesaria para Amazon ECS se encuentra en el archivo de especificaciones del agente de Amazon ECS en GitHub.

Al utilizar la AMI optimizada para Amazon ECS, Docker está previamente instalado y configurado para funcionar con el agente de contenedor de Amazon ECS. La AMI incluye una versión de Docker probada y compatible con Amazon ECS.

> **Note**
>
> Si bien Amazon ECS admite varias versiones de Docker, recomendamos usar la versión de Docker que viene con la AMI optimizada para Amazon ECS para obtener la mejor compatibilidad.

AMI optimizada para Amazon ECS

Las variantes de Linux de la AMI optimizada para Amazon ECS utilizan la AMI de Amazon Linux 2 como base. El nombre de AMI de origen de Amazon Linux 2 para cada variante se puede recuperar consultando la API de Systems Manager Parameter Store. Para obtener más información, consulte Recuperación de metadatos de las AMI de Linux optimizadas para Amazon ECS. Cuando lanza las instancias de contenedor desde la AMI de Amazon Linux 2 optimizada para Amazon ECS más reciente, recibirá la versión actual del agente de contenedor. Para lanzar una instancia de

contenedor con la AMI de Amazon Linux 2 optimizada para Amazon ECS, consulte Lanzamiento de una instancia de contenedor de Linux de Amazon ECS.

Información adicional

En las páginas siguientes se proporciona información adicional acerca de los cambios:

- Registro de cambios del agente de Amazon ECS en GitHub
- El código de origen de la aplicación `ecs-init` y los scripts y la configuración para empaquetar el agente ahora forman parte del repositorio de agentes. Para ver versiones anteriores de `ecs-init` y paquetes, consulte el registro de cambios de Amazon ecs-init en GitHub.
- Notas de la versión de Amazon Linux 2.
- Notas de la versión de Docker Engine en la documentación de Docker
- Documentación de controlador NVIDIA en la documentación de NVIDIA

Configuración del agente de contenedor de Amazon ECS

Se aplica a: instancias de EC2

El agente de contenedor de Amazon ECS admite una serie de opciones de configuración, la mayoría de las cuales se establecen a través de variables de entorno.

Si la instancia de contenedor se lanzó con la variante de Linux de la AMI optimizada para Amazon ECS, puede establecer estas variables de entorno en el archivo `/etc/ecs/ecs.config` y, a continuación, reiniciar el agente. También puede escribir estas variables de configuración en sus instancias de contenedor con datos de usuario de Amazon EC2 en el momento del lanzamiento. Para obtener más información, consulte Arranque de instancias de contenedor de Linux de Amazon ECS para la transferencia de datos.

Si la instancia de contenedor se lanzó con la variante de Windows de la AMI optimizada para Amazon ECS, puede establecer estas variables de entorno en el comando PowerShell SetEnvironmentVariable y luego reiniciar el agente. Para obtener más información, consulte Ejecución de comandos al lanzar una instancia de EC2 con la entrada de datos de usuario en la Guía del usuario de Amazon EC2 y the section called “Arranque de instancias de contenedor”.

Si está iniciando manualmente el agente de contenedor de Amazon ECS (para AMI no optimizadas para Amazon ECS), puede utilizar estas variables de entorno en el comando

docker run que utiliza para iniciar el agente. Utilice estas variables con la sintaxis `--env=VARIABLE_NAME=VARIABLE_VALUE`. En el caso de información confidencial, por ejemplo credenciales de autenticación para repositorios privados, debe almacenar las variables de entorno del agente en un archivo y transmitirlas a la vez con la opción `--env-file path_to_env_file`. Puede utilizar los siguientes comandos para agregar las variables.

```
sudo systemctl stop ecs
sudo vi /etc/ecs/ecs.config
# And add the environment variables with VARIABLE_NAME=VARIABLE_VALUE format.
sudo systemctl start ecs
```

Ejecución del agente de Amazon ECS con el espacio de nombres PID del host

De forma predeterminada, el agente de Amazon ECS se ejecuta con su propio espacio de nombres PID. En las siguientes configuraciones, puede configurar el agente de Amazon ECS para que se ejecute con el espacio de nombres PID del host:

- El modo de aplicación de SELinux está activado.
- La política de seguridad SELinux de Docker está establecida como verdadera.

Puede configurar este comportamiento estableciendo la variable de entorno `ECS_AGENT_PID_NAMESPACE_HOST` en `true` en su archivo `/etc/ecs/ecs.config`. Cuando esta variable esté habilitada, `ecs-init` iniciará el contenedor del agente de Amazon ECS con el espacio de nombres PID del host (`--pid=host`), lo que permitirá que el agente se inicie correctamente en entornos que utilizan SELinux. Esta característica está disponible en las versiones de agente `1.94.0` de Amazon ECS y posteriores.

Para habilitar esta característica, agregue la siguiente línea a su archivo `/etc/ecs/ecs.config`:

```
ECS_AGENT_PID_NAMESPACE_HOST=true
```

Tras realizar este cambio, reinicie el agente de Amazon ECS para que surta efecto:

```
sudo systemctl restart ecs
```

Las siguientes características no funcionarán cuando el modo de aplicación de SELinux esté activado y la política de seguridad de Docker esté configurada como verdadera, incluso cuando esté configurado `ECS_AGENT_PID_NAMESPACE_HOST=true`.

- Amazon ECS Exec
- Tarea adjunta de Amazon EBS
- Service Connect
- FireLens para Amazon ECS

Parámetros disponibles

Para obtener más información acerca de la utilización del agente de contenedor, consulte Parámetros de configuración del agente de contenedor de Amazon ECS.

Almacenamiento de la configuración de instancia de contenedor de Amazon ECS en Amazon S3

La configuración del agente de contenedor de Amazon ECS se controla mediante la variable de entorno. Las variantes de Linux de la AMI optimizada para Amazon ECS buscan estas variables en `/etc/ecs/ecs.config` cuando se inicia el agente de contenedor y el agente se configura en consecuencia. Las variables de entorno no sensibles, como `ECS_CLUSTER`, se pueden transmitir a la instancia de contenedor en el lanzamiento a través de los datos de usuario de Amazon EC2 y escribir en este archivo sin consecuencias. No obstante, otro tipo de información confidencial, como las credenciales de AWS o la variable `ECS_ENGINE_AUTH_DATA`, no deben pasarse nunca a una instancia en los datos de usuario ni escribirse en `/etc/ecs/ecs.config` de manera que les permita aparecer en un archivo `.bash_history`.

El almacenamiento de la información de configuración en un bucket privado de Amazon S3 y la concesión de acceso de solo lectura al rol de IAM de instancia de contenedor es una forma práctica y segura de permitir la configuración de instancia de contenedor en el momento del lanzamiento. Puede almacenar una copia del archivo `ecs.config` en un bucket privado. Puede utilizar los datos de usuario de Amazon EC2 para instalar la AWS CLI y copiar la información de configuración en `/etc/ecs/ecs.config` cuando se lance la instancia.

Para almacenar un archivo **`ecs.config`** en Amazon S3

1. Debe conceder permisos al rol de instancia de contenedor (ecsInstanceRole) para que tenga acceso de solo lectura a Amazon S3. Para ello, asigne AmazonS3ReadOnlyAccess al rol `ecsInstanceRole`. Para obtener información acerca de cómo asociar una política a un rol, consulte Actualización de los permisos de un rol en la Guía del usuario de AWS Identity and Access Management.

2. Cree un `ecs.config` archivo con variables de configuración del agente Amazon ECS válidas con el siguiente formato. En este ejemplo, se configura la autenticación de registros privados. Para obtener más información, consulte Uso de imágenes de contenedor que no sean de AWS en Amazon ECS.

```
ECS_ENGINE_AUTH_TYPE=dockercfg
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example.com"}}
```

> **Note**
>
> Para obtener una lista completa de las variables de configuración del agente Amazon ECS disponibles, consulte Amazon ECS Container Agent en GitHub.

3. Para almacenar el archivo de configuración, cree un bucket privado en Amazon S3. Para obtener más información, consulte Creación de un bucket en la Guía del usuario de Amazon Simple Storage Service.
4. Cargue el archivo `ecs.config` en el bucket de S3. Para obtener más información, consulte Carga de objetos en la Guía del desarrollador de Amazon Simple Storage Service.

Para cargar un archivo **`ecs.config`** desde Amazon S3 en el momento del lanzamiento

1. Complete los procedimientos indicados anteriormente en esta sección para permitir el acceso de solo lectura de Amazon S3 a las instancias de contenedor y almacenar un archivo `ecs.config` en un bucket de S3 privado.
2. Lance nuevas instancias de contenedor y utilice el siguiente script de ejemplo en Datos de usuario de EC2. El script instala la AWS CLI y copia el archivo de configuración en `/etc/ecs/ecs.config`. Para obtener más información, consulte Lanzamiento de una instancia de contenedor de Linux de Amazon ECS.

```
#!/bin/bash
yum install -y aws-cli
aws s3 cp s3://your_bucket_name/ecs.config /etc/ecs/ecs.config
```

Instalación del agente de contenedor de Amazon ECS

Si desea registrar una instancia de Amazon EC2 en su clúster de Amazon ECS y dicha instancia no utiliza una AMI basada en la AMI optimizada para Amazon ECS, puede instalar el agente de contenedor de Amazon ECS manualmente mediante el siguiente procedimiento. Para ello, puede descargar el agente desde uno de los buckets de Amazon S3 de la región o desde Amazon Elastic Container Registry Public. Si lo descarga de uno de los buckets de Amazon S3 de la región, puede verificar la validez del archivo del agente de contenedor mediante la firma PGP.

> **Note**
>
> Las unidades `systemd` de los servicios de Amazon ECS y de Docker tienen la instrucción de esperar a que `cloud-init` finalice antes de comenzar ambos servicios. El proceso `cloud-init` no se considera finalizado hasta que los datos de usuario de Amazon EC2 hayan terminado de ejecutarse. Por lo tanto, el inicio de Amazon ECS o de Docker a través de los datos de usuario de Amazon EC2 puede provocar un bloqueo. Para comenzar el agente de contenedor mediante los datos de usuario de Amazon EC2, puede utilizar `systemctl enable --now --no-block ecs.service`.

Instalación del agente de contenedor de Amazon ECS en una instancia de EC2 que no es de Amazon Linux

Para instalar el agente de contenedor de Amazon ECS en una instancia de Amazon EC2, puede descargar el agente desde uno de los buckets de Amazon S3 de la región e instalarlo.

> **Note**
>
> Cuando utilice una AMI que no sea de Amazon Linux, la instancia de Amazon EC2 requiere que el `cgroupfs` sea compatible con el controlador `cgroup` para que el agente de Amazon ECS admita los límites de recursos de nivel de tarea. Para obtener más información, consulte Agente de Amazon ECS en Github.

A continuación, se muestran los archivos del agente de contenedor de Amazon ECS más recientes por región para cada arquitectura del sistema, a modo de referencia.

Región	Nombre de la región	Archivos deb de init de Amazon ECS	Archivos rpm de init de Amazon ECS
us-east-2	Este de EE. UU. (Ohio)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
us-east-1	Este de EE. UU. (Norte de Virginia)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
us-west-1	Oeste de EE. UU. (Norte de California)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
us-west-2	Oeste de EE. UU. (Oregón)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-east-1	Asia-Pacífico (Hong Kong)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-northeast-1	Asia-Pacífico (Tokio)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Región	Nombre de la región	Archivos deb de init de Amazon ECS	Archivos rpm de init de Amazon ECS
ap-northeast-2	Asia-Pacífico (Seúl)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-south-1	Asia-Pacífico (Mumbai)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-southeast-1	Asia-Pacífico (Singapur)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-southeast-2	Asia-Pacífico (Sídney)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ca-central-1	Canadá (centro)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
eu-central-1	Europa (Fráncfort)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Región	Nombre de la región	Archivos deb de init de Amazon ECS	Archivos rpm de init de Amazon ECS
eu-west-1	Europa (Irlanda)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
eu-west-2	Europa (Londres)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
eu-west-3	Europa (París)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
sa-east-1	América del Sur (São Paulo)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 Amazon ECS init aarch64 (aarch64)
us-gov-east-1	AWS GovCloud (Este de EE. UU.)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
us-gov-west-1	AWS GovCloud (Oeste de EE. UU.)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Para instalar el agente de contenedor de Amazon ECS en una instancia de Amazon EC2 con una AMI que no sea de Amazon Linux

1. Lance una instancia de Amazon EC2 con un rol de IAM que permita obtener acceso a Amazon ECS. Para obtener más información, consulte Rol de IAM de instancia de contenedor de Amazon ECS.
2. Conecte con la instancia .
3. Instale la última versión de Docker en su instancia.
4. Compruebe la versión de Docker para verificar que su sistema cumpla con el requisito de la versión mínima.

> **Note**
>
> La versión mínima de Docker para obtener métricas fiables es la versión de Docker `v20.10.13` y versiones posteriores, que se incluyen en la AMI `20220607` optimizada para Amazon ECS y versiones posteriores.
> Los agentes de Amazon ECS versión `1.20.0` y posteriores ya no admiten versiones de Docker anteriores a la `18.01.0`.

```
docker --version
```

5. Descargue el archivo del agente correspondiente de Amazon ECS correspondiente correspondiente a su sistema y arquitectura del sistema del sistema e instálelo.

   Para arquitecturas `deb`:

```
ubuntu:~$ curl -O https://s3.us-west-2.amazonaws.com/amazon-ecs-agent-us-west-2/
amazon-ecs-init-latest.amd64.deb
ubuntu:~$ sudo dpkg -i amazon-ecs-init-latest.amd64.deb
```

   Para arquitecturas `rpm`:

```
fedora:~$ curl -O https://s3.us-west-2.amazonaws.com/amazon-ecs-agent-us-west-2/
amazon-ecs-init-latest.x86_64.rpm
fedora:~$ sudo yum localinstall -y amazon-ecs-init-latest.x86_64.rpm
```

6. Edite el archivo `/lib/systemd/system/ecs.service` y agregue la siguiente línea al final de la sección `[Unit]`.

```
After=cloud-final.service
```

7. (Opcional) Para registrar la instancia en un clúster distinto del clúster `default`, edite el archivo `/etc/ecs/ecs.config` y agregue los siguientes contenidos. Por ejemplo, lo siguiente especifica el clúster `MyCluster`:

```
ECS_CLUSTER=MyCluster
```

Para obtener más información acerca de estas y otras opciones de tiempo de ejecución de agente, consulte Configuración del agente de contenedor de Amazon ECS.

> **Note**
>
> Si lo desea, puede almacenar las variables de entorno del agente en Amazon S3 (se pueden descargar en las instancias de contenedor en el momento del lanzamiento utilizando datos de usuario de Amazon EC2). Se recomienda su uso para información confidencial como las credenciales de autenticación para repositorios privados. Para obtener más información, consulte Almacenamiento de la configuración de instancia de contenedor de Amazon ECS en Amazon S3 y Uso de imágenes de contenedor que no sean de AWS en Amazon ECS.

8. Inicie el servicio `ecs`.

```
ubuntu:~$ sudo systemctl start ecs
```

Ejecución del agente de Amazon ECS en el modo de red del host

Al ejecutar el agente de contenedor de Amazon ECS, `ecs-init` creará el contenedor del agente de contenedores con el modo de red `host`. Este es el único modo de red compatible para el contenedor de agente de contenedores.

Esto permite bloquear el acceso al punto de conexión de servicio de metadatos de la instancia de Amazon EC2 (`http://169.254.169.254`) para los contenedores que inicia el agente de contenedor. Esto garantiza que los contenedores no puedan obtener acceso a las credenciales del rol de IAM desde el perfil de instancia de contenedor y obliga a que las tareas utilicen solo las

credenciales del rol de tarea de IAM. Para obtener más información, consulte Rol de IAM de tarea de Amazon ECS.

Esto también lo hace para que el agente de contenedor no compita por las conexiones y el tráfico de red en el puente `docker0`.

Parámetros de configuración del registro del agente de contenedor de Amazon ECS

El agente de contenedor de Amazon ECS almacena registros en las instancias de contenedor.

Para el agente de contenedor versión 1.36.0 y posterior, los registros se encuentran de forma predeterminada en `/var/log/ecs/ecs-agent.log` en instancias de Linux y en `C:\ProgramData\Amazon\ECS\log\ecs-agent.log` en instancias de Windows.

Para el agente de contenedor versión 1.35.0 y anterior, los registros se encuentran de forma predeterminada en `/var/log/ecs/ecs-agent.log.`*`timestamp`* en instancias de Linux y en `C:\ProgramData\Amazon\ECS\log\ecs-agent.log.`*`timestamp`* en instancias de Windows.

De forma predeterminada, los registros de agente rotan cada hora y se almacena un máximo de 24 registros.

A continuación se muestran las variables de configuración del agente de contenedor que se pueden utilizar para cambiar el comportamiento de registro de agente predeterminado. Para obtener información detallada sobre todos los parámetros de configuración disponibles, consulte Configuración del agente de contenedor de Amazon ECS o el README del agente de Amazon ECS en GitHub.

Para el agente de contenedor versión 1.36.0 y posterior, a continuación se ofrece un archivo de registro de ejemplo cuando se utiliza el formato `logfmt`.

```
level=info time=2019-12-12T23:43:29Z msg="Loading configuration" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Image excluded from cleanup: amazon/amazon-
ecs-agent:latest" module=parse.go
level=info time=2019-12-12T23:43:29Z msg="Image excluded from cleanup: amazon/amazon-
ecs-pause:0.1.0" module=parse.go
level=info time=2019-12-12T23:43:29Z msg="Amazon ECS agent Version: 1.36.0, Commit:
 ca640387" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Creating root ecs cgroup: /ecs"
 module=init_linux.go
```

```
level=info time=2019-12-12T23:43:29Z msg="Creating cgroup /ecs"
 module=cgroup_controller_linux.go
level=info time=2019-12-12T23:43:29Z msg="Loading state!" module=statemanager.go
level=info time=2019-12-12T23:43:29Z msg="Event stream ContainerChange start
 listening..." module=eventstream.go
level=info time=2019-12-12T23:43:29Z msg="Restored cluster 'auto-robc'" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Restored from checkpoint file. I
 am running as 'arn:aws:ecs:us-west-2:0123456789:container-instance/auto-
robc/3330a8a91d15464ea30662d5840164cd' in cluster 'auto-robc'" module=agent.go
```

A continuación se ofrece un archivo de registro de ejemplo cuando se utiliza el formato JSON.

```
{"time": "2019-11-07T22:52:02Z", "level": "info", "msg": "Starting Amazon Elastic
 Container Service Agent", "module": "engine.go"}
```

Para el agente de contenedor versión 1.35.0 y anterior, a continuación se muestra el formato del archivo de registro.

```
2016-08-15T15:54:41Z [INFO] Starting Agent: Amazon ECS Agent - v1.12.0 (895f3c1)
2016-08-15T15:54:41Z [INFO] Loading configuration
2016-08-15T15:54:41Z [WARN] Invalid value for task cleanup duration, will be overridden
 to 3h0m0s, parsed value 0, minimum threshold 1m0s
2016-08-15T15:54:41Z [INFO] Checkpointing is enabled. Attempting to load state
2016-08-15T15:54:41Z [INFO] Loading state! module="statemanager"
2016-08-15T15:54:41Z [INFO] Detected Docker versions [1.17 1.18 1.19 1.20 1.21 1.22]
2016-08-15T15:54:41Z [INFO] Registering Instance with ECS
2016-08-15T15:54:41Z [INFO] Registered! module="api client"
```

Configuración de instancias de contenedor de Amazon ECS para imágenes de Docker privadas

El agente de contenedor de Amazon ECS puede realizar autenticaciones en registros privados, mediante autenticación básica. Cuando se habilita la autenticación de registros privados, puede utilizar las imágenes de Docker privadas en sus definiciones de tareas. Esta característica solo se admite en tareas que utilizan el tipo de lanzamiento de EC2.

Otro método para habilitar la autenticación de registros privados es usar AWS Secrets Manager para almacenar sus credenciales de registros privados de forma segura y hacer referencia a ellas en su definición de contenedor. Esto permite que las tareas usen imágenes de los repositorios privados. Este método es compatible con las tareas que utilizan los tipos de lanzamiento de EC2 o Fargate.

Para obtener más información, consulte Uso de imágenes de contenedor que no sean de AWS en Amazon ECS.

El agente de contenedor de Amazon ECS busca dos variables de entorno cuando se lanza:

- `ECS_ENGINE_AUTH_TYPE`, que especifica el tipo de datos de autenticación que se están enviando.
- `ECS_ENGINE_AUTH_DATA`, que contiene las credenciales de autenticación reales.

Las variantes Linux de la AMI optimizada para Amazon ECS exploran el archivo `/etc/ecs/ecs.config` en busca de estas variables cuando se lanza la instancia de contenedor y cada vez que se inicia el servicio (mediante el comando sudo start ecs). Las AMI no optimizadas para Amazon ECS deben almacenar estas variables de entorno en un archivo y pasarlas con la opción `--env-file` *`path_to_env_file`* al comando docker run que inicia el agente de contenedor.

> **Important**
>
> Recomendamos no introducir estas variables de entorno de autenticación en el momento del lanzamiento de la instancia con los datos de usuario de Amazon EC2 ni transferirlas con la opción `--env` al docker runcomando . Estos métodos no son adecuados para la información confidencial como las credenciales de autenticación. Para obtener información sobre cómo añadir de forma segura credenciales de autenticación a sus instancias de contenedor, consulte Almacenamiento de la configuración de instancia de contenedor de Amazon ECS en Amazon S3.

Formatos de autenticación

Existen dos formatos disponibles para autenticación de registros privados, `dockercfg` y `docker`.

Formato de autenticación dockercfg

El formato `dockercfg` utiliza la información de autenticación almacenada en el archivo de configuración que se crea cuando se ejecuta el comando docker login. Puede crear este archivo ejecutando el comando docker login en el sistema local y especificar el nombre de usuario, la contraseña y la dirección de correo electrónico del registro. También puede iniciar sesión en una instancia de contenedor y ejecutar el comando en ella. Dependiendo de su versión de Docker, este archivo se guarda como `~/.dockercfg` o `~/.docker/config.json`.

```
cat ~/.docker/config.json
```

Salida:

```
{
  "auths": {
    "https://index.docker.io/v1/": {
      "auth": "zq212MzEXAMPLE7o6T25Dk0i"
    }
  }
}
```

> **Important**
>
> Las versiones más nuevas de Docker crean un archivo de configuración como se muestra más arriba con un objeto `auths` exterior. El agente de Amazon ECS solo admite los datos de autenticación `dockercfg` que están en el formato siguiente, sin el objeto `auths`. Si tiene instalada la utilidad jq, puede extraer estos datos con el siguiente comando: cat ~/.docker/config.json | jq .auths

```
cat ~/.docker/config.json | jq .auths
```

Salida:

```
{
  "https://index.docker.io/v1/": {
    "auth": "zq212MzEXAMPLE7o6T25Dk0i",
    "email": "email@example.com"
  }
}
```

En el ejemplo anterior, se deben agregar las siguientes variables de entorno al archivo de variables de entorno (`/etc/ecs/ecs.config` para la AMI optimizada para Amazon ECS) que el agente de contenedor de Amazon ECS carga en tiempo de ejecución. Si no está utilizando la AMI optimizada para Amazon ECS e inicia el agente manualmente con docker run, especifique el archivo de variables de entorno con la opción `--env-file` *`path_to_env_file`* al iniciar el agente.

```
ECS_ENGINE_AUTH_TYPE=dockercfg
```

```
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example.com"}}
```

Puede configurar varios registros privados con la sintaxis siguiente:

```
ECS_ENGINE_AUTH_TYPE=dockercfg
ECS_ENGINE_AUTH_DATA={"repo.example-01.com":
{"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example-01.com"},"repo.example-02.com":
{"auth":"fQ172MzEXAMPLEoF7225DU0j","email":"email@example-02.com"}}
```

Formato de autenticación de Docker

El formato `docker` utiliza una representación JSON para el servidor de registro en el que el agente se debe autenticar. También incluye los parámetros de autenticación requeridos por dicho registro (por ejemplo, nombre de usuario, contraseña y la dirección de correo electrónico de dicha cuenta). Para una cuenta de Docker Hub, la representación JSON tiene el siguiente aspecto:

```
{
  "https://index.docker.io/v1/": {
    "username": "my_name",
    "password": "my_password",
    "email": "email@example.com"
  }
}
```

En este ejemplo, se deben agregar las siguientes variables de entorno al archivo de variables de entorno (`/etc/ecs/ecs.config` para la AMI optimizada para Amazon ECS) que el agente de contenedor de Amazon ECS carga en tiempo de ejecución. Si no está utilizando la AMI optimizada para Amazon ECS e inicia el agente manualmente con docker run, especifique el archivo de variables de entorno con la opción `--env-file` *`path_to_env_file`* al iniciar el agente.

```
ECS_ENGINE_AUTH_TYPE=docker
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"username":"my_name","password":"my_password","email":"email@example.com"}}
```

Puede configurar varios registros privados con la sintaxis siguiente:

```
ECS_ENGINE_AUTH_TYPE=docker
```

```
ECS_ENGINE_AUTH_DATA={"repo.example-01.com":
{"username":"my_name","password":"my_password","email":"email@example-01.com"},"repo.example-02
{"username":"another_name","password":"another_password","email":"email@example-02.com"}}
```

Procedimiento

Utilice el siguiente procedimiento para activar registros privados para las instancias de contenedor.

Para habilitar los registros privados en la AMI optimizada para Amazon ECS

1. Inicie sesión en su instancia de contenedor mediante SSH.
2. Abra el archivo `/etc/ecs/ecs.config` y añada los valores `ECS_ENGINE_AUTH_TYPE` y `ECS_ENGINE_AUTH_DATA` para su registro y cuenta:

   ```
   sudo vi /etc/ecs/ecs.config
   ```

   En este ejemplo se autentica una cuenta de usuario de Docker Hub:

   ```
   ECS_ENGINE_AUTH_TYPE=docker
   ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
   {"username":"my_name","password":"my_password","email":"email@example.com"}}
   ```

3. Compruebe si su agente utiliza la variable de entorno `ECS_DATADIR` para guardar su estado:

   ```
   docker inspect ecs-agent | grep ECS_DATADIR
   ```

   Salida:

   ```
   "ECS_DATADIR=/data",
   ```

   > **Important**
   >
   > Si el comando anterior no devuelve la variable de entorno `ECS_DATADIR`, debe detener las tareas en ejecución en esta instancia de contenedor antes de detener el agente. Los agentes más recientes con la variable de entorno `ECS_DATADIR` guardan su estado y usted puede detenerlos e iniciarlos mientras que las tareas se ejecuten sin problemas. Para obtener más información, consulte Actualización del agente de contenedor de Amazon ECS.

4. Detenga el servicio `ecs`:

```
sudo stop ecs
```

Salida:

```
ecs stop/waiting
```

5. Reinicie el servicio `ecs`.

   - Para la AMI de Amazon Linux 2 optimizada para Amazon ECS:

```
sudo systemctl restart ecs
```

   - Para la AMI de Amazon Linux optimizada para Amazon ECS:

```
sudo stop ecs && sudo start ecs
```

6. (Opcional) Puede verificar que el agente se está ejecutando y ver información acerca de la nueva instancia de contenedor consultando la operación de la API de introspección del agente. Para obtener más información, consulte the section called “Introspección de contenedor”.

```
curl http://localhost:51678/v1/metadata
```

Limpieza automática de tareas e imágenes de Amazon ECS

Cada vez que se coloca una tarea en una instancia de contenedor, el agente de contenedor de Amazon ECS comprueba si las imágenes a las que se hace referencia en la tarea son las más recientes de la etiqueta especificada en el repositorio. De lo contrario, el comportamiento predeterminado permite que el agente extraiga las imágenes desde sus repositorios respectivos. Si actualiza las imágenes con frecuencia en las tareas y servicios, el almacenamiento de la instancia de contenedor se puede rellenar rápidamente con imágenes de Docker que ya no utiliza y probablemente no volverá a utilizar. Por ejemplo, podría utilizar una canalización de integración e implementación continuas (CI/CD).

> **Note**
>
> El comportamiento de extracción de imágenes del agente de Amazon ECS se puede personalizar mediante el parámetro `ECS_IMAGE_PULL_BEHAVIOR`. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS.

Del mismo modo, los contenedores que pertenecen a tareas detenidas también pueden consumir almacenamiento de la instancia de contenedor con información de registro, volúmenes de datos y otros artefactos. Estos artefactos son útiles para la depuración de contenedores que se han detenido de forma inesperada, pero la mayor parte de este almacenamiento se puede liberar con seguridad tras un periodo de tiempo.

De forma predeterminada, el agente de contenedor de Amazon ECS limpia automáticamente las tareas detenidas y las imágenes de Docker que no está utilizando ninguna tarea en las instancias de contenedor.

> **Note**
>
> La característica de limpieza automatizada de imágenes requiere al menos la versión 1.13.0 del agente de contenedor de Amazon ECS. Para actualizar el agente a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS.

Las siguientes variables de configuración del agente están disponibles para ajustar la experiencia de limpieza de imágenes y tareas automatizada. Para obtener más información acerca de cómo se establecen estas variables en las instancias de contenedor, consulte Configuración del agente de contenedor de Amazon ECS.

`ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`

El tiempo de espera predeterminado para eliminar los contenedores de una tarea detenida. Si el valor se establece en menos que 1 segundo, el valor se ignora. De forma predeterminada, este parámetro se establece en 3 horas, pero puede reducir este período a solo 1 minuto, si lo necesita para su aplicación.

> **Note**
>
> El proceso de limpieza de imágenes no puede eliminar una imagen en tanto en cuando haya un contenedor que haga referencia a la misma. Una vez retirados los contenedores, las imágenes no referenciadas se convierten en candidatas para la limpieza en función de los parámetros de configuración de la limpieza de imágenes.

`ECS_DISABLE_IMAGE_CLEANUP`

Si establece esta variable como `true`, la limpieza de imagen automatizada se desactiva en la instancia de contenedor y no se elimina ninguna imagen automáticamente.

`ECS_IMAGE_CLEANUP_INTERVAL`

Esta variable especifica con qué frecuencia debe comprobar el proceso de limpieza de imagen las imágenes para eliminarlas. El valor predeterminado es cada 30 minutos, pero puede reducir este período a solo 10 minutos para eliminar las imágenes con más frecuencia.

`ECS_IMAGE_MINIMUM_CLEANUP_AGE`

Esta variable especifica la cantidad de tiempo mínima entre la extracción de una imagen y el momento en que puede convertirse en candidata a su eliminación. Esto se utiliza para evitar la limpieza de las imágenes que se acaban de extraer. El valor predeterminado es una hora.

`ECS_NUM_IMAGES_DELETE_PER_CYCLE`

Esta variable especifica cuántas imágenes se pueden eliminar durante un ciclo de limpieza único. El valor predeterminado es 5 y el mínimo es 1.

Cuando el agente de contenedor de Amazon ECS está en ejecución y no se ha desactivado la limpieza automatizada de imágenes, el agente comprueba si hay imágenes de Docker a las que no se hace referencia en los contenedores en ejecución o detenidos a una frecuencia determinada por la variable `ECS_IMAGE_CLEANUP_INTERVAL`. Si se encuentran imágenes sin utilizar y son más antiguas del tiempo de limpieza mínimo especificado por la variable `ECS_IMAGE_MINIMUM_CLEANUP_AGE`, el agente elimina hasta el número máximo de imágenes especificadas por la variable `ECS_NUM_IMAGES_DELETE_PER_CYCLE`. Se eliminan primero las imágenes a las que se ha hecho referencia menos recientemente. Una vez que las imágenes se han eliminado, el agente espera hasta el siguiente intervalo y vuelve a repetir el proceso.

Programación de los contenedores en Amazon ECS

Amazon Elastic Container Service (Amazon ECS) es un sistema de simultaneidad optimista de estado compartido que ofrece capacidades de programación flexibles para sus cargas de trabajo en contenedores. Los programadores de Amazon ECS utilizan la misma información de estado de clúster que proporciona la API de Amazon ECS para tomar decisiones adecuadas de colocación.

Amazon ECS proporciona un programador de servicio para las tareas y aplicaciones de ejecución prolongada. También ofrece la posibilidad de ejecutar tareas independientes o tareas programadas para trabajos por lotes o tareas que se ejecutan una sola vez. Puede especificar las estrategias de ubicación de tareas y las restricciones para ejecutar las tareas que mejor se adapten a sus necesidades. Por ejemplo, puede especificar si las tareas se ejecutan en varias zonas de disponibilidad o dentro de una sola. Tiene la opción de integrar las tareas con programadores propios personalizados o de terceros.

Opción	Cuándo se debe usar	Más información
Servicio	El programador de servicios es adecuado para servicios y aplicaciones sin estado, de ejecución prolongada. El programador de servicios también puede asegurarse de que las tareas se registren en una balanceador de carga de Elastic Load Balancing. Puede actualizar los servicios que mantiene el programador de servicios. Esto podría incluir la implementación de una nueva definición de tareas o el cambio del número de tareas deseadas que se ejecutan. De forma predeterminada, el programador de servicios distribuye las tareas en varias	Servicios de Amazon ECS

Opción	Cuándo se debe usar	Más información
	zonas de disponibilidad. Sin embargo, puede utilizar estrategias y restricciones de ubicación de tareas para personalizar las decisiones de ubicación de tareas.	
Tarea independiente	Una tarea individual es adecuada para procesos tales como trabajos por lotes que hacen el trabajo y, a continuación, se detienen. Por ejemplo, puede tener una llamada de procesos `RunTask` cuando el trabajo entra en una cola. La tarea extrae trabajo de la cola, realiza el trabajo y, a continuación, se cierra. Al utilizar `RunTask`, puede permitir que la estrategia predeterminada de ubicación de tareas distribuya las tareas aleatoriamente en el clúster. Esto minimiza las posibilidades de que una sola instancia reciba un número desproporcionado de tareas.	Tareas independientes de Amazon ECS

Opción	Cuándo se debe usar	Más información
Tareas programadas	Una tarea programada es adecuada cuando se tienen tareas que se deben ejecutar a intervalos fijos en el clúster; puede utilizar Programador de EventBridge para crear una programación. Puede ejecutar tareas para una operación de copia de seguridad o un análisis de registros. El programa del programador de EventBridge que crea puede ejecutar una o más tareas en el clúster en momentos específicos. El evento programado se puede configurar en un intervalo específico (ejecutar cada *N* minutos, horas o días). De lo contrario, para una programación más complicada, puede utilizar una expresión `cron`.	Uso de Programador de Amazon EventBridge para programar tareas de Amazon ECS

Opciones de computación

Con Amazon ECS, puede especificar la infraestructura en la que se ejecutan sus tareas o servicios. Puede utilizar una estrategia de proveedor de capacidad o un tipo de lanzamiento.

En el caso de Fargate, los proveedores de capacidad son Fargate y Fargate Spot. En EC2, el proveedor de capacidad es el grupo de escalado automático con las instancias de contenedor registradas.

La estrategia del proveedor de capacidad distribuye las tareas entre los proveedores de capacidad asociados al clúster.

Solo los proveedores de capacidad que ya estén asociados a un clúster y tengan un estado `ACTIVE` o `UPDATING` se pueden utilizar en una estrategia de proveedores de capacidad. Puede asociar un proveedor de capacidad a un clúster al crear un clúster.

En una estrategia de proveedores de capacidad, el valor de base opcional designa cuántas tareas, como mínimo, se ejecutan en un proveedor de capacidad especificado. Solo un proveedor de capacidad en una estrategia de proveedor de capacidad puede tener una base definida.

El valor de peso designa el porcentaje relativo del número total de tareas lanzadas que utiliza el proveedor de capacidad especificado. Considere el siguiente ejemplo. Tiene una estrategia que contiene dos proveedores de capacidad y ambos tienen un peso de `1`. Cuando se alcanza el porcentaje base, las tareas se dividen en partes iguales entre los dos proveedores de capacidad. Con la misma lógica, suponga que especifica un peso de `1` para capacityProviderA y un peso de `4` para capacityProviderB. Luego, para cada tarea que se ejecute con capacityProviderA, cuatro tareas utilizan capacityProviderB.

El tipo de lanzamiento lanza las tareas directamente en Fargate o en las instancias de Amazon EC2 que haya registrado manualmente en los clústeres.

Ciclo de vida de las tareas de Amazon ECS

Cuando una tarea se inicia, ya sea manualmente o como parte de un servicio, puede pasar por diversos estados antes de que finalice por sí misma o se detenga manualmente. Algunas tareas están destinadas a ejecutarse como tareas por lotes que progresan de forma natural de `PENDING` a `RUNNING` a `STOPPED`. Otras tareas, que pueden formar parte de un servicio, están destinadas a seguir en ejecución indefinidamente o a ampliarse o reducirse en función de las necesidades.

Cuando se solicitan cambios de estado de tareas, por ejemplo, detener una tarea o actualizar el recuento deseado de un servicio para ampliarla o reducirla, el agente de contenedor de Amazon ECS realiza el seguimiento de estos cambios como el último estado conocido (`lastStatus`) y el estado deseado (`desiredStatus`) de la tarea. Tanto el último estado conocido como el estado deseado de una tarea se pueden ver en la consola o mediante la descripción de la tarea con la API o AWS CLI.

El siguiente diagrama de flujo muestra el flujo del ciclo de vida de la tarea.

Estados del ciclo de vida

A continuación se indican las descripciones de cada uno de los estados de ciclo de vida de la tarea.

PROVISIONING

Amazon ECS tiene que realizar pasos adicionales antes de que se lance la tarea. Por ejemplo, para las tareas que utilizan el modo de red `awsvpc`, la interfaz de red elástica se tiene que aprovisionar.

PENDIENTE

Se trata de un estado de transición en el que Amazon ECS está a la espera de que el agente de contenedor realice otras acciones. Una tarea permanecerá en estado pendiente hasta que haya recursos disponibles para la tarea.

ACTIVATING

Este es un estado de transición en el que Amazon ECS tiene que realizar pasos adicionales después de la tarea se haya lanzado, pero antes de que pueda pasar al estado `RUNNING`. Este es el estado en el que Amazon ECS extrae las imágenes de los contenedores, crea los contenedores, configura la red de tareas, registra los grupos de destino del equilibrador de carga y configura la detección de servicios.

RUNNING

La tarea se ejecuta correctamente.

DEACTIVATING

Este es un estado de transición en el que Amazon ECS tiene que realizar pasos adicionales antes de que la tarea se haya detenido. Por ejemplo, en el caso de tareas que forman parte de un servicio configurado para utilizar varios grupos de destino de Elastic Load Balancing, durante este estado se produce la anulación del registro del grupo de destino.

STOPPING

Se trata de un estado de transición en el que Amazon ECS está a la espera de que el agente de contenedor realice otras acciones.

Para los contenedores de Linux, el agente de contenedor enviará la señal `SIGTERM` para notificar que la aplicación debe finalizarse y apagarse y, a continuación, enviará `SIGKILL` después de esperar la duración de `StopTimeout` establecida en la definición de tarea.

DEPROVISIONING

Amazon ECS tiene que realizar pasos adicionales después de que la tarea se haya detenido, pero antes de que pase al estado `STOPPED`. Por ejemplo, para las tareas que utilizan el modo de red `awsvpc`, la interfaz de red elástica se tiene que desasociar y eliminar.

STOPPED

La tarea se ha detenido correctamente.

Si la tarea se detuvo debido a un error, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

DELETED

Se trata de un estado de transición en el que se detiene una tarea. Este estado no se muestra en la consola, pero se muestra en `describe-tasks`.

Cómo coloca Amazon ECS las tareas en las instancias de contenedor

Puede usar la ubicación de tareas para configurar Amazon ECS de manera que coloque sus tareas en instancias de contenedor que cumplan ciertos requisitos, por ejemplo, una zona de disponibilidad o un tipo de instancia.

Los siguientes son componentes de ubicación de tareas:

- Estrategia de ubicación de tareas: el algoritmo para seleccionar instancias de contenedor para ubicación de tareas o tareas para terminación. Por ejemplo, Amazon ECS puede seleccionar las instancias de contenedor de manera aleatoria o de modo que las tareas se distribuyan de forma uniforme entre un grupo de instancias.
- Grupo de tareas: grupo de tareas relacionadas, por ejemplo, tareas de bases de datos.
- Restricción de ubicación de tareas: se trata de reglas que se deben cumplir para colocar una tarea en una instancia de contenedor. Si no se cumple la restricción, la tarea no se desplegará y permanecerá en el estado `PENDING`. Por ejemplo, puede utilizar una restricción para colocar tareas únicamente en un tipo de instancia concreto.

Amazon ECS tiene diferentes algoritmos para los tipos de lanzamiento.

Tipo de lanzamiento de EC2

Para las tareas que utilizan el tipo de lanzamiento de EC2, Amazon ECS debe determinar dónde se colocará la tarea en función de los requisitos especificados en la definición de la tarea, como la CPU y la memoria. Del mismo modo, cuando se reduce la escala del número de tareas, Amazon ECS debe determinar qué tareas debe terminar. Puede aplicar estrategias y restricciones de ubicación de tareas para personalizar la manera en la que Amazon ECS ubica y termina las tareas.

Las estrategias de ubicación de tareas por defecto dependen de si ejecuta las tareas manualmente (tareas independientes) o dentro de un servicio. Para las tareas que se ejecutan como parte de un servicio de Amazon ECS, la estrategia de ubicación de tareas es `spread` mediante `attribute:ecs.availability-zone`. No existe una restricción de ubicación de tareas predeterminada para las tareas que no están en los servicios. Para obtener más información, consulte Programación de los contenedores en Amazon ECS.

> **Note**
>
> Las estrategias de ubicación de tareas se realizan en la medida de lo posible. Amazon ECS sigue intentando ubicar tareas, incluso cuando la opción de ubicación más adecuada no está disponible. Sin embargo, las restricciones de ubicación de tareas son vinculantes, y pueden impedir la ubicación de tareas.

Puede utilizar juntas estrategias y restricciones de ubicación de tareas. Por ejemplo, puede utilizar una estrategia de ubicación de tareas y una delimitación de ubicación de tareas para distribuir tareas entre las zonas de disponibilidad y agruparlas en bin packing en función de la memoria de cada zona de disponibilidad, pero únicamente si se trata de instancias G2.

Cuando Amazon ECS ubica las tareas, utiliza este proceso para seleccionar instancias de contenedor:

1. Identificar las instancias de contenedor que satisfacen los requisitos de CPU, GPU, memoria y puerto en la definición de tareas.
2. Identificar las instancias de contenedor que satisfacen las restricciones de ubicación de tareas.
3. Identificar las instancias de contenedor que satisfacen las estrategias de ubicación de tareas.
4. Seleccionar las instancias de contenedor para ubicación de tareas.

Tipo de lanzamiento de Fargate

No se admiten estrategias ni restricciones de ubicación de tareas para tareas que utilizan el tipo de lanzamiento de Fargate. Fargate hará todo lo posible para distribuir las tareas entre las zonas de disponibilidad accesibles. Si el proveedor de capacidad incluye Fargate y Fargate Spot, el comportamiento de distribución es independiente para cada proveedor de capacidad.

Uso de estrategias para definir la ubicación de las tareas de Amazon ECS

Para las tareas que utilizan el tipo de lanzamiento de EC2, Amazon ECS debe determinar dónde se colocará la tarea en función de los requisitos especificados en la definición de la tarea, como la CPU y la memoria. Del mismo modo, cuando se reduce la escala del número de tareas, Amazon ECS debe determinar qué tareas debe terminar. Puede aplicar estrategias y restricciones de ubicación de tareas para personalizar la manera en la que Amazon ECS ubica y termina las tareas.

Las estrategias de ubicación de tareas por defecto dependen de si ejecuta las tareas manualmente (tareas independientes) o dentro de un servicio. Para las tareas que se ejecutan como parte de un servicio de Amazon ECS, la estrategia de ubicación de tareas es `spread` mediante `attribute:ecs.availability-zone`. No existe una restricción de ubicación de tareas predeterminada para las tareas que no están en los servicios. Para obtener más información, consulte Programación de los contenedores en Amazon ECS.

> **Note**
>
> Las estrategias de ubicación de tareas se realizan en la medida de lo posible. Amazon ECS sigue intentando ubicar tareas, incluso cuando la opción de ubicación más adecuada no está disponible. Sin embargo, las restricciones de ubicación de tareas son vinculantes, y pueden impedir la ubicación de tareas.

Puede utilizar juntas estrategias y restricciones de ubicación de tareas. Por ejemplo, puede utilizar una estrategia de ubicación de tareas y una delimitación de ubicación de tareas para distribuir tareas entre las zonas de disponibilidad y agruparlas en bin packing en función de la memoria de cada zona de disponibilidad, pero únicamente si se trata de instancias G2.

Cuando Amazon ECS ubica las tareas, utiliza este proceso para seleccionar instancias de contenedor:

1. Identificar las instancias de contenedor que satisfacen los requisitos de CPU, GPU, memoria y puerto en la definición de tareas.
2. Identificar las instancias de contenedor que satisfacen las restricciones de ubicación de tareas.
3. Identificar las instancias de contenedor que satisfacen las estrategias de ubicación de tareas.
4. Seleccionar las instancias de contenedor para ubicación de tareas.

Las estrategias de ubicación de tareas se especifican en la definición del servicio o en la definición de la tarea mediante el parámetro `placementStrategy`.

```
"placementStrategy": [
    {
        "field": "The field to apply the placement strategy against",
        "type": "The placement strategy to use"
    }
]
```

Puede especificar las estrategias al ejecutar una tarea (RunTask), crear un nuevo servicio (CreateService) o actualizar un servicio existente (UpdateService).

En la tabla siguiente, se describen los tipos y campos disponibles.

type	Valores de campo válidos	
`binpack` Las tareas se colocan en instancias de contenedor para dejar la menor cantidad de CPU o memoria sin usar. Esta estrategia minimiza el número de instancias de contenedor en uso. Cuando se utiliza esta estrategia y se realiza una acción de reducción horizontal, Amazon ECS termina las tareas. Lo hace en función de la cantidad de recursos que quedan en la instancia del contenedor una vez terminada la tarea. La instancia de contenedor que tenga la mayor cantidad de recursos disponibles después de la	• cpu • memoria	

type	Valores de campo válidos	
terminación de la tarea hace que esa tarea termine.		
`random` Las tareas se colocan aleatoriamente.	No se utiliza	

type	Valores de campo válidos	
`spread` Las tareas se colocan uniformemente en función del valor especificado. Las tareas de servicio se distribuyen en función de las tareas de dicho servicio. Las tareas independientes se distribuyen en función de las tareas del mismo grupo de tareas. Para obtener más información acerca de los grupos de tareas, consulte Agrupación de tareas relacionadas con Amazon ECS . Cuando se utiliza la estrategia `spread` y se realiza una acción de reducción horizontal, Amazon ECS selecciona las tareas que mantienen un balance entre las zonas de disponibilidad para terminarlas. Dentro de una zona de disponibilidad, las tareas se seleccionan de manera aleatoria.	• `instanceId` (o `host`, que tiene el mismo efecto) • cualquier plataforma o atributo personalizado que se aplique a una instancia de contenedor, como `attribute:ecs.availability-zone`	

Las estrategias de colocación de tareas también se pueden actualizar para los servicios existentes. Para obtener más información, consulte Cómo coloca Amazon ECS las tareas en las instancias de contenedor.

Puede crear una estrategia de ubicación de tareas que utilice varias estrategias mediante la creación de conjuntos de estrategias en el orden en que desee que se realicen. Por ejemplo, si desea distribuir las tareas entre las zonas de disponibilidad y, a continuación, agrupar las tareas en función de la memoria de cada zona de disponibilidad, especifique la estrategia de la zona de disponibilidad seguida de la estrategia de memoria. Para ver ejemplos de estrategias, consulte Ejemplo de estrategias de ubicación de tareas de Amazon ECS.

Ejemplo de estrategias de ubicación de tareas de Amazon ECS

Puede especificar estrategias de ubicación de tarea con las acciones siguientes: CreateService, UpdateService y RunTask.

Ejemplos

- Distribuir las tareas de manera uniforme entre zonas de disponibilidad
- Distribuir las tareas de manera uniforme en todas las instancias
- Agrupar tareas en bin packing en función de la memoria
- Ubicar las tareas de forma aleatoria
- Distribuir las tareas de forma uniforme en las zonas de disponibilidad y, a continuación, distribuir las tareas de forma uniforme entre las instancias dentro de cada zona de disponibilidad
- Distribuir las tareas de forma uniforme en las zonas de disponibilidad y, a continuación, agrupar en bin packing las tareas en función de la memoria dentro de cada zona de disponibilidad
- Distribuir las tareas de manera uniforme entre las instancias y, a continuación, agrupar las tareas en bin packing según la memoria

Distribuir las tareas de manera uniforme entre zonas de disponibilidad

La estrategia siguiente distribuye las tareas de forma uniforme entre las zonas de disponibilidad.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    }
]
```

Distribuir las tareas de manera uniforme en todas las instancias

La estrategia siguiente distribuye las tareas de forma uniforme entre todas las instancias.

```
"placementStrategy": [
    {
        "field": "instanceId",
        "type": "spread"
    }
]
```

Agrupar tareas en bin packing en función de la memoria

La estrategia siguiente agrupa las tareas en bin packing en función de la memoria.

```
"placementStrategy": [
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Ubicar las tareas de forma aleatoria

La siguiente estrategia ubica las tareas aleatoriamente.

```
"placementStrategy": [
    {
        "type": "random"
    }
]
```

Distribuir las tareas de forma uniforme en las zonas de disponibilidad y, a continuación, distribuir las tareas de forma uniforme entre las instancias dentro de cada zona de disponibilidad

La siguiente estrategia distribuye las tareas de forma uniforme en las zonas de disponibilidad y, a continuación, distribuye las tareas de forma uniforme entre las instancias dentro de cada zona de disponibilidad.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
```

```
    {
        "field": "instanceId",
        "type": "spread"
    }
]
```

Distribuir las tareas de forma uniforme en las zonas de disponibilidad y, a continuación, agrupar en bin packing las tareas en función de la memoria dentro de cada zona de disponibilidad

La siguiente estrategia distribuye las tareas de forma uniforme en las zonas de disponibilidad y, a continuación, agrupa en bin packing las tareas en función de la memoria dentro de cada zona de disponibilidad.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Distribuir las tareas de manera uniforme entre las instancias y, a continuación, agrupar las tareas en bin packing según la memoria

La siguiente estrategia distribuye las tareas de manera uniforme en todas las instancias y, a continuación, agrupa las tareas en bin packing en función de la memoria de cada instancia.

```
"placementStrategy": [
    {
        "field": "instanceId",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Agrupación de tareas relacionadas con Amazon ECS

Puede identificar un conjunto de tareas relacionadas y colocarlas en un grupo de tareas. Todas las tareas con el mismo nombre de grupo de tareas se consideran un conjunto cuando se utiliza la estrategia de ubicación de tareas `spread`. Por ejemplo, suponga que está ejecutando distintas aplicaciones en un clúster, tales como bases de datos y servidores web. Para asegurarse de que las bases de datos estén balanceadas en las zonas de disponibilidad, agréguelas a un grupo de tareas con el nombre `databases` y, a continuación, utilice la estrategia de ubicación de tareas `spread`. Para obtener más información, consulte Uso de estrategias para definir la ubicación de las tareas de Amazon ECS.

Los grupos de tareas también se pueden utilizar como restricción de ubicación de tareas. Al especificar un grupo de tareas en la restricción `memberOf`, las tareas solo se envían a instancias de contenedores que se encuentran en el grupo de tareas especificado. Para ver un ejemplo, consulta Ejemplos de restricciones para ubicación de tareas de Amazon ECS.

De forma predeterminada, las tareas independientes utilizan el nombre de familia de definición de tareas (por ejemplo, `family:my-task-definition`) como nombre del grupo de tareas si no se especifica un nombre de grupo de tareas personalizado. Las tareas lanzadas como parte de un servicio utilizan el nombre del servicio como nombre del grupo de tareas y no se pueden cambiar.

Se aplican los siguientes requisitos para el grupo de tareas.

- Un nombre de grupo de tareas debe tener 255 caracteres o menos.
- Cada tarea puede estar exactamente en un grupo.
- Después de lanzar una tarea, no puede modificar su grupo de tarea.

Definición de las instancias de contenedor que utiliza Amazon ECS para las tareas

Una restricción de ubicación de tareas es una regla sobre una instancia de contenedor que Amazon ECS utiliza para determinar si la tarea puede ejecutarse en la instancia. Al menos una instancia de contenedor debe cumplir con la restricción. Si no hay instancias que coincidan con la restricción, la tarea permanece en un estado `PENDING`. Cuando crea un servicio nuevo o actualiza uno existente, puede especificar las restricciones de ubicación de tareas para las tareas del servicio.

Puede especificar las restricciones de ubicación de tareas en la definición del servicio, la definición de la tarea o la tarea mediante el parámetro `placementConstraint`.

```
"placementConstraints": [
    {
        "expression": "The expression that defines the task placement constraints",
        "type": "The placement constraint type to use"
    }
]
```

En la tabla siguiente, se describe cómo usar los parámetros.

Constraint type (Tipo de restricción)	Se puede especificar cuándo	
`distinctInstance` Coloque cada tarea activa en una instancia de contenedor distinta. Amazon ECS analiza el estado deseado de las tareas para su colocación. Por ejemplo, si el estado deseado de la tarea existente es `STOPPED` (pero el último estado no lo es), se puede colocar una nueva tarea entrante en la misma instancia a pesar de la restricción de ubicación de `distinctInstance` . Por lo tanto, es posible que vea 2 tareas con el último estado de `RUNNING` en la misma instancia. **Important** Recomendamos que los clientes que	• Ejecución de una tarea RunTask • Creación de un nuevo servicio CreateService,	

Constraint type (Tipo de restricción)	Se puede especificar cuándo	
buscan un aislamiento sólido para sus tareas utilicen Fargate. Fargate ejecuta cada tarea en un entorno de virtualización de hardware. Esto garantiza que estas cargas de trabajo en contenedores no compartan interfaces de red, almacenamiento efímero de Fargate, CPU o memoria con otras tareas. Para obtener más información, consulte Security Overview of AWS Fargate.		

Constraint type (Tipo de restricción)	Se puede especificar cuándo	
`memberOf` Colocar tareas en instancias de contenedor que satisfacen una expresión.	• Ejecución de una tarea RunTask • Creación de un nuevo servicio CreateService, • Creación de una nueva definición de tarea RegisterTaskDefinition • Creación de una nueva revisión de una definición de tarea RegisterTaskDefinition • Actualización de un servicio UpdateService	

Cuando utiliza el tipo de restricción `memberOf`, puede crear una expresión mediante el lenguaje de consulta de clústeres que define las instancias de contenedor en las que Amazon ECS puede colocar tareas. La expresión es una forma de agrupar las instancias de contenedor por atributos. La expresión se incluye en el `expression` parámetro de `placementConstraint`.

Atributos de instancias de contenedor de Amazon ECS

Puede añadir metadatos personalizados a sus instancias de contenedor, conocidas como atributos. Cada atributo tiene un nombre y un valor de cadena opcional. Puede utilizar los atributos integrados que ofrece Amazon ECS o definir atributos personalizados.

Las secciones siguientes contienen ejemplos de atributos integrados, opcionales y personalizados.

Atributos integrados

Amazon ECS aplica automáticamente los siguientes atributos a las instancias de contenedor.

`ecs.ami-id`

El ID de la AMI utilizada para iniciar la instancia. Un valor de ejemplo para este atributo es `ami-1234abcd`.

`ecs.availability-zone`

La zona de disponibilidad de la instancia. Un valor de ejemplo para este atributo es `us-east-1a`.

`ecs.instance-type`

El tipo de instancia de la instancia. Un valor de ejemplo para este atributo es `g2.2xlarge`.

`ecs.os-type`

El sistema operativo de la instancia. Los valores posibles para este atributo son `linux` y `windows`.

`ecs.os-family`

La versión del sistema operativo de la instancia.

Para las instancias de Linux, el valor válido es `LINUX`. Para las instancias de Windows, ECS establece el valor en el formato `WINDOWS_SERVER_<OS_Release>_<FULL or CORE>`. Los valores válidos son `WINDOWS_SERVER_2022_FULL`, `WINDOWS_SERVER_2022_CORE`, `WINDOWS_SERVER_20H2_CORE`, `WINDOWS_SERVER_2019_FULL`, `WINDOWS_SERVER_2019_CORE` y `WINDOWS_SERVER_2016_FULL`.

Esto es importante para los contenedores de Windows y Windows containers on AWS Fargate porque la versión del sistema operativo de cada contenedor de Windows debe coincidir con la del host. Si la versión de Windows de la imagen del contenedor es diferente a la del host, el contenedor no se inicia. Para obtener más información, consulte Compatibilidad de versiones de contenedores Windows en el sitio web de documentación de Microsoft.

Si el clúster ejecuta varias versiones de Windows, puede asegurarse de que la tarea se coloque en una instancia de EC2 que se ejecute en la misma versión mediante la restricción de ubicación: `memberOf(attribute:ecs.os-family == WINDOWS_SERVER_<OS_Release>_<FULL or CORE>)`. Para obtener más información, consulte the section called “Recuperación de metadatos de las AMI de Windows optimizadas para Amazon ECS”.

`ecs.cpu-architecture`

La arquitectura de CPU de la instancia. Los valores de ejemplo para este atributo son `x86_64` y `arm64`.

`ecs.vpc-id`

La VPC en la que se lanzó la instancia. Un valor de ejemplo para este atributo es `vpc-1234abcd`.

`ecs.subnet-id`

La subred que está utilizando la instancia. Un valor de ejemplo para este atributo es `subnet-1234abcd`.

Atributos opcionales

Amazon ECS puede agregar los siguientes atributos a las instancias de contenedor.

`ecs.awsvpc-trunk-id`

Si este atributo existe, la instancia tiene una interfaz de red troncal. Para obtener más información, consulte Aumento de las interfaces de red de instancias de contenedor de Linux de Amazon ECS.

`ecs.outpost-arn`

Si este atributo existe, contiene el nombre de recurso de Amazon (ARN) del Outpost. Para obtener más información, consulte the section called “Amazon Elastic Container Service en AWS Outposts”.

`ecs.capability.external`

Si este atributo existe, la instancia se identifica como instancia externa. Para obtener más información, consulte Clústeres de Amazon ECS para el tipo de lanzamiento externo.

Custom attributes (Atributos personalizados)

Puede aplicar atributos personalizados a sus instancias de contenedor. Por ejemplo, puede definir un atributo con el nombre "stack" y un valor "prod".

Al especificar atributos personalizados, se deben tener en cuenta los siguientes aspectos.

- El `name` debe contener entre 1 y 128 caracteres, que pueden ser letras (mayúsculas y minúsculas), números, guiones, guiones bajos, barras diagonales, barras invertidas o puntos.
- El `value` debe contener entre 1 y 128 caracteres, que pueden ser letras (mayúsculas y minúsculas), números, guiones, guiones bajos, puntos, signos de arroba (@), barras diagonales, barras invertidas, dos puntos o espacios. El valor no puede contener ningún espacio en blanco inicial ni final.

Creación de expresiones para definir instancias de contenedor para las tareas de Amazon ECS

Las consultas de clúster son expresiones que permiten agrupar objetos. Por ejemplo, puede agrupar instancias de contenedor por atributos tales como zona de disponibilidad, tipo de instancia o metadatos personalizados. Para obtener más información, consulte Atributos de instancias de contenedor de Amazon ECS.

Después de haber definido un grupo de instancias de contenedor, puede personalizar Amazon ECS para que ubique tareas en instancias de contenedor basadas en grupo. Para obtener más información, consulte Ejecución de una aplicación como tarea de Amazon ECS y Creación de una implementación de actualización continua de Amazon ECS. También puede aplicar un filtro de grupo al mostrar una lista de instancias de contenedor.

Sintaxis de expresiones

Las expresiones tienen la siguiente sintaxis:

```
subject operator [argument]
```

Asunto

El atributo o campo que se va a evaluar.

`agentConnected`

Seleccione instancias de contenedor por el estado de conexión del agente de contenedor de Amazon ECS. Puede utilizar este filtro para buscar las instancias cuyos agentes de contenedor están desconectados.

Los operadores válidos son: equals (==), not_equals (!=), in, not_in (!in), matches (=~), not_matches (!~)

`agentVersion`

Seleccione instancias de contenedor según la versión del agente de contenedor de Amazon ECS. Puede utilizar este filtro para buscar las instancias que ejecutan versiones obsoletas del agente de contenedor de Amazon ECS.

Los operadores válidos son: equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

`attribute:attribute-name`

Selecciona instancias de contenedor según el atributo. Para obtener más información, consulte Atributos de instancias de contenedor de Amazon ECS.

`ec2InstanceId`

Seleccione instancias de contenedor por el ID de las instancias de Amazon EC2.

Los operadores válidos son: equals (==), not_equals (!=), in, not_in (!in), matches (=~), not_matches (!~)

`registeredAt`

Selecciona instancias de contenedor por la fecha de registro de la instancia de contenedor. Puede utilizar este filtro para encontrar las instancias que se acaban de registrar o las instancias que son muy antiguas.

Los operadores válidos son: equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

Los formatos de fecha válidos son: 2018-06-18T22:28:28+00:00, 2018-06-18T22:28:28Z, 2018-06-18T22:28:28, 2018-06-18

`runningTasksCount`

Selecciona instancias de contenedor según el número de tareas en ejecución. Puede utilizar este filtro para encontrar las instancias que estén vacías o casi vacías (con pocas tareas en ejecución).

Los operadores válidos son: equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

`task:group`

Selecciona instancias de contenedor según el grupo de tareas. Para obtener más información, consulte Agrupación de tareas relacionadas con Amazon ECS .

Operador

El operador de comparación. Se admiten los siguientes operadores.

Operador	Descripción
==, equals	Igualdad de cadena
!=, not_equals	Desigualdad de cadena
>, greater_than	Mayor que
>=, greater_than_equal	Mayor o igual que
<, less_than	Menor que
<=, less_than_equal	Menor o igual que
exists	El asunto existe
!exists, not_exists	El asunto no existe
in	El valor está en la lista de argumentos
!in, not_in	El valor no está en la lista de argumentos
=~, matches	Coincidencia de patrón
!~, not_matches	No hay coincidencia de patrón

> **Note**
>
> Una expresión única no puede contener paréntesis. Sin embargo, se pueden utilizar paréntesis para especificar la prioridad en las expresiones compuestas.

Argumento

Para muchos operadores, el argumento es un valor literal.

Los operadores `in` y `not_in` esperan una lista de argumentos como argumento. Una lista de argumentos se especifica del siguiente modo:

```
[argument1, argument2, ..., argumentN]
```

Los operadores matches y not_matches esperan un argumento que se ajuste a la sintaxis de la expresión regular Java. Para obtener más información, consulte java.util.regex.Pattern.

Expresiones compuestas

Puede combinar expresiones utilizando los siguientes operadores booleanos:

- &&, y
- ||, o bien
- !, not

Puede especificar prioridad utilizando paréntesis:

```
(expression1 or expression2) and expression3
```

Expresiones de ejemplo

A continuación se incluyen expresiones de ejemplo.

Ejemplo: igualdad de cadena

La expresión siguiente selecciona las instancias con el tipo de instancia especificado.

```
attribute:ecs.instance-type == t2.small
```

Ejemplo: lista de argumentos

La expresión siguiente selecciona las instancias en la zona de disponibilidad us-east-1a o us-east-1b.

```
attribute:ecs.availability-zone in [us-east-1a, us-east-1b]
```

Ejemplo: expresiones compuestas

La siguiente expresión selecciona las instancias G2 que no están en la zona de disponibilidad us-east-1d.

```
attribute:ecs.instance-type =~ g2.* and attribute:ecs.availability-zone != us-east-1d
```

Ejemplo: afinidad de tareas

La expresión siguiente selecciona las instancias que alojan tareas en el grupo `service:production`.

```
task:group == service:production
```

Ejemplo: antiafinidad de tareas

La siguiente expresión selecciona las instancias que no alojan tareas en el grupo de base de datos.

```
not(task:group == database)
```

Ejemplo: Recuento de tareas en ejecución

La expresión siguiente selecciona las instancias que solo están ejecutando una tarea.

```
runningTasksCount == 1
```

Ejemplo: versión del agente de contenedor de Amazon ECS

La expresión siguiente selecciona las instancias que están ejecutando una versión del agente de contenedor anterior a la 1.14.5.

```
agentVersion < 1.14.5
```

Ejemplo: Hora de registro de las instancias

La expresión siguiente selecciona las instancias que se han registrado antes del 13 de febrero de 2018.

```
registeredAt < 2018-02-13
```

Ejemplo: ID de instancias de Amazon EC2

Esta expresión selecciona las instancias con los siguientes ID de instancias de Amazon EC2.

```
ec2InstanceId in ['i-abcd1234', 'i-wxyx7890']
```

Ejemplos de restricciones para ubicación de tareas de Amazon ECS

A continuación, se muestran ejemplos de restricción de ubicación de tareas.

En este ejemplo, se utiliza la restricción `memberOf` para colocar tareas en instancias t2. Se puede especificar con las acciones siguientes: CreateService, UpdateService, RegisterTaskDefinition y RunTask.

```
"placementConstraints": [
    {
        "expression": "attribute:ecs.instance-type =~ t2.*",
        "type": "memberOf"
    }
]
```

En el ejemplo, se utiliza la restricción `memberOf` para ubicar tareas de réplica en instancias con otras tareas en el grupo de tareas `daemon-service` del servicio de daemon, respetando las estrategias de ubicación de tareas que también se especifican. Esta restricción garantiza que las tareas del servicio de daemon se coloquen en la instancia de EC2 antes que las tareas del servicio de réplica.

Sustituya `daemon-service` por el nombre del servicio de daemon.

```
"placementConstraints": [
    {
        "expression": "task:group == service:daemon-service",
        "type": "memberOf"
    }
]
```

En el ejemplo, se utiliza la restricción `memberOf` para ubicar tareas en instancias con otras tareas en el grupo de tareas `databases`, respetando las estrategias de ubicación de tareas que también se especifiquen. Para obtener más información acerca de los grupos de tareas, consulte Agrupación de tareas relacionadas con Amazon ECS . Se puede especificar con las acciones siguientes: CreateService, UpdateService, RegisterTaskDefinition y RunTask.

```
"placementConstraints": [
    {
        "expression": "task:group == databases",
        "type": "memberOf"
    }
]
```

La restricción `distinctInstance` ubica cada tarea del grupo en una instancia diferente. Se puede especificar mediante las siguientes acciones: CreateService, UpdateService y RunTask

Amazon ECS analiza el estado deseado de las tareas para su colocación. Por ejemplo, si el estado deseado de la tarea existente es `STOPPED` (pero el último estado no lo es), se puede colocar una nueva tarea entrante en la misma instancia a pesar de la restricción de ubicación de `distinctInstance`. Por lo tanto, es posible que vea 2 tareas con el último estado de `RUNNING` en la misma instancia.

```
"placementConstraints": [
    {
        "type": "distinctInstance"
    }
]
```

Tareas independientes de Amazon ECS

Puede ejecutar la aplicación como una tarea si tiene una aplicación que hace algún trabajo y, después, se detiene (por ejemplo, un proceso por lotes). Si quiere ejecutar una tarea una vez, puede usar la consola, la AWS CLI, las API o los SDK.

Si necesita ejecutar su aplicación en un programa basado en tasa, cron o único, puede crear una programación con el Programador de EventBridge.

Flujo de trabajo de tareas

Al lanzar tareas de Amazon ECS (tareas independientes o mediante los servicios de Amazon ECS), se crea una tarea e inicialmente se traslada al estado `PROVISIONING`. Cuando el estado de una tarea es `PROVISIONING`, ni la tarea ni los contenedores existen porque Amazon ECS tiene que encontrar la capacidad de computación para llevar a cabo la tarea.

Amazon ECS selecciona la capacidad de computación adecuada para su tarea en función del tipo de lanzamiento o de la configuración del proveedor de capacidad. Puede utilizar proveedores de capacidad y estrategias de proveedores de capacidad con los tipos de lanzamiento de Fargate y Amazon EC2. Con Fargate, no tiene que preocuparse por aprovisionar, configurar y escalar la capacidad de su clúster. Fargate se encarga de toda la gestión de la infraestructura para sus tareas. Para el tipo de lanzamiento de EC2, puede administrar la capacidad de su clúster registrando las instancias de Amazon EC2 en su clúster, o puede usar el escalado automático del clúster para simplificar la administración de la capacidad de cómputo. El escalado automático del clúster se encarga de escalar dinámicamente la capacidad del clúster para que pueda concentrarse en ejecutar las tareas. Amazon ECS determina dónde colocar la tarea en función de los requisitos que

especifique en la definición de tarea, como la CPU y la memoria, así como de sus restricciones y estrategias de ubicación. Para obtener más información, consulte Cómo coloca Amazon ECS las tareas en las instancias de contenedor.

Si utiliza un proveedor de capacidad con el escalado administrado habilitado, las tareas que no se puedan iniciar por falta de capacidad de computación se trasladarán al estado `PROVISIONING` en lugar de generar un error de inmediato. Tras encontrar la capacidad para colocar la tarea, Amazon ECS aprovisiona los adjuntos necesarios (por ejemplo, interfaces de red elásticas [ENI] para las tareas en modo `awsvpc`). Utiliza el agente de contenedor de Amazon ECS para extraer las imágenes de los contenedores y, a continuación, iniciar los contenedores. Una vez finalizado el aprovisionamiento y lanzados los contenedores correspondientes, Amazon ECS pasa la tarea al estado `RUNNING`. Para obtener más información sobre los estados de las tareas, consulte Ciclo de vida de las tareas de Amazon ECS.

Ejecución de una aplicación como tarea de Amazon ECS

Puede crear una tarea para un proceso único mediante la AWS Management Console.

Para crear una tarea independiente (AWS Management Console)

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. La consola de Amazon ECS le permite crear una tarea independiente desde la página de detalles del clúster o desde la lista de revisiones de definiciones de tareas. Siga estos pasos para crear una tarea independiente según la página de recursos que elija.

Para iniciar un servicio desde	Pasos
La página de detalles del clúster	a. En la página Clusters (Clústeres), seleccione el clúster que va a crear el servicio. b. En la pestaña Tareas, elija Ejecutar tarea.
La página de revisión de definiciones de tareas	a. En la página de Definiciones de tareas, elija la familia de definiciones de

Para iniciar un servicio desde	Pasos	
	tareas para mostrar las revisiones de esa familia. b. Seleccione la revisión que desee utilizar. c. En el menú Implementar, elija Ejecutar tarea.	

3. En Clúster existente, elija el clúster.

   Elija Crear clúster para ejecutar la tarea en un clúster nuevo

4. Elija cómo distribuir las tareas en la infraestructura de clúster. En Configuración de cómputos, elija su opción. Para utilizar una estrategia de proveedor de capacidad, debe configurar sus proveedores de capacidad por clúster.

   Si no ha configurado el clúster para usar un proveedor de capacidad, utilice un tipo de lanzamiento en su lugar.

Método de distribución	Pasos	
Estrategia de proveedores de capacidad	a. En la sección Compute options (Opciones de computación), seleccione Capacity provider strategy (Estrategia de proveedor es de capacidad). b. Elija una estrategia: • Para utilizar la estrategi a de proveedores de capacidad predeterm inada del clúster, elija Use cluster default (Usar clúster predeterm inado).	

Método de distribución	Pasos	
	• Si el clúster no tiene ninguna estrategia de proveedores de capacidad predeterminada o si desea utilizar una estrategia personalizada, elija Use custom (Utilizar predeterminada), Add capacity provider strategy (Agregar estrategia de proveedores de capacidad) y, para definir la estrategia de proveedores de capacidad personalizada, complete los campos Base (Base), Capacity provider (Proveedor de capacidad) y Weight (Peso). **Note** Para utilizar un proveedor de capacidad en una estrategia, el proveedor de capacidad debe estar asociado con el clúster.	

Método de distribución	Pasos	
Tipo de lanzamiento	a. En la sección Compute options (Opciones de computación), seleccion e Lauch type (Tipo de lanzamiento). b. En Launch type (Tipo de lanzamiento), elija un tipo de lanzamiento. c. (Opcional) Cuando se especifica el tipo de lanzamiento de Fargate, en Platform version (Versión de la plataform a) especifique la versión de la plataforma que se va a utilizar. Si no se especifica una versión de la plataforma, se utilizará la versión `LATEST`.	

5. En Configuración de implementación, haga lo siguiente:

   a. En Definición de tarea, ingrese la definición de la tarea.

   > **Important**
   >
   > La consola valida la selección para asegurarse de que la familia y la revisión de definiciones de tareas seleccionadas sean compatibles con la configuración de cómputos definida.

   b. En Desired tasks (Tareas deseadas), ingrese el número de tareas que se lanzarán.

   c. En Grupo de tareas, ingrese el nombre del grupo de tareas.

6. Si la definición de su tarea utiliza el modo de red de `awsvpc`, expanda la opción de Networking (Red). Siga estos pasos para especificar una configuración personalizada.

a. En VPC, seleccione la VPC que se va a usar.

b. En Subnets (Subredes), seleccione una o varias subredes de la VPC que el programador de tareas considera al ubicar sus tareas.

c. En Grupos de seguridad, puede seleccionar un grupo de seguridad existente o crear uno nuevo. Para utilizar un grupo de seguridad existente, seleccione el grupo de seguridad y continúe con el próximo paso. Para crear un grupo de seguridad, elija Create a new security group (Crear un grupo de seguridad nuevo). Debe especificar un nombre de grupo de seguridad, una descripción y, a continuación, agregar una o varias reglas de entrada para el grupo de seguridad.

d. En Public IP (IP pública), elija si desea asignar automáticamente una dirección IP pública a la interfaz de red elástica (ENI) de la tarea.

   Las tareas de AWS Fargate pueden recibir una dirección IP pública cuando se ejecuten mediante una subred pública para que tengan una ruta a Internet. A las tareas de EC2 no se les puede asignar una IP pública mediante este campo. Para obtener más información, consulte Opciones de redes de tareas de Amazon ECS para el tipo de lanzamiento de Fargate y Asignar una interfaz de red para una tarea de Amazon ECS.

7. Si su tarea usa un volumen de datos compatible con la configuración en el momento de la implementación, puede expandir Volume para configurar el volumen.

   El nombre y el tipo de volumen se configuran al crear una revisión de la definición de la tarea y no se pueden cambiar cuando se ejecuta una tarea independiente. Para actualizar el nombre y el tipo del volumen, debe crear una nueva revisión de la definición de tareas y ejecutar una tarea con la nueva revisión.

Para configurar este tipo de volumen	Haga lo siguiente	
Amazon EBS	a. En Tipo de volumen de EBS, elija el tipo de volumen de EBS que desee adjuntar a la tarea. b. En Tamaño (GiB), ingrese un valor válido para el tamaño de volumen en	

Para configurar este tipo de volumen	Haga lo siguiente	
	gibibytes (GiB). Puede especificar un tamaño de volumen mínimo de 1 GiB y máximo de 16 384 GiB. Este valor es obligatorio a menos que proporcione un ID de instantánea. c. En IOPS, ingrese el número máximo de operaciones de entrada/salida (IOPS) que debe proporcionar el volumen. Este valor solo puede configurarse para los tipos de volumen `io1`, `io2` y `gp3`. d. En Rendimiento (MiB/s), ingrese el rendimiento que debe proporcionar el volumen, en mebibytes por segundo (MiBps o MiB/s). Este valor solo puede configurarse para el tipo de volumen `gp3`. e. En ID de instantánea, elija una instantánea de volumen de Amazon EBS existente o ingrese el ARN de una instantánea si desea crear un volumen a partir de una instantánea. También puede crear un volumen nuevo y vacío sin	

Para configurar este tipo de volumen	Haga lo siguiente	
	elegir ni ingresar ningún ID de instantánea. f. Si especifica un ID de instantánea, puede especificar una velocidad de inicialización del volumen (MiB/s). Ingrese un valor entre 100 y 300, en MiB/s, que determina rá la rapidez con la que se cargan los datos de la instantánea especific ada mediante el ID de instantánea para la creación del volumen. g. En Política de terminaci ón, desactive la casilla de verificación si desea que el volumen configura do para adjuntarse a la tarea se conserve una vez finalizada la tarea. De manera predeterminada, los volúmenes de EBS asociados a las tareas se eliminan al finalizar la tarea. h. En Tipo de sistema de archivos, elija el tipo de sistema de archivos que se utilizará para almacenar y recuperar datos en el volumen.	

Para configurar este tipo de volumen	Haga lo siguiente	
	Puede elegir el sistema operativo predeterminado o un tipo de sistema de archivos específico. El valor predeterminado para Linux es `XFS`. En el caso de los volúmenes creados a partir de una instantán ea, debe especificar el mismo tipo de sistema de archivos que utilizaba el volumen cuando se creó la instantánea. Si hay un error de coincidencia con el tipo de sistema de archivos, la tarea no podrá iniciarse. i. En Rol de infraestructura, elija un rol de IAM con los permisos necesarios que permitan a Amazon ECS administrar los volúmenes de Amazon EBS para las tareas. Puede adjuntar la política de `AmazonECSInfrastructureRolePolicyForVolumes` administrada al rol, o puede utilizar la política como guía para crear y adjuntar su propia política con los permisos que cumplan sus necesidades	

Para configurar este tipo de volumen	Haga lo siguiente	
	específicas. Para obtener información sobre los permisos necesarios, consulte Rol de IAM de infraestructura de Amazon ECS. j. En Cifrado, elija Predeterminado si quiere usar el cifrado de Amazon EBS como configuración predeterminada. Si su cuenta tiene configurado Cifrado predeterminado, el volumen se cifrará con la clave AWS Key Management Service (AWS KMS) especific ada en la configura ción. Si selecciona Predeterminado y el cifrado predeterminado de Amazon EBS no está activado, el volumen se descifrará. Si elige Personalizado, puede especificar una AWS KMS key de su preferencia para el cifrado por volumen. Si selecciona Ninguno, el volumen no se cifrará a menos que tenga el	

Para configurar este tipo de volumen	Haga lo siguiente	
	cifrado configurado de forma predeterminada o si crea un volumen a partir de una instantánea cifrada. k. Si ha elegido Personalizado en Cifrado, debe especificar la AWS KMS key que desee utilizar. En Clave de KMS, elija una AWS KMS key o escriba un ARN. Si decide cifrar su volumen mediante una clave simétrica administrada por el cliente, asegúrese de tener los permisos correctos definidos en su política de AWS KMS key. Para obtener más información, consulte Data encryption for Amazon EBS volumes. l. (Opcional) En Etiquetas, puede propagar las etiquetas de la definición de la tarea o proporcionar sus propias etiquetas para agregar etiquetas a su volumen de Amazon EBS. Si desea propagar etiquetas desde la	

Para configurar este tipo de volumen	Haga lo siguiente	
	definición de la tarea, seleccione Definición de tarea en Propagar etiquetas desde. Si elige No propagar o si no elige un valor, las etiquetas no se propagarán. Si quiere proporcionar sus propias etiquetas , seleccione Agregar etiqueta y, a continuación, proporcione la clave y el valor de cada etiqueta que agregue. Para obtener más información acerca del etiquetado de volúmenes de Amazon EBS, consulte Tagging Amazon EBS volumes.	

8. (Opcional) Para utilizar una estrategia de ubicación de tareas distinta a la predeterminada, expanda Task Placement (Ubicación de tareas) y, a continuación, elija una de las siguientes opciones.

   Para obtener más información, consulte Cómo coloca Amazon ECS las tareas en las instancias de contenedor.

   - Reparto equilibrado en AZ: distribuya las tareas en las zonas de disponibilidad y entre las instancias de contenedor dentro de cada zona de disponibilidad.
   - BinPack equilibrado en AZ: distribuya las tareas en las zonas de disponibilidad y entre las instancias de contenedor con la menor memoria disponible.
   - BinPack: distribuya las tareas en función de la cantidad mínima de CPU o memoria disponible.

- Una tarea por Host: coloque como máximo una tarea del servicio en cada instancia de contenedor.
- Personalizado: defina su propia estrategia de colocación de tareas.

Si elige Custom (Personalizado), defina el algoritmo de ubicación de tareas y las reglas que se tienen en cuenta durante la ubicación de tareas.

- En Strategy (Estrategia), para Type (Tipo) y Field (Campo), elija el algoritmo y la entidad que quiere utilizar para el algoritmo.

  Puede ingresar un máximo de 5 estrategias.
- En Restricción, para Tipo y Expresión, elija la regla y el atributo para la restricción.

  Por ejemplo, para establecer la restricción de colocar las tareas en las instancias T2, para la Expresión, ingrese attribute:ecs.instance-type =~ t2.*.

  Puede ingresar un máximo de 10 restricciones.

9. (Opcional) Para anular el rol de IAM de la tarea, o el rol de ejecución de la tarea que está definido en su definición de la tarea, expanda Task overrides (Anulaciones de tareas) y, a continuación, complete los siguientes pasos:

   a. En Rol de tarea, elija un rol de IAM para esta tarea. Para obtener más información, consulte Rol de IAM de tarea de Amazon ECS.

      Solo se muestran os roles con la relación de confianza `ecs-tasks.amazonaws.com`. Para obtener instrucciones sobre cómo crear un rol de IAM para las tareas, consulte Creación del rol de IAM de tareas.

   b. En Rol de ejecución de tareas, elija un rol de ejecución de tareas. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

10. (Opcional) Para anular los comandos del contenedor y las variables de entorno, expanda Container Overrides (Anulaciones de contenedores) y, a continuación, expanda el contenedor.

    - Para enviar un comando al contenedor que no sea el comando de definición de tareas, en Anulación de comando, ingrese el comando de Docker.
    - Para agregar una variable de entorno, elija Add environment variable (Agregar variable de entorno). En Key (Clave), ingrese el nombre de la variable de entorno. En Value (Valor), ingrese un valor de cadena el valor de entorno (sin las comillas dobles [" "]).

AWS rodea las cadenas con comillas dobles (" ") y pasa la cadena al contenedor en el formato siguiente:

```
MY_ENV_VAR="This variable contains a string."
```

11. (Opcional) Para ayudar a identificar la tarea, expanda la sección Tags (Etiquetas) y, a continuación, configure sus etiquetas.

    Para que Amazon ECS etiquete automáticamente todas las tareas recién lanzadas con el nombre del clúster y las etiquetas de definición de tareas, seleccione Turn on Amazon ECS managed tags (Activar las etiquetas gestionadas de Amazon ECS) y, a continuación, seleccione Task definitions (Definiciones de tareas).

    Añada o elimine una etiqueta.

    - [Agregar una etiqueta] Seleccione Add tag (Agregar etiqueta), y, a continuación, haga lo siguiente:
      - En Clave, escriba el nombre de la clave.
      - En Valor, escriba el valor de la clave.
    - [Eliminar una etiqueta] Junto a la etiqueta, elija Remove tag (Quitar etiqueta).
12. Seleccione Crear.

Uso de Programador de Amazon EventBridge para programar tareas de Amazon ECS

El Programador de Amazon EventBridge es un programador sin servidor que le permite crear, ejecutar y administrar tareas desde un servicio administrado y centralizado. Proporciona una funcionalidad de programación única y recurrente, independientemente de las reglas y los buses de eventos. El programador de EventBridge es altamente personalizable y ofrece una escalabilidad mejorada en comparación con las reglas programadas de EventBridge, con un conjunto más amplio de operaciones de API y servicios de AWS de destino. El programador de EventBridge proporciona los siguientes programas que puede configurar para sus tareas en la consola del programador de EventBridge:

- Basada en frecuencia
- Basado en cron

Puede configurar programas basados en cron en cualquier zona horaria.

- Programas únicos

  Puede configurar programas únicos en cualquier zona horaria.

Puede programar su Amazon ECS mediante el Programador de Amazon EventBridge.

Aunque puede crear una tarea programada en la consola de Amazon ECS, actualmente la consola del programador de EventBridge proporciona más funcionalidad.

Lleve a cabo los pasos siguientes antes de programar una tarea:

1. Utilice la consola de VPC para obtener los ID de subred en los que se ejecutan las tareas y los ID de los grupos de seguridad de las subredes. Para obtener más información, consulte Subredes para la VPC y Controlar el tráfico hacia los recursos de AWS mediante grupos de seguridad en la Guía del usuario de Amazon VPC.
2. Configure el rol de ejecución del programador de EventBridge. Para obtener más información, consulte Configurar el rol de ejecución en la Guía del usuario del programador de Amazon EventBridge.
3. Para utilizar una estrategia de proveedor de capacidad, el proveedor de capacidad debe estar asociado con el clúster.

Para crear un programa nuevo con la consola

1. Abra la consola del Programador de Amazon EventBridge enhttps://console.aws.amazon.com/scheduler/home.
2. En la página Programaciones, elija Crear programación.
3. En la página Especificar los detalles de la programación, en la sección Nombre y descripción de la programación, proceda del modo siguiente:

   a. En Nombre de la programación, escriba un nombre para la programación. Por ejemplo, **`MyTestSchedule`**.

   b. (Opcional) En Descripción, escriba una descripción para su programación. Por ejemplo, **`TestSchedule`**.

c. En Grupo de programaciones, elija un grupo de programaciones. Si no tiene un grupo, elija predeterminado. Para crear un grupo de programaciones, elija crear mi propia programación.

   Los grupos de programaciones se utilizan para añadir etiquetas a grupos de programaciones.

4. Elija sus opciones de programación.

Ocurrencia	Haga lo siguiente...
Programación única Una programación única invoca solo una vez un destino en la fecha y hora que especifique.	En Fecha y hora, realice lo siguiente: • Introduzca una fecha válida con el formato `YYYY/MM/DD` . • Introduzca una marca de tiempo con el formato `hh:mm` de 24 horas. • En Zona horaria, elija la zona horaria.
Programación recurrente Una programación recurrente invoca un destino a la velocidad que especifique mediante una expresión cron o de frecuencia.	a. En Tipo de programac ión, realice una de las siguientes acciones: • Si quiere utilizar una expresión Cron para definir la programac ión, elija Programac ión basada en Cron e introduzca la expresión Cron. • Si quiere utilizar una expresión de frecuencia para definir

Ocurrencia	Haga lo siguiente...	
	la programación, elija Programación basada en la frecuencia e introduzca la expresión de frecuencia. Para obtener más información sobre las expresiones cron y de frecuencia, consulte Tipos de programación el Programador de EventBridge en la Guía del usuario del Programador de Amazon EventBridge. b. En Intervalo de tiempo flexible, elija Desactivado para desactivar la opción o elegir uno de los periodos de tiempo predefinidos. Por ejemplo, si elige 15 minutos y establece una programación recurrente para invocar su destino una vez cada hora, la programación se ejecuta 15 minutos después del inicio de cada hora.	

5. (Opcional) Si elige Programación recurrente en el paso anterior, en la sección de Periodo de tiempo, realice lo siguiente:

   a. En Zona horaria, elija una zona horaria.

b. En Fecha y hora de inicio, introduzca una fecha válida con el formato `YYYY/MM/DD` y, a continuación, especifique una marca de tiempo con el formato `hh:mm` de 24 horas.

c. En Fecha y hora de finalización, introduzca una fecha válida con el formato `YYYY/MM/DD` y, a continuación, especifique una marca de tiempo con el formato `hh:mm` de 24 horas.

6. Elija Siguiente.

7. En la página Seleccionar destino, haga lo siguiente:

   a. Seleccione Todas las API y, a continuación, en el cuadro de búsqueda escriba ECS.

   b. Seleccione Amazon ECS.

   c. En el cuadro de búsqueda, escriba Ejecutar tarea y, a continuación, seleccione Ejecutar tarea.

   d. En Clúster de ECS, elija el clúster.

   e. Para la tarea de ECS, elija la definición de tarea que se utilizará para la tarea.

   f. Elija cómo distribuir las tareas en la infraestructura de clúster. Expanda las Opciones de computación y, a continuación, elija una de las siguientes opciones:

Opción de computación	Pasos	
Estrategia de proveedores de capacidad	a. Elija Estrategia de proveedores de capacidad. b. Elija una estrategia: • Para utilizar la estrategia de proveedores de capacidad predeterminada, elija Usar clúster predeterminado. • Para usar una estrategia personalizada, seleccione Uso personalizado. A continuación,	

Opción de computación	Pasos
	introduzca la Base, el Proveedor de capacidad y el Peso. Para EC2, el proveedor de capacidad es el grupo de escalado automátic o.
Tipo de lanzamiento	a. Seleccione Tipo de inicialización. b. En Launch type (Tipo de lanzamiento), elija un tipo de lanzamiento. c. Cuando se especific a el tipo de inicializ ación de Fargate, en Versión de la plataforma, especifique la versión de la plataforma que se va a utilizar.

g. En el caso de las Subredes, introduzca los ID de subred en los que se ejecutará la tarea.

h. En el caso los Grupos de seguridad, introduzca los ID de los grupos de seguridad de la subred.

i. (Opcional) Para utilizar una estrategia de ubicación de tareas que no sea la predeterminada, expanda Restricción de ubicación y, a continuación, introduzca las restricciones.

   Para obtener más información, consulte Cómo coloca Amazon ECS las tareas en las instancias de contenedor.

j. (Opcional) Para ayudar a identificar las tareas, en Etiquetas, configure las etiquetas.

   Para que Amazon ECS etiquete automáticamente todas las tareas recién lanzadas con las etiquetas de definición de tareas, seleccione Activar las etiquetas administradas de Amazon ECS.

8. Elija Siguiente.
9. En la página Configuración, haga lo siguiente:

   a. Para activar la programación, en Estado de la programación, cambie a Habilitar programación.

   b. Para configurar una política de reintentos para su programación, en Política de reintento y cola de mensajes fallidos (DLQ), realice lo siguiente:

      - Cambie a Reintentar.
      - En Tiempo de retención máxima del evento, ingrese el máximo de horas y minutos que el programador de EventBridge debe mantener un evento sin procesar.
      - El tiempo máximo es de 24 horas.
      - En Cantidad máxima de reintentos, introduzca el número máximo de veces que el Programador de EventBridge reintenta la programación si el destino devuelve un error.

        El valor máximo es 185 reintentos.

      Con las políticas de reintentos, si un programa no puede invocar su destino, el Programador de EventBridge vuelve a ejecutar el programa. Si se encuentra configurado, debe establecer el tiempo máximo de retención y los reintentos máximos para la programación.

   c. Elija dónde almacena los eventos no entregados el Programador de EventBridge.

Opción Cola de mensajes fallidos (DLQ)	Haga lo siguiente...
No almacenar	Seleccione Ninguno.
Guardar el evento en la misma Cuenta de AWS donde crea la programación	a. Elija Seleccionar una cola de Amazon SQS en mi Cuenta de AWS como DLQ. b. Elija el Nombre de recurso de Amazon (ARN) para la cola de Amazon SQS.

Opción Cola de mensajes fallidos (DLQ)	Haga lo siguiente...	
Guardar el evento en una Cuenta de AWS diferente de donde crea la programación	a. Elija Especificar una cola de Amazon SQS en otras Cuentas de AWS como DLQ. b. Ingrese el Nombre de recurso de Amazon (ARN) para la cola de Amazon SQS.	

d. Para utilizar una clave administrada por el cliente a fin de cifrar la entrada de destino, en Cifrado, elija Personalizar la configuración de cifrado (avanzado).

   Si elige esta opción, ingrese un ARN de clave de KMS existente o elija Crear una AWS KMS key para navegar hasta la consola de AWS KMS. Para obtener más información sobre cómo el Programador de EventBridge cifra los datos en reposo, consulte Encryption at rest en Amazon EventBridge Scheduler User Guide.

e. En Permisos, seleccione Usar el rol existente y, a continuación, seleccione el rol.

   Para que el Programador de EventBridge cree un rol de ejecución nuevo en su nombre, elija Crear un nuevo rol para esta programación. A continuación, ingrese un nombre para el Nombre de rol. Si elige esta opción, el Programador de EventBridge adjunta al rol los permisos necesarios para el objetivo creado con la plantilla.

10. Elija Siguiente.
11. En la página de Revisar y crear una programación, revise los detalles de su programación. En cada sección, elija Editar para volver a ese paso y editar sus detalles.
12. Elija Crear programación.

    Puede ver una lista de sus programaciones nuevas y existentes en la página Programaciones. En la columna Estado, verifique que su programación nueva se encuentre Habilitada.

Pasos a seguir a continuación

Puede utilizar la consola del programador de EventBridge o la AWS CLI para administrar el programa. Para obtener más información, consulte Administración de un programa en la Guía del usuario del programador de Amazon EventBridge.

Detención de una tarea de Amazon ECS

Si ya no necesita mantener una tarea independiente en ejecución, puede detenerla. La consola de Amazon ECS facilita la detención de una o más tareas.

No puede reiniciar una tarea detenida individual.

Si necesita detener un servicio, consulte Eliminación de un servicio de Amazon ECS mediante la consola.

Para detener una tarea independiente (AWS Management Console)

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. En la página Clústeres, elija el clúster para ir a la página de detalles del clúster.
4. En la página de detalles del clúster, elija la pestaña Tareas.
5. Puede filtrar las tareas por tipo de lanzamiento mediante la lista Filtrar tipo de lanzamiento.

Tareas que detener	Pasos
Una o más	a. Seleccione las tareas y, a continuación, elija Detener y Detener lo seleccionado. b. En la página de confirmación de Detener tarea, elija Detener
Todos	**Important** Si decide detener todas las tareas mediante la consola,

Tareas que detener	Pasos	
	Amazon ECS detiene todas las tareas independi entes y las tareas que forman parte de un servicio. Por lo tanto, le recomenda mos que tenga cuidado al utilizar esta opción. a. Elija Detener, Detener todo. b. En la página de confirmac ión de Detener tarea, ingrese Detener todas las tareas y luego seleccione Detener.	

Servicios de Amazon ECS

Puede utilizar un servicio de Amazon ECS para ejecutar y mantener un número determinado de instancias de una definición de tarea de manera simultánea en un clúster de Amazon ECS. Si una de las tareas falla o se detiene, el programador de servicios de Amazon ECS lanza otra instancia de su definición de tarea para sustituirla. Esto ayuda a mantener el número deseado de tareas en el servicio.

También puede ejecutar el servicio detrás de un equilibrador de carga. El balanceador de carga distribuye el tráfico entre las tareas que están asociadas al servicio.

Se recomienda utilizar el programador de servicios para los servicios y aplicaciones sin estado de larga duración. El programador de servicios garantiza el cumplimiento de la estrategia de programación especificada y reprograma las tareas cuando alguna de ellas falla. Por ejemplo, si falla la infraestructura subyacente, el programador de servicios puede reprogramar una tarea.

Puede utilizar estrategias de ubicación de tareas y restricciones para personalizar el modo en que el programador ubica y termina las tareas. Si se detiene una tarea de un servicio, el programador lanza una nueva tarea para sustituirla. Este proceso continúa hasta que el servicio alcanza el número deseado de tareas en función de la estrategia de programación que utiliza el servicio. La estrategia de programación del servicio también se denomina tipo de servicio.

El programador de servicios también reemplaza las tareas que se determina que están en mal estado después de que se produzca un error en una comprobación de estado del contenedor o en una comprobación de estado del grupo objetivo del equilibrador de cargas. Este reemplazo depende de los parámetros de definición del servicio `maximumPercent` y `desiredCount`. Si una tarea está marcada como en mal estado, el programador de servicios iniciará primero una tarea de reemplazo. Luego, ocurrirá lo siguiente.

- Si la tarea de reemplazo tiene un estado de `HEALTHY`, el programador de servicios detiene la tarea en mal estado.
- Si la tarea de reemplazo tiene un estado de `UNHEALTHY`, el programador detendrá la tarea de reemplazo en mal estado o la tarea existente en mal estado para igualar el recuento total de tareas en `desiredCount`.

Si el parámetro `maximumPercent` impide que el programador inicie primero una tarea de reemplazo, detendrá las tareas en mal estado de forma aleatoria de una en una para liberar capacidad y, a continuación, iniciará una tarea de reemplazo. El proceso de inicio y parada continúa hasta que todas las tareas en mal estado se sustituyan por tareas en buen estado. Una vez que se hayan reemplazado todas las tareas en mal estado y solo se estén ejecutando las tareas en buen estado, si el recuento total de tareas supera el límite de `desiredCount`, las tareas en buen estado se detienen aleatoriamente hasta que el recuento total de tareas sea igual a `desiredCount`. Para obtener más información sobre `maximumPercent` y `desiredCount`, consulte Parámetros de definición de servicios.

El programador de servicios incluye una lógica que limita la frecuencia con la que se reinician las tareas si estas fallan de forma repetida Si una tarea se detiene sin haber entrado en estado `RUNNING`, el programador de servicios comienza a ralentizar los intentos de lanzamiento y envía un mensaje de evento de servicio. Este comportamiento impide que se utilicen recursos innecesarios para tareas fallidas antes de poder resolver el problema. Una vez que el servicio de actualiza, el programador de servicios retoma el comportamiento normal de programación. Para obtener más información, consulte Lógica de limitación controlada de servicios de Amazon ECS y Visualización de los mensajes de eventos del servicio de Amazon ECS.

Opción de computación de infraestructura

Hay dos opciones de computación que distribuyen sus tareas.

- Una estrategia de proveedor de capacidad hace que Amazon ECS distribuya sus tareas en uno o varios proveedores de capacidad.

  En capacity provider strategy (estrategia de proveedor de capacidad), la consola selecciona una opción de computación de forma predeterminada. A continuación, se describe el orden que utiliza la consola para seleccionar un valor predeterminado:

  - Si su clúster tiene definida una estrategia de proveedor de capacidad por defecto, se seleccionará esa.
  - Si el clúster no tiene definida una estrategia de proveedores de capacidad predeterminada, pero sí tiene los proveedores de capacidad de Fargate agregados al clúster, se selecciona una estrategia de proveedores de capacidad personalizada que utiliza al proveedor de capacidad de `FARGATE`.
  - Si su clúster no tiene definida una estrategia de proveedor de capacidad por defecto, pero tiene uno o varios proveedores de capacidad de grupo de escalado automático agregados al clúster, se seleccionará la opción Utilizar personalizado (Avanzado) y tendrá que definir manualmente la estrategia.
  - Si el clúster no tiene definida ninguna estrategia de proveedores de capacidad predeterminada ni tiene proveedores de capacidad agregados al clúster, se selecciona el tipo de lanzamiento de Fargate.
- Un tipo de lanzamiento hace que Amazon ECS lance nuestras tareas directamente en Fargate o en las instancias de EC2 registradas en sus clústeres.

  De forma predeterminada, el servicio se inicia en las subredes de la VPC de clúster.

Escalado automático de servicios

El escalado automático del servicio es la capacidad de aumentar o disminuir automáticamente la cantidad de tareas deseada en el servicio de Amazon ECS. Amazon ECS utiliza el servicio de Application Auto Scaling para proporcionar esta funcionalidad.

Para obtener más información, consulte Escalado automático de su servicio de Amazon ECS.

Equilibrio de carga de los servicios

Los servicios de Amazon ECS alojados en AWS Fargate admiten los equilibradores de carga de aplicación, los equilibradores de carga de red y los equilibradores de carga de puerta de enlace. Utilice la siguiente tabla para obtener información sobre el tipo de equilibrador de carga que se debe utilizar.

Tipo de equilibrador de carga	Utilizar en estos casos	
Equilibrador de carga de aplicación	Dirija el tráfico HTTP o HTTPS (o de capa 7). Los Application Load Balancers ofrecen varias características nuevas que los hacen interesantes para utilizar con los servicios Amazon ECS: • Cada servicio puede servir el tráfico desde varios balanceadores de carga y exponer varios puertos con balance de carga especific ando varios grupos de destino. • Se admiten en tareas alojadas en instancias de Fargate y de EC2. • Los Application Load Balancers permiten a los contenedores utilizar la asignación de puerto de host dinámico (de modo que se permitan varias tareas desde el mismo servicio por instancia de contenedor).	

Tipo de equilibrador de carga	Utilizar en estos casos	
	• Los balanceadores de carga de aplicaciones admiten el enrutamiento basado en rutas y las reglas de prioridad (de modo que varios servicios puedan utilizar el mismo agente de escucha en un único Application Load Balancer).	
Equilibrador de carga de red	Dirija el tráfico TCP o UDP (o de capa 4).	
Gateway Load Balancer	Dirija el tráfico TCP o UDP (o de capa 4). Utilice dispositivos virtuales , como firewalls, sistemas de prevención y detección de intrusiones, y sistemas de inspección profunda de paquetes.	

Para obtener más información, consulte Uso del equilibrador de carga para distribuir el tráfico de servicio de Amazon ECS.

Interconexión de servicios

Si necesita una aplicación para conectarse a otras que se ejecutan en los servicios de Amazon ECS, Amazon ECS proporciona las siguientes formas de lograrlo sin un equilibrador de carga:

- Service Connect: permite que se lleven a cabo las comunicaciones de servicio a servicio con detección automática mediante nombres abreviados y puertos estándar.
- Detección de servicios: la detección de servicios usa Route 53 para crear un espacio de nombres para el servicio, lo que permite detectarlo a través del DNS.

- Amazon VPC Lattice: VPC Lattice es un servicio de redes de aplicaciones completamente administrado que puede utilizar para conectar, proteger y monitorear sus servicios en varias cuentas y nubes privadas virtuales (VPC). Su uso conlleva un costo asociado.

Para obtener más información, consulte Interconexión de los servicios de Amazon ECS.

Estrategias y controladores de implementación de servicios de Amazon ECS

Antes de implementar el servicio, determine las opciones para implementarlo y las características que utilizará el servicio.

Estrategia de programación

Existen dos estrategias del programador de servicio:

- `REPLICA`: la estrategia de programación de réplicas sitúa y mantiene en el clúster el número de tareas deseado. De forma predeterminada, el programador de servicio distribuye las tareas en zonas de disponibilidad. Puede utilizar estrategias y restricciones de ubicación de tareas para personalizar las decisiones de ubicación de las tareas. Para obtener más información, consulte Estrategia de programación de réplicas.
- `DAEMON`: la estrategia de programación del daemon implementa exactamente una tarea en cada instancia de contenedor activa que cumpla todas las restricciones de ubicación de tareas que se especifiquen para el clúster. Cuando se utiliza esta estrategia, no es necesario especificar un número deseado de tareas, ni una estrategia de ubicación de tareas ni utilizar políticas de Auto Scaling de servicios. Para obtener más información, consulte Estrategia de programación de daemon.

> **Note**
>
> Las tareas de Fargate no admiten la estrategia de programación de `DAEMON`.

Estrategia de programación de réplicas

La estrategia de programación de réplicas sitúa y mantiene en el clúster el número de tareas deseado.

En el caso de un servicio que ejecuta tareas en Fargate, cuando el programador de servicios lanza nuevas tareas o deja de ejecutarlas, el programador de servicios hace lo mejor para mantener un equilibrio entre las zonas de disponibilidad. No es necesario especificar estrategias ni restricciones de ubicación de tareas.

Al crear un servicio que ejecuta tareas en instancias EC2, tiene la opción de especificar estrategias y restricciones de ubicación de tareas a fin de personalizar las decisiones de ubicación de tareas. Si no se especifican estrategias o restricciones de ubicación de tareas, el programador de servicios repartirá las tareas de forma predeterminada entre las zonas de disponibilidad. El programador de servicios utiliza la siguiente lógica:

- Determina cuál de las instancias de contenedor de su clúster puede admitir la definición de la tarea de su servicio (por ejemplo, la CPU, la memoria, los puertos y los atributos de la instancia de contenedor requeridos).
- Determina qué instancias de contenedor satisfacen las restricciones de ubicación definidas para el servicio.
- Si tiene un servicio de réplica que depende de un servicio de daemon (por ejemplo, una tarea del enrutador del registro de daemon que debe estar ejecutándose antes de que las tareas puedan utilizar el registro), cree una restricción de ubicación de tareas que garantice que las tareas del servicio de daemon se coloquen en la instancia de EC2 antes que las tareas del servicio de réplica. Para obtener más información, consulte Ejemplos de restricciones para ubicación de tareas de Amazon ECS.
- Cuando hay una estrategia de ubicación definida, utilice esa estrategia para seleccionar una instancia entre los candidatos restantes.
- Si no hay ninguna estrategia de ubicación definida, utilice la siguiente lógica para equilibrar las tareas entre las zonas de disponibilidad de su clúster:
  - Ordena las instancias de contenedor válidas. Da prioridad a las instancias que tienen el menor número de tareas en ejecución para este servicio en su respectiva zona de disponibilidad. Por ejemplo, si la zona A tiene una tarea de servicio en ejecución y las zonas B y C tienen cero cada una, las instancias de contenedor válidas en la zona B o C se consideran óptimas para colocación.
  - Ubica la nueva tarea de servicio en una instancia de contenedor válida en una zona de disponibilidad óptima en función de los pasos anteriores. Favorece las instancias de contenedores con el menor número de tareas en ejecución para este servicio.

Le recomendamos que utilice la característica de reequilibrio de servicios cuando utilice la estrategia de `REPLICA`, ya que ayuda a garantizar una alta disponibilidad del servicio.

Estrategia de programación de daemon

La estrategia de programación de daemon implementa exactamente una tarea en cada instancia de contenedor activa que cumpla todas las restricciones de ubicación de tareas especificadas en el clúster. El programador de servicios evalúa las restricciones de ubicación de las tareas en ejecución y detiene las tareas que no cumplen las restricciones de ubicación. Al utilizar esta estrategia, no es necesario especificar un número deseado de tareas ni una estrategia de ubicación de tareas, ni utilizar políticas de escalado automático de servicio.

Amazon ECS reserva los recursos de computación de la instancia de contenedor, incluido CPU, memoria e interfaces de red, para las tareas del daemon. Cuando se lanza un servicio daemon en un clúster con otros servicios de réplica, Amazon ECS prioriza la tarea del daemon. Esto significa que la tarea del daemon es la primera en lanzarse en las instancias y la última en detenerse después de que se detengan todas las tareas de réplica. Esta estrategia garantiza que las tareas de réplica pendientes no utilicen esos recursos y estén disponibles para las tareas del daemon.

El programador de servicios del daemon no ubica tareas en las instancias que tienen el estado `DRAINING`. Si una instancia de contenedor cambia al estado `DRAINING`, las tareas del daemon que incluya se detienen. El programador de servicios también monitorea cuándo se agregan nuevas instancias de contenedor al clúster y agrega las tareas de daemon en ellas.

Al especificar una configuración de implementación, el valor del parámetro `maximumPercent` debe ser `100` (especificado como porcentaje), que es el valor predeterminado que se utiliza si no se establece. El valor predeterminado del parámetro `minimumHealthyPercent` es `0` (especificado como porcentaje).

Debe reiniciar el servicio cuando cambie las restricciones de ubicación del servicio del daemon. Amazon ECS actualiza de forma dinámica los recursos reservados en instancias aptas para la tarea del daemon. Para las instancias existentes, el programador intenta ubicar la tarea en la instancia.

Una nueva implementación se inicia cuando hay un cambio en el tamaño de la tarea o en la reserva de recursos del contenedor en la definición de la tarea. Una nueva implementación también comienza cuando se actualiza un servicio o se establece una revisión diferente de la definición de la tarea. Amazon ECS recoge las reservas de memoria y CPU actualizadas para el daemon y, a continuación, bloquea esa capacidad para la tarea del daemon.

Si no hay recursos suficientes para cualquiera de los casos anteriores, ocurre lo siguiente:

- Se produce un error en la ubicación de la tarea.
- Se genera un evento de CloudWatch.
- Amazon ECS continúa intentando programar la tarea en la instancia a la espera de que los recursos estén disponibles.
- Amazon ECS libera todas las instancias reservadas que ya no cumplan con los criterios de restricción de ubicación y detiene las tareas de daemon correspondientes.

La estrategia de programación de daemon se puede utilizar en los siguientes casos:

- Ejecución de contenedores de aplicaciones
- Ejecución de contenedores de soporte para tareas de registro, monitoreo y seguimiento

Las tareas que utilizan el tipo de lanzamiento de Fargate o los tipos de controlador de implementación `CODE_DEPLOY` o `EXTERNAL` no admiten la estrategia de programación del daemon.

Cuando el programador de servicios detiene las tareas en ejecución, intenta mantener un balance entre las zonas de disponibilidad del clúster. El programador utiliza la siguiente lógica:

- Si se ha definido una estrategia de ubicación, utilice esta estrategia para seleccionar las tareas que deben terminar. Por ejemplo, si un servicio tiene definida una estrategia de distribución de zonas de disponibilidad, se seleccionará una tarea que deje a las demás tareas con la mejor distribución.
- Si no hay ninguna estrategia de ubicación definida, mantenga el equilibrio entre las zonas de disponibilidad de su clúster con la siguiente lógica:
  - Ordene las instancias de contenedor válidas. Dé prioridad a las instancias que tienen el mayor número de tareas en ejecución para este servicio en su respectiva zona de disponibilidad. Por ejemplo, si la zona A tiene una tarea de servicio en ejecución y las zonas B y C tienen dos cada una, las instancias de contenedor en la zona B o C se consideran óptimas para terminación.
  - Detenga la tarea en una instancia de contenedor en una zona de disponibilidad óptima en función de los pasos anteriores. Priorice las instancias de contenedores con el mayor número de tareas en ejecución para este servicio.

Controladores de implementación

El controlador de implementación es el mecanismo que determina cómo se implementan las tareas del servicio. Las opciones válidas son las siguientes:

- ECS

  Al crear un servicio que utiliza el controlador de implementación de `ECS`, puede elegir entre las siguientes estrategias de implementación:

  - `ROLLING`: cuando crea un servicio que utiliza la estrategia de implementación de actualización continua (`ROLLING`), el programador de servicios de Amazon ECS sustituye las tareas que se están ejecutando actualmente por unas nuevas. El número de tareas que Amazon ECS agrega o elimina del servicio durante una actualización continua se controla mediante la configuración de implementación del servicio.

    Las implementaciones de actualizaciones continuas son las más adecuadas para los siguientes escenarios:

    - Actualizaciones graduales del servicio: debe actualizar el servicio de forma gradual sin desconectar todo el servicio de una sola vez.
    - Requisitos de recursos limitados: desea evitar los costos generados por los recursos adicionales que supone ejecutar dos entornos completos de forma simultánea (como lo exigen las implementaciones azul/verde).
    - Tiempo de implementación aceptable: su aplicación puede tolerar un proceso de implementación más prolongado, ya que las actualizaciones continuas sustituyen las tareas una por una.
    - No es necesario realizar una reversión inmediata: su servicio puede tolerar un proceso de reversión que tarde minutos en lugar de segundos.
    - Proceso de implementación simple: prefiere un método de implementación sencillo sin la complejidad de administrar varios entornos, grupos de destino y oyentes.
    - No se requiere un equilibrador de carga: su servicio no usa ni requiere un equilibrador de carga, equilibrador de carga de aplicación, equilibrador de carga de red o Service Connect (que son necesarios para las implementaciones azul/verde).
    - Aplicaciones con estado: su aplicación mantiene un estado que dificulta la ejecución de dos entornos paralelos.
    - Sensibilidad a los costos: desea no tener que ejecutar entornos duplicados durante la implementación para minimizar los costos de implementación.

    Las actualizaciones continuas son la estrategia de implementación predeterminada de los servicios y proporcionan un equilibrio entre la seguridad de la implementación y la eficiencia de los recursos en muchos escenarios de aplicaciones comunes.

- `BLUE_GREEN`: una estrategia de implementación azul/verde (`BLUE_GREEN`) es una metodología de lanzamiento que reduce el tiempo de inactividad y el riesgo al ejecutar dos entornos de producción idénticos, denominados azul y verde. Con las implementaciones azul/verde de Amazon ECS, puede validar las nuevas revisiones de servicios antes de dirigir el tráfico de producción hacia ellas. Este método proporciona una forma más segura de implementar cambios con la capacidad de revertirlos rápidamente si es necesario.

  Las implementaciones azul/verde de Amazon ECS son las más adecuadas para los siguientes escenarios:

  - Validación de servicio: cuando tiene que validar nuevas revisiones de servicio antes de dirigir el tráfico de producción hacia ellas
  - Tiempo de inactividad cero: cuando su servicio requiere implementaciones sin tiempo de inactividad
  - Reversión instantánea: cuando necesita la capacidad de revertir rápidamente si se detectan problemas
  - Requisito de equilibrador de carga: cuando su servicio utiliza un equilibrador de carga de aplicación, equilibrador de carga de red o Service Connect
- Externo

  Utilice un controlador de implementación de terceros.
- Implementación azul/verde (con tecnología de AWS CodeDeploy)

  CodeDeploy instala una versión actualizada de la aplicación como un nuevo conjunto de tareas de sustitución y vuelve a enrutar el tráfico de producción del conjunto de tareas original al conjunto de tareas de sustitución. Cuando la implementación se realiza correctamente, se termina el conjunto de tareas original. Este tipo de implementación permite verificar la nueva implementación de un servicio antes de enviar el tráfico de producción hacia este.

Detección de errores de implementación de Amazon ECS

Amazon ECS proporciona dos métodos para detectar errores de implementación:

- Interruptor de circuitos de implementación
- Alarmas de CloudWatch

Ambos métodos se pueden configurar para revertir automáticamente las implementaciones con errores al último estado válido conocido.

Considere lo siguiente:

- Ambos métodos solo admiten los tipos de implementación de actualizaciones continuas y de implementación azul/verde.
- Cuando se utilizan ambos métodos, cualquiera de ellos puede provocar un error en la implementación.
- El método de reversión requiere una implementación previa en estado COMPLETADO.
- Se generan eventos de EventBridge para los cambios en el estado de la implementación.

Detección de errores por el interruptor de circuito de implementación de Amazon ECS

El disyuntor de implementación es el mecanismo de actualización continua que determina si las tareas alcanzan un estado estable. El interruptor de implementación tiene una opción que revertirá automáticamente una implementación con errores a la implementación con el estado `COMPLETED`.

Cuando una implementación de servicio cambia de estado, Amazon ECS envía un evento de cambio de estado de implementación del servicio a EventBridge. Esto proporciona una forma programática de monitorear el estado de las implementaciones de servicios. Para obtener más información, consulte Eventos de cambio de estado de implementación de servicios de Amazon ECS. Recomendamos que cree y supervise una regla de EventBridge con un `eventName` de `SERVICE_DEPLOYMENT_FAILED` para que pueda tomar medidas manuales para iniciar la implementación. Para obtener más información, consulte Introducción a EventBridge en la Guía del usuario de Amazon EventBridge.

Cuando el disyuntor de implementación determina que una implementación tiene errores, este busca la implementación más reciente que se encuentre en un estado `COMPLETED`. Esta es la implementación que utiliza como implementación restaurada. Cuando se inicia la reversión, la implementación cambia de `COMPLETED` a `IN_PROGRESS`. Esto significa que la implementación no es apta para otra reversión hasta que alcance el estado `COMPLETED`. Cuando el disyuntor de implementación no encuentra ninguna implementación que esté en estado `COMPLETED`, no inicia nuevas tareas y la implementación se detiene.

Al crear un servicio, el programador hace un seguimiento de las tareas que no se pudieron iniciar en dos etapas.

- Fase 1: el programador supervisa las tareas para comprobar si pasan al estado EN EJECUCIÓN.
  - Éxito: la implementación tiene posibilidades de pasar al estado COMPLETADO porque hay más de una tarea que ha pasado al estado EN EJECUCIÓN. Se omite el criterio de fallo y el disyuntor pasa a la fase 2.
  - Error: Hay tareas consecutivas que no pasaron al estado EN EJECUCIÓN y es posible que la implementación pase al estado ERROR.
- Fase 2: la implementación entra en esta etapa cuando hay al menos una tarea en ejecución. El disyuntor comprueba las comprobaciones de estado de las tareas de la implementación actual que se están evaluando. Las comprobaciones de estado validadas son Elastic Load Balancing, las comprobaciones de estado del servicio AWS Cloud Map y las comprobaciones de estado de los contenedores.
  - Correcto: hay al menos una tarea en ejecución cuyas comprobaciones de estado se han superado.
  - Error: Las tareas que se reemplazan debido a errores en las comprobaciones de estado han alcanzado el umbral de error.

Tenga en cuenta lo siguiente cuando utilice el método del interruptor de implementación en un servicio. EventBridge genera la regla.

- La respuesta de `DescribeServices` proporciona información sobre el estado de una implementación, el `rolloutState` y el `rolloutStateReason`. Cuando se inicia una nueva implementación, el estado de despliegue comienza en el estado `IN_PROGRESS`. Cuando el servicio alcanza un estado estable, el estado de implementación pasa a `COMPLETED`. Si el servicio no alcanza un estado estable y el interruptor está activado, la implementación pasará al estado `FAILED`. Una implementación en estado `FAILED` no lanzará ninguna tarea nueva.
- Además de los eventos de cambio de estado de implementación del servicio que Amazon ECS envía para implementaciones que se han iniciado y completado, Amazon ECS también envía un evento cuando falla una implementación con interruptor activado. Estos eventos proporcionan detalles acerca de por qué falló una implementación o si se inició debido a una restauración. Para obtener más información, consulte Eventos de cambio de estado de implementación de servicios de Amazon ECS.
- Si se inicia una nueva implementación porque se produjo un error en una implementación anterior y causó una restauración, el campo `reason` del evento de cambio de estado de implementación del servicio indica que la implementación se inició debido a una restauración.

- El interruptor de implementación solo es compatible con los servicios de Amazon ECS que utilizan el controlador de implementación de actualización continua (`ECS`).
- Debe utilizar la consola de Amazon ECS o la AWS CLI cuando utilice el interruptor de implementación con la opción CloudWatch. Para obtener más información, consulte the section called "Creación de un servicio a partir de los parámetros definidos" y create-service en la Referencia de la AWS Command Line Interface.

En los siguientes ejemplos de la AWS CLI de `create-service`, se muestra cómo crear un servicio de Linux cuando se usa el interruptor de implementación con la opción de restauración.

```
aws ecs create-service \
     --service-name MyService \
     --deployment-controller type=ECS \
     --desired-count 3 \
     --deployment-configuration "deploymentCircuitBreaker={enable=true,rollback=true}"
 \
     --task-definition sample-fargate:1 \
     --launch-type FARGATE \
     --platform-family LINUX \
     --platform-version 1.4.0 \
     --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
```

Ejemplo:

La implementación 1 está en estado `COMPLETED`.

La implementación 2 no puede iniciarse, por lo que el disyuntor vuelve a la implementación 1. La implementación 1 pasa al estado `IN_PROGRESS`.

La implementación 3 se inicia y no hay ninguna implementación en estado `COMPLETED`, por lo que la implementación 3 no puede revertir ni iniciar tareas.

Failure threshold

El disyuntor de despliegue calcula el valor umbral y, a continuación, lo utiliza para determinar cuándo mover la implementación a un estado `FAILED`.

El disyuntor de implementación tiene un umbral mínimo de 3 y un umbral máximo de 200, y utiliza los valores de la siguiente fórmula para determinar el error de implementación.

```
Minimum threshold <= 0.5 * desired task count => maximum threshold
```

Cuando el resultado del cálculo es superior al mínimo de 3, pero inferior al máximo de 200, el umbral de error se establece en el umbral calculado (redondeado al alza).

> **Note**
>
> No se puede cambiar ninguno de los valores de umbral.

Hay dos etapas para la comprobación del estado de la implementación.

1. El disyuntor de despliegue supervisa las tareas que forman parte de la implementación y comprueba las tareas que se encuentran en el estado `RUNNING`. El programador ignora los criterios de error cuando una tarea de la implementación actual se encuentra en el estado `RUNNING` y pasa a la siguiente etapa. Cuando las tareas no alcanzan en el estado `RUNNING`, el disyuntor de implementación aumenta el recuento de error en uno. Cuando el recuento de errores es igual al umbral, la implementación se marca como `FAILED`.
2. Se entra en esta etapa cuando hay una o más tareas en el estado `RUNNING`. El disyuntor de implementación realiza comprobaciones de estado de los siguientes recursos para las tareas de la implementación actual:
   - Balanceadores de carga de Elastic Load Balancing
   - Servicio de AWS Cloud Map
   - Comprobaciones de estado de los contenedores de Amazon ECS

   Cuando falla una comprobación de estado de la tarea, el disyuntor de implementación aumenta el recuento de errores en uno. Cuando el recuento de errores es igual al umbral, la implementación se marca como `FAILED`.

La siguiente tabla muestra algunos ejemplos.

Recuento deseado de tareas	Cálculo	Threshold
1	`3 <= 0.5 * 1 => 200`	3 (el valor calculado es inferior al mínimo)

Recuento deseado de tareas	Cálculo	Threshold
25	`3 <= 0.5 * 25 => 200`	13 (el valor se redondea hacia arriba)
400	`3 <= 0.5 * 400 => 200`	200
800	`3 <= 0.5 * 800 => 200`	200 (el valor calculado es mayor que el máximo)

Por ejemplo, cuando el umbral es 3, el disyuntor comienza con el recuento de fallos establecido en 0. Cuando una tarea no alcanza el estado `RUNNING`, el disyuntor de implementación aumenta el recuento de error en uno. Cuando el recuento de errores es igual 3, la implementación se marca como `FAILED`.

Para obtener más ejemplos acerca de cómo usar la opción de restauración, consulte Anuncio del interruptor de cirtcuito de implementación de Amazon ECS.

Detección de errores en la implementación de Amazon ECS por las alarmas de CloudWatch

Puede configurar Amazon ECS para que defina la implementación como fallida cuando detecte que una alarma de CloudWatch específica ha pasado al estado `ALARM`.

Si lo desea, puede establecer la configuración para restaurar una implementación con errores a la última implementación completada.

En los siguientes ejemplos de la AWS CLI de `create-service`, se muestra cómo crear un servicio de Linux cuando se utilizan las alarmas de implementación con la opción de restauración.

```
aws ecs create-service \
     --service-name MyService \
     --deployment-controller type=ECS \
     --desired-count 3 \
     --deployment-configuration
 "alarms={alarmNames=[alarm1Name,alarm2Name],enable=true,rollback=true}" \
     --task-definition sample-fargate:1 \
     --launch-type FARGATE \
     --platform-family LINUX \
     --platform-version 1.4.0 \
```

```
    --network-configuration
"awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
```

Tenga en cuenta lo siguiente al utilizar el método de alarmas de Amazon CloudWatch en un servicio.

- El tiempo durante el cual las revisiones de servicio azul y verde se ejecutan simultáneamente después de que se haya transferido el tráfico de producción. Amazon ECS calcula este periodo en función de la configuración de alarma asociada a la implementación. No puede establecer este valor.
- El parámetro de solicitud `deploymentConfiguration` ahora contiene el tipo de datos `alarms`. Puede especificar los nombres de las alarmas, si desea utilizar el método y si desea iniciar una restauración cuando las alarmas indiquen un error de implementación. Para obtener más información, consulte CreateService en la Referencia de la API de Amazon Elastic Container Service.
- La respuesta de `DescribeServices` proporciona información sobre el estado de una implementación, el `rolloutState` y el `rolloutStateReason`. Cuando se inicia una nueva implementación, el estado de implementación comienza en el estado `IN_PROGRESS`. Cuando el servicio alcanza un estado estable y se completa el tiempo de incorporación, el estado de implementación pasa a `COMPLETED`. Si el servicio no alcanza un estado estable y la alarma pasa al estado `ALARM`, la implementación pasará al estado `FAILED`. Si el estado de una implementación es `FAILED`, no lanzará ninguna tarea nueva.
- Además de los eventos de cambio de estado de implementación del servicio que Amazon ECS envía para implementaciones que se han iniciado y completado, Amazon ECS también envía un evento cuando se produce un error con una implementación que usa alarmas. Estos eventos proporcionan detalles acerca de por qué falló una implementación o si se inició debido a una restauración. Para obtener más información, consulte Eventos de cambio de estado de implementación de servicios de Amazon ECS.
- Si se inicia una nueva implementación porque se produjo un error en una implementación anterior y se activó la restauración, el campo `reason` del evento de cambio de estado de implementación de servicio indicará que la implementación se inició debido a una restauración.
- Si utiliza el interruptor de implementación y las alarmas de Amazon CloudWatch para detectar errores, los dos pueden iniciar un error de implementación tan pronto como se cumplan los criterios de cualquiera de los métodos. Se produce una restauración cuando se utiliza esta opción en el método que inició el error de implementación.
- Las alarmas de Amazon CloudWatch solo son compatibles con los servicios de Amazon ECS que utilizan el controlador de implementación de actualización continua (`ECS`).

- Puede configurar esta opción mediante la consola de Amazon ECS o la AWS CLI. Para obtener más información, consulte the section called “Creación de un servicio a partir de los parámetros definidos” y create-service en la Referencia de la AWS Command Line Interface.
- Notará que el estado de implementación permanece `IN_PROGRESS` durante un periodo prolongado. La razón de esto es que Amazon ECS no cambia el estado hasta que no haya eliminado la implementación activa y esto no ocurre hasta después del tiempo de incorporación. En función de la configuración de alarmas, puede parecer que la implementación tarda varios minutos más que si no se utilizan alarmas (aunque el nuevo conjunto de tareas principal escale verticalmente y la implementación anterior se reduzca verticalmente). Si usa los tiempos de espera de CloudFormation, considere aumentarlos. Para obtener más información, consulte Creación de condiciones de espera en una plantilla en la Guía del usuario de AWS CloudFormation.
- Amazon ECS llama a `DescribeAlarms` para sondear las alarmas. Las llamadas a `DescribeAlarms` contabilizarán para las cuotas de servicio de CloudWatch asociadas a su cuenta. Si tiene otros servicios de AWS que llamen a `DescribeAlarms`, Amazon ECS podría tener un impacto en el sondeo de las alarmas. Por ejemplo, si otro servicio hace suficientes llamadas a `DescribeAlarms` para alcanzar la cuota, ese servicio recibe una limitación y el de Amazon ECS también, y no puede sondear las alarmas. Si se genera una alarma durante el periodo de limitación, es posible que Amazon ECS no la detecte y que no se produzca la restauración. No hay ningún otro impacto en la implementación. Para obtener más información sobre las cuotas de servicio de CloudWatch, consulte Service Quotas de CloudWatch en la Guía del usuario de CloudWatch.
- Si hay una alarma en el estado `ALARM` al principio de una implementación, Amazon ECS no supervisará las alarmas mientras dure esa implementación (Amazon ECS ignora la configuración de la alarma). Este comportamiento aborda el caso en el que desee iniciar una nueva implementación para corregir un error de implementación inicial.

Alarmas recomendadas

Le recomendamos que utilice las siguientes métricas de alarma:

- Si usa un equilibrador de carga de aplicación, use las métricas de equilibrador de carga de aplicación `HTTPCode_ELB_5XX_Count` y `HTTPCode_ELB_4XX_Count`. Estas métricas comprueban si hay picos de HTTP. Para obtener más información acerca de las métricas del equilibrador de carga de aplicación, consulte Métricas de CloudWatch para su equilibrador de carga de aplicación en la Guía del usuario para equilibradores de carga de aplicaciones.

- Si ya tiene una aplicación, utilice las métricas `CPUUtilization` y `MemoryUtilization`. Estas métricas comprueban el porcentaje de CPU y memoria que utiliza el clúster o el servicio. Para obtener más información, consulte the section called “Consideraciones”.
- Si usa colas de Amazon Simple Queue Service en sus tareas, utilice la métrica `ApproximateNumberOfMessagesNotVisible` de Amazon SQS. Esta métrica comprueba el número de mensajes de la cola que van con retraso y no están disponibles para su lectura inmediata. Para obtener más información sobre las métricas de Amazon SQS, consulte Métricas de CloudWatch disponibles para Amazon SQS en la Guía para desarrolladores de Amazon Simple Queue Service.

Enlaces de ciclo de vida para las implementaciones de servicios de Amazon ECS

Cuando se inicia una implementación, pasa por las etapas del ciclo de vida. Estas etapas pueden estar en estados como IN_PROGRESS o SUCCESSFUL. Puede utilizar enlaces de ciclo de vida, que son funciones de Lambda que Amazon ECS ejecuta en su nombre en etapas del ciclo de vida específicas. Las funciones pueden ser cualquiera de las siguientes opciones:

- Una API asíncrona que valida la comprobación de estado en 15 minutos.
- Una API de sondeo que inicia otro proceso asíncrono que evalúa la finalización del enlace de ciclo de vida.

Una vez que la función haya terminado de ejecutarse, debe devolver un `hookStatus` para que la implementación continúe. Si no se devuelve un `hookStatus` o si la función falla, la implementación se revierte. Los siguientes son valores del `hookStatus`:

- `SUCCEEDED`: la implementación pasa a la siguiente fase del ciclo de vida.
- `FAILED`: la implementación se revierte a la última implementación correcta.
- `IN_PROGRESS`: Amazon ECS vuelve a ejecutar la función tras un breve periodo. De forma predeterminada, es un intervalo de 30 segundos; sin embargo, este valor se puede personalizar devolviendo un `callBackDelaySeconds` junto con el `hookStatus`.

En el siguiente ejemplo se muestra cómo devolver un `hookStatus` con un retraso de devolución de llamada personalizado. En este ejemplo, Amazon ECS volvería a intentar este enlace en 60 segundos en lugar de los 30 segundos predeterminados:

```
{
    "hookStatus": "IN_PROGRESS",
    "callBackDelaySeconds": 60
}
```

Cuando se produce una reversión, Amazon ECS ejecuta los enlaces de ciclo de vida para las siguientes etapas de dicho ciclo de vida:

- PRODUCTION_TRAFFIC_SHIFT
- TEST_TRAFFIC_SHIFT

Cargas útiles del ciclo de vida

Cuando configura los enlaces de ciclo de vida para las implementaciones de sus servicios de ECS, Amazon ECS invoca estos enlaces en etapas específicas del proceso de implementación. Cada etapa del ciclo de vida proporciona una carga útil de JSON con información acerca del estado actual de la implementación. En este documento se describe la estructura de carga útil de cada etapa del ciclo de vida.

Estructura común de las cargas útiles

Todas las cargas útiles de las etapas del ciclo de vida incluyen los siguientes campos comunes:

- `serviceArn`: el nombre de recurso de Amazon (ARN) del servicio.
- `targetServiceRevisionArn`: el ARN de la revisión de servicio de destino que se está implementando.
- `testTrafficWeights`: un mapa de los ARN de las revisiones de servicio con sus correspondientes porcentajes de ponderación de tráfico de prueba.
- `productionTrafficWeights`: un mapa de los ARN de las revisiones de servicio con sus correspondientes porcentajes de ponderación de tráfico de producción.

Cargas útiles de las etapas del ciclo de vida

RECONCILE_SERVICE

Esta etapa se produce al principio del proceso de implementación cuando se está conciliando el servicio. A continuación, se muestra un ejemplo de carga útil para esta etapa del ciclo de vida.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/01275892":
 100,
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/78652123": 0
  },
  "productionTrafficWeights": {
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/01275892":
 100,
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/78652123": 0
  }
}
```

Expectativas para esta etapa:

- El conjunto de tareas principal está en una escala del 0 %

PRE_SCALE_UP

Esta etapa se produce antes de que se escalen verticalmente las nuevas tareas. A continuación, se muestra un ejemplo de carga útil para esta etapa del ciclo de vida.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {}
}
```

Expectativas para esta etapa:

- Las tareas de revisión de servicio verde están en una escala del 0 %

POST_SCALE_UP

Esta etapa se produce una vez que las nuevas tareas se han escalado verticalmente y están en buen estado. A continuación, se muestra un ejemplo de carga útil para esta etapa del ciclo de vida.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {}
}
```

Expectativas para esta etapa:

- Las tareas de revisión de servicio verde están en una escala del 100 %
- Las tareas de revisión de servicio verde están en buen estado

TEST_TRAFFIC_SHIFT

Esta etapa se produce cuando el tráfico de prueba se está transfiriendo a las tareas de revisión de servicio verde.

A continuación, se muestra un ejemplo de carga útil para esta etapa del ciclo de vida.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/01275892":
 100,
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/78652123": 0
  },
  "productionTrafficWeights": {}
}
```

Expectativas para esta etapa:

- El tráfico de pruebas está pasando hacia las tareas de revisión de servicio verde.

POST_TEST_TRAFFIC_SHIFT

Esta etapa se produce después de que el tráfico de prueba se haya transferido por completo a las nuevas tareas.

A continuación, se muestra un ejemplo de carga útil para esta etapa del ciclo de vida.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {}
}
```

Expectativas para esta etapa:

- El 100 % del tráfico de prueba se destinó a las tareas de revisión de servicio verde.

PRODUCTION_TRAFFIC_SHIFT

Esta etapa se produce cuando el tráfico de producción se está transfiriendo a las tareas de revisión de servicio verde.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/01275892":
 100,
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/78652123": 0
  }
```

```
}
```

Expectativas para esta etapa:

- El tráfico de producción está pasando hacia la revisión de servicio verde.

POST_PRODUCTION_TRAFFIC_SHIFT

Esta etapa se produce cuando el tráfico de producción se ha transferido por completo a las tareas de revisión de servicio verde.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {}
}
```

Expectativas para esta etapa:

- El 100 % del tráfico de producción se destinó a las tareas de revisión de servicio verde.

Categorías de etapas del ciclo de vida

Las etapas del ciclo de vida se dividen en dos categorías:

1. Etapas de invocación única: estas etapas se invocan solo una vez durante la implementación de un servicio:
   - PRE_SCALE_UP
   - POST_SCALE_UP
   - POST_TEST_TRAFFIC_SHIFT
   - POST_PRODUCTION_TRAFFIC_SHIFT
2. Etapas de invocación recurrentes: estas etapas se pueden invocar varias veces durante la implementación de un servicio, por ejemplo, cuando se produce una operación de reversión:

- TEST_TRAFFIC_SHIFT
- PRODUCTION_TRAFFIC_SHIFT

Estado de la implementación durante los enlaces de ciclo de vida

Mientras los enlaces de ciclo de vida estén en ejecución, el estado de la implementación será `IN_PROGRESS` para todas las etapas del ciclo de vida.

Etapa del ciclo de vida	Estado de la implementación
RECONCILE_SERVICE	IN_PROGRESS
PRE_SCALE_UP	IN_PROGRESS
POST_SCALE_UP	IN_PROGRESS
TEST_TRAFFIC_SHIFT	IN_PROGRESS
POST_TEST_TRAFFIC_SHIFT	IN_PROGRESS
PRODUCTION_TRAFFIC_SHIFT	IN_PROGRESS
POST_PRODUCTION_TRAFFIC_SHIFT	IN_PROGRESS

Detención de las implementaciones de servicios de Amazon ECS

Puede detener de forma manual una implementación cuando el interruptor o las alarmas de CloudWatch no detecten una implementación fallida. Están disponibles los siguientes tipos de detención:

- Reversión: esta opción revierte la implementación del servicio a la revisión de servicio anterior.

  Puede usar esta opción incluso si no configuró la implementación del servicio con la opción de reversión.

Es posible detener una implementación que se encuentre en alguno de los estados siguientes. Para obtener más información sobre los estados de implementación de servicios, consulte Visualización del historial de servicios mediante las implementaciones de servicios de Amazon ECS.

- PENDING: la implementación del servicio pasa al estado ROLLBACK_REQUESTED y, a continuación, se inicia la operación de reversión.
- IN_PROGRESS: la implementación del servicio pasa al estado ROLLBACK_REQUESTED y, a continuación, comienza la operación de reversión.
- STOP_REQUESTED: la implementación del servicio continúa deteniéndose.
- ROLLBACK_REQUESTED: la implementación del servicio continúa con la operación de reversión.
- ROLLBACK_IN_PROGRESS: la implementación del servicio continúa con la operación de reversión.

Procedimiento

Antes de empezar, configure los permisos necesarios para ver las implementaciones de servicios. Para obtener más información, consulte Permisos necesarios para ver las implementaciones de servicios de Amazon ECS.

Amazon ECS Console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), elija el clúster.
3. En la página de detalles del clúster, en la sección Servicios, elija el servicio.

   Aparecerá la página de detalles del servicio.
4. En la página de detalles del servicio, elija Implementaciones.

   Se muestra la página de implementaciones.
5. En Implementación en curso, elija Revertir. A continuación, en la ventana de confirmación, elija Revertir.

AWS CLI

1. Ejecute `list-service-deployments` para recuperar el ARN de implementación del servicio.

   Sustituya las *`entradas del usuario`* por sus valores.

   ```
   aws ecs list-service-deployments --cluster cluster-name --service service-name
   ```

Anote el valor de `serviceDeploymentArn` para la implementación que quiere detener.

```
{
    "serviceDeployments": [
        {
            "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-deployment/cluster-name/service-name/NCWGC2ZR-taawPAYrIaU5",
            "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/cluster-name/service-name",
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/cluster-name",
            "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:123456789012:service-revision/cluster-name/service-name/4980306466373577095",
            "status": "SUCCESSFUL"
        }
    ]
}
```

2. Ejecute `stop-service-deployments`. Utilice el identificador `serviceDeploymentArn` que se devolvió de `list-service-deployments`.

   Sustituya las *`entradas del usuario`* por sus valores.

```
aws ecs stop-service-deployment --service-deployment-arn
 arn:aws:ecs:region:123456789012:service-deployment/cluster-name/service-name/NCWGC2ZR-taawPAYrIaU5 --stop-type ROLLBACK
```

Pasos a seguir a continuación

Decida qué cambios deben realizarse en el servicio y, a continuación, actualícelo. Para obtener más información, consulte Actualización de un servicio de Amazon ECS.

Implementación de los servicios de Amazon ECS mediante el reemplazo de tareas

Cuando crea un servicio que utiliza el tipo de implementación de actualización continua (`ECS`), el programador de servicios de Amazon ECS reemplaza las tareas que se ejecutan en ese momento por unas nuevas. El número de tareas que Amazon ECS agrega o elimina del servicio durante una actualización continua se controla mediante la configuración de implementación del servicio.

Amazon ECS utiliza los siguientes parámetros para determinar el número de tareas:

- El valor `minimumHealthyPercent` representa el límite mínimo del número de tareas que se deben ejecutar para un servicio durante una implementación o cuando una instancia de contenedor se está agotando, como un porcentaje del número deseado de tareas para el servicio. Este valor se redondea hacia arriba. Por ejemplo, si el porcentaje mínimo en buen estado es `50` y el número de tareas deseado es cuatro, entonces el programador puede detener dos tareas existentes antes de iniciar dos nuevas. Del mismo modo, si el porcentaje mínimo en buen estado es del 75 % y el recuento de tareas deseado es dos, entonces el programador no puede detener ninguna tarea debido a que el valor resultante también es dos.

  Si las tareas no funcionan correctamente, el programador de servicios de Amazon ECS iniciará primero las tareas de reemplazo y las mantendrá con el `minimumHealthyPercent` hasta que las tareas de reemplazo estén en buen estado. A medida que las tareas de reemplazo se inicien y vayan recuperando el estado correcto, las tareas en mal estado se detendrán gradualmente.
- El valor `maximumPercent` representa el límite máximo del número de tareas que se deben ejecutar para un servicio durante una implementación o cuando una instancia de contenedor se está agotando, como un porcentaje del número deseado de tareas para un servicio. Este valor se redondea hacia abajo. Por ejemplo, si el porcentaje máximo es `200` y el recuento de tareas deseado es cuatro, entonces el programador puede iniciar cuatro tareas nuevas antes de detener cuatro tareas existentes. Del mismo modo, si el porcentaje máximo es `125` y el recuento de tareas deseado es tres, el programador no puede iniciar ninguna tarea debido a que el valor resultante también es tres.

> **Important**
>
> Al establecer un porcentaje mínimo o uno máximo en buen estado, debe asegurarse de que el programador pueda detener o iniciar al menos una tarea cuando se inicia una implementación. Si la implementación del servicio está atascada debido a una configuración de implementación no válida, se enviará un mensaje de evento de servicio. Para obtener más información, consulte servicio (*`service-name`*) no pudo detener o iniciar tareas durante una implementación debido a la configuración de implementación del servicio. Actualice el valor minimumHealthyPercent o MaximumPercent y vuelva a intentarlo..

Las implementaciones continuas tienen dos métodos que proporcionan una forma de identificar rápidamente cuando se produce un error en la implementación de un servicio:

- the section called “ Detección de errores por el interruptor de circuito de implementación”
- the section called “Detección de errores en la implementación por las alarmas de CloudWatch”

Los métodos se pueden utilizar por separado o juntos. Si utiliza ambos métodos, la implementación se establece en un estado con errores en cuanto se cumplen los criterios de error de cualquiera de los métodos.

Utilice las siguientes directrices para determinar qué método debe utilizar:

- Interruptor: utilice este método cuando quiera detener una implementación en caso de que las tareas no puedan iniciarse.
- Alarmas de CloudWatch: utilice este método cuando quiera detener una implementación en función de las métricas de la aplicación.

Ambos métodos permiten revertir a la revisión de servicio anterior.

Resolución de imagen de contenedor

De forma predeterminada, Amazon ECS convierte las etiquetas de imágenes de contenedores especificadas en la definición de la tarea en resúmenes de imágenes de contenedores. Si crea un servicio que ejecuta y mantiene una sola tarea, esa tarea se utiliza para establecer los resúmenes de imágenes de los contenedores de la tarea. Si crea un servicio que ejecuta y mantiene varias tareas, la primera tarea iniciada por el programador de servicios durante la implementación se utiliza para establecer resúmenes de imágenes para los contenedores de las tareas.

Si se producen errores en tres o más intentos de establecer los resúmenes de imágenes del contenedor, la implementación continúa sin la resolución del resumen de la imagen. Si el interruptor de circuito de implementación está activado, la implementación también fallará y se revertirá.

Una vez establecidos los resúmenes de imágenes del contenedor, Amazon ECS los utiliza para iniciar cualquier otra tarea deseada y para cualquier futura actualización del servicio. Esto garantiza que todas las tareas de un servicio ejecuten siempre imágenes de contenedor idénticas, resultando en la consistencia de las versiones del software.

Puede configurar este comportamiento para cada contenedor de la tarea mediante el parámetro `versionConsistency` de la definición del contenedor. Para obtener más información, consulte `versionConsistency`.

Note

- Las versiones del agente de Amazon ECS inferiores a `1.31.0` no admiten la resolución de resúmenes de imágenes. Las versiones del agente `1.31.0` a `1.69.0` admiten la resolución de resúmenes de imágenes solo para las imágenes enviadas a los repositorios de Amazon ECR. Las versiones del agente `1.70.0` o superiores admiten la resolución de resúmenes de imágenes para todas las imágenes.
- La versión de la plataforma Fargate Linux mínima para la resolución de resumen de imágenes es `1.3.0`. La versión de la plataforma Fargate Windows mínima para la resolución de resumen de imágenes es `1.0.0`.
- Amazon ECS no captura resúmenes de contenedores sidecar administrados por Amazon ECS, como el agente de seguridad Amazon GuardDuty o el proxy Service Connect.
- Para reducir la latencia potencial asociada a la resolución de imágenes de contenedores en servicios con varias tareas, ejecute la versión del agente Amazon ECS `1.83.0` o superior en las instancias de contenedor EC2. Para evitar la latencia potencial, especifique los resúmenes de imágenes del contenedor en la definición de la tarea.
- Si se crea un servicio con un recuento de tareas deseado igual a cero, Amazon ECS no podrá establecer los resúmenes del contenedor hasta que active otra implementación del servicio con un recuento de tareas deseado superior a cero.
- Para establecer resúmenes de imágenes actualizados, puede forzar una nueva implementación. Los resúmenes actualizados se utilizarán para iniciar nuevas tareas y no afectarán a las tareas que ya estén en ejecución. Para obtener más información sobre cómo forzar nuevas implementaciones, consulte ForceNewDeployment en la referencia de la API de Amazon ECS.
- Cuando se utilizan proveedores de capacidad de EC2, si no hay suficiente capacidad para iniciar una tarea durante la implementación inicial, es posible que no se logre la coherencia de las versiones del software. Para garantizar que se mantenga la coherencia de las versiones incluso cuando la capacidad sea limitada, configure `versionConsistency: true` de forma explícita en la configuración del contenedor de definición de tareas en lugar de basarse en el comportamiento predeterminado. Esto hace que Amazon ECS espere hasta que haya capacidad disponible antes de continuar con la implementación.

Prácticas recomendadas para los parámetros de servicio de Amazon ECS

Para garantizar que no haya tiempo de inactividad de las aplicaciones, el proceso de implementación es el siguiente:

1. Inicie los nuevos contenedores de aplicaciones y mantenga en funcionamiento los contenedores existentes.
2. Compruebe que los nuevos contenedores estén en buen estado.
3. Detenga los contenedores antiguos.

En función de la configuración de implementación y de la cantidad de espacio libre y sin reservar en el clúster, es posible que se necesiten varias rondas para completar el proceso y sustituir todas las tareas antiguas por tareas nuevas.

Existen dos opciones de configuración del servicio que puede utilizar para modificar el número:

- `minimumHealthyPercent`: 100 % (predeterminado)

  El límite inferior del número de tareas de su servicio que deben permanecer en el estado `RUNNING` durante una implementación. Es un porcentaje de `desiredCount` que se redondea al número entero más cercano. Este parámetro le permite implementar sin utilizar capacidad de clúster adicional.
- `maximumPercent`: 200 % (predeterminado)

  El límite superior del número de tareas para su servicio que se permiten en el estado `RUNNING` o `PENDING` durante una implementación. Es un porcentaje de `desiredCount` que se redondea a la baja número entero más cercano.

Ejemplo: Opciones de configuración predeterminadas

Considere el siguiente servicio, que tiene seis tareas, implementado en un clúster con capacidad para ocho tareas en total. Las opciones de configuración del servicio predeterminadas no permiten que la implementación supere el 100 % de las seis tareas deseadas.

El proceso de implementación es el siguiente:

1. El objetivo es sustituir las seis tareas.

2. El programador inicia dos nuevas tareas porque la configuración predeterminada requiere que haya seis tareas en ejecución en todo momento.

   Ahora hay seis tareas existentes y dos nuevas en ejecución.
3. El programador detiene dos de las tareas existentes.

   Ahora hay cuatro tareas existentes y dos nuevas en ejecución.
4. El programador inicia dos nuevas tareas adicionales.

   Ahora hay cuatro tareas existentes y cuatro nuevas.
5. El programador detiene dos de las tareas existentes.

   Ahora hay dos tareas existentes y cuatro nuevas en ejecución.
6. El programador inicia dos nuevas tareas adicionales.

   Ahora hay dos tareas existentes y seis nuevas en ejecución.
7. El programador detiene las dos últimas tareas existentes.

   Ahora hay seis tareas nuevas en ejecución.

En el ejemplo anterior, si utiliza los valores predeterminados para las opciones, tendrá que esperar 2 minutos y medio para que se inicie una nueva tarea. Además, es posible que el equilibrador de carga tenga que esperar 5 minutos para que se detenga la tarea anterior.

Ejemplo: Modificar **`minimumHealthyPercent`**

Puede acelerar la implementación estableciendo el valor `minimumHealthyPercent` en un 50 %.

Considere el siguiente servicio, que tiene seis tareas, implementado en un clúster con capacidad para ocho tareas en total. El proceso de implementación es el siguiente:

1. El objetivo es sustituir seis tareas.
2. El programador detiene tres de las tareas existentes.

   Todavía hay tres tareas existentes en ejecución que cumplen con el valor `minimumHealthyPercent`.
3. El programador inicia cinco nuevas tareas.

   Ahora hay tres tareas existentes y cinco nuevas.

4. El programador detiene las tres tareas existentes restantes.

   Ahora hay cinco tareas nuevas.
5. El programador inicia las tareas nuevas finales.

   Ahora hay seis tareas nuevas.

Ejemplo: Modificar el espacio libre del clúster

También puede agregar espacio libre adicional para poder ejecutar tareas adicionales.

Considere el siguiente servicio, que tiene seis tareas, implementado en un clúster con capacidad para diez tareas en total. El proceso de implementación es el siguiente:

1. El objetivo es sustituir las tareas existentes.
2. El programador detiene tres de las tareas existentes.

   Ahora hay tres tareas existentes.
3. El programador inicia seis tareas nuevas.

   Ahora tiene las tareas existentes y seis nuevas.
4. El programador detiene las tres tareas existentes.

   Ahora hay seis tareas nuevas.

Recomendaciones

Utilice los siguientes valores para las opciones de configuración del servicio cuando sus tareas estén inactivas durante algún tiempo y no tengan una tasa de uso elevada.

- `minimumHealthyPercent`: 50 %
- `maximumPercent`: 200 %

Creación de una implementación de actualización continua de Amazon ECS

Cree un servicio para ejecutar y mantener simultáneamente un número determinado de instancias de una definición de tarea en un clúster. Si una de las tareas falla o se detiene, el programador de servicios de Amazon ECS lanza otra instancia de su definición de tarea para sustituirla. Esto ayuda a mantener el número deseado de tareas en el servicio.

Decida los siguientes parámetros de configuración antes de crear un servicio:

- Hay dos opciones de computación que distribuyen sus tareas.
  - Una estrategia de proveedor de capacidad hace que Amazon ECS distribuya sus tareas en uno o varios proveedores de capacidad.
  - Un tipo de lanzamiento hace que Amazon ECS lance nuestras tareas directamente en Fargate o en las instancias de EC2 registradas en sus clústeres.
- Las definiciones de tareas que utilizan el modo de red `awsvpc` o los servicios configurados para utilizar un balanceador de carga deben tener una configuración de redes. De forma predeterminada, la consola selecciona la Amazon VPC predeterminada junto con todas las subredes y el grupo de seguridad predeterminado dentro de la Amazon VPC predeterminada.
- La estrategia de ubicación. La estrategia de ubicación de tareas predeterminada distribuye uniformemente las tareas entre las zonas de disponibilidad.

  Le recomendamos que utilice el reequilibrio de zonas de disponibilidad para garantizar una alta disponibilidad del servicio. Para obtener más información, consulte Equilibrio de un servicio de Amazon ECS entre zonas de disponibilidad.
- Cuando utiliza el Tipo de lanzamiento para su implementación de servicios, el servicio se inicia en las subredes de su VPC de clúster de forma predeterminada.
- En capacity provider strategy (estrategia de proveedor de capacidad), la consola selecciona una opción de computación de forma predeterminada. A continuación, se describe el orden que utiliza la consola para seleccionar un valor predeterminado:
  - Si su clúster tiene definida una estrategia de proveedor de capacidad por defecto, se seleccionará esa.
  - Si el clúster no tiene definida una estrategia de proveedores de capacidad predeterminada, pero sí tiene los proveedores de capacidad de Fargate agregados al clúster, se selecciona una estrategia de proveedores de capacidad personalizada que utiliza al proveedor de capacidad de `FARGATE`.
  - Si su clúster no tiene definida una estrategia de proveedor de capacidad por defecto, pero tiene uno o varios proveedores de capacidad de grupo de escalado automático agregados al clúster, se seleccionará la opción Utilizar personalizado (Avanzado) y tendrá que definir manualmente la estrategia.
  - Si el clúster no tiene definida ninguna estrategia de proveedores de capacidad predeterminada ni tiene proveedores de capacidad agregados al clúster, se selecciona el tipo de lanzamiento de Fargate.

- Las opciones predeterminadas de detección de errores en implementación son utilizar el disyuntor de implementación de Amazon ECS con la opción de reversión en caso de errores.

  Para obtener más información, consulte Detección de errores por el interruptor de circuito de implementación de Amazon ECS.

- Si quiere utilizar la opción de implementación azul/verde, determine de qué manera mueve CodeDeploy las aplicaciones. Están disponibles las siguientes opciones:
  - CodeDeployDefault.ECSAllAtOnce: desplaza todo el tráfico al contenedor de Amazon ECS actualizado de una vez.
  - CodeDeployDefault.ecsLinear10PercentEvery1Minutes: desplaza el 10 % del tráfico cada minuto hasta que se haya desplazado todo el tráfico.
  - CodeDeployDefault.ecsLinear10PercentEvery3Minutes: desplaza el 10 % del tráfico cada tres minutos hasta que se haya desplazado todo el tráfico.
  - CodeDeployDefault.ECSCanary10Percent5Minutes: desplaza el 10 % del tráfico en el primer incremento. El 90 por ciento restante se implementa cinco minutos más tarde.
  - CodeDeployDefault.ECSCanary10Percent15Minutes: desplaza el 10 % del tráfico en el primer incremento. El 90 por ciento restante se implementa 15 minutos más tarde.
- Decida si desea que Amazon ECS aumente o disminuya automáticamente la cantidad deseada de tareas del servicio. Para obtener información, consulte Escalado automático de su servicio de Amazon ECS.
- Si necesita una aplicación para conectarse a otras aplicaciones que se ejecutan en Amazon ECS, determine la opción que se adapte a su arquitectura. Para obtener más información, consulte Interconexión de los servicios de Amazon ECS.
- Cuando crea un servicio que utiliza un interruptor de Amazon ECS, Amazon ECS crea una implementación y una revisión de servicios. Estos recursos le permiten ver información detallada sobre el historial del servicio. Para obtener más información, consulte Visualización del historial de servicios mediante las implementaciones de servicios de Amazon ECS.

  Para obtener información sobre cómo crear un servicio con la AWS CLI, consulte create-service en la Referencia de la AWS Command Line Interface.

  Para obtener información sobre cómo crear un servicio mediante AWS CloudFormation, consulte AWS::ECS::Service en la Guía del usuario de AWS CloudFormation.

Creación de un servicio con las opciones predeterminadas

Puede utilizar la consola para crear e implementar rápidamente un servicio. El servicio tiene la siguiente configuración:

- Se implementa en la VPC y en las subredes asociadas a su clúster
- Implementa una tarea
- Utiliza la implementación continua
- Utiliza la estrategia del proveedor de capacidad con su proveedor de capacidad predeterminado
- Utiliza el disyuntor de implementación para detectar errores y establece la opción de restaurar automáticamente la implementación en caso de error

Para que la implementación de un servicio se lleve a cabo con los parámetros predeterminados, siga estos pasos.

Para crear un servicio (consola de Amazon ECS)

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, elija Clústeres.
3. En la página Clústeres, seleccione el clúster en el que va a crear el servicio.
4. En la pestaña Services (Servicios), elija Create (Crear).

   Aparecerá la página Crear servicio.
5. En Detalles del servicio, haga lo siguiente:

   a. En Definición de tareas, ingrese la familia y la revisión de definiciones de tareas que se va a utilizar.

   b. En Service name (Nombre del servicio), ingrese un nombre para el servicio.
6. En Configuración de implementación, haga lo siguiente:

   - En Desired tasks (Tareas deseadas), ingrese el número de tareas que se lanzarán y mantendrán en el servicio.
7. (Opcional) Para ayudar a identificar el servicio y las tareas, expanda la sección Tags (Etiquetas) y, a continuación, configure sus etiquetas.

   Para que Amazon ECS etiquete automáticamente todas las tareas recién lanzadas con el nombre del clúster y las etiquetas de definición de tareas, seleccione Turn on Amazon ECS

managed tags (Activar las etiquetas gestionadas de Amazon ECS) y, a continuación, seleccione Task definitions (Definiciones de tareas).

Para que Amazon ECS etiquete automáticamente todas las tareas recién lanzadas con el nombre del clúster y las etiquetas de servicio, seleccione Turn on Amazon ECS managed tags (Activar las etiquetas gestionadas de Amazon ECS) y, a continuación, seleccione Service (Servicio).

Añada o elimine una etiqueta.

- [Agregar una etiqueta] Seleccione Add tag (Agregar etiqueta), y, a continuación, haga lo siguiente:
  - En Clave, escriba el nombre de la clave.
  - En Valor, escriba el valor de la clave.
- [Eliminar una etiqueta] Junto a la etiqueta, elija Remove tag (Quitar etiqueta).

Creación de un servicio a partir de los parámetros definidos

Para crear un servicio a partir de los parámetros definidos, siga estos pasos.

Para crear un servicio (consola de Amazon ECS)

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. Determine el recurso desde el que lanza el servicio.

Para iniciar un servicio desde	Pasos	
Clústeres	a. En la página Clusters (Clústeres), seleccione el clúster que va a crear el servicio. b. En la pestaña Services (Servicios), elija Create (Crear).	

Para iniciar un servicio desde	Pasos	
Definición de tarea	a. En la página de definiciones de tareas, seleccione el botón de opción situado junto a la definición de la tarea. b. En el menú Implementar, elija Crear servicio.	

Aparecerá la página Crear servicio.

3. En Detalles del servicio, haga lo siguiente:

   a. En Definición de tarea, ingrese la definición de la tarea que va a usar. A continuación, en Revisión, elija la revisión que desee utilizar.

   b. En Service name (Nombre del servicio), ingrese un nombre para el servicio.

4. En Clúster existente, elija el clúster.

   Elija Crear clúster para ejecutar la tarea en un clúster nuevo

5. Elija cómo distribuir las tareas en la infraestructura de clúster. En Configuración de computación, elija su opción.

Opción de computación	Pasos	
Estrategia de proveedores de capacidad	a. En Opciones de computación, elija Estrategia del proveedor de capacidad. b. Elija una estrategia: • Para utilizar la estrategia de proveedores de capacidad predeterminada del clúster, elija Use cluster default	

Opción de computación	Pasos	
	(Usar clúster predeterminado). • Si el clúster no tiene ninguna estrategia de proveedores de capacidad predeterminada o si desea utilizar una estrategia personalizada, elija Utilizar predeterminada, Agregar estrategia de proveedores de capacidad y, luego, defina la estrategia de proveedores de capacidad personalizada especificando una Base, Proveedor de capacidad y Peso. **Note** Para utilizar un proveedor de capacidad en una estrategia, el proveedor de capacidad debe estar asociado con el clúster.	

Opción de computación	Pasos	
Tipo de lanzamiento	a. En la sección Compute options (Opciones de computación), seleccion e Lauch type (Tipo de lanzamiento). b. En Launch type (Tipo de lanzamiento), elija un tipo de lanzamiento. c. (Opcional) Cuando se especifica el tipo de lanzamiento de Fargate, en Platform version (Versión de la plataform a) especifique la versión de la plataforma que se va a utilizar. Si no se especifica una versión de la plataforma, se utilizará la versión `LATEST`.	

6. En Configuración de implementación, haga lo siguiente:

   a. En Service type (Tipo de servicio), elija la estrategia de programación del servicio.

      - Para que el programador implemente exactamente una tarea en cada instancia de contenedor activa que cumpla con todas las restricciones de colocación de tareas, elija Daemon.
      - Para que el programador coloque y mantenga el número deseado de tareas en su clúster, elija Replica (Réplica).

   b. Si eligió Replica (Réplica), en Desired tasks (Tareas deseadas), ingrese el número de tareas que se lanzarán y mantendrán en el servicio.

   c. Si eligió Réplica para que Amazon ECS supervise la distribución de las tareas entre las zonas de disponibilidad y las redistribuya cuando haya un desequilibrio, en Reequilibrio del servicio de zonas de disponibilidad, seleccione Reequilibrio del servicio de zonas de disponibilidad.

d. En Periodo de gracia de la comprobación de estado, ingrese la cantidad de tiempo (en segundos) durante la cual el programador de servicios ignora las comprobaciones de estado de Elastic Load Balancing, VPC Lattice y contenedor en mal estado después de que se haya iniciado una tarea por primera vez. Si no se especifica ningún valor para el período de gracia de comprobación de estado, se utiliza el valor predeterminado: 0.

e. Determine el tipo de implementación de su servicio. Amplíe Opciones de implementación y, a continuación, especifique los siguientes parámetros.

Tipo de implementación	Pasos	
Actualización continua	a. En Max running tasks (Máximo de tareas en ejecución), ingrese el límite máximo del número de tareas del servicio que se permiten en el estado `RUNNING` durante una implement ación, como porcentaj e del número de tareas deseado del servicio (redondeado al entero inferior más próximo). Para obtener más información, consulte Configuración de la implementación. b. En Max running tasks (Máximo de tareas en ejecución), ingrese el límite máximo del número de tareas del servicio que se permiten en el estado `RUNNING` o `PENDING` durante una implementación, como porcentaje del número de tareas deseado del servicio (redondeado al entero inferior más próximo).	

Tipo de implementación	Pasos	
Implementación azul/verde	a. En Configuración de implementación, elija de qué manera dirige CodeDeploy el tráfico de producción al conjunto de tareas de reemplazo durante la implement ación. b. En Rol de servicio de CodeDeploy, elija el rol de IAM que utiliza el servicio para realizar solicitudes de API a los Servicios de AWS autorizados.	

f. Para configurar el modo en que Amazon ECS detecta y gestiona los errores de implementación, expanda Deployment failure detection (Detección de errores de implementación) y, a continuación, elija sus opciones.

   i. Para detener una implementación cuando las tareas no puedan iniciarse, seleccione Use the Amazon ECS deployment circuit breaker (Utilizar el interruptor de circuito de implementación de Amazon ECS).

      Para que el software restaure automáticamente la implementación a su último estado completado cuando el disyuntor de implementación establezca un estado con error, seleccione Restauración en caso de error.

   ii. Para detener una implementación en función de las métricas de la aplicación, seleccione Use CloudWatch alarms. A continuación, elija las alarmas en Nombre de la alarma de CloudWatch. Para crear una alarma nueva, vaya a la consola de CloudWatch.

      Para que el software restaure automáticamente la implementación a su último estado de implementación completada cuando una alarma de CloudWatch establezca un estado con error, seleccione Restauración en caso de error.

7. Si la definición de la tarea utiliza el modo de red `awsvpc`, puede especificar una configuración de red personalizada, expandir Redes y, a continuación, hacer lo siguiente.

   a. En VPC, seleccione la VPC que se va a usar.

   b. En Subnets (Subredes), seleccione una o varias subredes de la VPC que el programador de tareas considera al ubicar sus tareas.

   c. En Security groups (Grupos de seguridad), puede seleccionar un grupo de seguridad existente o crear uno nuevo. Para utilizar un grupo de seguridad existente, seleccione el grupo de seguridad y continúe con el próximo paso. Para crear un grupo de seguridad, elija Create a new security group (Crear un grupo de seguridad nuevo). Debe especificar un nombre de grupo de seguridad, una descripción y, a continuación, agregar una o varias reglas de entrada para el grupo de seguridad.

   d. En Public IP (IP pública), elija si desea asignar automáticamente una dirección IP pública a la interfaz de red elástica (ENI) de la tarea.

      Las tareas de AWS Fargate pueden recibir una dirección IP pública cuando se ejecuten mediante una subred pública para que tengan una ruta a Internet. A las tareas de EC2 no se les puede asignar una IP pública mediante este campo. Para obtener más información, consulte Opciones de redes de tareas de Amazon ECS para el tipo de lanzamiento de Fargate y Asignar una interfaz de red para una tarea de Amazon ECS.

8. (Opcional) Para interconectar su servicio con Service Connect, expanda Service Connect y, a continuación, especifique lo siguiente:

   a. Seleccione Activar Service Connect.

   b. En Service Connect configuration (Configuración de Service Connect), especifique el modo cliente.

      - Si su servicio ejecuta una aplicación cliente de red que solo necesita conectarse a otros servicios del espacio de nombres, elija Solo en el lado del cliente.
      - Si su servicio ejecuta una aplicación de servicio web o red y necesita proporcionar puntos de conexión para este servicio y se conecta a otros servicios del espacio de nombres, elija Client and server (Cliente y servidor).

   c. Para usar un espacio de nombres que no sea el espacio de nombres predeterminado del clúster, en Namespace (Espacio de nombres), elija el espacio de nombres del servicio.

   d. (Opcional) Especifique una configuración de registro. Seleccione Utilizar colección de registros. La opción predeterminada envía registros de contenedor a los Registros de

CloudWatch. Las demás opciones del controlador de registro se configuran mediante AWS FireLens. Para obtener más información, consulte Envío de registros de Amazon ECS a un servicio de AWS o AWS Partner.

A continuación, se describe con más detalle cada uno de los destinos de registro de contenedor.

- Amazon CloudWatch: configure la tarea para enviar registros de contenedor a CloudWatch Logs. Se proporcionan las opciones de controlador de registro predeterminadas que crean un grupo de registros de CloudWatch en su nombre. Para especificar otro nombre de grupo de registros, cambie los valores de las opciones del controlador.
- Amazon Data Firehose: configure la tarea para enviar registros de contenedor a Firehose. Se proporcionan las opciones de controlador de registro predeterminadas que envían registros a un flujo de entrega de Firehose. Para especificar un nombre de flujo de entrega distinto, cambie los valores de las opciones del controlador.
- Amazon Kinesis Data Streams: configure la tarea para enviar registros de contenedores a Kinesis Data Streams. Se proporcionan las opciones de controlador de registro predeterminadas que envían registros a un flujo de Kinesis Data Streams. Para especificar otro nombre de transmisión, cambie los valores de las opciones del controlador.
- Amazon OpenSearch Service: configure la tarea para enviar registros de contenedor a un dominio de OpenSearch Service. Se deben proporcionar las opciones del controlador de registros.
- Amazon S3: configure la tarea para enviar registros de contenedor a un bucket de Amazon S3. Se proporcionan las opciones de controlador de registro predeterminadas, pero debe especificar un nombre de bucket de Amazon S3 válido.

9. (Opcional) Para interconectar su servicio con Detección de servicios, expanda la opción Detección de servicios y, a continuación, haga lo siguiente.

   a. Seleccione Utilizar la detección de servicios.

   b. Para usar un espacio de nombres nuevo, seleccione Crear un nuevo espacio de nombres en Configurar el espacio de nombres y, a continuación, proporcione un nombre y una descripción del espacio de nombres. Para usar un espacio de nombres existente, elija Seleccione un espacio de nombres existente y, a continuación, elija el espacio de nombres que desea usar.

c. Proporcione la información del servicio de detección de servicios, como el nombre y la descripción del servicio.

d. Para que Amazon ECS ejecute comprobaciones de estado periódicas por contenedor, seleccione Habilitar la propagación del estado de las tareas de Amazon ECS.

e. En Tipo de registro DNS, seleccione el tipo de registro DNS para el servicio. La detección de servicios de Amazon ECS solo admite registros A y SRV, según el modo de red que se especifique en la definición de tareas. Para obtener más información sobre estos tipos de registros, consulte Tipos de registros de DNS que se admiten en la Guía para desarrolladores de Amazon Route 53.

   - Si la definición de tarea que especifica su tarea de servicio utiliza el modo de red `bridge` o `host`, solo se admiten los registros de tipo SRV. Elija una combinación de nombre y contenedor de contenedor para asociarla con el registro.
   - Si la definición de tarea que especifica su tarea de servicio utiliza el modo de red `awsvpc`, seleccione el tipo de registro A o SRV. Si eligió A, vaya al siguiente paso. Si está seleccionado el tipo SRV, especifique el puerto en el que se puede encontrar el servicio o una combinación de nombre y puerto de contenedor para asociarla con el registro.

   En el caso del TTL, introduzca el tiempo en segundos en el que los solucionadores de DNS y los navegadores web almacenan en caché un conjunto de registros.

10. (Opcional) Para interconectar su servicio con VPC Lattice, expanda la opción VPC Lattice y, a continuación, haga lo siguiente:

    a. Seleccione Activar VPC Lattice

    b. En Rol de infraestructura, elija el rol de infraestructura.

       Si no ha creado un rol, elija Crear rol de infraestructura.

    c. En Grupos de destino, elija el grupo o grupos de destino. Debe elegir un mínimo de un grupo de destino y un máximo de cinco. Elija Agregar grupo de destino para agregar grupos de destino adicionales. Elija el nombre del puerto, el protocolo y el puerto para cada grupo de destino que elija.

       Para eliminar un grupo de destino, elija Eliminar.

> **Note**
>
> - Si quiere agregar grupos de destino existentes, debe utilizar la AWS CLI. Para obtener instrucciones sobre cómo agregar grupos de destino mediante la AWS CLI, consulte register-targets en la Referencia de AWS Command Line Interface.
> - Si bien un servicio de VPC Lattice puede tener varios grupos de destino, cada grupo de destino solo se puede agregar a un servicio.

d. Para completar la configuración de VPC Lattice, mediante la inclusión de los nuevos grupos de destino en la acción predeterminada del agente de escucha o en las reglas de un servicio de VPC Lattice existente en la consola de VPC Lattice. Para obtener más información, consulte Reglas de oyente para el servicio de VPC Lattice.

11. (Opcional) Para configurar un equilibrador de carga para el servicio, expanda Load balancing (Equilibrio de carga).

    Elija el equilibrador de carga.

Para usar este equilibrador de carga	Haga lo siguiente	
Equilibrador de carga de aplicación	a. En Select load balancer type (Seleccionar tipo de balanceador de carga), elija Application Load Balancer. b. Elija Create a new load balancer (Crear un nuevo balanceador de carga) para crear un nuevo Application Load Balancer o Use an existing load balancer (Utilizar un balanceador de carga existente) para seleccion	

Para usar este equilibrador de carga	Haga lo siguiente	
	ar un Application Load Balancer existente. c. En Load balancer name (Nombre del equilibra dor de carga), ingrese un nombre único. d. En Choose container to load balance (Seleccio nar contenedor para equilibrar la carga), elija el contenedor que aloja el servicio. e. En Listener (Agente de escucha), ingrese un puerto y un protocolo que el equilibrador de carga de aplicación pueda utilizar para escuchar las solicitudes de conexión. De forma predeterminada, el balanceador de carga se configurará para utilizar el puerto 80 y HTTP. f. En Target group name (Nombre del grupo de destino), ingrese un nombre y un protocolo para el grupo de destino al que el equilibrador de carga de aplicación dirige las solicitudes. De forma predeterminada, el grupo de destino dirige	

Para usar este equilibrador de carga	Haga lo siguiente	
	las solicitudes al primer contenedor definido en la definición de la tarea. g. En Retardo de anulación del registro, ingrese el número de segundos al que el equilibrador de carga debe cambiar el estado de destino a `UNUSED`. El valor predeterminado es de 300 segundos. h. En Health check path (Ruta de comprobación de estado), ingrese una ruta existente dentro de su contenedor donde el equilibrador de carga de aplicación envíe periódica mente solicitudes para verificar el estado de la conexión entre el equilibra dor de carga de aplicació n y el contenedor. El valor predeterminado es el directorio raíz (`/`).	

Para usar este equilibrador de carga	Haga lo siguiente	
Equilibrador de carga de red	a. En Load balancer type (Tipo de equilibrador de carga), elija Network Load Balancer (Equilibrador de carga de red). b. En Load Balancer (Equilibrador de carga), elija un equilibrador de carga de red existente. c. En Choose container to load balance (Seleccio nar contenedor para equilibrar la carga), elija el contenedor que aloja el servicio. d. En Target group name (Nombre del grupo de destino), ingrese un nombre y un protocolo para el grupo de destino al que el equilibrador de carga de red dirige las solicitudes. De forma predeterminada, el grupo de destino dirige las solicitudes al primer contenedor definido en la definición de la tarea. e. En Retardo de anulación del registro, ingrese el número de segundos al que el equilibrador de carga debe cambiar	

Para usar este equilibrador de carga	Haga lo siguiente	
	el estado de destino a `UNUSED`. El valor predeterminado es de 300 segundos. f. En Health check path (Ruta de comprobación de estado), ingrese una ruta existente dentro de su contenedor donde el equilibrador de carga de red envíe periódicamente solicitudes para verificar el estado de la conexión entre el equilibrador de carga de aplicación y el contenedor. El valor predeterminado es el directorio raíz (`/`).	

12. (Opcional) Para configurar el escalado automático de servicios, expanda Escalado automático de servicios y, a continuación, especifique los siguientes parámetros. Para utilizar el escalado automático predictivo, que analiza los datos de cargas anteriores de los flujos de tráfico, configúrelo después de crear el servicio. Para obtener más información, consulte Uso de patrones históricos para escalar los servicios de Amazon ECS con escalado predictivo.

    a. Para utilizar el escalado automático de servicios, seleccione Service auto scaling (Escalado automático de servicios).

    b. En Cantidad mínima de tareas, ingrese el límite mínimo del número de tareas que se va a utilizar para el escalado automático del servicio. El recuento deseado no será inferior a este recuento.

    c. En Cantidad máxima de tareas, ingrese el límite máximo del número de tareas que se va a utilizar para el escalado automático del servicio. El recuento deseado no será superior a este recuento.

    d. Elija el tipo de política. En Tipo de política de escalado, elija una de las siguientes opciones.

Para utilizar este tipo de política	Haga lo siguiente	
Seguimiento de destino	a. Para Scaling policy type (Tipo de política de escalado), elija Target tracking (Seguimiento de destino). b. En Policy name (Nombre de la política), ingrese el nombre de la política. c. En Métrica de servicio de ECS, seleccione una de las siguientes métricas. • ECSServiceAverageCPUUtilization: uso medio de la CPU del servicio. • ECSServiceAverageMemoryUtilization: uso medio de la memoria del servicio. • ALBRequestCountPerTarget: número de peticiones completadas por destino en un grupo de destinos del equilibrador de carga de aplicación. d. En Target value (Valor de destino), ingrese el valor que el servicio	

Para utilizar este tipo de política	Haga lo siguiente	
	mantiene para la métrica seleccionada. e. En Periodo de recuperación de escalado horizontal, ingrese la cantidad de tiempo, en segundos, después de una actividad de escalado horizontal (agregar tareas) que debe transcurrir antes de que pueda iniciarse otra actividad de escalado horizontal. f. En Periodo de recuperación de desescalado horizontal, ingrese la cantidad de tiempo, en segundos, después de una actividad de reducción horizontal (eliminar tareas) que debe transcurrir antes de que pueda iniciarse otra actividad de reducción horizontal. g. Para evitar que la política lleve a cabo una actividad de reducción horizontal, seleccione Turn off scale-in (Desactivar la reducción horizontal).	

Para utilizar este tipo de política	Haga lo siguiente	
	h. • (Opcional) Seleccione Desactivar la acción de desescalar horizontalmente si desea que la política de escalado se escale horizontalmente para adaptarse al aumento del tráfico, pero no necesita que se reduzca horizontalmente cuando el tráfico disminuya.	

Para utilizar este tipo de política	Haga lo siguiente	
Escalado por pasos	a. Para Scaling policy type (Tipo de política de escalado), elija Step scaling (Escalado de pasos). b. En Nombre de política, ingrese el nombre de la política. c. En Alarm name, escriba un nombre único para la alarma. d. En Métrica de servicio de Amazon ECS, elija la métrica de servicio que desea utilizar para su alarma. e. En Estadística, elija la estadística de la alarma. f. En Período, elija el período de evaluación de la alarma. g. En Estado de alarma, elija cómo comparar la métrica seleccionada con el umbral definido. h. En Umbral para comparar las métricas y Período de evaluació n para iniciar la alarma, introduzca el umbral utilizado para la alarma	

Para utilizar este tipo de política	Haga lo siguiente	
	y durante cuánto tiempo se evaluará el umbral. i. En Acciones de escalado, haga lo siguiente: • Elija Acción si desea agregar, quitar o establecer un recuento específico para su servicio. • Si decide agregar o quitar las tareas, en Valor, ingrese el número de tareas (o porcentaje de las tareas existentes) que desea agregar o quitar cuando se inicie la acción de escalado. Si opta por establece r el recuento deseado, introduzca el número de tareas. En Tipo, seleccione si el Valor es un número entero o un porcentaje del recuento deseado existente. • En Límite inferior y Límite superior, introduzca el límite inferior y el límite superior del ajuste de	

Para utilizar este tipo de política	Haga lo siguiente	
	escalado de pasos. De forma predeterminada, el límite inferior para Agregar política es el umbral de la alarma y el límite superior es infinito positivo (+). De forma predeterminada, el límite superior de Quitar política es el límite de la alarma y el límite inferior es infinito negativo (-). • (Opcional) Agregue opciones de escalado adicionales. Seleccione Agregar nueva acción de escalado y, a continuación, repita los pasos de Acciones de escalado. • En Periodo de recuperación, ingrese la cantidad de tiempo, en segundos, que debe esperarse para que surta efecto una actividad de reducción horizontal anterior. En el caso de una política de ampliación, este es el momento en el que, después de una	

Para utilizar este tipo de política	Haga lo siguiente	
	actividad de escalado horizontal, la política de escalado bloquea las actividades de reducción horizontal y limita el número de tareas que se pueden escalar horizonta lmente a la vez. En el caso de una política de reducción horizonta l, este es el tiempo que debe transcurrir tras completarse una actividad de reducción horizontal antes de que pueda comenzar otra actividad de reducción horizontal.	

13. (Opcional) Para utilizar una estrategia de ubicación de tareas distinta a la predeterminada, expanda Task Placement (Ubicación de tareas) y, a continuación, elija una de las siguientes opciones.

    Para obtener más información, consulte Cómo coloca Amazon ECS las tareas en las instancias de contenedor.

    - Reparto equilibrado en AZ: distribuya las tareas en las zonas de disponibilidad y entre las instancias de contenedor dentro de cada zona de disponibilidad.
    - BinPack equilibrado en AZ: distribuya las tareas en las zonas de disponibilidad y entre las instancias de contenedor con la menor memoria disponible.
    - BinPack: distribuya las tareas en función de la cantidad mínima de CPU o memoria disponible.
    - Una tarea por Host: coloque como máximo una tarea del servicio en cada instancia de contenedor.

- Personalizado: defina su propia estrategia de colocación de tareas.

Si elige Custom (Personalizado), defina el algoritmo de ubicación de tareas y las reglas que se tienen en cuenta durante la ubicación de tareas.

- En Strategy (Estrategia), para Type (Tipo) y Field (Campo), elija el algoritmo y la entidad que quiere utilizar para el algoritmo.

  Puede ingresar un máximo de 5 estrategias.
- En Restricción, para Tipo y Expresión, elija la regla y el atributo para la restricción.

  Por ejemplo, para establecer la restricción de colocar las tareas en las instancias T2, para la Expresión, ingrese attribute:ecs.instance-type =~ t2.*.

  Puede ingresar un máximo de 10 restricciones.

14. Si su tarea usa un volumen de datos compatible con la configuración en el momento de la implementación, puede expandir Volume para configurar el volumen.

    El nombre del volumen y el tipo de volumen se configuran al crear una revisión de definición de tareas y no se pueden cambiar al crear un servicio. Para actualizar el nombre y el tipo del volumen, debe crear una nueva revisión de la definición de tareas y crear un servicio con la nueva revisión.

Para configurar este tipo de volumen	Haga lo siguiente	
Amazon EBS	a. En Tipo de volumen de EBS, elija el tipo de volumen de EBS que desee adjuntar a la tarea. b. En Tamaño (GiB), ingrese un valor válido para el tamaño de volumen en gibibytes (GiB). Puede especificar un tamaño de volumen mínimo de 1 GiB y máximo de 16 384 GiB.	

Para configurar este tipo de volumen	Haga lo siguiente	
	Este valor es obligatorio a menos que proporcione un ID de instantánea. c. En IOPS, ingrese el número máximo de operaciones de entrada/salida (IOPS) que debe proporcionar el volumen. Este valor solo puede configurarse para los tipos de volumen `io1`, `io2` y `gp3`. d. En Rendimiento (MiB/s), ingrese el rendimiento que debe proporcionar el volumen, en mebibytes por segundo (MiBps o MiB/s). Este valor solo puede configurarse para el tipo de volumen `gp3`. e. En ID de instantánea, elija una instantánea de volumen de Amazon EBS existente o ingrese el ARN de una instantánea si desea crear un volumen a partir de una instantánea. También puede crear un volumen nuevo y vacío sin elegir ni ingresar ningún ID de instantánea. f. Si especifica un ID de instantánea, puede	

Para configurar este tipo de volumen	Haga lo siguiente	
	especificar una velocidad de inicialización del volumen (MiB/s). Ingrese un valor entre 100 y 300, en MiB/s, que determina rá la rapidez con la que se cargan los datos de la instantánea especific ada mediante el ID de instantánea para la creación del volumen. g. En Tipo de sistema de archivos, elija el tipo de sistema de archivos que se utilizará para almacenar y recuperar datos en el volumen. Puede elegir el sistema operativo predeterminado o un tipo de sistema de archivos específico. El valor predeterminado para Linux es `XFS`. En el caso de los volúmenes creados a partir de una instantán ea, debe especificar el mismo tipo de sistema de archivos que utilizaba el volumen cuando se creó la instantánea. Si hay un error de coincidencia con el tipo de sistema de archivos, la tarea no podrá iniciarse.	

Para configurar este tipo de volumen	Haga lo siguiente	
	h. En Rol de infraestructura, elija un rol de IAM con los permisos necesarios que permitan a Amazon ECS administrar los volúmenes de Amazon EBS para las tareas. Puede adjuntar la política de `AmazonECSInfrastructureRolePolicyForVolumes` administrada al rol, o puede utilizar la política como guía para crear y adjuntar su propia política con los permisos que cumplan sus necesidades específicas. Para obtener información sobre los permisos de necesarios, consulte Rol de IAM de infraestructura de Amazon ECS. i. En Cifrado, elija Predeterminado si quiere usar el cifrado de Amazon EBS como configuración predeterminada. Si su cuenta tiene configurado Cifrado predeterminado, el volumen se cifrará con la clave AWS Key Management Service	

Para configurar este tipo de volumen	Haga lo siguiente	
	(AWS KMS) especificada en la configuración. Si selecciona Predeterminado y el cifrado predeterminado de Amazon EBS no está activado, el volumen se descifrará. Si elige Personalizado, puede especificar una AWS KMS key de su preferencia para el cifrado por volumen. Si selecciona Ninguno, el volumen no se cifrará a menos que tenga el cifrado configurado de forma predeterminada o si crea un volumen a partir de una instantánea cifrada. j. Si ha elegido Personalizado en Cifrado, debe especificar la AWS KMS key que desee utilizar. En Clave de KMS, elija una AWS KMS key o escriba un ARN. Si decide cifrar su volumen mediante una clave simétrica administrada por el cliente, asegúrese	

Para configurar este tipo de volumen	Haga lo siguiente	
	de tener los permisos correctos definidos en su política de AWS KMS key. Para obtener más información, consulte Data encryption for Amazon EBS volumes. k. (Opcional) En Etiquetas , puede propagar las etiquetas de la definición de la tarea o del servicio o proporcionar sus propias etiquetas para agregar etiquetas a su volumen de Amazon EBS. Si desea propagar etiquetas desde la definición de la tarea, seleccione Definición de tarea en Propagar etiquetas desde. Si desea propagar etiquetas desde el servicio, elija Servicio en Propagar etiquetas desde. Si elige No propagar o si no elige un valor, las etiquetas no se propagarán. Si quiere proporcionar sus propias etiquetas , seleccione Agregar etiqueta y, a continuación,	

Para configurar este tipo de volumen	Haga lo siguiente	
	proporcione la clave y el valor de cada etiqueta que agregue. Para obtener más información acerca del etiquetado de volúmenes de Amazon EBS, consulte Tagging Amazon EBS volumes.	

15. (Opcional) Para ayudar a identificar el servicio y las tareas, expanda la sección Tags (Etiquetas) y, a continuación, configure sus etiquetas.

    Para que Amazon ECS etiquete automáticamente todas las tareas recién lanzadas con el nombre del clúster y las etiquetas de definición de tareas, seleccione Activar las etiquetas administradas de Amazon ECS y, a continuación, en Propagar etiquetas de, seleccione Definiciones de tareas.

    Para que Amazon ECS etiquete automáticamente todas las tareas recién lanzadas con el nombre del clúster y las etiquetas de servicio, seleccione Activar las etiquetas administradas de Amazon ECS y, a continuación, en Propagar etiquetas de, seleccione Servicio.

    Añada o elimine una etiqueta.

    - [Agregar una etiqueta] Seleccione Add tag (Agregar etiqueta), y, a continuación, haga lo siguiente:
      - En Clave, escriba el nombre de la clave.
      - En Valor, escriba el valor de la clave.
    - [Eliminar una etiqueta] Junto a la etiqueta, elija Remove tag (Quitar etiqueta).
16. Seleccione Crear.

Pasos a seguir a continuación

Las siguientes son acciones adicionales que están disponibles después de crear un servicio.

- Configure el escalado automático predictivo, que analiza los datos de cargas anteriores de los flujos de tráfico. Para obtener más información, consulte Uso de patrones históricos para escalar los servicios de Amazon ECS con escalado predictivo.
- Haga un seguimiento de la implementación y consulte el historial de los servicios que utiliza el interruptor de Amazon ECS. Para obtener más información, consulte Visualización del historial de servicios mediante las implementaciones de servicios de Amazon ECS.

Implementaciones “blue/green” de Amazon ECS

Una estrategia de implementación azul/verde es una metodología de lanzamiento que reduce el tiempo de inactividad y el riesgo al ejecutar dos entornos de producción idénticos, denominados azul y verde. Con las implementaciones azul/verde de Amazon ECS, puede validar las nuevas revisiones de servicios antes de dirigir el tráfico de producción hacia ellas. Este método proporciona una forma más segura de implementar cambios con la capacidad de revertirlos rápidamente si es necesario.

Ventajas

El uso de las implementaciones azul/verde ofrece los siguientes beneficios:

- Reduce el riesgo mediante pruebas con el tráfico de producción antes de cambiar de producción. Puede validar la nueva implementación con tráfico de prueba antes de dirigir el tráfico de producción hacia él.
- Implementaciones sin tiempo de inactividad. El entorno de producción permanece disponible durante todo el proceso de implementación, lo que garantiza la disponibilidad continua del servicio.
- Reversión sencilla si se detectan problemas. Si surgen problemas con la implementación verde, puede volver rápidamente a la implementación azul sin interrumpir el servicio por periodos prolongados.
- Entorno de pruebas controlado. El entorno verde proporciona un espacio aislado para comprobar nuevas características con patrones de tráfico reales antes de su implementación completa.
- Proceso de implementación predecible. El método estructurado con etapas del ciclo de vida definidas hace que las implementaciones sean más coherentes y fiables.
- Validación automatizada mediante enlaces de ciclo de vida. Puede incorporar pruebas automatizadas en varias etapas de la implementación para verificar la funcionalidad.

Terminología

A continuación se indican los términos de implementación azul/verde de Amazon ECS:

- Tiempo de incorporación: el tiempo durante el cual las revisiones de servicio azul y verde se ejecutan simultáneamente después de que se haya transferido el tráfico de producción.
- Implementación azul: la revisión de servicio de producción actual que desea sustituir.
- Implementación verde: la nueva revisión de servicio que desea implementar.
- Etapas del ciclo de vida: serie de eventos de la operación de implementación, como “una transferencia de tráfico posterior a la producción”.
- Enlace de ciclo de vida: una función de Lambda que verifica la implementación en una etapa específica del ciclo de vida.
- Oyente: un recurso de Elastic Load Balancing que comprueba las solicitudes de conexión utilizando el protocolo y el puerto configurados. Las reglas que defina para un oyente determinan cómo Amazon ECS va a enrutar las solicitudes hacia sus destinos registrados.
- Regla: un recurso de Elastic Load Balancing asociado a un oyente. Una regla define cómo se enrutan las solicitudes y consta de una acción, una condición y una prioridad.
- Grupo de destino: un recurso de Elastic Load Balancing que se utiliza para enrutar las solicitudes a uno o más destinos registrados (por ejemplo, instancias de EC2). Cuando se crea un agente de escucha, especifica un grupo de destino para su acción predeterminada. El tráfico se reenvía al grupo de destino especificado en la regla del agente de escucha.
- Transferencia de tráfico: el proceso que Amazon ECS utiliza para transferir el tráfico de la implementación azul a la implementación verde. Para las implementaciones azul/verde de Amazon ECS, todo el tráfico se transfiere del servicio azul al servicio verde a la vez.

Consideraciones

Tenga en cuenta lo siguiente al elegir un tipo de implementación:

- Uso de recursos: las implementaciones azul/verde ejecutan temporalmente las revisiones de servicio azul y verde simultáneamente, lo que puede duplicar el uso de recursos durante las implementaciones.
- Monitoreo de la implementación: las implementaciones azul/verde proporcionan información más detallada sobre el estado de la implementación, lo que le permite monitorear cada etapa del proceso de implementación.
- Reversión: las implementaciones azul/verde facilitan la reversión a la versión anterior si se detectan problemas, ya que la revisión azul se mantiene activa hasta que vence el tiempo de incorporación.

Flujo de trabajo de implementación de servicios azul/verde de Amazon ECS

El proceso de implementación azul/verde de Amazon ECS sigue un método estructurado con seis fases distintas que garantizan actualizaciones de aplicaciones seguras y fiables. Cada fase tiene un propósito específico al validar y hacer la transición de la aplicación de la versión actual (azul) a la nueva versión (verde).

1. Fase de preparación: cree el entorno verde junto con el entorno azul existente. Esto incluye el aprovisionamiento de nuevas revisiones de los servicios y la preparación de los grupos de destino.
2. Fase de implementación: implemente la nueva revisión de servicio en un entorno verde. Amazon ECS lanza nuevas tareas mediante la revisión de servicio actualizada, mientras que el entorno azul sigue sirviendo al tráfico de producción.
3. Fase de prueba: valide el entorno verde mediante el enrutamiento del tráfico de prueba. El equilibrador de carga de aplicación dirige las solicitudes de prueba al entorno verde mientras que el tráfico de producción permanece en el entorno azul.
4. Fase de transferencia de tráfico: transfiera el tráfico de producción de azul a verde en función de la estrategia de implementación que haya configurado. Esta fase incluye puntos de control de monitoreo y validación.
5. Fase de monitoreo: monitoree el estado de las aplicaciones, las métricas de rendimiento y los estados de alarma durante el periodo correspondiente al tiempo de incorporación. Cuando se detectan problemas, se inicia una operación de reversión.
6. Fase de finalización: finalice la implementación cerrando el entorno azul o manteniéndolo para posibles escenarios de reversión, según la configuración.

Flujo de trabajo

El siguiente diagrama ilustra el flujo de trabajo integral de implementación azul/verde y muestra la interacción entre Amazon ECS y el equilibrador de carga de aplicación:

El flujo de trabajo de implementación mejorado incluye los siguientes pasos detallados:

1. Estado inicial: el servicio azul (producción actual) administra el 100 % del tráfico de producción. El equilibrador de carga de aplicación tiene un único oyente con reglas que enrutan todas las solicitudes al grupo de destino azul que contiene tareas azules en buen estado.
2. Aprovisionamiento de un entorno verde: Amazon ECS crea nuevas tareas mediante la definición de tareas actualizada. Estas tareas se registran en un nuevo grupo de destino verde, pero inicialmente no reciben tráfico.
3. Validación de comprobaciones de estado: el equilibrador de carga de aplicación realiza comprobaciones de estado de las tareas verdes. La implementación solo pasa a la siguiente fase cuando las tareas verdes superan las comprobaciones de estado.
4. Enrutamiento de tráfico de prueba: si están configuradas, las reglas de oyente del equilibrador de carga de aplicación enrutan patrones de tráfico específicos (como las solicitudes con encabezados de prueba) al entorno verde para su validación, mientras que el tráfico de producción permanece en el entorno azul. Esto lo controla el mismo oyente que gestiona el tráfico de producción mediante reglas diferentes en función de los atributos de las solicitudes.

5. Transferencia de tráfico de producción: transfiera el tráfico de producción de azul a verde en función de la configuración de implementación. En las implementaciones azul/verde de ECS, se trata de una transferencia inmediata (de una sola vez) en la que el 100 % del tráfico se traslada del entorno azul al verde. El equilibrador de carga de aplicación utiliza un único oyente con reglas correspondientes que controlan la distribución del tráfico entre los grupos de destino azules y verdes en función de las ponderaciones.
6. Monitoreo y validación: durante toda la transferencia de tráfico, Amazon ECS monitorea las métricas de CloudWatch, los estados de alarma y el estado de la implementación. Los activadores automáticos de reversión se activan si se detectan problemas.
7. Tiempo de incorporación: el tiempo durante el cual las revisiones de servicio azul y verde se ejecutan simultáneamente después de que se haya transferido el tráfico de producción.
8. Cierre del entorno azul: tras la validación y transferencia de tráfico correctos, se ciera el entorno azul para liberar recursos del clúster o se mantiene para poder revertirlo rápidamente.
9. Estado final: el entorno verde se convierte en el nuevo entorno de producción y gestiona el 100 % del tráfico. La implementación está marcada como exitosa.

Etapas del ciclo de vida de la implementación

El proceso de implementación azul/verde progresa a través de distintas etapas del ciclo de vida (una serie de eventos en la operación de implementación, como “una transferencia de tráfico posterior a la producción”), cada una con responsabilidades específicas y puntos de control de validación. Conocer estas etapas le ayuda a monitorear el progreso de la implementación y a solucionar los problemas de forma eficaz.

Cada etapa del ciclo de vida puede durar hasta 24 horas. Recomendamos que el valor se mantenga por debajo de la marca de 24 horas. Esto se debe a que los procesos asíncronos necesitan tiempo para activar los enlaces. El sistema agota el tiempo de espera, no se realiza la implementación y, después, inicia una reversión cuando una etapa alcanza las 24 horas. Las implementaciones de AWS CloudFormation tienen restricciones de tiempo de espera adicionales. Si bien el límite por etapas de 24 horas sigue vigente, AWS CloudFormation impone un límite de 36 horas para toda la implementación. Si el proceso no se completa en 36 horas, AWS CloudFormation marca un error en la implementación y, a continuación, se inicia una reversión.

Etapas del ciclo de vida	Descripción	¿Desea utilizar esta etapa para el enlace de ciclo de vida?
RECONCILE_SERVICE	Esta etapa solo ocurre cuando se inicia una nueva implementación de servicio con más de una revisión de servicio en estado ACTIVO.	Sí
PRE_SCALE_UP	La revisión de servicio verde no ha comenzado. La revisión de servicio azul gestiona el 100 % del tráfico de producció n. No hay tráfico de prueba.	Sí
SCALE_UP	El momento en que la revisión de servicio verde se escala verticalmente al 100 % e inicia nuevas tareas. La revisión de servicio verde no sirve ningún tráfico en este momento.	No
POST_SCALE_UP	La revisión de servicio verde ha comenzado. La revisión de servicio azul gestiona el 100 % del tráfico de producció n. No hay tráfico de prueba.	Sí
TEST_TRAFFIC_SHIFT	Las revisiones de servicio azul y verde están en curso. La revisión de servicio azul gestiona el 100 % del tráfico de producción. La revisión de servicio verde está migrando del 0 % al 100 % del tráfico de prueba.	Sí
POST_TEST_TRAFFIC_ SHIFT	Se ha completado la transferencia de tráfico de prueba. La revisión de servicio verde gestiona el 100 % del tráfico de prueba.	Sí

Etapas del ciclo de vida	Descripción	¿Desea utilizar esta etapa para el enlace de ciclo de vida?
PRODUCTION_TRAFFIC_SHIFT	El tráfico de producción se está transfiriendo a la revisión de servicio verde. La revisión de servicio verde está migrando del 0 % al 100 % del tráfico de producción.	Sí
POST_PRODUCTION_TRAFFIC_SHIFT	Se ha completado la transferencia de tráfico de producción.	Sí
BAKE_TIME	El tiempo durante el cual las revisiones de servicio azul y verde se ejecutan simultáneamente.	No
CLEAN_UP	La revisión de servicio azul se ha reducido verticalmente por completo a 0 tareas en curso. Tras esta etapa, la revisión de servicio verde es ahora la revisión de servicio de producción.	No

Cada etapa del ciclo de vida incluye puntos de control de validación integrados que deben superarse antes de pasar a la siguiente etapa. Si se produce un error en la validación, la implementación se puede revertir automáticamente para mantener la disponibilidad y la fiabilidad del servicio.

Recursos necesarios para las implementaciones azul/verde de Amazon ECS

Para utilizar una implementación azul/verde con transferencia de tráfico administrada, su servicio debe utilizar una de las siguientes características:

- Elastic Load Balancing
- Service Connect

Los servicios que no utilizan Detección de servicios, Service Connect, VPC Lattice o Elastic Load Balancing también pueden utilizar implementaciones azul/verde, pero no obtienen ninguno de los beneficios de la transferencia de tráfico administrada.

En la siguiente lista se proporciona una descripción general de alto nivel de lo que se debe configurar para las implementaciones azul/verde de Amazon ECS:

- Su servicio utiliza un equilibrador de carga de aplicación, un equilibrador de carga de red o Service Connect. Configure los recursos adecuados.
  - Equilibrador de carga de aplicación: para más información, consulte Recursos de equilibrador de carga de aplicación para implementaciones azul/verde.
  - Equilibrador de carga de red: para más información, consulte Recursos del equilibrador de carga de red para las implementaciones azul/verde de Amazon ECS.
  - Service Connect: para más información, consulte Recursos de Service Connect para implementaciones azul/verde de Amazon ECS.
- Establezca el controlador de implementación del servicio en `ECS`.
- Configure la estrategia de implementación como `blue/green` en su definición de servicio.
- Opcionalmente, configure parámetros adicionales, como:
  - Tiempo de incorporación para la nueva implementación
  - Alarmas de CloudWatch para la reversión automática
  - Enlaces de ciclo de vida de la implementación para realizar pruebas (son funciones de Lambda que se ejecutan en etapas de implementación específicas)

Prácticas recomendadas

Siga estas prácticas recomendadas para una implementación azul/verde de Amazon ECS exitosa:

- Configure las comprobaciones de estado adecuadas que reflejen con precisión el estado de su aplicación.
- Establezca un tiempo de incorporación que permita realizar pruebas suficientes de la implementación verde.
- Implemente alarmas de CloudWatch para detectar automáticamente los problemas y activar las reversiones.
- Utilice los enlaces de ciclo de vida para realizar pruebas automatizadas en cada etapa de la implementación.

- Asegúrese de que su aplicación pueda gestionar entornos azules y verdes que se ejecuten simultáneamente.
- Planifique una capacidad de clúster suficiente para gestionar ambos entornos durante la implementación.
- Compruebe sus procedimientos de reversión antes de implementarlos en producción.

Recursos de equilibrador de carga de aplicación para implementaciones azul/verde

Para utilizar los equilibradores de carga de aplicación con las implementaciones azul/verde de Amazon ECS, debe configurar recursos específicos que permitan el enrutamiento del tráfico entre las revisiones de servicio azul y verde.

Limitaciones

Las siguientes limitaciones se aplican actualmente a las implementaciones azul/verde con equilibradores de carga de aplicación:

- Las implementaciones azul/verde no se admiten cuando se utilizan oyentes de equilibrador de carga de aplicación que tienen varios tipos de acciones configurados con acciones que no son de remisión. Si las reglas de oyentes del equilibrador de carga de aplicación contienen una combinación de acciones de remisión y no remisión (como acciones de redireccionamiento, respuesta fija o autenticación), la implementación fallará.

Grupos de destino

Para las implementaciones azul/verde con Elastic Load Balancing, debe crear dos grupos de destino:

- Un grupo de destino principal para la revisión de servicio azul (tráfico de producción actual)
- Un grupo de destino alternativo para la revisión de servicio verde (nueva versión)

Ambos grupos de destino deben configurarse con los siguientes ajustes:

- Tipo de destino: `IP` (para Fargate o EC2 con modo de red `awsvpc`)
- Protocolo: `HTTP` (o el protocolo que utilice su aplicación)
- Puerto: el puerto en el que escucha su aplicación (normalmente `80` para HTTP)
- VPC: la misma VPC que sus tareas de Amazon ECS

- Configuración de comprobación de estado: configurada para comprobar correctamente el estado de la aplicación

Durante una implementación azul/verde, Amazon ECS registra automáticamente las tareas con el grupo de destino adecuado en función de la etapa de implementación.

Example Creación de grupos de destino para un equilibrador de carga de aplicación

Los siguientes comandos de la CLI crean dos grupos de destino para usarlos con un equilibrador de carga de aplicación en una implementación azul/verde:

```
aws elbv2 create-target-group \
    --name blue-target-group \
    --protocol HTTP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-path / \
    --health-check-protocol HTTP \
    --health-check-interval-seconds 30 \
    --health-check-timeout-seconds 5 \
    --healthy-threshold-count 2 \
    --unhealthy-threshold-count 2

aws elbv2 create-target-group \
    --name green-target-group \
    --protocol HTTP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-path / \
    --health-check-protocol HTTP \
    --health-check-interval-seconds 30 \
    --health-check-timeout-seconds 5 \
    --healthy-threshold-count 2 \
    --unhealthy-threshold-count 2
```

Equilibrador de carga de aplicación

Debe crear un equilibrador de carga de aplicación con la siguiente configuración:

- Esquema: con acceso a Internet o interno, según sus necesidades

- Tipo de dirección IP: IPv4
- VPC: la misma VPC que sus tareas de Amazon ECS
- Subredes: al menos dos subredes en diferentes zonas de disponibilidad
- Grupos de seguridad: un grupo de seguridad que permita el tráfico en los puertos de oyentes

El grupo de seguridad adjunto al equilibrador de carga de aplicación debe tener una regla de salida que permita el tráfico al grupo de seguridad adjunto a las tareas de Amazon ECS.

Example Creating an Application Load Balancer (Crear un equilibrador de carga de aplicación)

El siguiente comando de la CLI crea un equilibrador de carga de aplicación para su uso en una implementación azul/verde:

```
aws elbv2 create-load-balancer \
    --name my-application-load-balancer \
    --type application \
    --security-groups sg-abcd1234 \
    --subnets subnet-12345678 subnet-87654321
```

Oyentes y reglas

En el caso de las implementaciones azul/verde, debe configurar los oyentes en su equilibrador de carga de aplicación:

- Oyente de producción: gestiona el tráfico de producción (normalmente en los puertos 80 o 443)
  - Inicialmente remite el tráfico hacia el grupo de destino principal (revisión de servicio azul)
  - Tras la implementación, remite el tráfico al grupo de destino alternativo (revisión de servicio verde)
- Oyente de prueba (opcional): gestiona el tráfico de prueba para validar la revisión de servicio verde antes de transferir el tráfico de producción
  - Se puede configurar en un puerto diferente (por ejemplo, 8080 o 8443)
  - Remite el tráfico al grupo de destino alternativo (revisión de servicio verde) durante la fase de prueba

Durante una implementación azul/verde, Amazon ECS actualiza automáticamente las reglas de oyente para que enruten el tráfico hacia el grupo de destino adecuado en función de la etapa de implementación.

Example Creación de un oyente de producción

El siguiente comando de la CLI crea un oyente de producción en el puerto 80 que remite el tráfico al grupo de destino principal (azul):

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
app/my-application-load-balancer/abcdef123456 \
    --protocol HTTP \
    --port 80 \
    --default-actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
blue-target-group/abcdef123456
```

Example Creación de un oyente de prueba

El siguiente comando de la CLI crea un oyente de prueba en el puerto 8080 que remite el tráfico al grupo de destino alternativo (verde):

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
app/my-application-load-balancer/abcdef123456 \
    --protocol HTTP \
    --port 8080 \
    --default-actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
green-target-group/ghijkl789012
```

Example Creación de una regla de oyente para el enrutamiento basado en rutas

El siguiente comando de la CLI crea una regla que remite el tráfico de una ruta específica al grupo de destino verde para llevar a cabo pruebas:

```
aws elbv2 create-rule \
    --listener-arn arn:aws:elasticloadbalancing:region:123456789012:listener/app/my-
application-load-balancer/abcdef123456/ghijkl789012 \
    --priority 10 \
    --conditions Field=path-pattern,Values='/test/*' \
    --actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
green-target-group/ghijkl789012
```

Example Creación de una regla de oyente para el enrutamiento basado en encabezados

El siguiente comando de la CLI crea una regla que remite el tráfico con un encabezado específico al grupo de destino verde para llevar a cabo pruebas:

```
aws elbv2 create-rule \
    --listener-arn arn:aws:elasticloadbalancing:region:123456789012:listener/app/my-application-load-balancer/abcdef123456/ghijkl789012 \
    --priority 20 \
    --conditions Field=http-header,HttpHeaderConfig='{Name=X-Environment,Values=[test]}' \
    --actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/green-target-group/ghijkl789012
```

Configuración del servicio

Debe tener permisos para autorizar a Amazon ECS que en su nombre administre los recursos del equilibrador de carga en sus clústeres. Para obtener más información, consulte Rol de IAM de infraestructura de Amazon ECS para los equilibradores de carga.

Al crear o actualizar un servicio de Amazon ECS para las implementaciones azul/verde con Elastic Load Balancing, debe especificar la siguiente configuración.

Sustituya las *entradas del usuario* por sus valores.

Los componentes clave de esta configuración son:

- `targetGroupArn`: el ARN del grupo de destino principal (revisión de servicio azul).
- `alternateTargetGroupArn`: el ARN del grupo de destino alternativo (revisión de servicio verde).
- `productionListenerRule`: el ARN de la regla de oyente para el tráfico de producción.
- `roleArn`: el ARN del rol que permite que Amazon ECS administre los recursos de Elastic Load Balancing.
- `strategy`: configúrelo en `BLUE_GREEN` para habilitar las implementaciones azul/verde.
- `bakeTimeInMinutes`: el tiempo durante el cual las revisiones de servicio azul y verde se ejecutan simultáneamente después de que se haya transferido el tráfico de producción.
- `TestListenerRule`: el ARN de la regla de oyente para el tráfico de prueba. Se trata de un parámetro opcional.

```
{
    "loadBalancers": [
        {
            "targetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/primary-target-group/
abcdef123456",
            "containerName": "container-name",
            "containerPort": 80,
            "advancedConfiguration": {
                "alternateTargetGroupArn":
 "arn:aws:elasticloadbalancing:region:account-id:targetgroup/alternate-target-group/
ghijkl789012",
                "productionListenerRule": "arn:aws:elasticloadbalancing:region:account-
id:listener-rule/app/load-balancer-name/abcdef123456/listener/ghijkl789012/rule/
mnopqr345678",
                "roleArn": "arn:aws:iam::123456789012:role/ecs-elb-role"
            }
        }
    ],
    "deploymentConfiguration": {
        "strategy": "BLUE_GREEN",
        "maximumPercent": 200,
        "minimumHealthyPercent": 100,
        "bakeTimeInMinutes": 5
    }
}
```

Flujo de tráfico durante la implementación

Durante una implementación azul/verde con Elastic Load Balancing, el tráfico fluye por el sistema de la siguiente manera:

1. Estado inicial: todo el tráfico de producción se enruta al grupo de destino principal (revisión de servicio azul).
2. Implementación de la revisión de servicio verde: Amazon ECS implementa las nuevas tareas y las registra en el grupo de destino alternativo.
3. Tráfico de prueba: si se configura un oyente, el tráfico de prueba se enruta al grupo de destino alternativo para validar la revisión de servicio verde.
4. Transferencia de tráfico de producción: Amazon ECS actualiza la regla de oyentes de producción para enrutar el tráfico al grupo de destino alternativo (revisión de servicio verde).

5. Tiempo de incorporación: el tiempo durante el cual las revisiones de servicio azul y verde se ejecutan simultáneamente después de que se haya transferido el tráfico de producción.
6. Finalización: tras una implementación correcta, se da por finalizada la revisión de servicio azul.

Si se detectan problemas durante la implementación, Amazon ECS puede revertirlos automáticamente enrutando nuevamente el tráfico al grupo de destino principal (revisión de servicio azul).

Recursos del equilibrador de carga de red para las implementaciones azul/verde de Amazon ECS

Para usar un equilibrador de carga de red con las implementaciones azul/verde de Amazon ECS, debe configurar recursos específicos que permitan el enrutamiento del tráfico entre las revisiones de servicio azul y verde. En esta sección se explican los componentes necesarios y su configuración.

Cuando su configuración incluye un equilibrador de carga de red, Amazon ECS agrega un retraso de 10 minutos a las siguientes etapas del ciclo de vida:

- PRE_SCALE_UP
- TEST_TRAFFIC_SHIFT
- PRODUCTION_TRAFFIC_SHIFT

Este retraso explica los problemas de tiempo de ejecución del equilibrador de carga de red que pueden provocar una discordancia entre las ponderaciones de tráfico configuradas y el enrutamiento de tráfico real en el plano de datos.

Grupos de destino

Para las implementaciones azul/verde con un equilibrador de carga de red, debe crear dos grupos de destino:

- Un grupo de destino principal para la revisión de servicio azul (tráfico de producción actual)
- Un grupo de destino alternativo para la revisión de servicio verde (nueva revisión del servicio)

Ambos grupos de destino deben configurarse con los siguientes ajustes:

- Tipo de destino: `ip` (para Fargate o EC2 con modo de red `awsvpc`)
- Protocolo: `TCP` (o el protocolo que utilice su aplicación)
- Puerto: el puerto en el que escucha su aplicación (normalmente `80` para HTTP)

- VPC: la misma VPC que sus tareas de Amazon ECS
- Configuración de comprobación de estado: configurada para comprobar correctamente el estado de la aplicación

  Para las comprobaciones de estado del TCP, el equilibrador de carga de red establece una conexión TCP con el destino. Si la conexión se realiza correctamente, se considera que el destino está en buen estado.

  Para las comprobaciones de estado de HTTP/HTTPS, el equilibrador de carga de red envía una solicitud HTTP/HTTPS al destino y verifica la respuesta.

Durante una implementación azul/verde, Amazon ECS registra automáticamente las tareas con el grupo de destino adecuado en función de la etapa de implementación.

Example Creación de grupos de destino para el equilibrador de carga de red

Los siguientes comandos de la AWS CLI crean dos grupos de destino para usarlos con un equilibrador de carga de red en una implementación azul/verde:

```
aws elbv2 create-target-group \
    --name blue-target-group \
    --protocol TCP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-protocol TCP

aws elbv2 create-target-group \
    --name green-target-group \
    --protocol TCP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-protocol TCP
```

Equilibrador de carga de red

Debe crear un equilibrador de carga de red con la siguiente configuración:

- Esquema: con acceso a Internet o interno, según sus necesidades
- Tipo de dirección IP: IPv4

- VPC: la misma VPC que sus tareas de Amazon ECS
- Subredes: al menos dos subredes en diferentes zonas de disponibilidad

A diferencia de los equilibradores de carga de aplicaciones, los equilibradores de carga de red funcionan en la capa de transporte (capa 4) y no utilizan grupos de seguridad. En su lugar, debe asegurarse de que los grupos de seguridad asociados a sus tareas de Amazon ECS permitan el tráfico del equilibrador de carga de red en los puertos de oyente.

Example Creación de un Network Load Balancer

El siguiente comando de la AWS CLI crea un equilibrador de carga de red para utilizarlo en una implementación azul/verde:

```
aws elbv2 create-load-balancer \
    --name my-network-load-balancer \
    --type network \
    --subnets subnet-12345678 subnet-87654321
```

Consideraciones acerca del uso de equilibradores de carga de red con las implementaciones azul/verde

Cuando use un equilibrador de carga de red para implementaciones azul/verde, tenga en cuenta lo siguiente:

- Funcionamiento de la capa 4: los equilibradores de carga de red funcionan en la capa de transporte (capa 4) y no inspeccionan el contenido de la capa de aplicación (capa 7). Esto significa que no puede usar encabezados o rutas HTTP para tomar decisiones respecto al enrutamiento.
- Comprobaciones de estado: las comprobaciones de estado de un equilibrador de carga de red se limitan a los protocolos TCP, HTTP o HTTPS. Para las comprobaciones de estado de TCP, el equilibrador de carga de red solo verifica que se pueda establecer la conexión.
- Conservación de la conexión: los equilibradores de carga de red conservan la dirección IP de origen del cliente, lo que puede resultar útil por motivos de seguridad y registro.
- Direcciones IP estáticas: los equilibradores de carga de red proporcionan direcciones IP estáticas para cada subred, lo que puede resultar útil para su inclusión en listas blancas o cuando los clientes necesitan conectarse a una dirección IP fija.
- Tráfico de prueba: dado que los equilibradores de carga de red no admiten el enrutamiento basado en el contenido, el tráfico de prueba debe enviarse a un puerto diferente al del tráfico de producción.

Oyentes y reglas

Para las implementaciones azul/verde con un equilibrador de carga de red, debe configurar oyentes:

- Oyente de producción: gestiona el tráfico de producción (normalmente en los puertos 80 o 443)
  - Inicialmente remite el tráfico hacia el grupo de destino principal (revisión de servicio azul)
  - Tras la implementación, remite el tráfico al grupo de destino alternativo (revisión de servicio verde)
- Oyente de prueba (opcional): gestiona el tráfico de prueba para validar la revisión de servicio verde antes de transferir el tráfico de producción
  - Se puede configurar en un puerto diferente (por ejemplo, 8080 o 8443)
  - Remite el tráfico al grupo de destino alternativo (revisión de servicio verde) durante la fase de prueba

A diferencia de los equilibradores de carga de aplicación, los equilibradores de carga de red no admiten reglas de enrutamiento basadas en el contenido. En su lugar, el tráfico se enruta en función del puerto y el protocolo del oyente.

Los siguientes comandos de la AWS CLI crean oyentes de producción y de prueba para un equilibrador de carga de red:

Sustituya las *entradas del usuario* por sus valores.

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
net/my-network-lb/1234567890123456 \
    --protocol TCP \
    --port 80 \
    --default-actions Type=forward,
 TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/blue-
target-group/1234567890123456

aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
net/my-network-lb/1234567890123456 \
    --protocol TCP \
    --port 8080 \
```

```
    --default-actions Type=forward,
 TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/green-
target-group/1234567890123456
```

Configuración del servicio

Debe tener permisos para autorizar a Amazon ECS que en su nombre administre los recursos del equilibrador de carga en sus clústeres. Para obtener más información, consulte Rol de IAM de infraestructura de Amazon ECS para los equilibradores de carga.

Al crear o actualizar un servicio de Amazon ECS para implementaciones azul/verde con un equilibrador de carga de red, debe especificar la siguiente configuración:

Sustituya las *entradas del usuario* por sus valores.

Los componentes clave de esta configuración son:

- `targetGroupArn`: el ARN del grupo de destino principal (revisión de servicio azul)
- `alternateTargetGroupArn`: el ARN del grupo de destino alternativo (revisión de servicio verde)
- `productionListenerArn`: el ARN del oyente para el tráfico de producción
- `testListenerArn`: (opcional) el ARN del oyente para el tráfico de prueba
- `roleArn`: el ARN del rol que permite que Amazon ECS administre los recursos del equilibrador de carga de red
- `strategy`: se configura en `BLUE_GREEN` para habilitar las implementaciones azul/verde
- `bakeTimeInMinutes`: el tiempo que se debe esperar después de que se implemente la revisión de servicio verde antes de transferir el tráfico de producción

```
{
    "loadBalancers": [
        {
            "targetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/blue-target-
group/1234567890123456",
            "containerName": "container-name",
            "containerPort": 80,
            "advancedConfiguration": {
                "alternateTargetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/green-target-
group/1234567890123456",
```

```
                "productionListenerArn":
 "arn:aws:elasticloadbalancing:region:123456789012:listener/net/my-network-
lb/1234567890123456/1234567890123456",
                "testListenerArn":
 "arn:aws:elasticloadbalancing:region:123456789012:listener/net/my-network-
lb/1234567890123456/2345678901234567",
                "roleArn": "arn:aws:iam::123456789012:role/ecs-nlb-role"
            }
        }
    ],
    "deploymentConfiguration": {
        "strategy": "BLUE_GREEN",
        "maximumPercent": 200,
        "minimumHealthyPercent": 100,
        "bakeTimeInMinutes": 5
    }
}
```

Flujo de tráfico durante la implementación

Durante una implementación azul/verde con un equilibrador de carga de red, el tráfico fluye por el sistema de la siguiente manera:

1. Estado inicial: todo el tráfico de producción se enruta al grupo de destino principal (revisión de servicio azul).
2. Implementación de la revisión de servicio verde: Amazon ECS implementa las nuevas tareas y las registra en el grupo de destino alternativo.
3. Tráfico de prueba: si se configura un oyente, el tráfico de prueba se enruta al grupo de destino alternativo para validar la revisión de servicio verde.
4. Transferencia de tráfico de producción: Amazon ECS actualiza el oyente de producción para enrutar el tráfico al grupo de destino alternativo (revisión de servicio verde).
5. Tiempo de incorporación: el tiempo durante el cual las revisiones de servicio azul y verde se ejecutan simultáneamente después de que se haya transferido el tráfico de producción.
6. Finalización: tras una implementación correcta, se da por finalizada la revisión de servicio azul.

Si se detectan problemas durante la implementación, Amazon ECS puede revertirlos automáticamente enrutando nuevamente el tráfico al grupo de destino principal (revisión de servicio azul).

Recursos de Service Connect para implementaciones azul/verde de Amazon ECS

Al utilizar Service Connect con implementaciones azul/verde, debe configurar componentes específicos para habilitar el enrutamiento de tráfico adecuado entre las revisiones de servicio azul y verde. En esta sección se explican los componentes necesarios y su configuración.

Información general de la arquitectura

Service Connect crea capacidades tanto de detección de servicios como de malla de servicios a través de un proxy sidecar administrado que se incorpora automáticamente a las tareas de Amazon ECS. Estos proxies gestionan las decisiones de enrutamiento, los reintentos y la recopilación de métricas, a la vez que AWS Cloud Map proporciona el backend del registro de servicios. Al implementar un servicio con Service Connect habilitado, el servicio se registra en AWS Cloud Map y los servicios de cliente lo detectan a través del espacio de nombres.

En una implementación estándar de Service Connect, los servicios de cliente se conectan a nombres de servicios lógicos y el proxy sidecar gestiona el enrutamiento a las instancias de servicio reales. Con las implementaciones azul/verde, este modelo se amplía para incluir el enrutamiento del tráfico de prueba a través de la configuración de `testTrafficRules`.

Durante una implementación azul/verde, los siguientes componentes clave trabajan juntos:

- Proxy de Service Connect: todo el tráfico entre los servicios pasa a través del proxy de Service Connect, que toma las decisiones de enrutamiento en función de la configuración.
- Registro de AWS Cloud Map: las implementaciones azul y verde se registran con AWS Cloud Map, pero la implementación verde se registra inicialmente como un punto de conexión de “prueba”.
- Enrutamiento de tráfico de prueba: las `testTrafficRules` en la configuración de Service Connect determinan cómo identificar y enrutar el tráfico de prueba a la implementación verde. Esto se logra mediante el enrutamiento basado en encabezados, donde los encabezados HTTP específicos de las solicitudes dirigen el tráfico a la revisión de prueba. De forma predeterminada, Service Connect reconoce el encabezado `x-amzn-ecs-blue-green-test` de los protocolos basados en HTTP cuando no se especifican reglas personalizadas.
- Configuración del cliente: todos los clientes del espacio de nombres reciben automáticamente las rutas de producción y de prueba, pero solo las solicitudes que coincidan con las reglas de prueba se destinarán a la implementación verde.

Lo que hace que este método sea eficaz es que gestiona la complejidad de la detección de servicios durante las transiciones. A medida que el tráfico pasa de la implementación azul a la verde, todos

los mecanismos de conectividad y detección se actualizan automáticamente. No es necesario actualizar los registros de DNS, reconfigurar los equilibradores de carga ni implementar los cambios de detección de servicios por separado, ya que la malla de servicios se encarga de todo.

Enrutamiento y comprobación del tráfico

Service Connect proporciona capacidades avanzadas de enrutamiento de tráfico para implementaciones azul/verde, incluido el enrutamiento basado en encabezados y la configuración de alias de cliente para escenarios de prueba.

Reglas de encabezado de tráfico de prueba

Durante las implementaciones azul/verde, puede configurar las reglas de los encabezados de tráfico de prueba para enrutar solicitudes específicas a la (nueva) revisión de servicio verde con fines de prueba. Esto le permite validar la nueva versión con tráfico controlado antes de completar la implementación.

Service Connect utiliza el enrutamiento basado en encabezados para identificar el tráfico de prueba. De forma predeterminada, Service Connect reconoce el encabezado `x-amzn-ecs-blue-green-test` de los protocolos basados en HTTP cuando no se especifican reglas personalizadas. Cuando este encabezado está presente en una solicitud, el proxy de Service Connect enruta automáticamente la solicitud a la implementación verde para su prueba.

Las reglas de los encabezados de tráfico de prueba le permiten hacer lo siguiente:

- Enrutar solicitudes con encabezados específicos a la revisión de servicio verde
- Comprobar la nueva funcionalidad con un subconjunto de tráfico
- Validar el comportamiento del servicio antes de la transición total del tráfico
- Implementar estrategias de pruebas canarias
- Realizar pruebas de integración en un entorno similar al de producción

El mecanismo de enrutamiento basado en encabezados funciona a la perfección con la arquitectura de aplicaciones existente. Los servicios de cliente no necesitan conocer el proceso de implementación azul/verde, simplemente incluyen los encabezados apropiados al enviar las solicitudes de prueba y el proxy de Service Connect gestiona la lógica de enrutamiento automáticamente.

Para más información acerca de cómo configurar reglas de encabezado de tráfico de prueba, consulte ServiceConnectTestTrafficHeaderRules en la Referencia de la API de Amazon Elastic Container Service.

Reglas de coincidencia de encabezados

Las reglas de coincidencia de encabezados definen los criterios para enrutar el tráfico de prueba durante las implementaciones azul/verde. Puede configurar varias condiciones de coincidencia para controlar con precisión qué solicitudes se enrutan a la revisión de servicio verde.

La coincidencia de encabezados admite lo siguiente:

- Coincidencias exactas con el valor del encabezado
- Comprobación de la presencia de encabezados
- Coincidencias basadas en patrones
- Múltiples combinaciones de encabezados

Los ejemplos de casos de uso incluyen el enrutamiento de solicitudes con cadenas de agentes de usuario específicas, versiones de API o indicadores de características al servicio verde para llevar a cabo pruebas.

Para más información sobre la configuración de coincidencias de encabezados, consulte ServiceConnectTestTrafficHeaderMatchRules en la Referencia de la API de Amazon Elastic Container Service.

Alias de cliente para implementaciones azul/verde

Los alias de cliente proporcionan puntos de conexión de DNS estables para los servicios durante las implementaciones azul/verde. Permiten enrutar el tráfico sin problemas entre las revisiones de servicio azul y verde sin necesidad de que las aplicaciones de cliente cambien sus puntos de conexión.

Durante una implementación azul/verde, los alias de cliente:

- Mantienen nombres de DNS coherentes para las conexiones de cliente
- Habilitan la transferencia automática del tráfico entre las revisiones de servicio
- Respaldan estrategias de migración gradual del tráfico
- Proporcionan capacidades de reversión al redireccionar el tráfico a la revisión azul

Puede configurar varios alias de cliente para distintos puertos o protocolos, lo que permite que las arquitecturas de servicios complejas mantengan la conectividad durante las implementaciones.

Para más información acerca de la configuración de alias de cliente, consulte ServiceConnectClientAlias en la Referencia de la API de Amazon Elastic Container Service.

Prácticas recomendadas para el enrutamiento de tráfico

Al implementar el enrutamiento de tráfico para las implementaciones azul/verde con Service Connect, tenga en cuenta las siguientes prácticas recomendadas:

- Comience con las pruebas basadas en encabezados: utilice las reglas de encabezado de tráfico de prueba para validar el servicio verde con tráfico controlado antes de transferir todo el tráfico.
- Configure las comprobaciones de estado: asegúrese de que los servicios azul y verde tengan configuradas las comprobaciones de estado adecuadas para evitar que el tráfico se enrute a instancias en mal estado.
- Monitoree las métricas del servicio: realice un seguimiento de los indicadores clave de rendimiento de ambas revisiones de servicio durante la implementación para identificar los problemas de forma temprana.
- Planifique una estrategia de reversión: configure los alias de los clientes y las reglas de enrutamiento para permitir una reversión rápida al servicio azul en caso de que se detecten problemas.
- Compruebe la lógica de coincidencia de encabezados: valide las reglas de coincidencia de encabezados en un entorno que no sea de producción antes de aplicarlas a las implementaciones de producción.

Flujo de trabajo de las implementaciones azul/verde de Service Connect

Comprender cómo Service Connect administra el proceso de implementación azul/verde le ayuda a implementar y solucionar los problemas de las implementaciones de forma eficaz. El siguiente flujo de trabajo muestra cómo interactúan los distintos componentes durante cada fase de la implementación.

Fases de la implementación

La implementación azul/verde de Service Connect pasa por varias fases distintas:

1. Estado inicial: el servicio azul gestiona el 100 % del tráfico de producción. Todos los servicios de cliente del espacio de nombres se conectan al servicio azul mediante el nombre lógico del servicio configurado en Service Connect.
2. Registro de servicio verde: cuando se inicia la implementación verde, se registra con AWS Cloud Map como punto de conexión de “prueba”. El proxy de Service Connect en los servicios de cliente recibe automáticamente las configuraciones de ruta de producción y prueba.
3. Enrutamiento de tráfico de prueba: el proxy de Service Connect enruta automáticamente las solicitudes que contienen los encabezados de tráfico de prueba (por ejemplo, `x-amzn-ecs-blue-green-test`) al servicio verde. El tráfico de producción sigue fluyendo hacia el servicio azul.
4. Preparación para la transferencia de tráfico: tras realizar satisfactoriamente las pruebas, el proceso de implementación se prepara para la transferencia de tráfico de producción. Tanto los servicios azules como los verdes permanecen registrados y en buen estado.
5. Transferencia de tráfico de producción: la configuración de Service Connect se actualiza para enrutar el tráfico de producción al servicio verde. Esto ocurre automáticamente sin necesidad de actualizar el servicio del cliente ni cambiar el DNS.
6. Tiempo de incorporación: el tiempo durante el cual las revisiones de servicio azul y verde se ejecutan simultáneamente después de que se haya transferido el tráfico de producción.
7. Cancelación del registro de servicio azul: tras la validación y la transferencia de tráfico correctos, se anula el registro de servicio azul de AWS Cloud Map y se cancela, completándose así la implementación.

Comportamiento del proxy de Service Connect

El proxy de Service Connect desempeña un papel crucial en la administración del tráfico durante las implementaciones azul/verde. Comprender su comportamiento le ayuda a diseñar estrategias de implementación y comprobación eficaces.

Comportamientos de proxy clave durante las implementaciones azul/verde:

- Detección automática de rutas: el proxy detecta automáticamente las rutas de producción y de prueba de AWS Cloud Map sin necesidad de reiniciar la aplicación ni cambiar la configuración.
- Enrutamiento basado en encabezados: el proxy examina los encabezados de las solicitudes entrantes y enruta el tráfico a la revisión de servicio correspondiente en función de las reglas de tráfico de prueba configuradas.

- Integración de comprobaciones de estado: el proxy solo enruta el tráfico a las instancias de servicio en buen estado y excluye automáticamente las tareas en mal estado del grupo de enrutamiento.
- Reintento y desconexión de circuitos: el proxy ofrece funciones integradas de lógica de reintento y desconexión de circuitos, lo que mejora la resiliencia durante las implementaciones.
- Recopilación de métricas: el proxy recopila métricas detalladas de los servicios azules y verdes, lo que permite un monitoreo exhaustivo durante las implementaciones.

Actualizaciones de la detección de servicios

Una de las principales ventajas de usar Service Connect para las implementaciones azul/verde es la gestión automática de las actualizaciones de detección de servicios. Las implementaciones azul/verde tradicionales suelen requerir actualizaciones de DNS complejas o una reconfiguración del equilibrador de carga, pero Service Connect administra estos cambios de forma transparente.

Durante una implementación, Service Connect gestiona:

- Actualizaciones del espacio de nombres: el espacio de nombres de Service Connect incluye automáticamente los puntos de conexión de servicio azules y verdes, con las reglas de enrutamiento adecuadas.
- Configuración del cliente: todos los servicios de cliente del espacio de nombres reciben automáticamente la información de enrutamiento actualizada sin necesidad de reiniciarlos ni volver a distribuirlos.
- Transición gradual: las actualizaciones de detección de servicios se realizan de forma gradual y segura, lo que garantiza que no se interrumpan las solicitudes en curso.
- Compatibilidad con las reversiones: si es necesario llevar a cabo una reversión, Service Connect puede revertir rápidamente las configuraciones de detección de servicios para enrutar el tráfico hacia el servicio azul.

Creación de una implementación azul/verde de Amazon ECS

Con las implementaciones azul/verde de Amazon ECS, puede realizar y comprobar cambios en el servicio antes de implementarlos en un entorno de producción.

Requisitos previos

Realice las siguientes operaciones antes de iniciar una implementación azul/verde.

1. Configure los permisos adecuados.
   - Para obtener información sobre los permisos de Elastic Load Balancing, consulte Rol de IAM de infraestructura de Amazon ECS para los equilibradores de carga.
   - Para obtener información acerca de los permisos de Lambda, consulte Permisos necesarios para las funciones de Lambda en las implementaciones azul/verde de Amazon ECS
2. Las implementaciones azul/verde de Amazon ECS requieren que su servicio utilice una de las siguientes características: configure los recursos adecuados.
   - Equilibrador de carga de aplicación: para más información, consulte Recursos de equilibrador de carga de aplicación para implementaciones azul/verde.
   - Equilibrador de carga de red: para más información, consulte Recursos del equilibrador de carga de red para las implementaciones azul/verde de Amazon ECS.
   - Service Connect: para más información, consulte Recursos de Service Connect para implementaciones azul/verde de Amazon ECS.
3. Cree una regla para enrutar el tráfico a la revisión de su servicio verde. Para más información, consulte Reglas de oyente en la Guía del usuario de equilibradores de carga de red.
4. Cree un grupo de destino para la revisión de servicio verde. Cuando utilice el modo de red `awsvpc` para sus tareas, el tipo de destino debe ser `ip`. Para más información, consulte Grupos de destino en la Guía del usuario de equilibradores de carga de red.
5. Decida si quiere ejecutar las funciones de Lambda para los eventos del ciclo de vida.
   - Antes de escalar verticalmente
   - Después de escalar verticalmente
   - Transferencia de tráfico de prueba
   - Después de la transferencia de tráfico de prueba
   - Transferencia de tráfico de producción
   - Después de la transferencia de tráfico de producción

   Cree funciones de Lambda para cada evento del ciclo de vida. Para más información, consulte Cree una función de Lambda con la consola en la Guía para desarrolladores de AWS Lambda.

Procedimiento

Puede utilizar la consola o la AWS CLI para crear un servicio azul/verde de Amazon ECS.

Console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. Determine el recurso desde el que lanza el servicio.

Para iniciar un servicio desde	Pasos
Clústeres	a. En la página Clusters (Clústeres), seleccione el clúster que va a crear el servicio. Aparecerá la página de detalles del clúster. b. En la pestaña Services (Servicios), elija Create (Crear).
Definición de tarea	a. En la página Definiciones de tareas, seleccione la definición de tarea. b. En el menú Implementar, elija Crear servicio.

Aparecerá la página Crear servicio.

3. En Detalles del servicio, haga lo siguiente:
   a. En Familia de definiciones de tareas, elija la definición de tarea que se va a utilizar. A continuación, en Revisión de la definición de tareas, introduzca la revisión que desee utilizar.
   b. En Service name (Nombre del servicio), ingrese un nombre para el servicio.
4. Para ejecutar el servicio en un clúster existente, en Clúster existente, elija el clúster. Para ejecutar el servicio en un clúster nuevo, seleccione Crear clúster
5. Elija cómo distribuir las tareas en la infraestructura de clúster. En Configuración de computación, elija su opción.

Opción de computación	Pasos	
Estrategia de proveedores de capacidad	a. En Opciones de computación, elija Estrategia del proveedor de capacidad. b. Elija una estrategia: • Para utilizar la estrategia de proveedores de capacidad predeterminada del clúster, elija Use cluster default (Usar clúster predeterminado). • Si el clúster no tiene ninguna estrategia de proveedores de capacidad predeterminada o si desea utilizar una estrategia personalizada, elija Utilizar predeterminada, Agregar estrategia de proveedores de capacidad y, luego, defina la estrategia de proveedores de capacidad personalizada especificando una Base, Proveedor de capacidad y Peso.	

Opción de computación	Pasos
	Note Para utilizar un proveedor de capacidad en una estrategia, el proveedor de capacidad debe estar asociado con el clúster.
Tipo de lanzamiento	a. En la sección Compute options (Opciones de computación), seleccion e Lauch type (Tipo de lanzamiento). b. En Launch type (Tipo de lanzamiento), elija un tipo de lanzamiento. c. (Opcional) Cuando se especifica el tipo de lanzamiento de Fargate, en Platform version (Versión de la plataform a) especifique la versión de la plataforma que se va a utilizar. Si no se especifica una versión de la plataforma, se utilizará la versión `LATEST`.

6. En Configuración de implementación, haga lo siguiente:

a. En Tipo de servicio, seleccione Réplica.

b. En Desired tasks (Tareas deseadas), ingrese el número de tareas que se lanzarán y mantendrán en el servicio.

c. Para que Amazon ECS monitoree la distribución de las tareas entre las zonas de disponibilidad y las redistribuya cuando haya un desequilibrio, en Reequilibrio del servicio de zonas de disponibilidad, seleccione Reequilibrio del servicio de zonas de disponibilidad.

d. En Periodo de gracia de la comprobación de estado, introduzca la cantidad de tiempo (en segundos) durante la cual el programador de servicios ignora las comprobaciones de estado de Elastic Load Balancing, VPC Lattice y contenedores en mal estado después de que se haya iniciado una tarea por primera vez. Si no se especifica ningún valor para el período de gracia de comprobación de estado, se utiliza el valor predeterminado: 0.

7. a. En Tiempo de incorporación, introduzca el número de minutos que las revisiones de servicio azul y verde durarán simultáneamente antes de que finalice la revisión azul. Esto permite disponer de tiempo para la verificación y la comprobación.

   b. (Opcional) Ejecute las funciones de Lambda que se van a ejecutar en etapas específicas de la implementación. En Enlaces de ciclo de vida de la implementación, seleccione las etapas para ejecutar los enlaces de ciclo de vida.

      Para agregar un enlace de ciclo de vida:

      a. Elija Agregar.

      b. En Función de Lambda, introduzca el nombre o el ARN de la función.

      c. En Rol, selecione el rol de IAM que tiene permiso para invocar la función de Lambda.

      d. En Etapas del ciclo de vida, seleccione las etapas en las que debe ejecutarse la función de Lambda.

8. Para configurar el modo en que Amazon ECS detecta y gestiona los errores de implementación, expanda Deployment failure detection (Detección de errores de implementación) y, a continuación, elija sus opciones.

   a. Para detener una implementación cuando las tareas no puedan iniciarse, seleccione Use the Amazon ECS deployment circuit breaker (Utilizar el interruptor de circuito de implementación de Amazon ECS).

Para que el software restaure automáticamente la implementación a su último estado completado cuando el disyuntor de implementación establezca un estado con error, seleccione Restauración en caso de error.

b. Para detener una implementación en función de las métricas de la aplicación, seleccione Use CloudWatch alarms. A continuación, elija las alarmas en Nombre de la alarma de CloudWatch. Para crear una alarma nueva, vaya a la consola de CloudWatch.

   Para que el software restaure automáticamente la implementación a su último estado de implementación completada cuando una alarma de CloudWatch establezca un estado con error, seleccione Restauración en caso de error.

9. (Opcional) Para interconectar su servicio con Service Connect, expanda Service Connect y, a continuación, especifique lo siguiente:

   a. Seleccione Activar Service Connect.

   b. En Service Connect configuration (Configuración de Service Connect), especifique el modo cliente.

      - Si su servicio ejecuta una aplicación cliente de red que solo necesita conectarse a otros servicios del espacio de nombres, elija Solo en el lado del cliente.
      - Si su servicio ejecuta una aplicación de servicio web o red y necesita proporcionar puntos de conexión para este servicio y se conecta a otros servicios del espacio de nombres, elija Client and server (Cliente y servidor).

   c. Para usar un espacio de nombres que no sea el espacio de nombres predeterminado del clúster, en Namespace (Espacio de nombres), elija el espacio de nombres del servicio.

   d. (Opcional) Configure las reglas de encabezado de tráfico de prueba para las implementaciones azul/verde. En Enrutamiento de tráfico de prueba, especifique lo siguiente:

      i. Seleccione Habilitar las reglas de encabezado de tráfico de prueba para enrutar solicitudes específicas a la revisión de servicio verde durante las pruebas.

      ii. En Reglas de coincidencia de encabezados, configure los criterios para enrutar el tráfico de prueba:

         - Nombre del encabezado: introduzca el nombre del encabezado HTTP que desee que coincida (por ejemplo, `X-Test-Version` o `User-Agent`).
         - Tipo de coincidencia: elija los criterios de coincidencia:

- Coincidencia exacta: se enrutan las solicitudes en las que el valor del encabezado coincida exactamente con el valor especificado.
- Encabezado presente: se enrutan las solicitudes que contienen el encabezado especificado, independientemente del valor.
- Coincidencia de patrones: se enrutan las solicitudes en las que el valor del encabezado coincide con un patrón especificado.

- Valor del encabezado (si se utiliza una coincidencia exacta o una coincidencia de patrón): introduzca el valor o patrón con el que desea que haya coincidencia.

Puede agregar varias reglas de coincidencia de encabezados para crear una lógica de enrutamiento compleja. Las solicitudes que coincidan con alguna de las reglas configuradas se enrutarán a la revisión de servicio verde para su comprobación.

iii. Elija Agregar regla de encabezado para configurar condiciones de coincidencia de encabezados adicionales.

> **Note**
>
> Las reglas de encabezado de tráfico de prueba le permiten validar la nueva funcionalidad con tráfico controlado antes de finalizar la implementación completa. Esto le permite comprobar la revisión de servicio verde con solicitudes específicas (como las de las herramientas de pruebas internas o de los usuarios de la versión beta) y, al mismo tiempo, mantener un flujo de tráfico normal hacia la revisión de servicio azul.

e. (Opcional) Especifique una configuración de registro. Seleccione Utilizar colección de registros. La opción predeterminada envía registros de contenedor a los Registros de CloudWatch. Las demás opciones del controlador de registro se configuran mediante AWS FireLens. Para obtener más información, consulte Envío de registros de Amazon ECS a un servicio de AWS o AWS Partner.

A continuación, se describe con más detalle cada uno de los destinos de registro de contenedor.

- Amazon CloudWatch: configure la tarea para enviar registros de contenedor a CloudWatch Logs. Se proporcionan las opciones de controlador de registro predeterminadas que crean un grupo de registros de CloudWatch en su nombre. Para

especificar otro nombre de grupo de registros, cambie los valores de las opciones del controlador.

- Amazon Data Firehose: configure la tarea para enviar registros de contenedor a Firehose. Se proporcionan las opciones de controlador de registro predeterminadas que envían registros a un flujo de entrega de Firehose. Para especificar un nombre de flujo de entrega distinto, cambie los valores de las opciones del controlador.
- Amazon Kinesis Data Streams: configure la tarea para enviar registros de contenedores a Kinesis Data Streams. Se proporcionan las opciones de controlador de registro predeterminadas que envían registros a un flujo de Kinesis Data Streams. Para especificar otro nombre de transmisión, cambie los valores de las opciones del controlador.
- Amazon OpenSearch Service: configure la tarea para enviar registros de contenedor a un dominio de OpenSearch Service. Se deben proporcionar las opciones del controlador de registros.
- Amazon S3: configure la tarea para enviar registros de contenedor a un bucket de Amazon S3. Se proporcionan las opciones de controlador de registro predeterminadas, pero debe especificar un nombre de bucket de Amazon S3 válido.

10. (Opcional) Configure el equilibrio de carga para una implementación azul/verde.

Tipo de Elastic Load Balancing	Pasos	
Equilibrador de carga de aplicación	a. En Tipo de equilibrador de carga, elija Equilibrador de carga de aplicación. b. Elija Create a new load balancer (Crear un nuevo balanceador de carga) para crear un nuevo Application Load Balancer o Use an existing load balancer (Utilizar un balanceador de carga	

Tipo de Elastic Load Balancing	Pasos	
	existente) para seleccionar un Application Load Balancer existente. c. En Contenedor, elija el contenedor que aloja el servicio. d. En Load balancer name (Nombre del equilibrador de carga), ingrese un nombre único. e. En Listener (Agente de escucha), ingrese un puerto y un protocolo que el equilibrador de carga de aplicación pueda utilizar para escuchar las solicitudes de conexión. De forma predeterminada, el balanceador de carga se configurará para utilizar el puerto 80 y HTTP. • En Regla de producción, introduzca el orden de evaluación y el patrón de ruta de la regla. Esta regla es para el tráfico de revisión de servicio (azul) de producción.	

Tipo de Elastic Load Balancing	Pasos	
	• En Regla de prueba, introduzca el orden de evaluación y el patrón de ruta de la regla. Esta regla es para el tráfico de revisión del servicio (verde) de prueba. f. En Grupo de destino, configure lo siguiente: • En Target group name (Nombre del grupo de destino), ingrese un nombre y un protocolo para el grupo de destino al que el equilibrador de carga de aplicación dirige las solicitudes. • En Protocolo, elija el protocolo para el grupo de destino hacia el que el equilibrador de carga de aplicación enruta las solicitudes. De forma predeterm inada, el grupo de destino dirige las solicitudes al primer contenedor definido en la definición de la tarea.	

Tipo de Elastic Load Balancing	Pasos	
	• En Retardo de anulación del registro, ingrese el número de segundos al que el equilibrador de carga debe cambiar el estado de destino a `UNUSED`. El valor predeterminado es de 300 segundos. • En Health check path (Ruta de comprobación de estado), ingrese una ruta existente dentro de su contenedor donde el equilibrador de carga de aplicación envíe periódicamente solicitudes para verificar el estado de la conexión entre el equilibrador de carga de aplicación y el contenedor. El valor predeterminado es el directorio raíz (`/`). • En Nombre de grupo alternativo, introduzca el nombre del grupo de destino para la revisión de servicio (verde) de prueba.	

Tipo de Elastic Load Balancing	Pasos	
Equilibrador de carga de red	a. En Load balancer type (Tipo de equilibrador de carga), elija Network Load Balancer (Equilibrador de carga de red). b. En Load Balancer (Equilibrador de carga), elija un equilibrador de carga de red existente. c. En Choose container to load balance (Seleccionar contenedor para equilibrar la carga), elija el contenedor que aloja el servicio. d. En Oyente de producción, elija el puerto de oyente de producción y el protocolo de oyente de producción. Este es el oyente del tráfico de revisión de su servicio (azul) de producción. e. En Oyente de prueba, elija el puerto de oyente de prueba y el protocolo del oyente de prueba. Este es el oyente del tráfico de revisión del	

Tipo de Elastic Load Balancing	Pasos	
	servicio (verde) de prueba. f. En Grupo de destino, configure lo siguiente: • En Target group name (Nombre del grupo de destino), ingrese un nombre y un protocolo para el grupo de destino al que el equilibrador de carga de red dirige las solicitudes. • En Protocolo, elija el protocolo para el grupo de destino al que el equilibrador de carga de red enruta las solicitudes. De forma predeterminada, el grupo de destino dirige las solicitudes al primer contenedor definido en la definición de la tarea. • En Retardo de anulación del registro, ingrese el número de segundos al que el equilibrador de carga debe cambiar el estado de destino a `UNUSED`. El valor	

Tipo de Elastic Load Balancing	Pasos	
	predeterminado es de 300 segundos. • En Health check path (Ruta de comprobación de estado), ingrese una ruta existente dentro de su contenedor donde el equilibrador de carga de aplicación envíe periódicamente solicitudes para verificar el estado de la conexión entre el equilibrador de carga de aplicación y el contenedor. El valor predeterminado es el directorio raíz (/). • En Nombre de grupo alternativo, introduzca el nombre del grupo de destino para la revisión de servicio (verde) de prueba.	

11. (Opcional) Para ayudar a identificar el servicio y las tareas, expanda la sección Tags (Etiquetas) y, a continuación, configure sus etiquetas.

    Para que Amazon ECS etiquete automáticamente todas las tareas recién lanzadas con el nombre del clúster y las etiquetas de definición de tareas, seleccione Activar las etiquetas administradas de Amazon ECS y, a continuación, en Propagar etiquetas de, seleccione Definiciones de tareas.

Para que Amazon ECS etiquete automáticamente todas las tareas recién lanzadas con el nombre del clúster y las etiquetas de servicio, seleccione Activar las etiquetas administradas de Amazon ECS y, a continuación, en Propagar etiquetas de, seleccione Servicio.

Añada o elimine una etiqueta.

- [Agregar una etiqueta] Seleccione Add tag (Agregar etiqueta), y, a continuación, haga lo siguiente:
  - En Clave, escriba el nombre de la clave.
  - En Valor, escriba el valor de la clave.
- [Eliminar una etiqueta] Junto a la etiqueta, elija Remove tag (Quitar etiqueta).

12. Seleccione Crear.

AWS CLI

1. Cree un archivo llamado `service-definition.json` con el siguiente contenido.

   Sustituya las *`entradas del usuario`* por sus valores.

```
{
  "serviceName": "myBlueGreenService",
  "cluster": "arn:aws:ecs:us-west-2:123456789012:cluster/sample-fargate-
cluster",
  "taskDefinition": "sample-fargate:1",
  "desiredCount": 5,
  "launchType": "FARGATE",
  "networkConfiguration": {
    "awsvpcConfiguration": {
      "subnets": [
        "subnet-09ce6e74c116a2299",
        "subnet-00bb3bd7a73526788",
        "subnet-0048a611aaec65477"
      ],
      "securityGroups": [
        "sg-09d45005497daa123"
      ],
      "assignPublicIp": "ENABLED"
    }
  },
```

```
  "deploymentController": {
    "type": "ECS"
  },
  "deploymentConfiguration": {
    "strategy": "BLUE_GREEN",
    "maximumPercent": 200,
    "minimumHealthyPercent": 100,
    "bakeTimeInMinutes": 2,
    "alarms": {
      "alarmNames": [
        "myAlarm"
      ],
      "rollback": true,
      "enable": true
    },
    "lifecycleHooks": [
      {
        "hookTargetArn": "arn:aws:lambda:us-
west-2:7123456789012:function:checkExample",
        "roleArn": "arn:aws:iam::123456789012:role/ECSLifecycleHookInvoke",
        "lifecycleStages": [
          "PRE_SCALE_UP"
        ]
      }
    ]
  },
  "loadBalancers": [
    {
      "targetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/blue-target-group/54402ff563af1197",
      "containerName": "fargate-app",
      "containerPort": 80,
      "advancedConfiguration": {
        "alternateTargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/green-target-group/cad10a56f5843199",
        "productionListenerRule": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener-rule/app/my-blue-green-
demo/32e0e4f946c3c05b/9cfa8c482e204f7d/831dbaf72edb911",
        "roleArn": "arn:aws:iam::123456789012:role/LoadBalancerManagementforECS"
      }
    }
  ]
}
```

2. Ejecute `create-service`.

   Sustituya las *`entradas del usuario`* por sus valores.

```
aws ecs create-service --cli-input-json file://service-definition.json
```

   Como alternativa, puede usar el siguiente ejemplo, que crea un servicio de implementación azul/verde con una configuración de equilibrador de carga:

```
aws ecs create-service \
   --cluster "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster" \
   --service-name "blue-green-example-service" \
   --task-definition "nginxServer:1" \
   --launch-type "FARGATE" \
   --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12345,subnet-67890,subnet-abcdef,subnet-
fedcba],securityGroups=[sg-12345],assignPublicIp=ENABLED}" \
   --desired-count 3 \
   --deployment-controller "type=ECS" \
   --deployment-configuration
 "strategy=BLUE_GREEN,maximumPercent=200,minimumHealthyPercent=100,bakeTimeInMinutes=0"
 \
   --load-balancers "targetGroupArn=arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/MyBGtg1/
abcdef1234567890,containerName=nginx,containerPort=80,advancedConfiguration={alternateTa
west-2:123456789012:targetgroup/
MyBGtg2/0987654321fedcba,productionListenerRule=arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener-rule/app/
MyLB/1234567890abcdef/1234567890abcdef,roleArn=arn:aws:iam::123456789012:role/
ELBManagementRole}"
```

Pasos a seguir a continuación

- Actualice el servicio para iniciar la implementación. Para obtener más información, consulte Actualización de un servicio de Amazon ECS.
- Monitoree el proceso de implementación para asegurarse de que sigue el patrón azul/verde:
  - Se crea y escala verticalmente la revisión de servicio verde
  - El tráfico de prueba se enruta a la revisión verde (si está configurada)
  - El tráfico de producción se transfiere a la revisión verde

- Transcurrido el tiempo de incorporación, la revisión azul finaliza

Implementación de los servicios de Amazon ECS mediante un controlador de terceros

El tipo de implementación externa le permite utilizar cualquier controlador de implementación de terceros para tener un control completo del proceso de implementación de un servicio de Amazon ECS. Los detalles del servicio se administran por medio de las acciones de la API de administración de servicios (`CreateService`, `UpdateService` y `DeleteService`) o las acciones de la API de administración de conjunto de tareas (`CreateTaskSet`, `UpdateTaskSet`, `UpdateServicePrimaryTaskSet` y `DeleteTaskSet`). Cada acción de la API administra un subconjunto de los parámetros de definición del servicio.

La acción de la API `UpdateService` actualiza los parámetros de período de gracia de comprobación de estado y recuento deseados de un servicio. Si se tienen que actualizar el tipo de lanzamiento, la versión de la plataforma, los detalles del balanceador de carga, la configuración de red o la definición de tarea, debe crear una nueva tarea.

La acción de la API `UpdateTaskSet` actualiza únicamente el parámetro de escala de un conjunto de tarea.

La acción de la API `UpdateServicePrimaryTaskSet` modifica qué conjunto de tareas de un servicio es el conjunto de tareas principal. Cuando llama a la acción de la API `DescribeServices`, devuelve todos los campos especificados para un conjunto de tareas principal. Si se actualiza el conjunto de tareas principal para un servicio, cualquier valor de parámetro del conjunto de tareas que exista en el nuevo conjunto de tareas principal que difieran del conjunto de tareas principal antiguo en un servicio se actualiza al nuevo valor cuando se define un nuevo conjunto de tareas principal. Si no se ha definido ningún conjunto de tareas principal para un servicio, al describir el servicio, los campos del conjunto de tareas son nulos.

Consideraciones acerca de la implementación externa

Tenga en cuenta lo siguiente al utilizar el tipo de implementación externa:

- Los tipos de balanceador de carga admitidos son balanceador de carga de aplicaciones o balanceador de carga de red.
- El tipo de lanzamiento de Fargate o los tipos de controladores de implementación `EXTERNAL` no son compatibles con la estrategia de programación de `DAEMON`.

- La integración del escalado automático de aplicaciones con Amazon ECS solo admite el servicio Amazon ECS como destino. La dimensión escalable admitida para Amazon ECS es `ecs:service:DesiredCount`, el recuento de tareas de un servicio de Amazon ECS. No existe una integración directa entre los conjuntos de tareas de escalado automático de aplicaciones y Amazon ECS. Los conjuntos de tareas de Amazon ECS calculan el `ComputedDesiredCount` en función del `DesiredCount` del servicio de Amazon ECS.

Flujo de trabajo de implementación externa

A continuación, se muestra el flujo de trabajo básico para administrar una implementación externa en Amazon ECS.

Para administrar un servicio de Amazon ECS mediante un controlador de implementación externo

1. Creación de un servicio de Amazon ECS. El único parámetro obligatorio es el nombre de servicio. Puede especificar los siguientes parámetros al crear un servicio con un controlador de implementación externo. Todos los demás parámetros de servicio se especifican al crear un conjunto de tareas dentro del servicio.

   `serviceName`

   Tipo: cadena

   Obligatorio: sí

   El nombre de su servicio. Se admiten hasta 255 letras (mayúsculas y minúsculas), números, guiones y caracteres de subrayado. Los nombres de servicio deben ser únicos dentro de un clúster, pero puede tener servicios con el mismo nombre en varios clústeres dentro de una región o en varias regiones.

   `desiredCount`

   El número de instancias de la definición de tarea de conjunto de tareas especificada para colocar y seguir ejecutando en el servicio.

   `deploymentConfiguration`

   Parámetros de implementación opcionales que controlan cuántas tareas se ejecutan durante una implementación y la ordenación de tareas de parada e inicio.

`tags`

Tipo: Matriz de objetos

Obligatorio: no

Los metadatos que se aplican al servicio para ayudarle a categorizarlas y organizarlas. Cada etiqueta está formada por una clave y un valor opcional, ambos definidos por el usuario. Cuando se elimina un servicio, también se eliminan las etiquetas. Se puede aplicar un máximo de 50 etiquetas al servicio. Para obtener más información, consulte Etiquetado de los recursos de Amazon ECS.

`key`

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Obligatorio: no

Una parte de un par clave-valor que compone una etiqueta. Un clave es una etiqueta general que actúa como una categoría para valores de etiqueta más específicos.

`value`

Tipo: cadena

Limitaciones de longitud: longitud mínima de 0. La longitud máxima es de 256 caracteres.

Obligatorio: no

La parte opcional de un par clave-valor que compone una etiqueta. Un valor actúa como un descriptor en una categoría de etiquetas (clave).

`enableECSManagedTags`

Especifica si se deben usar etiquetas administradas por Amazon ECS para las tareas dentro del servicio. Para obtener más información, consulte Uso de etiquetas para facturación.

`propagateTags`

Tipo: cadena

Valores válidos: `TASK_DEFINITION | SERVICE`

Obligatorio: no

Especifica si se deben copiar las etiquetas de la definición de tareas o el servicio en las tareas del servicio. Si no se especifica ningún valor, las etiquetas no se copian. Solo se pueden copiar las etiquetas en las tareas del servicio durante la creación del servicio. Para agregar etiquetas a una tarea tras la creación del servicio, utilice la acción de la API `TagResource`.

`schedulingStrategy`

La estrategia de programación que se va a utilizar. Los servicios que utilizan un controlador de implementación solo admiten la estrategia de programación `REPLICA`.

`placementConstraints`

Una matriz de objetos de restricción de colocación que utilizar para tareas en su servicio. Puede especificar 10 restricciones como máximo por tarea (este límite incluye restricciones en la definición de tareas y las especificadas en tiempo de ejecución). Si utiliza el tipo de lanzamiento de Fargate, no se admiten las restricciones de ubicación de tareas.

`placementStrategy`

Los objetos de estrategia colocación que utilizar para tareas en su servicio. Puede especificar un máximo de cuatro reglas de estrategia por servicio.

A continuación, se muestra un ejemplo de definición de servicio para crear un servicio mediante un controlador de implementación externo.

```
{
    "cluster": "",
    "serviceName": "",
    "desiredCount": 0,
    "role": "",
    "deploymentConfiguration": {
        "maximumPercent": 0,
        "minimumHealthyPercent": 0
    },
    "placementConstraints": [
        {
            "type": "distinctInstance",
            "expression": ""
```

```
        }
    ],
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION"
}
```

2. Crear un conjunto de tareas inicial. El conjunto de tareas contiene los siguientes detalles sobre el servicio:

   `taskDefinition`

   La definición de tareas en el conjunto de tareas que se va a utilizar.

   `launchType`

   Tipo: cadena

   Valores válidos: `EC2` | `FARGATE` | `EXTERNAL`

   Obligatorio: no

   El tipo de lanzamiento en el que ejecutar su servicio. Si no se especifica ningún tipo de lanzamiento, se usará `capacityProviderStrategy` de forma predeterminada. Para obtener más información, consulte Tipos de lanzamiento de Amazon ECS.

   Si se especifica `launchType`, se debe omitir el parámetro `capacityProviderStrategy`.

`platformVersion`

Tipo: cadena

Requerido: no

La versión de la plataforma en la que se ejecutan sus tareas en el servicio. La versión de la plataforma solo se especifica para las tareas que utilizan el tipo de lanzamiento de Fargate. Si no se especifica ninguna, se usará la versión más reciente (`LATEST`) de forma predeterminada.

Las versiones de la plataforma de AWS Fargate se utilizan para hacer referencia a un entorno en tiempo de ejecución específico para la infraestructura de tareas de Fargate. Cuando se especifica la versión de la plataforma `LATEST` al ejecutar una tarea o crear un servicio, se obtiene la versión más actual de la plataforma disponible para las tareas. Cuando se escala un servicio, esas tareas reciben la versión de la plataforma especificada en la implementación actual del servicio. Para obtener más información, consulte Versiones de la plataforma Fargate para Amazon ECS.

> **Note**
>
> No se especifican las versiones de la plataforma para las tareas que utilizan el tipo de lanzamiento de EC2.

`loadBalancers`

Un objeto de balanceador de carga que representa el balanceador de carga que utilizar con su servicio. Cuando se utiliza un controlador de implementación externo, solo se admiten Application Load Balancers y Network Load Balancers. Si utiliza un Application Load Balancer, solo se permite un grupo de destino del Application Load Balancer por conjunto de tareas.

El siguiente fragmento muestra un objeto `loadBalancer` de ejemplo que se va a utilizar.

```
"loadBalancers": [
        {
            "targetGroupArn": "",
            "containerName": "",
```

```
            "containerPort": 0
        }
]
```

> **Note**
>
> Al especificar un objeto `loadBalancer`, debe especificar un `targetGroupArn` y omitir los parámetros `loadBalancerName`.

`networkConfiguration`

La configuración de red del servicio. Este parámetro es necesario para definiciones de tareas que usan el modo de red `awsvpc` para recibir su propia interfaz de red elástica y no se admite para otros modos de red. Para obtener más información acerca de las redes para el tipo de lanzamiento de Fargate, consulte Opciones de redes de tareas de Amazon ECS para el tipo de lanzamiento de Fargate.

`serviceRegistries`

Los detalles de los registros de detección de servicios que asignar a este servicio. Para obtener más información, consulte Uso de la detección de servicios para conectar los servicios de Amazon ECS con nombres de DNS.

`scale`

Un porcentaje de punto flotante del número de tareas deseado que colocar y seguir ejecutando en el conjunto de tareas. El valor se especifica como un total de porcentaje del `desiredCount` de un servicio. Los valores aceptados son números entre 0 y 100.

A continuación, se muestra un ejemplo JSON para crear un conjunto de tareas para un controlador de implementación externo.

```
{
    "service": "",
    "cluster": "",
    "externalId": "",
    "taskDefinition": "",
    "networkConfiguration": {
        "awsvpcConfiguration": {
```

```
            "subnets": [
                ""
            ],
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "launchType": "EC2",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "scale": {
        "value": null,
        "unit": "PERCENT"
    },
    "clientToken": ""
}
```

3. Cuando es necesario cambiar el servicio, utilice la acción de la API `CreateTaskSet`, `UpdateTaskSet` o `UpdateService` en función de los parámetros que esté actualizando. Si ha creado un conjunto de tareas, utilice el parámetro `scale` para cada conjunto de tareas en

un servicio para determinar cuántas tareas hay que mantener en ejecución en el servicio. Por ejemplo, si tiene un servicio que contiene `tasksetA` y crea un `tasksetB`, puede probar la validez de transición `tasksetB` antes de que desear pasar el tráfico de producción al mismo. Podría establecer el valor `scale` de ambos conjunto de tareas en `100` y cuando esté listo para pasar todo el tráfico de producción a `tasksetB`, podría actualizar el valor de `scale` de `tasksetA` a `0` para reducirlo.

Implementaciones azul/verde de CodeDeploy para Amazon ECS

Le recomendamos que utilice la implementación azul/verde de Amazon ECS. Para obtener más información, consulte Creación de una implementación azul/verde de Amazon ECS .

El tipo de implementación blue/green utiliza el modelo de implementación blue/green controlado por CodeDeploy. Este tipo de implementación permite verificar una nueva implementación de un servicio antes de enviar el tráfico de producción a este. Para obtener más información, consulte What Is CodeDeploy en la Guía del usuario de AWS CodeDeploy. Valide el estado de un servicio de Amazon ECS antes de la implementación.

Hay tres formas de desviar el tráfico en una implementación azul/verde:

- canario: el tráfico se desvía en dos incrementos. Puede elegir opciones de valor controlado predefinidas que especifiquen el porcentaje de tráfico desviado al conjunto de tareas actualizado en el primer incremento y el intervalo, en minutos, antes de que el tráfico restante se desvíe en el segundo incremento.
- Lineal: el tráfico se desvía en incrementos iguales con el mismo número de minutos entre incrementos. Puede elegir opciones lineales predefinidas que especifiquen el porcentaje de tráfico desviado en cada incremento y el número de minutos entre cada incremento.
- Todo a la vez: todo el tráfico se desvía del conjunto de tareas original al conjunto de tareas actualizado a la vez.

A continuación, se indican los componentes de CodeDeploy que emplea Amazon ECS cuando un servicio utiliza el tipo de implementación blue/green:

Aplicación CodeDeploy

Colección de recursos de CodeDeploy. Se compone de uno o varios grupos de implementación.

Grupo de implementaciones de CodeDeploy

Configuración de la implementación. Consta de los elementos siguientes:

- Servicio y clúster de Amazon ECS
- Información del grupo de destino del balanceador de carga y del agente de escucha
- Estrategia de reversión de implementación
- Configuración de redireccionamiento del tráfico
- Configuración de terminación de revisión original
- Configuración de implementación
- Configuración de alarmas de CloudWatch que se pueden configurar para detener las implementaciones
- Configuración de SNS o CloudWatch Events para notificaciones

Para obtener más información, consulte Utilización de grupos de implementación en la Guía del usuario de AWS CodeDeploy.

Configuración de implementación de CodeDeploy

Especifica de qué manera dirige CodeDeploy el tráfico de producción al conjunto de tareas de sustitución durante la implementación. La siguiente configuración de implementación lineal y de valor controlado predefinida está disponible. También puede crear implementaciones lineales y de valor controlado definidas personalizadas. Para obtener más información, consulte Utilización de configuraciones de implementación en la Guía del usuario de AWS CodeDeploy.

- CodeDeployDefault.ECSAllAtOnce: desplaza todo el tráfico al contenedor de Amazon ECS actualizado de una vez.
- CodeDeployDefault.ecsLinear10PercentEvery1Minutes: desplaza el 10 % del tráfico cada minuto hasta que se haya desplazado todo el tráfico.
- CodeDeployDefault.ecsLinear10PercentEvery3Minutes: desplaza el 10 % del tráfico cada tres minutos hasta que se haya desplazado todo el tráfico.
- CodeDeployDefault.ECSCanary10Percent5Minutes: desplaza el 10 % del tráfico en el primer incremento. El 90 por ciento restante se implementa cinco minutos más tarde.
- CodeDeployDefault.ECSCanary10Percent15Minutes: desplaza el 10 % del tráfico en el primer incremento. El 90 por ciento restante se implementa 15 minutos más tarde.

Revisión

Una revisión es el archivo de especificación de la aplicación CodeDeploy (archivo AppSpec). En el archivo AppSpec, debe especificar el ARN completo de la definición de la tarea, el contenedor y el puerto del conjunto de tareas de sustitución hacia donde debe dirigirse el tráfico cuando se crea una implementación nueva. El nombre del contenedor debe ser uno de los nombres de contenedor al que se hace referencia en la definición de tarea. Si la configuración de red o versión de la plataforma se ha actualizado en la definición del servicio, también debe especificar esos detalles en el archivo AppSpec. También puede especificar las funciones de Lambda que se deben ejecutar durante los eventos del ciclo de vida de la implementación. Las funciones de Lambda le permiten ejecutar pruebas y obtener métricas durante la implementación. Para obtener más información, consulte Referencia del archivo AppSpec en la Guía del usuario de AWS CodeDeploy.

Consideraciones

Tenga en cuenta lo siguiente al utilizar el tipo de implementación blue/green:

- Cuando un servicio de Amazon ECS que utiliza el tipo de implementación blue/green se crea inicialmente, se crea un conjunto de tareas de Amazon ECS.
- Debe configurar el servicio para que utilice un Application Load Balancer o un Network Load Balancer. A continuación se indican los requisitos del balanceador de carga:
  - Debe añadir un agente de escucha de producción al balanceador de carga, que se utiliza para dirigir el tráfico de producción.
  - Es posible añadir un agente de escucha de pruebe opcional al balanceador de carga, que se utiliza para dirigir el tráfico de prueba. Si especifica un agente de escucha de prueba, CodeDeploy dirige el tráfico de prueba al conjunto de tareas de sustitución durante una implementación.
  - Tanto los agentes de escucha de prueba como de producción deben pertenecer al mismo balanceador de carga.
  - Debe definir un grupo de destino para el balanceador de carga. El grupo de destino dirige el tráfico al conjunto de tareas original en un servicio a través del agente de escucha de producción.
  - Cuando se utiliza un Network Load Balancer, solo se admite la configuración de implementación `CodeDeployDefault.ECSAllAtOnce`.

- Para los servicios configurados para que utilicen el escalado automático del servicio y el tipo de implementación “green/blue”, el escalado automático no se bloquea durante una implementación, pero la implementación puede fallar en algunas circunstancias. A continuación, se describe este comportamiento con más detalle.
  - Si un servicio está escalando y se inicia una implementación, se crea el conjunto de tareas “green” (verdes) y CodeDeploy esperará hasta una hora para que el conjunto de tareas verdes alcance el estado constante, y no cambiará el tráfico hasta que lo haga.
  - Si un servicio está en proceso de implementación “blue/green” y se produce un evento de escalado, el tráfico continuará cambiando durante 5 minutos. Si el servicio no alcanza el estado estable en 5 minutos, CodeDeploy detendrá la implementación y la marcará como fallida.
- Las tareas que utilizan el tipo de lanzamiento de Fargate o los tipos de controlador de implementación `CODE_DEPLOY` no son compatibles con la estrategia de programación `DAEMON`.
- Cuando crea inicialmente una aplicación y un grupo de implementaciones de CodeDeploy, debe especificar lo siguiente:
  - Debe definir dos grupos de destino para el balanceador de carga. Un grupo de destino debe ser el grupo de destino inicial que se definió para el balanceador de carga al crear el servicio de Amazon ECS. El único requisito del segundo grupo de destino es que no puede asociarse con un balanceador de carga diferente al utilizado por el servicio.
- Cuando crea una implementación de CodeDeploy para un servicio de Amazon ECS, CodeDeploy crea un conjunto de tareas de sustitución (o un conjunto de tareas “green” [verdes]) en la implementación. Si ha agregado un agente de escucha de prueba al balanceador de carga, CodeDeploy dirige el tráfico de prueba al conjunto de tareas de sustitución. Esto es cuando puede ejecutar cualquier prueba de validación. A continuación, CodeDeploy redirige el tráfico de producción del conjunto de tareas original al conjunto de tareas de sustitución de acuerdo con la configuración de redireccionamiento del tráfico para el grupo de implementaciones.

Permisos de IAM necesarios

Las implementaciones azul/verde son posibles gracias a la combinación de las API de Amazon ECS y CodeDeploy. Los usuarios deben tener los permisos adecuados para estos servicios antes de que puedan utilizar las implementaciones azul/verde de Amazon ECS desde la AWS Management Console o los SDK de AWS CLI.

Además de los permisos estándar de IAM para crear y actualizar servicios, Amazon ECS requiere los siguientes permisos. Estos permisos se han añadido a la política de IAM `AmazonECS_FullAccess`. Para obtener más información, consulte AmazonECS_FullAccess.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codedeploy:CreateApplication",
                "codedeploy:CreateDeployment",
                "codedeploy:CreateDeploymentGroup",
                "codedeploy:GetApplication",
                "codedeploy:GetDeployment",
                "codedeploy:GetDeploymentGroup",
                "codedeploy:ListApplications",
                "codedeploy:ListDeploymentGroups",
                "codedeploy:ListDeployments",
                "codedeploy:StopDeployment",
                "codedeploy:GetDeploymentTarget",
                "codedeploy:ListDeploymentTargets",
                "codedeploy:GetDeploymentConfig",
                "codedeploy:GetApplicationRevision",
                "codedeploy:RegisterApplicationRevision",
                "codedeploy:BatchGetApplicationRevisions",
                "codedeploy:BatchGetDeploymentGroups",
                "codedeploy:BatchGetDeployments",
                "codedeploy:BatchGetApplications",
                "codedeploy:ListApplicationRevisions",
                "codedeploy:ListDeploymentConfigs",
                "codedeploy:ContinueDeployment",
                "sns:ListTopics",
                "cloudwatch:DescribeAlarms",
                "lambda:ListFunctions"
            ],
            "Resource": ["*"]
        }
    ]
}
```

> **Note**
>
> Además de los permisos estándar de Amazon ECS requeridos para ejecutar tareas y servicios, los usuarios también requieren permisos `iam:PassRole` para utilizar roles de IAM para las tareas.

CodeDeploy necesita permisos para llamar a las API de Amazon ECS, modificar Elastic Load Balancing, invocar funciones de Lambda y describir alarmas de CloudWatch, y permisos para modificar el recuento deseado del servicio en su nombre. Antes de crear un servicio de Amazon ECS que utilice el tipo de implementación “blue/green”, debe crear un rol de IAM (`ecsCodeDeployRole`). Para obtener más información, consulte Rol de IAM de CodeDeploy de Amazon ECS.

En los ejemplos de política de IAM Ejemplo de creación de servicios de Amazon ECS y Ejemplo de actualización de servicios de Amazon ECS, se muestran los permisos que necesitan los usuarios para utilizar las implementaciones azul/verde de Amazon ECS en la AWS Management Console.

Migre las implementaciones azul/verde de CodeDeploy a las implementaciones azul/verde de Amazon ECS

Las implementaciones azul/verde de CodeDeploy y las implementaciones azul/verde de Amazon ECS ofrecen una funcionalidad similar, pero difieren en su configuración y administración.

Descripción general de las implementaciones azul/verde de CodeDeploy

Al crear un servicio de Amazon ECS mediante CodeDeploy, debe:

1. Crear un equilibrador de carga con un oyente de producción y (opcionalmente) un oyente de prueba. Cada oyente está configurado con una única regla (predeterminada) que enruta todo el tráfico a un único grupo de destino (el grupo de destino principal).
2. Crear un servicio de Amazon ECS, configurado para usar el oyente y el grupo de destino, con el tipo `deploymentController` establecido en `CODE_DEPLOY`. La creación del servicio da como resultado la creación de un conjunto de tareas (azul) registrado en el grupo de destino especificado.
3. Cree un grupo de implementación de CodeDeploy (como parte de una aplicación CodeDeploy) y configúrelo con detalles del clúster de Amazon ECS, el nombre del servicio, los oyentes del equilibrador de carga, dos grupos de destino (el grupo de destino principal utilizado en la regla de oyente de producción y un grupo de destino secundario que se utilizará para tareas

de sustitución), un rol de servicio (para conceder permisos a CodeDeploy a fin de que pueda manipular los recursos de Amazon ECS y de Elastic Load Balancing) y varios parámetros que controlan el comportamiento de la implementación.

Con CodeDeploy, las nuevas versiones de un servicio se implementan mediante `CreateDeployment()`, especificando el nombre de la aplicación de CodeDeploy, el nombre del grupo de implementación y un archivo AppSpec que proporciona detalles de la nueva revisión y los enlaces de ciclo de vida opcionales. La implementación de CodeDeploy crea un conjunto de tareas de sustitución (verde) y registra sus tareas en el grupo de destino secundario. Cuando este recupere el buen estado, estará disponible para pruebas (opcional) y para producción. En ambos casos, el redireccionamiento se logra cambiando la regla de oyente correspondiente para que apunte al grupo de destino secundario asociado al conjunto de tareas verde. La reversión se logra haciendo que la regla de oyente de producción pase a ser la del grupo de destino principal.

Descripción general de las implementaciones azul/verde de Amazon ECS

En el caso de las implementaciones azul/verde de Amazon ECS, la configuración de implementación forma parte del propio servicio de Amazon ECS:

1. Debe preconfigurar el oyente de producción del equilibrador de carga con una regla que incluya dos grupos de destino con ponderaciones de 1 y 0.
2. Debe especificar los siguientes recursos o actualizar los recursos del servicio:
   - El ARN de esta regla de oyente
   - Los dos grupos de destino
   - Un rol de IAM para conceder a Amazon ECS permiso para llamar a las API de Elastic Load Balancing
   - Un rol de IAM opcional para ejecutar funciones de Lambda
   - Defina el tipo `deploymentController` en `ECS` y `deploymentConfiguration.strategy` en `BLUE_GREEN`. Esto da como resultado la creación de una implementación de servicio (azul) cuyas tareas se registran en el grupo de destino principal.

Con las implementaciones azul/verde de Amazon ECS, se crea una nueva revisión de servicio llamando a `UpdateService()` de Amazon ECS y pasando los detalles de la nueva revisión. La implementación del servicio crea nuevas tareas de revisión de servicio (verde) y las registra en el grupo de destino secundario. Amazon ECS administra las operaciones de redireccionamiento y reversión cambiando las ponderaciones de la regla de oyente.

Diferencias de implementación claves

Si bien ambos métodos dan como resultado la creación de un conjunto de tareas inicial, la implementación subyacente es diferente:

- CodeDeploy utiliza un conjunto de tareas, mientras que Amazon ECS utiliza una revisión de servicio. Los conjuntos de tareas de Amazon ECS son un constructo antiguo que ha sido sustituido por las revisiones e implementaciones de servicios de Amazon ECS. Estas últimas ofrecen una mayor visibilidad del proceso de implementación, así como del historial de implementación y revisión de servicio.
- Con CodeDeploy, los enlaces de ciclo de vida se especifican como parte del archivo AppSpec que se suministra a `CreateDeployment()`. Esto significa que los enlaces se pueden cambiar de una implementación a otra. Con las implementaciones azul/verde de Amazon ECS, los enlaces se especifican como parte de la configuración del servicio y cualquier actualización requerirá una llamada a `UpdateService()`.
- Tanto las implementaciones azul/verde de CodeDeploy como las de Amazon ECS utilizan Lambda para la implementación de enlaces, pero las entradas y salidas esperadas son diferentes.

  Con CodeDeploy, la función debe llamar a `PutLifecycleEventHookExecutionStatus()` para devolver el estado del enlace, que puede ser `SUCCEEDED` o `FAILED`. Con Amazon ECS, la respuesta de Lambda se utiliza para indicar el estado del enlace.
- CodeDeploy invoca cada enlace como una llamada única y espera que se devuelva el estado de ejecución final en una hora. Los enlaces de Amazon ECS son más flexibles, ya que pueden devolver un indicador `IN_PROGRESS`, lo que significa que el enlace debe volver a invocarse repetidamente hasta que dé como resultado `SUCCEEDED` o `FAILED`. Para obtener más información, consulte Enlaces de ciclo de vida para las implementaciones de servicios de Amazon ECS.

Enfoques de migración

Existen tres métodos principales para migrar de las implementaciones azul/verde de CodeDeploy a las implementaciones azul/verde de Amazon ECS. Cada método tiene características diferentes en términos de complejidad, riesgo, capacidad de reversión y posible tiempo de inactividad.

Reutilice los mismos recursos de Elastic Load Balancing utilizados para CodeDeploy

Debe actualizar el servicio de Amazon ECS existente para utilizar el controlador de implementación de Amazon ECS con una estrategia de implementación azul/verde en lugar del controlador de implementación de CodeDeploy. Tenga en cuenta lo siguiente cuando utilice este método:

- El procedimiento de migración es más sencillo porque está actualizando el controlador de implementación del servicio de Amazon ECS y la estrategia de implementación existentes.
- Si se configura y migra correctamente, no hay tiempo de inactividad.
- Una reversión requiere que se revierta la revisión de servicio.
- El riesgo es alto porque no hay una configuración azul/verde paralela.

Utiliza el mismo oyente y los mismos grupos de destino del equilibrador de carga que se utilizan para CodeDeploy. Si está usando AWS CloudFormation, consulta Migración de una plantilla de implementación azul/verde de CodeDeploy de AWS CloudFormation a una plantilla de AWS CloudFormation azul/verde de Amazon ECS.

1. Modifique la regla predeterminada de los oyentes de producción o de prueba para incluir el grupo de destino alternativo y establezca la ponderación del grupo de destino principal en 1 y del grupo de destino alternativo en 0.

   En el caso de CodeDeploy, los oyentes del equilibrador de carga adjunto al servicio se configuran con una sola regla (predeterminada) que enruta todo el tráfico a un único grupo de destino. En el caso de Amazon ECS (azul/verde), los oyentes del equilibrador de carga deben estar preconfigurados con una regla que incluya los dos grupos de destino con ponderaciones.
2. Actualice el servicio de Amazon ECS existente llamando a la API `UpdateService` y configurando el parámetro `deploymentController` en `ECS` y el parámetro `deploymentStrategy` en `BLUE_GREEN`. Especifique los ARN del grupo de destino, el grupo de destino alternativo, el oyente de producción y un oyente de prueba opcional.
3. Compruebe que el servicio funciona según lo previsto.
4. Elimine la configuración de CodeDeploy para este servicio de Amazon ECS, ya que ahora utiliza implementaciones azul/verde de Amazon ECS.

Servicio nuevo con un equilibrador de carga existente

Este método utiliza la estrategia azul/verde para la migración.

Tenga en cuenta lo siguiente cuando utilice este método:

- La interrupción es mínima. Solo se produce durante el intercambio de puertos de Elastic Load Balancing.
- Una reversión requiere que restituya el intercambio de puertos de Elastic Load Balancing.
- El riesgo es bajo porque hay configuraciones en paralelo. Por lo tanto, puede realizar la prueba antes de la transferencia de tráfico.

1. Deje intactos los oyentes, los grupos de destino y el servicio de Amazon ECS para la configuración de CodeDeploy a fin de que pueda revertirse fácilmente a esta configuración si es necesario.
2. Cree nuevos grupos de destino y nuevos oyentes (con puertos diferentes a los de los oyentes originales) en el equilibrador de carga existente. A continuación, cree un nuevo servicio de Amazon ECS que coincida con el servicio de Amazon ECS existente, excepto si utiliza `ECS` como controlador de implementación, `BLUE_GREEN` como estrategia de implementación, y pase los ARN a los nuevos grupos de destino y las nuevas reglas de oyente.
3. Verifique la nueva configuración enviando manualmente el tráfico HTTP al servicio. Si todo va bien, cambie los puertos de los oyentes originales por los de los nuevos para enrutar el tráfico a la nueva configuración.
4. Compruebe la nueva configuración y, si todo sigue funcionando según lo previsto, elimine la configuración de CodeDeploy.

Servicio nuevo con un nuevo equilibrador de carga

Al igual que el método anterior, este utiliza la estrategia azul/verde para la migración. La diferencia clave es que el cambio de la configuración de CodeDeploy a la configuración azul/verde de Amazon ECS se produce en una capa de proxy inversa por encima del equilibrador de carga. Algunos ejemplos de implementaciones para la capa de proxy inverso son Route 53 y CloudFront.

Este método es adecuado para los clientes que ya disponen de esta capa de proxy inverso y si toda la comunicación con el servicio se realiza a través de ella (por ejemplo, si no hay comunicación directa a nivel del equilibrador de carga).

Tenga en cuenta lo siguiente cuando utilice este método:

- Esto requiere una capa de proxy inversa.

- El procedimiento de migración es más complejo porque es necesario actualizar el controlador de implementación del servicio de Amazon ECS y la estrategia de implementación existentes.
- La interrupción es mínima. Solo se produce durante el intercambio de puertos de Elastic Load Balancing.
- Una reversión requiere que se reviertan los cambios en la configuración del proxy.
- El riesgo es bajo porque hay configuraciones en paralelo. Por lo tanto, puede realizar la prueba antes de la transferencia de tráfico.

1. No modifique la configuración de CodeDeploy existente intacta (equilibrador de carga, oyentes, grupos de destino, servicio de Amazon ECS y grupo de implementación de CodeDeploy).
2. Cree un nuevo equilibrador de carga, grupos de destino y oyentes configurados para implementaciones azul/verde de Amazon ECS.

   Configure los recursos adecuados.

   - Equilibrador de carga de aplicación: para más información, consulte Recursos de equilibrador de carga de aplicación para implementaciones azul/verde.
   - Equilibrador de carga de red: para más información, consulte Recursos del equilibrador de carga de red para las implementaciones azul/verde de Amazon ECS.
3. Cree un nuevo servicio con `ECS` como controlador de implementación y `BLUE_GREEN` como estrategia de implementación, apuntando a los nuevos recursos del equilibrador de carga.
4. Verifique la nueva configuración probándola con el nuevo equilibrador de carga.
5. Actualice la configuración del proxy inverso para enrutar el tráfico al nuevo equilibrador de carga.
6. Observe la nueva revisión de servicio y, si todo sigue funcionando según lo esperado, elimine la configuración de CodeDeploy.

Pasos a seguir a continuación

Después de migrar a las implementaciones azul/verde de Amazon ECS:

- Actualice los scripts de implementación y las canalizaciones de CI/CD para usar la API `UpdateService` de Amazon ECS en lugar de la API `CreateDeployment` de CodeDeploy.
- Actualice el monitoreo y las alertas para realizar un seguimiento de las implementaciones del servicio de Amazon ECS en lugar de las implementaciones de CodeDeploy.

- Considere la posibilidad de implementar pruebas automatizadas de su nuevo proceso de implementación para asegurarse de que funciona según lo esperado.

Migración de una implementación azul/verde de CodeDeploy a una implementación de servicio azul/verde de Amazon ECS

Con las implementaciones azul/verde de Amazon ECS, puede realizar y comprobar cambios en el servicio antes de implementarlos en un entorno de producción.

Debe crear nuevos enlaces de ciclo de vida para su implementación azul/verde de Amazon ECS.

Requisitos previos

Realice las siguientes operaciones antes de iniciar una implementación azul/verde.

1. Sustituya el rol de IAM de CodeDeploy de Amazon ECS por los siguientes permisos.
   - Para obtener información sobre los permisos de Elastic Load Balancing, consulte Rol de IAM de infraestructura de Amazon ECS para los equilibradores de carga.
   - Para obtener información acerca de los permisos de Lambda, consulte Permisos necesarios para las funciones de Lambda en las implementaciones azul/verde de Amazon ECS.
2. Desactive la automatización de CodeDeploy. Para más información, consulte Working with deployment groups in CodeDeploy en la Guía del usuario de CodeDeploy.
3. Asegúrese de disponer de la siguiente información de su implementación azul/verde de CodeDeploy. Puede reutilizar esta información para la implementación azul/verde de Amazon ECS:
   - El grupo de destino de producción
   - El oyente de producción
   - La regla de producción
   - El grupo de destino de prueba

     Este es el grupo de destino de la revisión de servicio verde,
4. Asegúrese de que los grupos de destino del equilibrador de carga de aplicación estén asociados correctamente a las reglas de oyente:
   - Si no utiliza oyentes de prueba, ambos grupos de destino (de producción y de prueba) deben estar asociados a las reglas de oyente de producción.

- Si utiliza oyentes de prueba, un grupo de destino debe estar vinculado a las reglas de oyente de producción y el otro grupo de destino debe estar vinculado a las reglas de oyente de prueba.

  Si no se cumple este requisito, la implementación del servicio fallará y se generará el siguiente error: `Service deployment rolled back because of invalid networking configuration. Both targetGroup and alternateTargetGroup must be associated with the productionListenerRule or testListenerRule.`

5. Compruebe que no haya implementaciones de servicio en curso para el servicio. Para obtener más información, consulte Visualización del historial de servicios mediante las implementaciones de servicios de Amazon ECS.
6. Las implementaciones azul/verde de Amazon ECS requieren que su servicio utilice una de las siguientes características: configure los recursos adecuados.
   - Equilibrador de carga de aplicación: para más información, consulte Recursos de equilibrador de carga de aplicación para implementaciones azul/verde.
   - Equilibrador de carga de red: para más información, consulte Recursos del equilibrador de carga de red para las implementaciones azul/verde de Amazon ECS.
   - Service Connect: para más información, consulte Recursos de Service Connect para implementaciones azul/verde de Amazon ECS.
7. Decida si desea ejecutar funciones de Lambda para las etapas del ciclo de vida de la implementación azul/verde de Amazon ECS.
   - Antes de escalar verticalmente
   - Después de escalar verticalmente
   - Transferencia de tráfico de prueba
   - Después de la transferencia de tráfico de prueba
   - Transferencia de tráfico de producción
   - Después de la transferencia de tráfico de producción

   Cree funciones de Lambda para cada etapa del ciclo de vida. Para más información, consulte Cree una función de Lambda con la consola en la Guía para desarrolladores de AWS Lambda.

Para más información sobre la actualización de un controlador de implementación de un servicio, consulte Actualización de los parámetros de servicio de Amazon ECS.

Procedimiento

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), elija el clúster.

   Aparecerá la página de detalles del clúster.
3. En la pestaña Servicios, elija el servicio.

   Aparecerá la página de detalles del servicio.
4. En el encabezado, seleccione Actualizar el tipo de controlador de implementación.

   Aparecerá la página Migrar tipo de controlador de implementación.
5. Expanda Nuevo y, a continuación, especifique los siguientes parámetros.

   a. En Tipo de controlador de implementación, elija ECS.

   b. En Estrategia de implementación, seleccione Azul/verde.

   c. En Tiempo de incorporación, introduzca la hora a la que se ejecutan las revisiones de servicio azul y verde.

   d. Para ejecutar las funciones de Lambda en una etapa del ciclo de vida, en Enlaces de ciclo de vida de la implementación, haga lo siguiente para cada función de Lambda única:

      i. Elija Agregar.

         Repita este procedimiento para cada función única que desee ejecutar.

      ii. En Función de Lambda, ingrese el nombre de la función.

      iii. En Rol, elija el rol que creó en los requisitos previos con los permisos azul/verde.

         Para obtener más información, consulte Permisos necesarios para las funciones de Lambda en las implementaciones azul/verde de Amazon ECS.

      iv. En Etapas del ciclo de vida, seleccione las etapas que ejecuta la función de Lambda.

      v. (Opcional) En Detalles del enlace, introduzca un par clave-valor que proporcione información sobre el enlace.
6. Expanda Equilibrio de carga y, a continuación, configure lo siguiente:

   a. En Rol, elija el rol que creó en los requisitos previos con los permisos azul/verde.

Para obtener más información, consulte Permisos necesarios para las funciones de Lambda en las implementaciones azul/verde de Amazon ECS.

b. En Oyente, elija el oyente de producción de su implementación azul/verde de CodeDeploy.

c. En Regla de producción, elija la regla de producción de su implementación azul/verde de CodeDeploy.

d. En Regla de prueba, elija la regla de prueba de su implementación azul/verde de CodeDeploy.

e. En Grupo de destino, elija el grupo de destino de producción de su implementación azul/verde de CodeDeploy.

f. En Grupo de destino alternativo, elija el grupo de destino de prueba de su implementación azul/verde de CodeDeploy.

7. Elija Actualizar.

Pasos a seguir a continuación

- Actualice el servicio para iniciar la implementación. Para obtener más información, consulte Actualización de un servicio de Amazon ECS.
- Monitoree el proceso de implementación para asegurarse de que sigue el patrón azul/verde:
  - Se crea y escala verticalmente la revisión de servicio verde
  - El tráfico de prueba se enruta a la revisión verde (si está configurada)
  - El tráfico de producción se transfiere a la revisión verde
  - Transcurrido el tiempo de incorporación, la revisión azul finaliza

Migración de una plantilla de implementación azul/verde de CodeDeploy de AWS CloudFormation a una plantilla de AWS CloudFormation azul/verde de Amazon ECS

Migre una plantilla de AWS CloudFormation que utilice las implementaciones azul/verde de CodeDeploy para los servicios de Amazon ECS a una que utilice la estrategia de implementación azul/verde nativa de Amazon ECS. La migración sigue el método “Reutilizar los mismos recursos de Elastic Load Balancing utilizados para CodeDeploy”. Para obtener más información, consulte la documentación de the section called “Migre las implementaciones azul/verde de CodeDeploy a las implementaciones azul/verde de Amazon ECS”.

Plantilla de origen

Esta plantilla utiliza la transformación `AWS::CodeDeployBlueGreen` y el enlace `AWS::CodeDeploy::BlueGreen` para poner en marcha implementaciones azul/verde para un servicio de Amazon ECS.

Origen

Esta es la plantilla de AWS CloudFormation completa que utiliza la implementación azul/verde de CodeDeploy. Para obtener más información, consulte el ejemplo de plantilla de implementación azul/verde en la Guía del usuario de AWS CloudFormation:

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "Vpc": {
      "Type": "AWS::EC2::VPC::Id"
    },
    "Subnet1": {
      "Type": "AWS::EC2::Subnet::Id"
    },
    "Subnet2": {
      "Type": "AWS::EC2::Subnet::Id"
    }
  },
  "Transform": [
    "AWS::CodeDeployBlueGreen"
  ],
  "Hooks": {
    "CodeDeployBlueGreenHook": {
      "Type": "AWS::CodeDeploy::BlueGreen",
      "Properties": {
        "TrafficRoutingConfig": {
          "Type": "TimeBasedCanary",
          "TimeBasedCanary": {
            "StepPercentage": 15,
            "BakeTimeMins": 5
          }
        },
        "Applications": [
          {
            "Target": {
              "Type": "AWS::ECS::Service",
```

```
              "LogicalID": "ECSDemoService"
            },
            "ECSAttributes": {
              "TaskDefinitions": [
                "BlueTaskDefinition",
                "GreenTaskDefinition"
              ],
              "TaskSets": [
                "BlueTaskSet",
                "GreenTaskSet"
              ],
              "TrafficRouting": {
                "ProdTrafficRoute": {
                  "Type": "AWS::ElasticLoadBalancingV2::Listener",
                  "LogicalID": "ALBListenerProdTraffic"
                },
                "TargetGroups": [
                  "ALBTargetGroupBlue",
                  "ALBTargetGroupGreen"
                ]
              }
            }
          }
        ]
      }
    }
  },
  "Resources": {
    "ExampleSecurityGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "GroupDescription": "Security group for ec2 access",
        "VpcId": {"Ref": "Vpc"},
        "SecurityGroupIngress": [
          {
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": 8080,
            "ToPort": 8080,
```

```
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIp": "0.0.0.0/0"
          }
        ]
      }
    },
    "ALBTargetGroupBlue": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
        "HealthyThresholdCount": 2,
        "Matcher": {
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ALBTargetGroupGreen": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
```

```
        "HealthyThresholdCount": 2,
        "Matcher": {
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ExampleALB": {
      "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
      "Properties": {
        "Scheme": "internet-facing",
        "SecurityGroups": [
          {"Ref": "ExampleSecurityGroup"}
        ],
        "Subnets": [
          {"Ref": "Subnet1"},
          {"Ref": "Subnet2"}
        ],
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "Type": "application",
        "IpAddressType": "ipv4"
      }
    },
    "ALBListenerProdTraffic": {
      "Type": "AWS::ElasticLoadBalancingV2::Listener",
      "Properties": {
        "DefaultActions": [
          {
            "Type": "forward",
```

```
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                  "Weight": 1
                }
              ]
            }
          }
        ],
        "LoadBalancerArn": {"Ref": "ExampleALB"},
        "Port": 80,
        "Protocol": "HTTP"
      }
    },
    "ALBListenerProdRule": {
      "Type": "AWS::ElasticLoadBalancingV2::ListenerRule",
      "Properties": {
        "Actions": [
          {
            "Type": "forward",
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                  "Weight": 1
                }
              ]
            }
          }
        ],
        "Conditions": [
          {
            "Field": "http-header",
            "HttpHeaderConfig": {
              "HttpHeaderName": "User-Agent",
              "Values": [
                "Mozilla"
              ]
            }
          }
        ],
        "ListenerArn": {"Ref": "ALBListenerProdTraffic"},
        "Priority": 1
```

```
      }
    },
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Sid": "",
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
      }
    },
    "BlueTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
        "ContainerDefinitions": [
          {
            "Name": "DemoApp",
            "Image": "nginxdemos/hello:latest",
            "Essential": true,
            "PortMappings": [
              {
                "HostPort": 80,
                "Protocol": "tcp",
                "ContainerPort": 80
              }
            ]
          }
        ],
        "RequiresCompatibilities": [
          "FARGATE"
        ],
```

```
        "NetworkMode": "awsvpc",
        "Cpu": "256",
        "Memory": "512",
        "Family": "ecs-demo"
      }
    },
    "ECSDemoCluster": {
      "Type": "AWS::ECS::Cluster",
      "Properties": {}
    },
    "ECSDemoService": {
      "Type": "AWS::ECS::Service",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "DesiredCount": 1,
        "DeploymentController": {
          "Type": "EXTERNAL"
        }
      }
    },
    "BlueTaskSet": {
      "Type": "AWS::ECS::TaskSet",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "LaunchType": "FARGATE",
        "NetworkConfiguration": {
          "AwsVpcConfiguration": {
            "AssignPublicIp": "ENABLED",
            "SecurityGroups": [
              {"Ref": "ExampleSecurityGroup"}
            ],
            "Subnets": [
              {"Ref": "Subnet1"},
              {"Ref": "Subnet2"}
            ]
          }
        },
        "PlatformVersion": "1.4.0",
        "Scale": {
          "Unit": "PERCENT",
          "Value": 100
        },
        "Service": {"Ref": "ECSDemoService"},
        "TaskDefinition": {"Ref": "BlueTaskDefinition"},
```

```
        "LoadBalancers": [
          {
            "ContainerName": "DemoApp",
            "ContainerPort": 80,
            "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"}
          }
        ]
      }
    },
    "PrimaryTaskSet": {
      "Type": "AWS::ECS::PrimaryTaskSet",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "Service": {"Ref": "ECSDemoService"},
        "TaskSetId": {"Fn::GetAtt": ["BlueTaskSet", "Id"]}
      }
    }
  }
}
```

Pasos para realizar la migración

Eliminación de recursos específicos de CodeDeploy

Ya no necesita las siguientes propiedades:

- La transformación `AWS::CodeDeployBlueGreen`
- El enlace `CodeDeployBlueGreenHook`
- Los recursos `GreenTaskDefinition` y `GreenTaskSet` (los administrará Amazon ECS)
- El recurso `PrimaryTaskSet` (Amazon ECS administrará los conjuntos de tareas internamente)

Reconfiguración del oyente del equilibrador de carga

Modifique el recurso `ALBListenerProdTraffic` para utilizar una acción de remisión con dos grupos de destino:

```
{
  "DefaultActions": [
    {
      "Type": "forward",
      "ForwardConfig": {
```

```
        "TargetGroups": [
          {
            "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
            "Weight": 1
          },
          {
            "TargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
            "Weight": 0
          }
        ]
      }
    }
  ]
}
```

Actualización de las propiedades de implementación

Actualice y agregue lo siguiente:

- Cambie la propiedad `DeploymentController` de `EXTERNAL` a `ECS`.
- Agregue la propiedad `Strategy` y establézcala en BLUE_GREEN.
- Agregue la propiedad `BakeTimeInMinutes`.

```
{
  "DeploymentConfiguration": {
    "MaximumPercent": 200,
    "MinimumHealthyPercent": 100,
    "DeploymentCircuitBreaker": {
      "Enable": true,
      "Rollback": true
    },
    "BakeTimeInMinutes": 5,
    "Strategy": "BLUE_GREEN"
  }
}
```

- Agregue la configuración del equilibrador de carga al servicio:

```
{
  "LoadBalancers": [
    {
      "ContainerName": "DemoApp",
```

```
      "ContainerPort": 80,
      "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
      "AdvancedConfiguration": {
        "AlternateTargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
        "ProductionListenerRule": {"Ref": "ALBListenerProdRule"},
        "RoleArn": {"Fn::GetAtt": ["ECSInfrastructureRoleForLoadBalancers", "Arn"]}
      }
    }
  ]
}
```

- Agregue la referencia de definición de tareas al servicio:

```
{
  "TaskDefinition": {"Ref": "BlueTaskDefinition"}
}
```

Creación del rol AmazonECSInfrastructureRolePolicyForLoadBalancers

Agregue un nuevo rol de IAM que permita que Amazon ECS administre los recursos del equilibrador de carga:

Sustituya la *entrada del usuario* por su valor.

```
{
  "ECSInfrastructureRoleForLoadBalancers": {
    "Type": "AWS::IAM::Role",
    "Properties": {
      "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Sid": "AllowAccessToECSForInfrastructureManagement",
            "Effect": "Allow",
            "Principal": {
              "Service": "ecs.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
          }
        ]
      },
      "ManagedPolicyArns": [
```

```
        "arn:aws:iam::aws:policy/AmazonECSInfrastructureRolePolicyForLoadBalancers"
      ]
    }
  }
}
```

Comprobación de recomendaciones

1. Implemente la plantilla migrada en un entorno que no sea de producción.
2. Compruebe que el servicio se implementa correctamente con la configuración inicial.
3. Compruebe una implementación actualizando la definición de la tarea y observando el proceso de implementación azul/verde.
4. Compruebe que el tráfico se transfiere correctamente entre las implementaciones azul y verde.
5. Para probar la funcionalidad de reversión, fuerce un error en la implementación.

Plantilla posterior a la migración

Plantilla final

Esta es la plantilla de AWS CloudFormation completa que utiliza la implementación azul/verde de Amazon ECS:

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "Vpc": {
      "Type": "AWS::EC2::VPC::Id"
    },
    "Subnet1": {
      "Type": "AWS::EC2::Subnet::Id"
    },
    "Subnet2": {
      "Type": "AWS::EC2::Subnet::Id"
    }
  },
  "Resources": {
    "ExampleSecurityGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "GroupDescription": "Security group for ec2 access",
        "VpcId": {"Ref": "Vpc"},
```

```
        "SecurityGroupIngress": [
          {
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": 8080,
            "ToPort": 8080,
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIp": "0.0.0.0/0"
          }
        ]
      }
    },
    "ALBTargetGroupBlue": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
        "HealthyThresholdCount": 2,
        "Matcher": {
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
```

```
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ALBTargetGroupGreen": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
        "HealthyThresholdCount": 2,
        "Matcher": {
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ExampleALB": {
      "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
      "Properties": {
        "Scheme": "internet-facing",
        "SecurityGroups": [
          {"Ref": "ExampleSecurityGroup"}
        ],
        "Subnets": [
          {"Ref": "Subnet1"},
          {"Ref": "Subnet2"}
        ],
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
```

```
        ],
        "Type": "application",
        "IpAddressType": "ipv4"
      }
    },
    "ALBListenerProdTraffic": {
      "Type": "AWS::ElasticLoadBalancingV2::Listener",
      "Properties": {
        "DefaultActions": [
          {
            "Type": "forward",
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                  "Weight": 1
                },
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
                  "Weight": 0
                }
              ]
            }
          }
        ],
        "LoadBalancerArn": {"Ref": "ExampleALB"},
        "Port": 80,
        "Protocol": "HTTP"
      }
    },
    "ALBListenerProdRule": {
      "Type": "AWS::ElasticLoadBalancingV2::ListenerRule",
      "Properties": {
        "Actions": [
          {
            "Type": "forward",
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                  "Weight": 1
                },
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
```

```
                  "Weight": 0
                }
              ]
            }
          }
        ],
        "Conditions": [
          {
            "Field": "http-header",
            "HttpHeaderConfig": {
              "HttpHeaderName": "User-Agent",
              "Values": [
                "Mozilla"
              ]
            }
          }
        ],
        "ListenerArn": {"Ref": "ALBListenerProdTraffic"},
        "Priority": 1
      }
    },
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Sid": "",
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
      }
    },
    "ECSInfrastructureRoleForLoadBalancers": {
      "Type": "AWS::IAM::Role",
```

```
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Sid": "AllowAccessToECSForInfrastructureManagement",
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/AmazonECSInfrastructureRolePolicyForLoadBalancers"
        ]
      }
    },
    "BlueTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
        "ContainerDefinitions": [
          {
            "Name": "DemoApp",
            "Image": "nginxdemos/hello:latest",
            "Essential": true,
            "PortMappings": [
              {
                "HostPort": 80,
                "Protocol": "tcp",
                "ContainerPort": 80
              }
            ]
          }
        ],
        "RequiresCompatibilities": [
          "FARGATE"
        ],
        "NetworkMode": "awsvpc",
        "Cpu": "256",
        "Memory": "512",
        "Family": "ecs-demo"
```

```
      }
    },
    "ECSDemoCluster": {
      "Type": "AWS::ECS::Cluster",
      "Properties": {}
    },
    "ECSDemoService": {
      "Type": "AWS::ECS::Service",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "DesiredCount": 1,
        "DeploymentController": {
          "Type": "ECS"
        },
        "DeploymentConfiguration": {
          "MaximumPercent": 200,
          "MinimumHealthyPercent": 100,
          "DeploymentCircuitBreaker": {
            "Enable": true,
            "Rollback": true
          },
          "BakeTimeInMinutes": 5,
          "Strategy": "BLUE_GREEN"
        },
        "NetworkConfiguration": {
          "AwsvpcConfiguration": {
            "AssignPublicIp": "ENABLED",
            "SecurityGroups": [
              {"Ref": "ExampleSecurityGroup"}
            ],
            "Subnets": [
              {"Ref": "Subnet1"},
              {"Ref": "Subnet2"}
            ]
          }
        },
        "LaunchType": "FARGATE",
        "PlatformVersion": "1.4.0",
        "TaskDefinition": {"Ref": "BlueTaskDefinition"},
        "LoadBalancers": [
          {
            "ContainerName": "DemoApp",
            "ContainerPort": 80,
            "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
```

```
            "AdvancedConfiguration": {
              "AlternateTargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
              "ProductionListenerRule": {"Ref": "ALBListenerProdRule"},
              "RoleArn": {"Fn::GetAtt": ["ECSInfrastructureRoleForLoadBalancers",
 "Arn"]}
            }
          }
        ]
      }
    }
  }
}
```

Migración de una implementación azul/verde de CodeDeploy a una implementación de servicio de actualización continua de Amazon ECS

Puede migrar las implementaciones de sus servicios de una implementación azul/verde de CodeDeploy a una implementación de actualización continua de Amazon ECS. Esto lo aleja de la dependencia de CodeDeploy y pasa a utilizar una implementación integrada.

El programador de servicios de Amazon ECS sustituye las tareas que se están ejecutando actualmente por unas nuevas. El número de tareas que Amazon ECS agrega o elimina del servicio durante una actualización continua se controla mediante la configuración de implementación del servicio.

Requisitos previos

Realice las siguientes operaciones antes de iniciar una implementación azul/verde.

1. Ya no necesita el rol de IAM de CodeDeploy en Amazon ECS.
2. Desactive la automatización de CodeDeploy. Para más información, consulte Working with deployment groups in CodeDeploy en la Guía del usuario de CodeDeploy.
3. Compruebe que no haya implementaciones de servicio en curso para el servicio. Para obtener más información, consulte Visualización del historial de servicios mediante las implementaciones de servicios de Amazon ECS.

Para más información sobre la actualización de un controlador de implementación de un servicio, consulte Actualización de los parámetros de servicio de Amazon ECS.

Procedimiento

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), elija el clúster.

   Aparecerá la página de detalles del clúster.
3. En la pestaña Servicios, elija el servicio.

   Aparecerá la página de detalles del servicio.
4. En el banner, elija Migrar.

   Aparece la página Actualizar la configuración de implementación.
5. Amplíe Opciones de implementación y, a continuación, especifique los siguientes parámetros.

   a. En Tipo de controlador de implementación, elija ECS.

   b. En Estrategia de implementación, seleccione Actualización continua.

   c. En Max running tasks (Máximo de tareas en ejecución), ingrese el límite máximo del número de tareas del servicio que se permiten en el estado `RUNNING` durante una implementación, como porcentaje del número de tareas deseado del servicio (redondeado al entero inferior más próximo). Para obtener más información, consulte Configuración de la implementación.

   d. En Max running tasks (Máximo de tareas en ejecución), ingrese el límite máximo del número de tareas del servicio que se permiten en el estado `RUNNING` o `PENDING` durante una implementación, como porcentaje del número de tareas deseado del servicio (redondeado al entero inferior más próximo).
6. Expanda Equilibrio de carga y, a continuación, configure lo siguiente:

   a. En Rol, elija el rol que creó en los requisitos previos con los permisos azul/verde.

      Para obtener más información, consulte Permisos necesarios para las funciones de Lambda en las implementaciones azul/verde de Amazon ECS.

   b. En Oyente, elija el oyente de producción de su implementación azul/verde de CodeDeploy.

   c. En Grupo de destino, elija el grupo de destino de producción de su implementación azul/verde de CodeDeploy.
7. Elija Actualizar.

Pasos a seguir a continuación

Debe actualizar el servicio para que los cambios surtan efecto. Para obtener más información, consulte Actualización de un servicio de Amazon ECS.

Actualización de la estrategia de implementación de Amazon ECS

Amazon ECS admite varias estrategias de implementación para actualizar sus servicios. Puede migrar entre estas estrategias en función de los requisitos de su aplicación. Este tema explica cómo migrar entre las implementaciones azul/verde y las implementaciones continuas.

Información sobre las estrategias de implementación de Amazon ECS

Antes de migrar de una estrategia de implementación a otra, es importante entender cómo funciona cada estrategia y cuáles son sus principales diferencias:

Implementaciones continuas

En una implementación continua, Amazon ECS sustituye la versión continua actual de la aplicación por una nueva versión. El programador de servicio utiliza los parámetros de porcentaje de buen estado mínimo y máximo para determinar la estrategia de implementación.

Las implementaciones continuas son más sencillas de configurar, pero ofrecen menos control sobre el proceso de implementación y el enrutamiento del tráfico.

Implementaciones blue/green

En una implementación azul/verde, Amazon ECS crea una nueva versión de su servicio (verde) junto con la versión existente (azul). Esto permite verificar la nueva versión antes de enrutar el tráfico de producción hacia ella.

Las implementaciones azul/verde proporcionan un mayor control sobre el proceso de implementación, incluidas las capacidades de transferencia, comprobación y reversión del tráfico.

Prácticas recomendadas

Siga estas prácticas recomendadas al migrar entre estrategias de implementación:

- Realice pruebas en un entorno que no sea de producción: pruebe siempre la actualización en un entorno que no sea de producción antes de aplicar cambios a los servicios de producción.
- Planifique la reversión: tenga un plan de reversión en caso de que la actualización no funcione como se esperaba.

- Monitoree durante la transición: monitoree de cerca su servicio durante y después de la migración para asegurarse de que siga funcionando correctamente.
- Actualice la documentación: actualice la documentación de implementación para que refleje la nueva estrategia de implementación.
- Tenga en cuenta el impacto en el tráfico: comprenda cómo la actualización podría afectar al tráfico de su servicio y planifique según corresponda.

Actualización de la estrategia de implementación de la actualización continua a la implementación azul/verde de Amazon ECS

Puede migrar de una implementación de actualizaciones continuas a una implementación azul/verde de Amazon ECS cuando desee realizar y comprobar cambios en el servicio antes de implementarlos en un entorno de producción.

Requisitos previos

Antes de migrar su servicio de una implementación continua a una implementación azul/verde, asegúrese de contar con lo siguiente:

- Espere a que se completen las implementaciones actuales.
- Un servicio de Amazon ECS existente que utiliza la estrategia de implementación continua.
- Si tiene varias revisiones de servicio que sirven el tráfico, Amazon ECS intenta consolidar el tráfico en una sola revisión durante la migración. Si esto no funciona, es posible que tenga que actualizar manualmente el servicio para utilizar una sola revisión antes de migrar.
- Configure los permisos adecuados.
  - Para obtener información sobre los permisos de Elastic Load Balancing, consulte Rol de IAM de infraestructura de Amazon ECS para los equilibradores de carga.
  - Para obtener información acerca de los permisos de Lambda, consulte Permisos necesarios para las funciones de Lambda en las implementaciones azul/verde de Amazon ECS.
- En función de la configuración, tendrá que llevar a cabo una de las siguientes opciones:
  - Si su servicio usa Elastic Load Balancing, actualícelo con la nueva “advancedConfiguration” e inicie una implementación continua.
  - Si su servicio usa Service Connect, actualícelo e inicie una implementación continua.
  - Si su servicio usa Elastic Load Balancing y Service Connect, lleve a cabo los dos pasos anteriores (puede usar una sola solicitud de UpdateService).

- Si su servicio no utiliza ninguna de las opciones anteriores, no será necesaria ninguna operación adicional.
- Las implementaciones azul/verde de Amazon ECS requieren que su servicio utilice una de las siguientes características. Configure los recursos adecuados.
  - Equilibrador de carga de aplicación: para más información, consulte Recursos de equilibrador de carga de aplicación para implementaciones azul/verde.
  - Equilibrador de carga de red: para más información, consulte Recursos del equilibrador de carga de red para las implementaciones azul/verde de Amazon ECS.
  - Service Connect: para más información, consulte Recursos de Service Connect para implementaciones azul/verde de Amazon ECS.

Procedimiento

1. Abra la consola de Amazon ECS en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. En la página Clústeres, elija el clúster que contiene el servicio que desea migrar.

   Se mostrará la página de detalles del clúster.
4. En la página Detalles del clúster, seleccione la pestaña Servicios.
5. Elija el servicio y, a continuación, elija Actualizar.

   Se mostrará la página del servicio de actualización
6. Expanda Opciones de implementación y, a continuación, haga lo siguiente:
7. En Estrategia de implementación, seleccione Azul/verde.
8. Configure los ajustes de implementación azul/verde:

   a. En Tiempo de incorporación, introduzca el número de minutos que las revisiones de servicio azul y verde durarán simultáneamente antes de que finalice la revisión azul.

      Esto permite disponer de tiempo para la verificación y la comprobación.

   b. (Opcional) Configure las funciones de Lambda que se van a ejecutar en etapas específicas de la implementación. En Enlaces de ciclo de vida de implementación, configure las funciones de Lambda para las siguientes etapas:

      - Antes de escalar verticalmente: se ejecuta antes de escalar verticalmente la revisión de servicio verde

- Después de escalar verticalmente: se ejecuta después de escalar verticalmente la revisión de servicio verde
- Transferencia de tráfico de prueba: se ejecuta durante el enrutamiento de tráfico de prueba hacia la revisión de servicio verde
- Después de la transferencia de tráfico de prueba: se ejecuta después de que el tráfico de prueba se enruta a la revisión del servicio verde
- Transferencia de tráfico de producción: se ejecuta durante el tráfico de producción y se enruta a la revisión de servicio verde
- Después de la transferencia de tráfico de producción: se ejecuta después de que el tráfico de producción se enruta a la revisión del servicio verde

Para agregar un enlace de ciclo de vida:

a. Elija Agregar.
b. En Función de Lambda, introduzca el nombre o el ARN de la función.
c. En Rol, elija el rol de IAM que tiene permiso para invocar la función de Lambda.
d. En Etapas del ciclo de vida, seleccione las etapas en las que debe ejecutarse la función de Lambda.
e. Opcional: en Detalles del enlace, introduzca los pares clave-valor para proporcionar información adicional al enlace.

9. Configure los ajustes del equilibrador de carga:

   a. En Equilibrio de carga, compruebe que el servicio está configurado para utilizar un equilibrador de carga.
   b. En Grupo de destino, elija el grupo de destino principal para su entorno (azul) de producción.
   c. En Grupo de destino alternativo, elija el grupo de destino para su entorno (verde) de prueba.
   d. En Regla de oyente de producción, elija la regla de oyente para enrutar el tráfico de producción.
   e. Opcional: En Regla de oyente de prueba, elija una regla de oyente para enrutar el tráfico de prueba a su entorno verde.
   f. En Rol, elija el rol de IAM que permite a Amazon ECS administrar el equilibrador de carga.

10. Revise los cambios de configuración y, a continuación, seleccione Actualizar.

Pasos a seguir a continuación

- Actualice el servicio para iniciar la implementación. Para obtener más información, consulte Actualización de un servicio de Amazon ECS.
- Monitoree el proceso de implementación para asegurarse de que sigue el patrón azul/verde:
  - Se crea y escala verticalmente la revisión de servicio verde
  - El tráfico de prueba se enruta a la revisión verde (si está configurada)
  - El tráfico de producción se transfiere a la revisión verde
  - Transcurrido el tiempo de incorporación, la revisión azul finaliza

Actualización de la estrategia de implementación azul/verde de Amazon ECS a una actualización continua

Puede migrar una implementación azul/verde a una implementación de actualización continua.

Tenga en cuenta las siguientes consideraciones al migrar a implementaciones continuas:

- Gestión del tráfico: con las implementaciones continuas, las nuevas tareas comienzan a recibir tráfico en cuanto pasan las comprobaciones de estado. No hay una fase de prueba independiente como ocurre con las implementaciones azul/verde.
- Eficiencia de los recursos: las implementaciones continuas suelen utilizar menos recursos que las implementaciones azul/verde porque sustituyen las tareas de forma gradual en lugar de crear un entorno completamente duplicado.
- Complejidad de la reversión: las implementaciones continuas hacen que las reversiones sean más complejas en comparación con las implementaciones azul/verde. Si necesita realizar una reversión, debe iniciar una nueva implementación con la definición de tarea anterior.
- Velocidad de implementación: las implementaciones continuas pueden tardar más en completarse que las implementaciones azul/verde, especialmente en el caso de los servicios con muchas tareas.
- Configuración del equilibrador de carga: la configuración del equilibrador de carga actual seguirá funcionando con las implementaciones continuas, pero el comportamiento al transferir el tráfico será diferente.

Requisitos previos

Antes de migrar el servicio de una implementación azul/verde a una implementación continua, asegúrese de tener lo siguiente:

- Un servicio de Amazon ECS existente que utiliza la estrategia de implementación azul/verde
- No hay implementaciones continuas del servicio (espere a que se completen las implementaciones actuales)
- Una comprensión clara de cómo se comportará su servicio con las implementaciones continuas

> **Note**
>
> No puede migrar un servicio a una implementación continua si tiene una implementación en curso. Espere a que se complete cualquier implementación actual antes de continuar.

Procedimiento de migración

Siga estos pasos para migrar su servicio de Amazon ECS de una implementación azul/verde a una implementación continua:

1. Abra la consola de Amazon ECS en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. En la página Clústeres, elija el clúster que contiene el servicio que desea migrar.
4. En la página Detalles del clúster, seleccione la pestaña Servicios.
5. Seleccione el servicio que desee migrar y, a continuación, elija Actualizar.
6. En la página Actualizar servicio, vaya a la sección Opciones de implementación y expándala si es necesario.
7. En Estrategia de implementación, seleccione Actualización continua.
8. Configure los ajustes de la implementación continua:
    a. En Porcentaje mínimo en buen estado, ingrese el porcentaje mínimo de tareas que el servicio debe mantener en el estado `RUNNING` durante una implementación. Este valor se especifica como un porcentaje del número deseado de tareas para el servicio.

b. En Porcentaje máximo, ingrese el porcentaje máximo de tareas que se permiten en el estado `RUNNING` o `PENDING` durante una implementación. Este valor se especifica como un porcentaje del número deseado de tareas para el servicio.

9. Opcional: en Detección de errores de implementación, configure el modo en que Amazon ECS detecta y gestiona los errores de implementación:

   a. Para utilizar el interruptor de circuito de implementación, seleccione Utilizar el interruptor de circuito de implementación de Amazon ECS.

   b. Para revertir automáticamente las implementaciones fallidas, seleccione Revertir en caso de error.

10. Revise los cambios de configuración y, a continuación, seleccione Actualizar para guardar los cambios y migrar el servicio a una implementación continua.

Amazon ECS actualizará la configuración del servicio para utilizar la estrategia de implementación continua. La próxima vez que actualice su servicio, utilizará el proceso de implementación continua.

> **Note**
>
> Al migrar de una implementación azul/verde a una implementación continua, Amazon ECS gestiona la transición de la siguiente manera:
>
> 1. Identifica la revisión de servicio activa actual que sirve el tráfico.
> 2. Mantiene la configuración del equilibrador de carga existente, pero cambia la forma en que se gestionan las nuevas implementaciones.
> 3. Prepara el servicio para futuras implementaciones continuas.

Pasos a seguir a continuación

- Actualice el servicio para iniciar la implementación. Para obtener más información, consulte Actualización de un servicio de Amazon ECS.

Resolución de problemas de actualizaciones de la estrategia de implementación de Amazon ECS

En esta sección se brindan soluciones a problemas comunes que puede encontrar al migrar estrategias de implementación.

Múltiples revisiones de servicio o conjuntos de tareas

Los siguientes problemas se refieren a tener varias revisiones de servicio para una sola implementación.

Varios conjuntos de tareas al actualizar el controlador de implementación de ECS

Mensaje de error: `Updating the deployment controller is not supported when there are multiple tasksets in the service. Please ensure your service has only one taskset and try again.`

Solución: este error se produce al intentar cambiar el tipo de controlador de implementación de un servicio con varios conjuntos de tareas activos. Para resolver este problema en el controlador de implementación `CODE_DEPLOY` o `EXTERNAL`:

1. Compruebe los conjuntos de tareas actuales:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].taskSets"
```

2. Espere a que se complete cualquier implementación en curso.
3. Fuerce una nueva implementación para limpiar los conjuntos de tareas:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --force-new-deployment
```

4. Si es necesario, elimine manualmente los conjuntos de tareas adicionales:

```
aws ecs delete-task-set --cluster your-cluster-name --service your-service-name
 --task-set task-set-id
```

5. Cuando solo quede un conjunto de tareas, vuelva a intentar actualizar el controlador de implementación.

Para obtener más información, consulte the section called “Controladores y estrategias de implementación de servicios”.

Falta el conjunto de tareas principal al actualizar el controlador de implementación ECS

Mensaje de error: `Updating the deployment controller requires a primary taskset in the service. Please ensure your service has a primary taskset and try again.`

Solución: este error se produce al intentar cambiar el tipo de controlador de implementación de un servicio que no tiene un conjunto de tareas principal. Para resolver este problema, siga estos pasos:

1. Verifique el estado del servicio y los conjuntos de tareas.). Si existe un conjunto de tareas en el servicio, debe marcarse como `ACTIVE`.

   ```
   aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].taskSets[*].[status,id]
   ```

   Si no hay ningún conjunto de tareas en estado `ACTIVE`, migre la implementación. Para más información, consulte Métodos de migración.

2. Si el servicio no tiene tareas en ejecución, implemente al menos una tarea actualizando el servicio:

   ```
   aws ecs update-service-primary-task-set --cluster your-cluster-name --service your-service-name --primary-task-set your-taskset-id
   ```

   Esto marcará el conjunto de tareas (anteriormente `ACTIVE`) en el servicio como en estado `PRIMARY`.

3. Espere a que la tarea alcance un estado de ejecución estable. Puede comprobar el estado con:

   ```
   aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].deployments"
   ```

4. Una vez que el servicio tenga una tarea principal configurada con tareas en ejecución, vuelva a intentar actualizar el controlador de implementación.

Para obtener más información, consulte the section called “Controladores y estrategias de implementación de servicios”.

Discrepancia entre el tipo de detección de errores de implementación y el controlador de implementación

Los problemas siguientes se relacionan con una discrepancia entre el tipo de detección de errores de implementación y el controlador de implementación.

Interruptor de circuito de implementación con un controlador que no es de ECS

Mensaje de error: `Deployment circuit breaker feature is only supported with ECS deployment controller. Update to ECS deployment controller and try again.`

Solución: este error se produce al intentar habilitar la característica de interruptor de circuito de implementación en un servicio que no utiliza el controlador de implementación `ECS`. El interruptor de circuito de implementación solo es compatible con el controlador de implementación `ECS`.

1. Compruebe el controlador de implementación actual del servicio:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].deploymentController"
```

2. Actualice su servicio para usar el controlador de implementación `ECS`:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

3. Una vez que el servicio utilice el controlador de implementación `ECS`, active el interruptor de ccircuito de implementación:

```
aws ecs update-service --cluster your-cluster-name --
service your-service-name --deployment-configuration
 "deploymentCircuitBreaker={enable=true,rollback=true}"
```

Para obtener más información, consulte the section called “ Detección de errores por el interruptor de circuito de implementación”.

Reversión basada en alarmas con un controlador que no es de ECS

Mensaje de error: `Alarm based rollback feature is only supported with ECS deployment controller. Update to ECS deployment controller and try again.`

Solución: este error se produce al intentar configurar la reversión basada en alarmas en un servicio que no utiliza el controlador de implementación `ECS`. La característica de reversión basada en alarmas solo es compatible con el controlador de implementación `ECS`.

1. Compruebe el controlador de implementación actual del servicio:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].deploymentController"
```

2. Actualice su servicio para usar el controlador de implementación `ECS`:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

3. Una vez que el servicio utilice el controlador de implementación `ECS`, configure la reversión basada en alarmas:

```
aws ecs update-service --cluster your-cluster-name --services your-service-name --deployment-configuration "alarms={alarmNames=[your-alarm-name],enable=true,rollback=true}"
```

Para obtener más información, consulte the section called “Detección de errores en la implementación por las alarmas de CloudWatch”.

Discrepancia entre Service Connect y el controlador de implementación

Los siguientes problemas se relacionan con una discrepancia entre Service Connect y el controlador de implementación.

Controlador `EXTERNAL` con Service Connect

Mensaje de error: `The EXTERNAL deployment controller type is not supported for services using Service Connect.`

Solución: este error se produce al intentar utilizar el controlador de implementación `EXTERNAL` con un servicio que tiene activado Service Connect. El controlador `EXTERNAL` no es compatible con Service Connect.

1. Compruebe si su servicio tiene activado Service Connect:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].serviceConnectConfiguration"
```

2. Si necesita usar el controlador de implementación `EXTERNAL`, desactive Service Connect actualizando el servicio:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --service-connect-configuration "{}"
```

3. Como alternativa, si debe usar Service Connect, utilice el controlador de implementación `ECS` en su lugar:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

Para obtener más información, consulte the section called “Controladores y estrategias de implementación de servicios”.

Service Connect con un controlador que no es de ECS

Mensaje de error: `Service Connect feature is only supported with ECS (rolling update) deployment controller. Update to ECS deployment controller and try again.`

Solución: este error se produce al intentar configurar Service Connect en un servicio que no utiliza el controlador de implementación `ECS`. La característica Service Connect solo es compatible con el controlador de implementación `ECS`.

1. Compruebe el controlador de implementación actual del servicio:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].deploymentController"
```

2. Actualice su servicio para usar el controlador de implementación de ECS:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

3. Una vez que el servicio utilice el controlador de implementación de ECS, habilite Service Connect:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --service-connect-configuration "enabled=true,namespace=your-namespace"
```

Para obtener más información, consulte the section called “Controladores y estrategias de implementación de servicios”.

Discrepancia entre el tipo de controlador y la estrategia de programación

Los siguientes problemas se relacionan con una discrepancia entre el tipo de controlador y la estrategia de programación.

Controlador `CODE_DEPLOY` con estrategia de programación `DAEMON`

Mensaje de error: `The CODE_DEPLOY deployment controller type is not supported for services using the DAEMON scheduling strategy.`

Solución: este error se produce al intentar utilizar el controlador de implementación CODE_DEPLOY con un servicio que utiliza la estrategia de programación `DAEMON`. El controlador `CODE_DEPLOY` solo es compatible con la estrategia de programación `REPLICA`.

1. Compruebe la estrategia de programación actual de su servicio:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].schedulingStrategy"
```

2. Si necesita implementaciones azul/verde, cambie su servicio para usar la estrategia de programación `REPLICA`:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --scheduling-strategy REPLICA
```

3. Como alternativa, si debe utilizar la estrategia de programación `DAEMON`, utilice el controlador de implementación `ECS` en su lugar:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

Para obtener más información, consulte the section called “Controladores y estrategias de implementación de servicios”.

Controlador EXTERNAL con la estrategia de programación DAEMON

Mensaje de error: `The EXTERNAL deployment controller type is not supported for services using the DAEMON scheduling strategy.`

Solución: este error se produce al intentar utilizar el controlador de implementación EXTERNAL con un servicio de ECS que utiliza la estrategia de programación DAEMON. El controlador EXTERNAL solo es compatible con la estrategia de programación REPLICA.

1. Compruebe la estrategia de programación actual de su servicio:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].schedulingStrategy"
```

2. Si necesita usar el controlador de implementación `EXTERNAL`, cambie su servicio para usar la estrategia de programación `REPLICA`:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --scheduling-strategy REPLICA
```

3. Como alternativa, si debe utilizar la estrategia de programación `DAEMON`, utilice el controlador de implementación `ECS` en su lugar:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

Para obtener más información, consulte the section called “Controladores y estrategias de implementación de servicios”.

Registros de servicios con tipo de lanzamiento externo

Mensaje de error: `Service registries are not supported for external launch type.`

Solución: este error se produce al intentar configurar la detección de servicios (registros de servicios) para un servicio que utiliza el tipo de lanzamiento `EXTERNAL`. La detección de servicios no es compatible con el tipo de lanzamiento `EXTERNAL`.

1. Compruebe el tipo de lanzamiento actual de su servicio:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].launchType"
```

2. Si necesita la detección de servicio, cámbielo para usar el tipo de lanzamiento `EC2` o `FARGATE`:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --launch-type FARGATE
```

3. Como alternativa, si debe usar el tipo de lanzamiento `EXTERNAL`, elimine la configuración del registro de servicio:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --service-registries "[]"
```

Para obtener más información, consulte the section called “Controladores y estrategias de implementación de servicios”.

Reversión de una actualización del controlador de implementación

Si decide que desea volver al controlador de implementaciones anterior, puede optar por uno de los siguientes procedimientos:

- Si utilizó AWS CloudFormation, puede utilizar la plantilla anterior para crear una nueva pila. Para más información, consulte Creación de una pila desde en la Guía del usuario de AWS CloudFormation.
- Si utilizó la consola de Amazon ECS o la AWS CLI, puede actualizar el servicio. Para obtener más información, consulte the section called “Actualización de un servicio”.

  Si utiliza el comando update-service, utilice la opción `--deployment-controller` y configúrela en el controlador de implementación anterior.

Visualización del historial de servicios mediante las implementaciones de servicios de Amazon ECS

Las implementaciones de servicios proporcionan una visión completa de las implementaciones. Las implementaciones de servicio proporcionan la siguiente información acerca del servicio:

- La configuración de la carga de trabajo actualmente implementada (la revisión de servicios de origen)
- La configuración de la carga de trabajo que se está implementando (la revisión de servicios de destino)
- Estado de las implementaciones
- El número de tareas con errores que detectó el interruptor
- Las alarmas de CloudWatch que están en alarma
- Cuando se inició y finalizó la implementación del servicio
- Los detalles de una reversión, si se ha producido

Para obtener información sobre las propiedades de implementación del servicio, consulte Propiedades incluidas en la implementación de un servicio de Amazon ECS.

Las implementaciones de servicios son de solo lectura y cada una tiene un identificador único.

Hay tres etapas de implementación del servicio:

Etapa	Definición	Estados asociados
Pendiente	Se ha creado una implementación de servicios, pero no se ha iniciado	PENDIENTE
Continuo	La implementación de servicios está en curso.	• IN_PROGRESS • STOP_REQUESTED • ROLLBACK_REQUESTED • ROLLBACK_IN_PROGRESS

Etapa	Definición	Estados asociados
Completado	La implementación de un servicio ha finalizado (correcta mente o con errores)	• SUCCESSFUL • STOPPED • ROLLBACK_SUCCESSFUL • ROLLBACK_FAILED

Utilice las implementaciones de servicios para comprender el ciclo de vida de su servicio y determinar si hay alguna acción que deba tomar. Por ejemplo, si se ha producido una reversión, es posible que tenga que investigar la implementación del servicio y analizar los eventos del servicio.

Puede ver el historial de 90 días más reciente de las implementaciones creadas a partir del 25 de octubre de 2024 mediante la consola, la API y la AWS CLI.

Puede detener una implementación que no se haya completado. Para obtener más información, consulte Detención de las implementaciones de servicios de Amazon ECS.

Ciclo de la implementación de servicios

Amazon ECS crea una nueva implementación de servicio automáticamente cuando se produce alguna de las siguientes acciones:

- Un usuario crea un servicio.
- Un usuario actualiza el servicio y utiliza la opción de forzar una nueva implementación.
- Un usuario actualiza una o varias propiedades del servicio que requieren una implementación.

Mientras la implementación está en curso, Amazon ECS actualiza las siguientes propiedades de implementación de servicios para reflejar el progreso de la implementación de servicios:

- El estado
- El número de tareas en ejecución

  Es posible que el número de tareas en ejecución indicado en la revisión de servicios no sea igual al número real de tareas en ejecución. Este número representa el número de tareas en ejecución cuando se completó la implementación. Por ejemplo, si ha lanzado tareas independientes de la implementación del servicio, esas tareas no se incluyen en el recuento de tareas en ejecución para la revisión de servicios.

- Detección de error en el interruptor:
  - El número de tareas que han presentado un error al iniciarse
- Detección de un error de alarma de CloudWatch
  - Las alarmas activas
- Información de reversión:
  - La hora de inicio
  - El motivo de la reversión
  - El ARN de la revisión de servicio que se utilizó para la reversión
- El motivo del estado

Amazon ECS elimina la implementación del servicio cuando usted elimina un servicio.

Estados de implementaciones de servicios

La implementación de un servicio comienza en el estado `PENDING`.

En la siguiente ilustración se muestran los estados de implementación del servicio que pueden producirse después del estado `PENDING`: `IN_PROGRESS`, `ROLLBACK_REQUESTED`, `SUCCESSFUL`, `STOP_REQUESTED`, `ROLLBACK_IN_PROGRESSS`, `ROLLBACK_FAILED`, `ROLLBACK_SUCCESSFUL` y `STOPPED`.

La siguiente información proporciona detalles sobre los estados de implementación de servicios:

- `PENDING`: se ha creado la implementación de servicios, pero no se ha iniciado.

  El estado puede pasar a ser `IN_PROGRESS`, `ROLLBACK_REQUESTED`, `STOP_REQUESTED` o `STOPPED`.
- `IN_PROGRESS`: la implementación del servicio está en curso.

  El estado puede pasar a ser `SUCCESSFUL`, `STOP_REQUESTED`, `ROLLBACK_REQUESTED`, `ROLLBACK_IN_PROGRESS` y `STOPPED`.
- `STOP_REQUESTED`: el estado de implementación del servicio pasa a ser `STOP_REQUESTED` cuando ocurre alguna de las siguientes situaciones:
  - Un usuario inicia una nueva implementación de servicios.
  - La opción de reversión no se utiliza para el mecanismo de detección de errores (el interruptor o la alarma) y el servicio no alcanza el estado `SUCCESSFUL`.

  El estado pasa a ser `STOPPED`.
- `ROLLBACK_REQUESTED`: el estado de implementación del servicio pasa a `ROLLBACK_REQUESTED` cuando un usuario solicita una reversión a través de la consola, la API o la CLI.

El estado puede pasar a ser `SUCCESSFUL`, `ROLLBACK_IN_PROGRESS` y `STOPPED`.

- `SUCCESSFUL`: el estado de implementación del servicio pasa a ser `SUCCESSFUL` cuando la implementación del servicio se completa correctamente.
- `ROLLBACK_IN_PROGRESS`: el estado de implementación del servicio pasa a ser `ROLLBACK_IN_PROGRESS` cuando se utiliza la opción de reversión para el mecanismo de detección de errores (el interruptor o la alarma) y el servicio produce un error.

  El estado pasa a ser `ROLLBACK_SUCCESSFUL` o `ROLLBACK_FAILED`.

Propiedades incluidas en la implementación de un servicio de Amazon ECS

Las siguientes propiedades se incluyen en la implementación de un servicio.

Propiedad	Descripción
ARN de implementación de servicios	El ARN de la implementación de servicios.
ARN de servicio	El ARN del servicio para esta implementación de servicios.
ARN del clúster	El ARN del clúster que aloja el servicio.
Hora de creación de la implementación de servicios	La hora a la que se creó la implementación de servicios.
Hora de inicio de la implement ación de servicios	La hora a la que se inició la implementación de servicios.
Hora de finalización de la implementación de servicios	La hora a la que finalizó la implementación de servicios.
Hora de detención de la implementación de servicios	La hora a la que se detuvo la implementación de servicios.

Propiedad	Descripción
Hora de actualización de la implementación de servicios	La hora de la última actualización de la implementación de servicios.
Revisiones de servicios de origen	Las revisiones del servicio que se están ejecutando actualmente. Para obtener más información sobre las propiedades incluidas, consulte Propiedades incluidas en una revisión de servicios de Amazon ECS.

Propiedad	Descripción	
Configuración de implement ación	Los parámetros de implement ación incluyen la configuración del interruptor y las alarmas que determinan. • El límite mínimo del número de tareas que se deben ejecutar para un servicio durante una implement ación o cuando una instancia de contenedor se está agotando, como un porcentaje del número deseado de tareas para el servicio. • El límite máximo del número de tareas que se deben ejecutar para un servicio durante una implementación o cuando una instancia de contenedo r se está agotando, como un porcentaje del número deseado de tareas para el servicio. • La configuración del interruptor. • Las alarmas utilizadas para la detección de errores.	

Propiedad	Descripción
Revisión de servicios de destino	La revisión de servicios que se debe implementar. Una vez que la implement ación se complete correctam ente, la revisión de servicios de destino es la revisión de servicios en ejecución.
Estados de implementación de servicios	El estado de la implement ación de servicios. Los valores válidos son PENDING, SUCCESSFU L, STOP_REQUESTED, STOP_IN_PROGRESS, IN_PROGRESS, ROLLBACK_ IN_PROGRESS, ROLLBACK_ SUCCESSFUL y ROLLBACK_ FAILED.
Información de estado de la implementación de servicios	Información sobre por qué la implementación del servicio se encuentra en el estado actual. Por ejemplo, el interruptor detectó un error.
Información de reversión	Las opciones de reversión que utiliza la implementación del servicio cuando se produce un error en la implementación.
Opciones de interruptor de implementación de servicios	El interruptor que determina que se produjo un error en la implementación de un servicio.

Propiedad	Descripción	
Alarmas de CloudWatch para la implementación de servicios	Las alarmas de CloudWatch que determinan cuándo se produce un error en la implementación de un servicio.	

Permisos necesarios para ver las implementaciones de servicios de Amazon ECS

Si sigue la práctica recomendada de concesión de privilegios mínimos, debe agregar más permisos para ver las implementaciones de servicios en la consola.

Debe acceder a las siguientes acciones:

- ListServiceDeployments
- DescribeServiceDeployments
- DescribeServiceRevisions

Debe acceder a los siguientes recursos:

- Servicio
- Implementación de servicios
- Revisión de servicios

La siguiente política de ejemplo contiene los permisos necesarios y limita las acciones a un servicio especificado.

Sustituya `account`, `cluster-name` y `service-name` por sus valores.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "ecs:ListServiceDeployments",
              "ecs:DescribeServiceDeployments",
```

```
                "ecs:DescribeServiceRevisions"
            ],
            "Resource": [
             "arn:aws:ecs:us-east-1:123456789012:service/cluster-name/service-name",
             "arn:aws:ecs:us-east-1:123456789012:service-deployment/cluster-name/
service-name/*",
             "arn:aws:ecs:us-east-1:123456789012:service-revision/cluster-name/service-
name/*"
            ]
        }
    ]
}
```

Visualización de las implementaciones de servicios de Amazon ECS

Puede consultar el historial de 90 días más reciente de las implementaciones creadas a partir del 25 de octubre de 2024. Las implementaciones de servicios pueden tener alguno de los estados que se indican a continuación:

- En curso
- Pendiente
- Completado

Puede usar esta información para determinar si debe actualizar la forma en que se implementa el servicio o la revisión de servicios. Para obtener más información sobre las propiedades incluidas, consulte Propiedades incluidas en la implementación de un servicio de Amazon ECS.

Antes de empezar, configure los permisos necesarios para ver las implementaciones de servicios. Para obtener más información, consulte Permisos necesarios para ver las implementaciones de servicios de Amazon ECS.

Amazon ECS Console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), elija el clúster.
3. En la página de detalles del clúster, en la sección Servicios, elija el servicio.

   Aparecerá la página de detalles del servicio.
4. En la página de detalles del servicio, elija Implementaciones.

5. Elija la implementación de servicios que desea ver.

Visualización de las implementaciones de servicios de este tipo de implementación	Haga lo siguiente
Implementación en curso	En Implementaciones en curso, elija el ID de implementación.
Última implementación	En Última implementación, elija el ID de implement ación.
Implementaciones finalizad as	En Implementaciones de servicios, elija el ID de implementación.

Se abrirá la página de detalles de la implementación de servicios.

6. (Opcional) Compare las revisiones del servicio para ver las diferencias.

   En Revisiones del servicio, elija Comparar revisiones y, a continuación, seleccione 2 revisiones para comparar.

   Las revisiones del servicio se muestran una al lado de la otra con las diferencias resaltadas.

AWS CLI

1. Ejecute `list-service-deployments` para recuperar el ARN de implementación del servicio.

   Sustituya las variables por valores propios.

```
aws ecs list-service-deployments --cluster cluster-name --service service-name
```

   Anote el identificador serviceDeploymentArn de la implementación que quiere ver.

```
{
    "serviceDeployments": [
        {
            "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-
deployment/example/sd-example/NCWGC2ZR-taawPAYrIaU5",
            "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/example/
sd-example",
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/example",
            "targetServiceRevisionArn": "arn:aws:ecs:us-
west-2:123456789012:service-revision/example/sd-example/4980306466373577095",
            "status": "SUCCESSFUL"
        }
    ]
}
```

2. Ejecute `describe-service-deployments`. Utilice el identificador `serviceDeploymentArn` que se devolvió de `list-service-deployments`.

   Sustituya las variables por valores propios.

```
aws ecs describe-service-deployments --service-deployment-arns
 arn:aws:ecs:region:123456789012:service-deployment/cluster-name/service-
name/NCWGC2ZR-taawPAYrIaU5
```

Pasos a seguir a continuación

Puede ver los detalles de las revisiones de servicios en la implementación. Para obtener más información, consulte Visualización de los detalles de la revisión de servicios de Amazon ECS

Revisiones de servicios de Amazon ECS

Una revisión de servicios contiene un registro de la configuración de carga de trabajo que Amazon ECS intenta implementar. Cada vez que crea o implementa un servicio, Amazon ECS crea y captura automáticamente la configuración que intenta implementar en la revisión de servicios.

Las revisiones de servicios son de solo lectura y tienen identificadores únicos. Para obtener más información sobre las propiedades incluidas, consulte Propiedades incluidas en una revisión de servicios de Amazon ECS.

Las revisiones de servicios proporcionan los siguientes beneficios:

- Durante una implementación de un servicio, puede comparar la revisión de servicios actualmente implementada (origen) con la que se está implementando (destino).
- Cuando utiliza la opción de reversión para una implementación de un servicio, Amazon ECS restablece automáticamente la implementación del servicio hasta la última revisión de servicios implementada correctamente.
- Las revisiones de servicios contienen el registro de la configuración de la carga de trabajo en un recurso.

Ciclo de vida de las revisiones de servicios

Amazon ECS crea automáticamente una nueva revisión de servicios al crear un servicio o al actualizar una propiedad del servicio que inicia una implementación.

Amazon ECS no crea una nueva revisión de servicios para una operación de reversión. Amazon ECS utiliza la última revisión de servicios correcta para la reversión.

Una revisión de servicios no se puede cambiar.

Amazon ECS elimina la revisión de servicios cuando usted elimina un servicio.

Puede ver las revisiones de servicios creadas a partir del 25 de octubre de 2024 mediante la consola, la API y la CLI.

Propiedades incluidas en una revisión de servicios de Amazon ECS

Las siguientes propiedades se incluyen en una revisión de servicios.

Recurso	Descripción
ARN de servicio	El ARN que identifica el servicio.
ARN del clúster	El ARN del clúster que aloja el servicio.
ARN de definición de tarea	El ARN de la definición de tareas que se utiliza para las tareas del servicio.

Recurso	Descripción	
Registros de servicios	Los detalles de los registros de servicios que se utilizan para la detección de servicios. • El nombre del contenedor El valor del nombre del contenedor que se utilizará para el servicio de detección de servicios. • El puerto del contenedor El valor del puerto que se utilizará para el servicio de detección de servicios. Este valor también se especifica en la definición de tareas. • El puerto El valor del puerto utilizado si el servicio de detección de servicios se especifica en un registro de SRV. • El ARN del registro El nombre de recurso de Amazon (ARN) del registro de servicios.	

Recurso	Descripción	
Proveedores de capacidad	Los detalles de la estrategia del proveedor de capacidad. • El nombre del proveedor de capacidad El nombre del proveedor de capacidad que se utiliza para el servicio. • Peso del proveedor de capacidad El porcentaje relativo del número total de tareas lanzadas que debe utilizar el proveedor de capacidad especificado. • La base del proveedor de capacidad El número de tareas, como mínimo, que se ejecutará n en el proveedor de capacidad especificado.	
Imágenes de contenedor	Los detalles acerca de las imágenes del contenedor. • El nombre del contenedor • El resumen de imágenes • La imagen del contenedor	

Recurso	Descripción
Red	La configuración de red del servicio. • Si hay o no una dirección IP pública • Las subredes en las que se ejecutan las tareas • Los grupos de seguridad que controlan el tráfico del servicio
Tipo de lanzamiento	El tipo de lanzamiento que se utiliza para el servicio.

Propiedades específicas de Fargate	Si el tipo de lanzamien to es Fargate, se trata de información sobre la versión de Fargate.
	• Versión de la plataforma y la familia de plataformas
	• Almacenamiento efímero
	  • Cantidad de almacenam iento efímero que se asignará a la implement ación.
	  • Clave de KMS que se va a utilizar para el cifrado del almacenamiento.

Recurso	Descripción	
Los volúmenes de Amazon EBS que se configuran en la implementación	La configuración de un volumen que se especifica en la definición de la tarea como un volumen que se configura en el momento del lanzamien to. • Encriptado Indica si el volumen está cifrado. • Tipo de sistema de archivos El tipo de sistema de archivos de Linux para el volumen. • IOPS Número de operaciones de E/S por segundo (IOPS). • Clave de KMS • rol de IAM El ARN del rol de IAM asociado a este volumen. • Tamaño El tamaño del volumen de en GiB. • ID de instantánea El ID de la instantánea del volumen. • Especificaciones de etiquetas	

Recurso	Descripción
	La configuración de etiquetas del volumen. • Rendimiento El rendimiento aprovisionado para un volumen. • Tipo de volumen El tipo de volumen.
Service Connect	La configuración de Service Connect. • La indicación de si Service Connect está en uso • El espacio de nombres • La lista de servicios interconectados • La configuración de registro de Service Connect
Equilibradores de carga de los servicios	Los equilibradores de carga que enrutan el tráfico del servicio. • Nombre • El nombre del contenedor • El puerto del contenedor • ARN del grupo de destino
Supervisión en tiempo de ejecución	Indica si la supervisión del tiempo de ejecución está activada.

Recurso	Descripción
Fecha de creación	La fecha en que se creó la revisión de servicios.
VPC Lattice	La configuración de VPC Lattice para la revisión de servicios.

Visualización de los detalles de la revisión de servicios de Amazon ECS

Puede ver información sobre los siguientes tipos de revisiones de servicios que se crearon el 25 de octubre de 2024 o después:

- Origen: configuración de carga de trabajo actualmente implementada
- Destino: configuración de la carga de trabajo que se implementa

Amazon ECS Console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), elija el clúster.
3. En la página de detalles del clúster, en la sección Servicios, elija el servicio.

   Aparecerá la página de detalles del servicio.
4. En la página de detalles del servicio, elija Implementaciones.
5. Elija la revisión de servicios que quiere ver.

Visualización de las revisiones del servicio de este tipo de implementación	Haga lo siguiente
Implementaciones continuas	En Implementaciones en curso, haga lo siguiente: • Para ver la revisión de servicios de origen, en

Visualización de las revisiones del servicio de este tipo de implementación	Haga lo siguiente
	Revisiones de servicios, elija el ID de revisión de servicios para el tipo de revisión de origen. • Para ver la revisión de servicios de destino, en Revisiones de servicios, elija el ID de revisión de servicios para el tipo de revisión de destino.
Última implementación	En Última implementación, seleccione la revisión de servicios de destino.
Implementaciones finalizadas	En Implementaciones de servicios, haga lo siguiente: • En Revisión de servicios de destino, elija el ID.

AWS CLI

1. Ejecute `describe-service-deployments` para recuperar el ARN de la revisión de servicios.

   Sustituya las variables por valores propios.

   ```
   aws ecs describe-service-deployments --service-deployment-arns
    arn:aws:ecs:region:account-id:service/cluster-name/service-name/NCWGC2ZR-
   taawPAYrIaU5
   ```

   Anote el `arn` de `sourceServiceRevisions` o `targetServiceRevisions`.

```
{
    "serviceDeployments": [
        {
            "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-
deployment/example/sd-example/NCWGC2ZR-taawPAYrIaU5",
            "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/example/
sd-example",
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/example",
            "updatedAt": "2024-09-10T16:49:35.572000+00:00",
            "sourceServiceRevision": {
                "arn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
example/sd-example/4980306466373578954",
                "requestedTaskCount": 0,
                "runningTaskCount": 0,
                "pendingTaskCount": 0
            },
            "targetServiceRevision": {
                "arn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
example/sd-example/4980306466373577095",
                "requestedTaskCount": 0,
                "runningTaskCount": 0,
                "pendingTaskCount": 0
            },
            "status": "IN_PROGRESS",
            "deploymentConfiguration": {
                "deploymentCircuitBreaker": {
                    "enable": false,
                    "rollback": false
                },
                "maximumPercent": 200,
                "minimumHealthyPercent": 100
            }
        }
    ],
    "failures": []
}
```

2. Ejecute `describe-service-revisions`. Utilice el identificador `arn` que se devolvió de `describe-service-deployments`.

   Sustituya las variables por valores propios.

```
aws ecs describe-service-revisions --service-revision-arns
 arn:aws:ecs:region:123456789012:service-revision/cluster-name/service-
name/4980306466373577095
```

Equilibrio de un servicio de Amazon ECS entre zonas de disponibilidad

Para ayudar a que sus aplicaciones alcancen una alta disponibilidad, le recomendamos configurar sus servicios multitarea para que se ejecuten en varias zonas de disponibilidad. En el caso de los servicios que especifican que su primera estrategia de ubicación es la distribución por zonas de disponibilidad, AWS hace todo lo posible por distribuir uniformemente las tareas de servicio entre las zonas de disponibilidad disponibles.

Sin embargo, puede haber ocasiones en las que la cantidad de tareas que se ejecutan en una zona de disponibilidad no sea la misma que en otras zonas de disponibilidad, como, por ejemplo, después de una interrupción en una zona de disponibilidad. Para corregir este desequilibrio de tareas, puede habilitar la característica de reequilibrio de las zonas de disponibilidad.

Con el reequilibrio de las zonas de disponibilidad, Amazon ECS supervisa de forma continua la distribución de las tareas entre las zonas de disponibilidad para cada uno de los servicios. Cuando Amazon ECS detecta una distribución desigual de las tareas, toma medidas automáticamente para reequilibrar la carga de trabajo entre las zonas de disponibilidad. Esto implica lanzar nuevas tareas en las zonas de disponibilidad con el menor número de tareas y finalizar las tareas en las zonas de disponibilidad sobrecargadas.

Esta redistribución garantiza que ninguna zona de disponibilidad se convierta en un punto de error, lo que ayuda a mantener la disponibilidad general de las aplicaciones en contenedores. El proceso de reequilibrio automatizado elimina la necesidad de intervención manual, lo que acelera el tiempo de recuperación después de un evento.

A continuación, se muestra el proceso de reequilibrio de zonas de disponibilidad:

1. Amazon ECS comienza a supervisar un servicio una vez que alcanza el estado estable y analiza el número de tareas que se ejecutan en cada zona de disponibilidad.
2. Amazon ECS realiza las siguientes operaciones cuando detecta un desequilibrio en el número de tareas que se ejecutan en cada zona de disponibilidad:
   - Envía un evento de servicio que indica que se está iniciando el reequilibrio de las zonas de disponibilidad.

- Inicia tareas en las zonas de disponibilidad con el menor número de tareas en ejecución.
- Detiene las tareas en las zonas de disponibilidad con el mayor número de tareas en ejecución.
- El programador espera a que las tareas recién iniciadas tengan el estado `HEALTHY` y `RUNNING` antes de detenerlas en la zona de disponibilidad sobredimensionada.
- Envía un evento de servicio con el resultado del reequilibrio de la zona de disponibilidad.

Cómo detecta Amazon ECS la distribución desigual de las tareas

Amazon ECS determina un desequilibrio en el número de tareas que se ejecutan en cada zona de disponibilidad dividiendo el recuento de tareas deseado del servicio entre el número de zonas de disponibilidad configuradas. Si el recuento de tareas deseado no se divide de manera uniforme, Amazon ECS distribuye el resto de las tareas de manera uniforme entre las zonas de disponibilidad configuradas. Cada zona de disponibilidad debe tener al menos una tarea.

Por ejemplo, considere un servicio de Amazon ECS con un recuento deseado de dos tareas configurado para dos zonas de disponibilidad. En este escenario, el recuento de tareas deseado se divide de manera uniforme. Una distribución equilibrada sería una tarea por zona de disponibilidad. Si hay dos tareas en la zona de disponibilidad 1 y cero tareas en la zona de disponibilidad 2, Amazon ECS aplicaría el reequilibrio iniciando una tarea en la zona de disponibilidad 2 antes de detener una tarea en la zona de disponibilidad 1.

Ahora, considere un servicio de Amazon ECS con un recuento deseado de tres tareas configurado para dos zonas de disponibilidad. En este escenario, el recuento de tareas deseado no se divide de manera uniforme. Una distribución equilibrada sería una tarea en la zona de disponibilidad 1 y dos tareas en la zona de disponibilidad 2, ya que cada zona de disponibilidad tiene al menos una tarea y la tarea restante se coloca en la zona de disponibilidad 2.

Considere un servicio de Amazon ECS que tenga un recuento deseado de cinco tareas configuradas para tres zonas de disponibilidad. En este escenario, el recuento de tareas deseado no se divide de manera uniforme. Una distribución equilibrada sería una tarea en la zona de disponibilidad 1 y dos tareas en cada una de las zonas de disponibilidad 2 y 3. Después de asignar una tarea a cada zona de disponibilidad, las dos tareas restantes se distribuyen de manera uniforme entre las zonas de disponibilidad.

Consideraciones para configurar el reequilibrio de zonas de disponibilidad

Tenga en cuenta lo siguiente cuando desee configurar el reequilibrio de zonas de disponibilidad:

- El reequilibrio de la zona de disponibilidad es compatible con los tipos de lanzamiento de Fargate y EC2. En el caso de Fargate, Amazon ECS redistribuirá automáticamente las tareas entre las zonas de disponibilidad disponibles para mantener el equilibrio. Para el tipo de lanzamiento de EC2, Amazon ECS reequilibra las tareas entre las instancias de contenedores existentes en la medida en que sea posible, y respeta las estrategias y restricciones de ubicación definidas. Sin embargo, Amazon ECS no puede lanzar nuevas instancias en zonas de disponibilidad infrautilizadas como parte del proceso de reequilibrio, lo que limita el reequilibrio a las instancias de contenedor existentes.
- El reequilibrio de las zonas de disponibilidad funciona en las siguientes configuraciones:
  - Servicios que utilizan la estrategia `Replica`
  - Los servicios que especifican la zona de disponibilidad se distribuyen como primera estrategia de ubicación de tareas o no especifican una estrategia de ubicación.
- No puede utilizar el reequilibrio de zonas de disponibilidad con servicios que cumplan alguno de los siguientes criterios:
  - Utiliza la estrategia `Daemon`
  - Utiliza el tipo de lanzamiento `EXTERNAL` (ECS Anywhere)
  - Utiliza el 100 % para el valor de `maximumPercent`
  - Utiliza un equilibrador de carga clásico
  - Utiliza `attribute:ecs.availability-zone` como restricción de ubicación de tareas

Estrategias y restricciones de ubicación con el reequilibrio de las zonas de disponibilidad

Las estrategias de ubicación determinan la forma en que Amazon ECS selecciona las instancias de contenedor y las zonas de disponibilidad para finalizar la ubicación de tareas. Las restricciones de ubicación de las tareas son reglas que determinan si una tarea puede ejecutarse en una instancia de contenedor específica.

Para el tipo de lanzamiento de EC2, puede usar estrategias de ubicación y restricciones de ubicación junto con el reequilibrio de la zona de disponibilidad. Sin embargo, para que el reequilibrio de las zonas de disponibilidad funcione, la estrategia de distribución de las zonas de disponibilidad debe ser la primera estrategia especificada.

El reequilibrio de la zona de disponibilidad es compatible con varias combinaciones de estrategias de ubicación. Por ejemplo, puede crear una estrategia que primero distribuye las tareas de forma

uniforme en las zonas de disponibilidad y, a continuación, agrupa las tareas en contenedores en función de la memoria de cada zona de disponibilidad. En este caso, el reequilibrio de las zonas de disponibilidad funciona porque primero se especifica la estrategia de distribución de las zonas de disponibilidad.

Es importante tener en cuenta que el reequilibrio de las zonas de disponibilidad no funcionará si la primera estrategia de la matriz de estrategias de ubicación no es un componente de distribución de las zonas de disponibilidad. Este requisito garantiza que el objetivo principal de la distribución de tareas sea mantener el equilibrio entre las zonas de disponibilidad, lo cual es crucial para una alta disponibilidad.

Para obtener más información sobre las estrategias y restricciones de ubicación de tareas, consulte Cómo coloca Amazon ECS las tareas en las instancias de contenedor.

No se admiten estrategias ni restricciones de ubicación de tareas para tareas que utilizan el tipo de lanzamiento de Fargate. Fargate hará todo lo posible para distribuir las tareas entre las zonas de disponibilidad accesibles. Si el proveedor de capacidad incluye Fargate y Fargate Spot, el comportamiento de distribución es independiente para cada proveedor de capacidad.

La estrategia de ejemplo siguiente distribuye las tareas de forma uniforme en las zonas de disponibilidad y, a continuación, agrupa las tareas en contenedores en función de la memoria de cada zona de disponibilidad. El reequilibrio de la zona de disponibilidad es compatible con el servicio porque la estrategia de `spread` es lo primero.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Activación del reequilibrio de la zona de disponibilidad

Debe habilitar el reequilibrio de la zona de disponibilidad para los servicios nuevos y existentes.

Puede habilitar y deshabilitar el reequilibrio de zonas de disponibilidad mediante la consola, las API o la AWS CLI.

Tipo de servicio	API	Consola	CLI
Existentes	UpdateService	Actualización de un servicio de Amazon ECS	update-service
New	CreateService	Creación de una implementación de actualización continua de Amazon ECS	create-service

En el siguiente ejemplo se muestra cómo habilitar el reequilibrio de servicios al crear un nuevo servicio:

```
aws ecs create-service \
    --cluster my-cluster \
    --service-name my-service \
    --task-definition my-task-definition:1 \
    --desired-count 6 \
    --availability-zone-rebalancing ENABLED
```

Seguimiento del reequilibrio de las zonas de disponibilidad de Amazon ECS

Puede comprobar si el reequilibrio de las zonas de disponibilidad está habilitado para un servicio en la consola o mediante una llamada a `describe-services`. El siguiente ejemplo se puede utilizar para ver el estado con la CLI.

La respuesta será `ENABLED` o `DISABLED`.

```
aws ecs describe-services \
    --services service-name \
    --cluster cluster-name \
    --query services[0].availabilityZoneRebalancing
```

Eventos de servicio

Amazon ECS envía eventos de acciones de servicio para ayudar a comprender el ciclo de vida del reequilibrio de la zona de disponibilidad.

Evento	Escenario	Tipo	Más información
`SERVICE_REBALANCING_STARTED`	Amazon ECS inicia una operación de reequilibrio de la zona de disponibilidad.	INFO	service (*nombre-del-servicio*) is not AZ balanced with *número-de-tareas* tasks in *zona de disponibilidad 1*, *número-de-tareas* in *zona de disponibilidad 2*, and *número-de-tareas* in *zona de disponibilidad 3*. El reequilibrio de la zona de disponibilidad está en curso.
`SERVICE_REBALANCING_COMPLETED`	Se completa la operación de reequilibrio de la zona de disponibilidad.	INFO	service (*nombre-del-servicio*) is AZ balanced with *número-de-tareas* tasks in *zona de disponibilidad 1*, *número-de-tareas* tasks in *zona de disponibilidad 2*, and *número-de-tareas* tasks in *zona de disponibilidad 3*.

Evento	Escenario	Tipo	Más información
`TASKS_STARTED`	Amazon ECS inicia correctamente las tareas como parte de la operación de reequilibrio de la zona de disponibilidad	INFO	*nombre-del-servicio* has started *número-de-tareas* tasks in *zona de disponibilidad* to AZ Rebalance: *id-de-tareas* .
`TASKS_STOPPED`	Amazon ECS detiene correctamente las tareas como parte de la operación de reequilibrio de la zona de disponibilidad.	INFO	*nombre-del-servicio* has stopped *número-de-tareas* running tasks in *zona de disponibilidad* due to AZ rebalancing: *id-de-tarea* .

Evento	Escenario	Tipo	Más información
`SERVICE_TASK_PLACEMENT_FAILURE`	Amazon ECS no pudo iniciar una tarea como parte de la operación de reequilibrio de la zona de disponibilidad.	ERROR	Para el tipo de lanzamiento de EC2, consulte service (*nombre-del-servicio*) is unable to place a task in *zona de disponibilidad* because no container instance met all of its requirements.. Para el tipo de lanzamiento de Fargate, consulte service (*nombre-del-servicio*) is unable to place a task in *zona de disponibilidad*..
`TASKSET_SCALE_IN_FAILURE_BY_TASK_PROTECTION`	La operación de reequilibrio de la zona de disponibilidad está bloqueada porque se utiliza la protección de tareas.	INFO	service (*nombre-del-servicio*) was unable to AZ Rebalance because *nombre-del-conjunto-de-tareas* was unable to scale in due to *motivo*.

Evento	Escenario	Tipo	Más información
`SERVICE_REBALANCING_STOPPED`	La operación de reequilibrio de la zona de disponibilidad se detuvo. Amazon ECS envía eventos adicionales que proporcionan más información.	INFO	service (*nombre-del-servicio*) stopped AZ Rebalancing.

Eventos de cambio de estado de tarea

Amazon ECS envía un evento de cambio de estado de la tarea (`START`) para cada tarea que se inicia como parte del proceso de reequilibrio.

Amazon ECS envía un evento de cambio de estado de la tarea (`STOPPED`) por cada tarea que detiene como parte del proceso de reequilibrio. El motivo está establecido en `Availability-zone rebalancing initiated by (deployment ecs-svc/`*`deployment-id`*`)`.

Para obtener más información acerca de los eventos, consulte Eventos de cambio de estado de tarea de Amazon ECS.

Solución de problemas de reequilibrio de servicios

Si tiene problemas con el reequilibrio de los servicios, tenga en cuenta los siguientes pasos de solución de problemas:

El reequilibrio no se inicia

Verifique lo siguiente:

- El reequilibrio de servicios está activado para su servicio
- El servicio usa una configuración compatible (consulte the section called “Consideraciones para configurar el reequilibrio de zonas de disponibilidad”)
- El servicio alcanzó un estado estable

Errores en la ubicación de las tareas durante el reequilibrio

Si ve eventos `SERVICE_TASK_PLACEMENT_FAILURE`:

- Para el tipo de lanzamiento de EC2: compruebe si tiene instancias de contenedor disponibles en la zona de disponibilidad de destino
- Para el tipo de lanzamiento de Fargate: compruebe si hay restricciones de recursos o cuotas de servicio que limiten la ubicación de las tareas
- Revise las restricciones de ubicación de tareas para asegurarse de que no impidan una distribución adecuada de estas

El reequilibrio se detiene inesperadamente

Si ve eventos `SERVICE_REBALANCING_STOPPED`:

- Compruebe si hay alguna protección de tareas que pueda estar bloqueando la operación
- Búsqueda de implementaciones de servicios simultáneas que puedan interrumpir el reequilibrio
- Revise los eventos de servicio para obtener información adicional sobre por qué se detuvo el reequilibrio

Prácticas recomendadas para el reequilibrio de servicios

Siga estas prácticas recomendadas para sacar el máximo provecho del reequilibrio de servicio:

- Monitoree las operaciones de reequilibrio: configure las alarmas de CloudWatch para monitorear los eventos de servicio relacionados con el reequilibrio e identificar rápidamente cualquier problema.
- Tenga en cuenta el impacto en el rendimiento: tenga en cuenta que las operaciones de reequilibrio pueden aumentar temporalmente el uso de los recursos, ya que se inician nuevas tareas antes de que se detengan las antiguas.
- Utilice la protección de tareas de forma estratégica: si tiene tareas críticas que no deberían finalizarse durante el reequilibrio, considere la posibilidad de utilizar la protección de tareas.
- Planifique la capacidad de EC2: para el tipo de lanzamiento de EC2, asegúrese de tener suficientes instancias de contenedor en todas las zonas de disponibilidad para permitir un reequilibrio efectivo.
- Compruebe el comportamiento de reequilibrio: antes de confiar en el reequilibrio en producción, compruebe cómo se comportan sus servicios durante las operaciones de reequilibrio en un entorno que no sea de producción.

Uso del equilibrador de carga para distribuir el tráfico de servicio de Amazon ECS

Opcionalmente, su servicio se puede configurar para usar Elastic Load Balancing a fin de distribuir el tráfico de manera uniforme entre las tareas de su servicio.

> **Note**
>
> Cuando utiliza conjuntos de tareas, todas las tareas del conjunto deben configurarse para utilizar Elastic Load Balancing o para no utilizar Elastic Load Balancing.

Los servicios de Amazon ECS alojados en AWS Fargate admiten los equilibradores de carga de aplicación, los equilibradores de carga de red y los equilibradores de carga de puerta de enlace. Utilice la siguiente tabla para obtener información sobre el tipo de equilibrador de carga que se debe utilizar.

Tipo de equilibrador de carga	Utilizar en estos casos	
Equilibrador de carga de aplicación	Dirija el tráfico HTTP o HTTPS (o de capa 7). Los Application Load Balancers ofrecen varias características nuevas que los hacen interesantes para utilizar con los servicios Amazon ECS: • Cada servicio puede servir el tráfico desde varios balanceadores de carga y exponer varios puertos con balance de carga especific ando varios grupos de destino.	

Tipo de equilibrador de carga	Utilizar en estos casos	
	• Se admiten en tareas alojadas en instancias de Fargate y de EC2. • Los Application Load Balancers permiten a los contenedores utilizar la asignación de puerto de host dinámico (de modo que se permitan varias tareas desde el mismo servicio por instancia de contenedor). • Los balanceadores de carga de aplicaciones admiten el enrutamiento basado en rutas y las reglas de prioridad (de modo que varios servicios puedan utilizar el mismo agente de escucha en un único Application Load Balancer).	
Equilibrador de carga de red	Dirija el tráfico TCP o UDP (o de capa 4).	
Gateway Load Balancer	Dirija el tráfico TCP o UDP (o de capa 4). Utilice dispositivos virtuales , como firewalls, sistemas de prevención y detección de intrusiones, y sistemas de inspección profunda de paquetes.	

Recomendamos utilizar los equilibradores de carga de aplicación para los servicios de Amazon ECS de manera que pueda aprovechar estas características más recientes, a menos que el servicio requiera una característica que solo esté disponible en los equilibradores de carga de aplicación o equilibradores de carga de puerta de enlace. Para obtener más información acerca de Elastic Load Balancing y las diferencias entre estos tipos de balanceadores de carga, consulte la Guía del usuario de Elastic Load Balancing.

Con el equilibrador de carga, solo se paga por lo que se usa. Para obtener más información, consulte Precios de Elastic Load Balancing.

Optimización de los parámetros de comprobación de estado del equilibrador de carga para Amazon ECS

Los equilibradores de carga dirigen las solicitudes únicamente a los destinos en buen estado de las zonas de disponibilidad del equilibrador de carga. Cada destino está registrado en un grupo de destino. El equilibrador de carga comprueba el estado de cada destino; para ello, utiliza la configuración de comprobación de estado de los grupos de destino. Una vez que registra el destino, debe superar una comprobación de estado para que se considere que se encuentra en buen estado. Amazon ECS supervisa el equilibrador de carga. El equilibrador de carga envía periódicamente comprobaciones de estado al contenedor de Amazon ECS. El agente de Amazon ECS supervisa el equilibrador de carga y espera a que este informe sobre el estado del contenedor. Lo hace antes de considerar que el contenedor esté en buen estado.

Dos parámetros de comprobación de estado de Elastic Load Balancing afectan a la velocidad de implementación:

- Intervalo de comprobaciones de estado: determina la cantidad aproximada de tiempo, en segundos, entre comprobaciones de estado de un contenedor individual. De forma predeterminada, el equilibrador de carga se comprueba cada 30 segundos.

  Este parámetro se denomina:

  - `HealthCheckIntervalSeconds` en la API de Elastic Load Balancing
  - Intervalo en la consola de Amazon EC2
- Recuento de umbral en buen estado: determina la cantidad de comprobaciones de estado consecutivas correctas que deben superarse para considerar que un contenedor con un estado incorrecto vuelve a estar en estado correcto. De forma predeterminada, el equilibrador de carga requiere pasar cinco comprobaciones de estado antes de informar que el contenedor de destino está en buen estado.

Este parámetro se denomina:

- `HealthyThresholdCount` en la API de Elastic Load Balancing
- Umbral en buen estado en la consola de Amazon EC2

Importante: Para los objetivos recién registrados, solo se requiere una comprobación de estado correcta para considerar que el objetivo está en buen estado, independientemente del valor establecido para el umbral de estado correcto. El umbral de estado correcto solo se aplica cuando un objetivo está pasando de un estado incorrecto a un estado correcto.

Con la configuración predeterminada, si el objetivo se vuelve incorrecto y luego se recupera, el tiempo total para determinar el estado de un contenedor es de 2 minutos y 30 segundos (`30 seconds * 5 = 150 seconds`).

Puede acelerar el proceso de comprobación de estado si el servicio se inicia y se estabiliza en menos de 10 segundos. Para acelerar el proceso, reduzca el intervalo de comprobación de estado y el recuento de umbral de estado.

- `HealthCheckIntervalSeconds` (nombre de la API de Elastic Load Balancing) o Intervalo (nombre de la consola de Amazon EC2): 5
- `HealthyThresholdCount` (nombre de la API de Elastic Load Balancing) o Umbral de estado correcto (nombre de la consola de Amazon EC2): 2

Con esta configuración, el proceso de comprobación de estado tarda 10 segundos, en lugar de los 2 minutos y 30 segundos predeterminados.

Para obtener más información acerca de las comprobaciones de estado de Elastic Load Balancing, consulte Health checks for your target groups en la Guía de usuario de Elastic Load Balancing.

Optimización de los parámetros de drenaje de conexiones del equilibrador de carga para Amazon ECS

Para permitir la optimización, los clientes mantienen una conexión permanente con el servicio de contenedores. Esto permite a las solicitudes posteriores de ese cliente reutilizar la conexión existente. Cuando quiera detener el tráfico hacia un contenedor, notifica al equilibrador de carga. El equilibrador de carga hace una comprobación periódica para ver si el cliente cerró la conexión persistente. El agente de Amazon ECS supervisa el equilibrador de carga y espera a que este informe que la conexión Keep Alive está cerrada (el objetivo está en un estado `UNUSED`).

El tiempo que el equilibrador de carga espera para mover el objetivo al estado `UNUSED` es el retraso en la cancelación del registro. Puede configurar el siguiente parámetro del equilibrador de carga para acelerar las implementaciones.

- `deregistration_delay.timeout_seconds`: 300 (predeterminado)

Si tiene un servicio con un tiempo de respuesta inferior a 1 segundo, establezca el parámetro en el siguiente valor para que el equilibrador de carga solo espere 5 segundos antes de interrumpir la conexión entre el cliente y el servicio de backend:

- `deregistration_delay.timeout_seconds`: 5

> **Note**
>
> No establezca el valor en 5 segundos cuando tenga un servicio con solicitudes de larga duración, como cargas lentas de archivos o conexiones de streaming.

Capacidad de respuesta de SIGTERM

Amazon ECS envía primero una señal SIGTERM a la tarea para notificar que la aplicación debe finalizar y cerrarse. A continuación, Amazon ECS envía un mensaje SIGKILL. Cuando las aplicaciones ignoran el SIGTERM, el servicio Amazon ECS debe esperar a enviar la señal SIGKILL para terminar el proceso.

El tiempo que Amazon ECS espera para enviar el mensaje SIGKILL viene determinado por la siguiente opción de agente de Amazon ECS:

- `ECS_CONTAINER_STOP_TIMEOUT`: 30 (predeterminado)

  Para obtener más información sobre el parámetro de agente contenedor, consulte Amazon ECS Container Agent en GitHub.

Para acelerar el periodo de espera, defina el parámetro del agente de Amazon ECS en el siguiente valor:

- `ECS_CONTAINER_STOP_TIMEOUT`: 2

Si la solicitud tarda más de 1 segundo, multiplique el valor por 2 y utilice ese número como valor.

En este caso, Amazon ECS espera 2 segundos a que se cierre el contenedor y, a continuación, envía un mensaje SIGKILL cuando la aplicación no se detiene.

También puede modificar el código de la aplicación para capturar la señal SIGTERM y reaccionar ante ella. El siguiente es un ejemplo en JavaScript:

```
process.on('SIGTERM', function() {
  server.close();
})
```

Este código hace que el servidor HTTP deje de aceptar nuevas solicitudes, termine de responder a las solicitudes en curso y, a continuación, finalice el proceso de Node.js porque el bucle de eventos no tiene más tareas pendientes. Por lo tanto, si el proceso tarda solo 500 ms en finalizar sus solicitudes en tránsito, finaliza antes de tiempo sin tener que esperar a que acabe el tiempo de espera y recibir un SIGKILL.

Uso de un equilibrador de carga de aplicaciones para Amazon ECS

Un Application Load Balancer toma decisiones de enrutamiento en la capa de aplicación (HTTP/HTTPS), admite el enrutamiento basado en rutas y puede dirigir las solicitudes a uno o varios puertos de cada instancia de contenedor del clúster. Los Application Load Balancers admiten el mapeo de puertos de host dinámico. Por ejemplo, si la definición de contenedor de la tarea especifica el puerto 80 para un puerto de contenedor NGINX y el puerto 0 para el puerto de host, el puerto de host se elige dinámicamente en el rango de puertos efímeros de la instancia de contenedor (como, por ejemplo, del 32768 al 61000 en las AMI optimizadas para Amazon ECS más recientes). Cuando se lanza la tarea, el contenedor NGINX se registra en el equilibrador de carga de aplicación como una combinación de ID de instancia y puerto, y el tráfico se distribuye al ID de la instancia y al puerto correspondientes a dicho contenedor. Este mapeo dinámico le permite tener varias tareas desde un servicio único en la misma instancia de contenedor. Para obtener más información, consulte la Guía del usuario de Application Load Balancers.

Para obtener información sobre las prácticas recomendadas para establecer parámetros que aceleren las implementaciones, consulte los siguientes recursos:

- Optimización de los parámetros de comprobación de estado del equilibrador de carga para Amazon ECS

- Optimización de los parámetros de drenaje de conexiones del equilibrador de carga para Amazon ECS

Al utilizar los equilibradores de carga de aplicación con Amazon ECS, tenga en cuenta lo siguiente:

- Amazon ECS requiere el rol de IAM vinculado al servicio, que proporciona los permisos necesarios para registrar y anular el registro de destinos en el balanceador de carga al crear y detener tareas. Para obtener más información, consulte Uso de roles vinculados al servicio para Amazon ECS.
- El grupo de destino debe tener el tipo de dirección IP establecido en IPv4.

  Aunque las tareas admiten IPv6, no se admite la integración de servicios con Elastic Load Balancing IPv6.
- Para los servicios con tareas que utilizan el modo de red `awsvpc`, al crear un grupo de destino para el servicio, debe elegir `ip` como tipo de destino, no `instance`. Esto se debe a que las tareas que utilizan el modo de red `awsvpc` están asociadas a una interfaz de red elástica, no a una instancia de Amazon EC2.
- Si el servicio requiere acceso a varios puertos con equilibrador de carga, como el puerto 80 y el puerto 443 para un servicio HTTP o HTTPS, puede configurar dos oyentes. Un agente de escucha es responsable de las solicitudes HTTPS que reenvían la solicitud al servicio y otro agente de escucha es responsable de redirigir las solicitudes HTTP al puerto HTTPS adecuado. Para obtener más información, consulte Creación de un agente de escucha para el Application Load Balancer en la Guía del usuario de Application Load Balancers.
- La configuración de subred del balanceador de carga debe incluir todas las zonas de disponibilidad en las que residen las instancias de contenedor.
- Después de crear un servicio, la configuración del equilibrador de carga no se puede cambiar desde la AWS Management Console. Puede usar el Copiloto de AWS, AWS CloudFormation, AWS CLI o SDK para modificar la configuración del equilibrador de carga solo del controlador de implementación progresiva `ECS`, no AWS CodeDeploy azul/verde o exterior. Cuando agrega, actualiza o elimina una configuración de equilibrador de carga, Amazon ECS inicia una nueva implementación con la configuración actualizada de Elastic Load Balancing. Esto hace que las tareas se registren y eliminen el registro de los equilibradores de carga. Le recomendamos que lo verifique en un entorno de prueba antes de actualizar la configuración de Elastic Load Balancing. Para obtener más información acerca de cómo modificar la configuración, consulte UpdateService en la Referencia de la API de Amazon Elastic Container Service.

- Si la tarea de un servicio no supera los criterios de comprobación de estado del equilibrador de carga, la tarea se detiene y se reinicia. Este proceso continúa hasta que el servicio alcanza el número de tareas en ejecución deseadas.
- Si está teniendo problemas con los servicios habilitados para el balanceador de carga, consulte Solución de problemas de los equilibradores de carga de servicio en Amazon ECS.
- Cuando use el tipo de destino `instance`, las tareas y el equilibrador de carga deben estar en la misma VPC. Cuando se utiliza el tipo de destino `ip`, se admite la conectividad entre VPC.
- Utilice un grupo de destino único para cada servicio.

  El uso del mismo grupo de destino para varios servicios puede provocar problemas durante la implementación del servicio.

Para obtener información sobre cómo crear un equilibrador de carga de aplicaciones, consulte Create an Application Load Balancer en Application Load Balancers

Uso de un equilibrador de carga de red para Amazon ECS

Un Network Load Balancer toma las decisiones de enrutamiento en la capa de transporte (TCP/SSL). Puede atender millones de solicitudes por segundo. Una vez que el balanceador de carga ha recibido una conexión, selecciona un destino del grupo de destinos para la regla predeterminada por medio de un algoritmo hash de flujo de direccionamiento. Intenta abrir una conexión TCP con el destino seleccionado en el puerto especificado en la configuración del agente de escucha. Reenvía la solicitud sin modificar los encabezados. Los Network Load Balancers admiten el mapeo de puertos de host dinámico. Por ejemplo, si la definición de contenedor de la tarea especifica el puerto 80 para un puerto de contenedor NGINX y el puerto 0 para el puerto de host, el puerto de host se elige dinámicamente en el rango de puertos efímeros de la instancia de contenedor (como, por ejemplo, del 32768 al 61000 en las AMI optimizadas para Amazon ECS más recientes). Cuando se lanza la tarea, el contenedor NGINX se registra en el Network Load Balancer como una combinación de ID de instancia y puerto, y el tráfico se distribuye al ID de la instancia y al puerto correspondiente a ese contenedor. Este mapeo dinámico le permite tener varias tareas desde un servicio único en la misma instancia de contenedor. Para obtener más información, consulte la Guía del usuario de balanceadores de carga de red.

Para obtener información sobre las prácticas recomendadas para establecer parámetros que aceleren las implementaciones, consulte los siguientes recursos:

- Optimización de los parámetros de comprobación de estado del equilibrador de carga para Amazon ECS
- Optimización de los parámetros de drenaje de conexiones del equilibrador de carga para Amazon ECS

Al utilizar los equilibradores de carga de red con Amazon ECS, tenga en cuenta lo siguiente:

- Amazon ECS requiere el rol de IAM vinculado al servicio, que proporciona los permisos necesarios para registrar y anular el registro de destinos en el balanceador de carga al crear y detener tareas. Para obtener más información, consulte Uso de roles vinculados al servicio para Amazon ECS.
- No se pueden asociar más de cinco grupos de destino a un servicio.
- Para los servicios con tareas que utilizan el modo de red `awsvpc`, al crear un grupo de destino para el servicio, debe elegir `ip` como tipo de destino, no `instance`. Esto se debe a que las tareas que utilizan el modo de red `awsvpc` están asociadas a una interfaz de red elástica, no a una instancia de Amazon EC2.
- La configuración de subred del balanceador de carga debe incluir todas las zonas de disponibilidad en las que residen las instancias de contenedor.
- Después de crear un servicio, la configuración del equilibrador de carga no se puede cambiar desde la AWS Management Console. Puede usar el Copiloto de AWS, AWS CloudFormation, AWS CLI o SDK para modificar la configuración del equilibrador de carga solo del controlador de implementación progresiva `ECS`, no AWS CodeDeploy azul/verde o exterior. Cuando agrega, actualiza o elimina una configuración de equilibrador de carga, Amazon ECS inicia una nueva implementación con la configuración actualizada de Elastic Load Balancing. Esto hace que las tareas se registren y eliminen el registro de los equilibradores de carga. Le recomendamos que lo verifique en un entorno de prueba antes de actualizar la configuración de Elastic Load Balancing. Para obtener más información acerca de cómo modificar la configuración, consulte UpdateService en la Referencia de la API de Amazon Elastic Container Service.
- Si la tarea de un servicio no supera los criterios de comprobación de estado del equilibrador de carga, la tarea se detiene y se reinicia. Este proceso continúa hasta que el servicio alcanza el número de tareas en ejecución deseadas.
- Cuando se utilizan equilibradores de carga de puerta de enlace configurados con direcciones IP como destinos y la conservación de la IP del cliente está desactivada, las solicitudes se ven como procedentes de la dirección IP privada de los equilibradores de carga de puerta de enlace. Esto significa que los servicios detrás de un equilibrador de carga de puerta de enlace se abren

de manera efectiva al mundo en cuanto se permiten solicitudes entrantes y comprobaciones de estado en el grupo de seguridad de destino.

- Para las tareas de Fargate, debe utilizar la versión de la plataforma `1.4.0` (Linux) o `1.0.0` (Windows).
- Si está teniendo problemas con los servicios habilitados para el balanceador de carga, consulte Solución de problemas de los equilibradores de carga de servicio en Amazon ECS.
- Cuando use el tipo de destino `instance`, las tareas y el equilibrador de carga deben estar en la misma VPC. Cuando se utiliza el tipo de destino `ip`, se admite la conectividad entre VPC.
- La conservación de la dirección IP del cliente del equilibrador de carga de red también es compatible con los destinos de Fargate.
- Utilice un grupo de destino único para cada servicio.

  El uso del mismo grupo de destino para varios servicios puede provocar problemas durante la implementación del servicio.

Para obtener información sobre cómo crear un equilibrador de carga de red, consulte Create a Network Load Balancer en Equilibrador de carga de red

> **Important**
>
> Si la definición de tareas del servicio utiliza el modo de red `awsvpc` (que se requiere para el tipo de lanzamiento de Fargate), se debe elegir `ip` como tipo de destino, no `instance`. Esto se debe a que las tareas que utilizan el modo de red `awsvpc` están asociadas a una interfaz de red elástica, no a una instancia de Amazon EC2.
> No puede registrar instancias por ID de instancia si tienen los siguientes tipos de instancia: C1, CC1, CC2, CG1, CG2, CR1, G1, G2, HI1, HS1, M1, M2, M3 y T1. Puede registrar las instancias de estos tipos por dirección IP.

Uso de un equilibrador de carga de puerta de enlace para Amazon ECS

Un equilibrador de carga de puerta de enlace opera en la tercera capa del modelo de interconexión de sistemas abiertos (OSI), es decir, la capa de red. Escucha todos los paquetes IP en todos los puertos y reenvía el tráfico al grupo de destino especificado en la regla de oyentes. Mantiene la persistencia de los flujos en un dispositivo de destino específico mediante 5 tuplas (para flujos TCP/UDP) o 3 tuplas (para flujos que no son TCP/UDP). Por ejemplo, si la definición de contenedor de

la tarea especifica el puerto 80 para un puerto de contenedor NGINX y el puerto 0 para el puerto de host, el puerto de host se elige dinámicamente en el rango de puertos efímeros de la instancia de contenedor (como, por ejemplo, del 32768 al 61000 en las AMI optimizadas para Amazon ECS más recientes). Cuando se lanza la tarea, el contenedor NGINX se registra en el equilibrador de carga de puerta de enlace como una combinación de ID de instancia y puerto, y el tráfico se distribuye al ID de la instancia y al puerto correspondientes a ese contenedor. Este mapeo dinámico le permite tener varias tareas desde un servicio único en la misma instancia de contenedor. Para obtener más información, consulte What is a Gateway Load Balancer en Gateway Load Balancers.

Para obtener información sobre las prácticas recomendadas para establecer parámetros que aceleren las implementaciones, consulte los siguientes recursos:

- Optimización de los parámetros de comprobación de estado del equilibrador de carga para Amazon ECS
- Optimización de los parámetros de drenaje de conexiones del equilibrador de carga para Amazon ECS

Tenga en cuenta lo siguiente al utilizar los equilibradores de carga de puerta de enlace con Amazon ECS:

- Amazon ECS requiere el rol de IAM vinculado al servicio, que proporciona los permisos necesarios para registrar y anular el registro de destinos en el balanceador de carga al crear y detener tareas. Para obtener más información, consulte Uso de roles vinculados al servicio para Amazon ECS.
- Para los servicios con tareas que utilizan el modo de red `awsvpc`, al crear un grupo de destino para el servicio, debe elegir `ip` como tipo de destino, no `instance`. Esto se debe a que las tareas que utilizan el modo de red `awsvpc` están asociadas a una interfaz de red elástica, no a una instancia de Amazon EC2.
- En el caso de los servicios con tareas que utilizan un modo de red que no sea el de `awsvpc`, no se admiten los equilibradores de carga de puerta de enlace.
- La configuración de subred del balanceador de carga debe incluir todas las zonas de disponibilidad en las que residen las instancias de contenedor.
- Después de crear un servicio, la configuración del equilibrador de carga no se puede cambiar desde la AWS Management Console. Puede usar el Copiloto de AWS, AWS CloudFormation, AWS CLI o SDK para modificar la configuración del equilibrador de carga solo del controlador de implementación progresiva `ECS`, no AWS CodeDeploy azul/verde o exterior. Cuando agrega, actualiza o elimina una configuración de equilibrador de carga, Amazon ECS inicia una nueva

implementación con la configuración actualizada de Elastic Load Balancing. Esto hace que las tareas se registren y eliminen el registro de los equilibradores de carga. Le recomendamos que lo verifique en un entorno de prueba antes de actualizar la configuración de Elastic Load Balancing. Para obtener más información acerca de cómo modificar la configuración, consulte UpdateService en la Referencia de la API de Amazon Elastic Container Service.

- Si la tarea de un servicio no supera los criterios de comprobación de estado del equilibrador de carga, la tarea se detiene y se reinicia. Este proceso continúa hasta que el servicio alcanza el número de tareas en ejecución deseadas.
- Al utilizar el equilibrador de carga de puerta de enlace configurado con direcciones IP como destinos, las solicitudes se ven como procedentes de la dirección IP privada de los equilibradores de carga de puerta de enlace. Esto significa que los servicios detrás de un equilibrador de carga de puerta de enlace se abren de manera efectiva al mundo en cuanto se permiten solicitudes entrantes y comprobaciones de estado en el grupo de seguridad de destino.
- Para las tareas de Fargate, debe utilizar la versión de la plataforma `1.4.0` (Linux) o `1.0.0` (Windows).
- Si está teniendo problemas con los servicios habilitados para el balanceador de carga, consulte Solución de problemas de los equilibradores de carga de servicio en Amazon ECS.
- Cuando use el tipo de destino `instance`, las tareas y el equilibrador de carga deben estar en la misma VPC. Cuando se utiliza el tipo de destino `ip`, se admite la conectividad entre VPC.
- Utilice un grupo de destino único para cada servicio.

  El uso del mismo grupo de destino para varios servicios puede provocar problemas durante la implementación del servicio.

Para obtener información sobre cómo crear un equilibrador de carga de puerta de enlace, consulte Introducción a los equilibradores de carga de puertas de enlace en los equilibradores de carga de puerta de enlace .

> **Important**
>
> Si la definición de tareas del servicio utiliza el modo de red `awsvpc` (que se requiere para el tipo de lanzamiento de Fargate), se debe elegir `ip` como tipo de destino, no `instance`. Esto se debe a que las tareas que utilizan el modo de red `awsvpc` están asociadas a una interfaz de red elástica, no a una instancia de Amazon EC2.

No puede registrar instancias por ID de instancia si tienen los siguientes tipos de instancia: C1, CC1, CC2, CG1, CG2, CR1, G1, G2, HI1, HS1, M1, M2, M3 y T1. Puede registrar las instancias de estos tipos por dirección IP.

Registro de varios grupos de destino en un servicio de Amazon ECS

El servicio de Amazon ECS puede atender el tráfico procedente de varios balanceadores de carga y exponer varios puertos con de carga balanceada cuando se especifican varios grupos de destino en una definición de servicio.

Actualmente, si desea crear un servicio que especifique varios grupos de destino, debe crear el servicio mediante la API de Amazon ECS, el SDK, la AWS CLI o una plantilla de AWS CloudFormation. Una vez creado el servicio, puede ver el servicio y los grupos de destino registrados con él con la AWS Management Console. Se debe utilizar `UpdateService` para modificar la configuración del balanceador de carga de un servicio existente.

Se puede especificar en una definición de servicio varios grupos de destino utilizando el siguiente formato. Para obtener la sintaxis completa de una definición de servicio, consulte Plantilla de definición de servicio.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"container_name",
      "containerPort":container_port
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"container_name",
      "containerPort":container_port
   }
]
```

Consideraciones

Cuando se especifiquen varios grupos de destino en una definición de servicio, debe tenerse en cuenta lo siguiente.

- En el caso de los servicios que utilizan un Application Load Balancer o un Network Load Balancer, no se pueden asociar más de cinco grupos de destino a un servicio.
- La especificación de varios grupos de destino en una definición de servicio solo se admite en las siguientes condiciones:
  - El servicio debe utilizar un Application Load Balancer o un Network Load Balancer.
  - El servicio debe utilizar el tipo de controlador de implementación de actualización continua (`ECS`).
- Se admite la especificación de varios grupos de destino para servicios que contengan tareas que utilicen los tipos de lanzamiento de Fargate y EC2.
- Al crear un servicio que especifica varios grupos de destino, se debe crear el rol vinculado al servicio de Amazon ECS. Para crear el rol, omita el parámetro `role` en las solicitudes de API o la propiedad `Role` en AWS CloudFormation. Para obtener más información, consulte Uso de roles vinculados al servicio para Amazon ECS.

Ejemplos de definiciones de servicio

A continuación se muestran unos ejemplos de casos de uso para especificar varios grupos de destino en una definición de servicio. Para obtener la sintaxis completa de una definición de servicio, consulte Plantilla de definición de servicio.

Configuración de equilibradores de carga independientes para el tráfico interno y externo

En el siguiente caso de uso, un servicio utiliza dos balanceadores de carga independientes, uno para el tráfico interno y otro para el tráfico de Internet, para el mismo contenedor y puerto.

```
"loadBalancers":[
   //Internal ELB
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"nginx",
      "containerPort":8080
   },
```

```
   //Internet-facing ELB
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"nginx",
      "containerPort":8080
   }
]
```

Exposición de varios puertos desde el mismo contenedor

En el siguiente caso de uso, un servicio utiliza un balanceador de carga pero expone varios puertos desde el mismo contenedor. Por ejemplo, un contenedor Jenkins podría exponer el puerto 8080 para la interfaz web de Jenkins y el puerto 50000 para la API.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"jenkins",
      "containerPort":8080
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"jenkins",
      "containerPort":50000
   }
]
```

Exposición de puertos desde varios contenedores

En el siguiente caso de uso, un servicio utiliza un balanceador de carga y dos grupos de destino para exponer puertos desde contenedores independientes.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
```

```
      "containerName":"webserver",
      "containerPort":80
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"database",
      "containerPort":3306
   }
]
```

Escalado automático de su servicio de Amazon ECS

El escalado automático es la posibilidad de aumentar o disminuir automáticamente la cantidad de tareas deseada en el servicio de Amazon ECS. Amazon ECS utiliza el servicio de Application Auto Scaling para proporcionar esta funcionalidad. Para obtener más información, consulte la Guía del usuario de Application Auto Scaling.

Amazon ECS publica métricas de CloudWatch de la utilización de memoria y CPU promedio del servicio. Para obtener más información, consulte Métricas de uso de los servicios de Amazon ECS. Puede usar estas y otras métricas de CloudWatch para escalar horizontalmente el servicio (agregar más tareas) a fin de abordar demandas elevadas en horas pico y para reducir horizontalmente el servicio (ejecutar menos tareas) a fin de disminuir los costos durante períodos de poco uso.

Service Auto Scaling de Amazon ECS admite los siguientes tipos de escalado automático:

- Uso de una métrica objetivo para escalar los servicios de Amazon ECS: aumenta o reduce el número de tareas que ejecuta el servicio en función del valor de destino para una métrica determinada. Se asemeja a los termostatos que se utilizan para mantener la temperatura del hogar. Se selecciona la temperatura y el termostato hace el resto.
- Uso de incrementos predefinidos en función de las alarmas de CloudWatch para escalar los servicios de Amazon ECS: aumenta o reduce el número de tareas que ejecuta el servicio en función de una serie de ajustes de escalado, denominados ajustes por pasos, que variarán en función del tamaño de la interrupción de alarma.
- Uso de acciones programadas para escalar los servicios de Amazon ECS: aumente o reduzca el número de tareas que ejecuta el servicio en función de la fecha y la hora.

- Uso de patrones históricos para escalar los servicios de Amazon ECS con escalado predictivo: aumenta o disminuye el número de tareas que ejecuta el servicio en función del análisis de datos de carga históricos para detectar patrones diarios o semanales en los flujos de tráfico.

Consideraciones

Cuando utilice las políticas de escalado, tenga en cuenta lo siguiente:

- Amazon ECS envía métricas a CloudWatch a intervalos de 1 minuto. Las métricas no están disponibles hasta que los clústeres y servicios envían las métricas a CloudWatch, y no puede crear alarmas de CloudWatch para métricas que no existen.
- Las políticas de escalado admiten un período de recuperación. Este es la cantidad de tiempo, en segundos, que se debe esperar a que surta efecto una actividad de escalado anterior.
  - En el caso de las políticas de escalado horizontal, la intención es realizar el escalado horizontal de manera continua (pero no excesiva). Después de que Service Auto Scaling efectúa correctamente el escalado horizontal a través de una política de escalado por pasos, comienza a calcular el tiempo de recuperación. La política de escalado no volverá a aumentar la capacidad deseada a menos que se inicie el escalado horizontal o finalice el periodo de recuperación. Mientras el periodo de recuperación del escalado ascendente esté en vigor, la capacidad agregada por la actividad inicial de escalado ascendente se considerará parte de la capacidad deseada para la siguiente actividad de escalado ascendente.
  - En el caso de eventos de reducción horizontal, la intención es efectuar un reducción horizontal de forma conservadora a fin de proteger la disponibilidad de la aplicación, de modo que las actividades de reducción horizontal se bloqueen hasta que haya transcurrido el período de recuperación. No obstante, si otra alarma inicia una actividad de escalado horizontal durante el periodo de recuperación de la reducción horizontal, Service Auto Scaling escala horizontalmente el destino de inmediato. En este caso, el periodo de recuperación de reducción horizontal se detiene y no se completa.
- El programador de servicios respeta el recuento deseado en todo momento, pero si dispone de políticas y alarmas de escalado activas en un servicio, Service Auto Scaling podría cambiar el recuento deseado que se estableció manualmente.
- Si el recuento deseado de un servicio se establece por debajo de su valor de capacidad mínimo y una alarma desencadena una actividad de escalado horizontal, Service Auto Scaling aumenta el recuento deseado al valor de capacidad mínimo y, luego, continúa con el escalado horizontal, según corresponda, en función de la política de escalado asociada a la alarma. No obstante, una

actividad de reducción horizontal no ajusta el recuento deseado, porque ya se encuentra por debajo del valor de capacidad mínimo.

- Si el recuento deseado de un servicio se establece por encima de su valor de capacidad máximo y una alarma desencadena una actividad de reducción horizontal, Service Auto Scaling escala el recuento deseado al valor de capacidad máximo y, luego, continúa con la reducción horizontal, según corresponda, en función de la política de escalado asociada a la alarma. No obstante, una actividad de reducción de escala no ajusta el recuento deseado, porque ya se encuentra por encima del valor de capacidad máximo.
- Durante las actividades de escalado, el recuento de tareas en ejecución real de un servicio es el valor que Service Auto Scaling utiliza como punto de inicio, en contraposición con el recuento deseado. Esto es lo que se supone que es la capacidad de procesamiento. Esto impide un escalado excesivo (descontrolado) que podría no satisfacerse, por ejemplo, si no hubiera suficientes recursos de instancia de contenedor para colocar las tareas adicionales. Si la capacidad de la instancia de contenedor está disponible más tarde, la actividad de escalado pendiente podría tener éxito y, entonces, las actividades de escalado adicionales podrían continuar después del periodo de recuperación.
- Si desea que el recuento de tareas se escale a cero cuando no haya trabajo por hacer, establezca una capacidad mínima de 0. Con las políticas de escalado de seguimiento de destino, cuando la capacidad real es 0 y la métrica indica que hay demanda de carga de trabajo, Service Auto Scaling espera que se envíe un punto de datos antes del escalado horizontal. En este caso, se escala horizontalmente la cantidad mínima posible como punto de partida y, a continuación, reanuda el escalado en función del recuento real de tareas en ejecución.
- El escalado automático de aplicaciones desactiva los procesos de reducir horizontalmente mientras se llevan a cabo las implementaciones de Amazon ECS. Sin embargo, los procesos de escalado horizontal continúan produciéndose durante una implementación, a menos que se suspendan. Este comportamiento no se aplica a los servicios de Amazon ECS que utilizan el controlador de implementación externo. Para obtener más información, consulte Escalado automático de servicios e implementaciones.
- Dispone de varias opciones de escalado automático de aplicaciones para las tareas de Amazon ECS. El seguimiento de destinos es el modo más fácil de utilizar. Con él, todo lo que necesita hacer es establecer un valor objetivo para una métrica, como la utilización media de la CPU. Luego, el escalador automático administra de forma automática la cantidad de tareas necesarias para alcanzar ese valor. Con el escalado por pasos, puede reaccionar más rápidamente a los cambios en la demanda, ya que define los umbrales específicos para sus métricas de escalado y cuántas tareas agregar o eliminar cuando se superen los umbrales. Y, lo que es más importante,

puede reaccionar muy rápidamente ante los cambios en la demanda al minimizar el tiempo que una alarma supera el umbral.

Para más información sobre las prácticas recomendadas para el escalado automático de servicios, consulte Optimización del escalado automático de servicios de Amazon ECS.

Escalado automático de servicios e implementaciones

El escalado automático de aplicaciones desactiva los procesos de reducir horizontalmente mientras se llevan a cabo las implementaciones de Amazon ECS. Sin embargo, los procesos de escalado horizontal continúan produciéndose durante una implementación, a menos que se suspendan. Este comportamiento no se aplica a los servicios de Amazon ECS que utilizan el controlador de implementación externo. Si desea suspender los procesos de escalado horizontal mientras las implementaciones están en curso, siga estos pasos.

1. Ejecute el comando describe-scalable-targets, especificando el ID de recurso del servicio asociado al destino escalable en Application Auto Scaling (Ejemplo: `service/default/sample-webapp`). Registre el resultado. Lo necesitará cuando ejecute el próximo comando.
2. Ejecute el comando register-scalable-target, especificando el ID de recurso, el espacio de nombres y la dimensión escalable. Especifique `true` tanto para `DynamicScalingInSuspended` como para `DynamicScalingOutSuspended`.
3. Una vez completada la implementación, puede ejecutar el comando register-scalable-target para reanudar el escalado.

Para obtener más información, consulte Suspensión y reanudación del escalado para Application Auto Scaling.

Uso de una métrica objetivo para escalar los servicios de Amazon ECS

Las políticas de escalado de seguimiento de destino le permiten seleccionar una métrica y establecer un valor de destino. El escalado automático de servicios de Amazon ECS crea y administra las alarmas de CloudWatch que controlan la política de escalado y calcula el ajuste de escalado en función de la métrica y el valor objetivo. La política de escalado amplía o reduce las tareas de servicio en función de las necesidades para mantener la métrica en el valor objetivo especificado o en un valor próximo. Además de mantener la métrica próxima al valor objetivo, la política de escalado de seguimiento de destino también se ajusta a las fluctuaciones de la métrica producidas por un

patrón de carga fluctuante y minimiza las fluctuaciones rápidas en el número de tareas que se ejecutan en su servicio.

Las políticas de seguimiento de objetivo eliminan la necesidad de definir manualmente alarmas en CloudWatch y en los ajustes de escalamiento. Amazon ECS gestiona esto automáticamente en función del objetivo que establezca.

Tenga en cuenta lo siguiente al utilizar las políticas de seguimiento de destino:

- En las políticas de escalado de seguimiento de destino, se presupone que el escalado ascendente se realiza cuando la métrica está por encima del valor objetivo. No puede utilizar una política de escalado de seguimiento de destino si la métrica especificada está por debajo del valor objetivo.
- Las políticas de escalado de seguimiento de destino no realizan el escalado cuando la métrica especificada no tiene datos suficientes. No realiza el escalado porque la carencia de datos no se interpreta como una infrautilización de recursos.
- Es posible que haya diferencias entre el valor objetivo y los puntos de datos de la métrica real. Esto se debe a que Service Auto Scaling siempre actúa de forma conservadora y redondea hacia arriba o hacia abajo a la hora de determinar la cantidad de capacidad que debe agregar o quitar. Con esto se evita que se agrega capacidad insuficiente o se elimine demasiada capacidad.
- Para garantizar la disponibilidad de la aplicación, el servicio se escala en horizontal proporcionalmente a la métrica tan rápido como puede, pero se escala de forma descendente más gradualmente.
- El escalado automático de aplicaciones desactiva los procesos de reducir horizontalmente mientras se llevan a cabo las implementaciones de Amazon ECS. Sin embargo, los procesos de escalado horizontal continúan produciéndose durante una implementación, a menos que se suspendan. Este comportamiento no se aplica a los servicios de Amazon ECS que utilizan el controlador de implementación externo. Para obtener más información, consulte Escalado automático de servicios e implementaciones.
- Se pueden tener varias políticas de escalado de seguimiento de destino para un servicio de Amazon ECS, siempre que cada una de ellas utilice una métrica diferente. El objetivo de Service Auto Scaling siempre es dar prioridad a la disponibilidad, por lo que su comportamiento varía en función de si las políticas de seguimiento de destino están listas para el escalado o la reducción horizontal. Realizará un escalado horizontal del servicio si cualquiera de las políticas de seguimiento de destino está lista para el escalado horizontal, pero solo realizará la reducción horizontal si todas las políticas de seguimiento de destino (que tienen la parte de reducción horizontal activada) están listas para la reducción horizontal.

- No modifique ni elimine las alarmas de CloudWatch que administra Service Auto Scaling para una política de escalado de seguimiento de destino. Service Auto Scaling elimina automáticamente las alarmas cuando se elimina la política de escalado.
- La métrica `ALBRequestCountPerTarget` para las políticas de escalado de seguimiento de destino no es compatible con el tipo de implementación azul/verde.

Para obtener más información acerca de políticas de escalado de seguimiento de destino, consulte Políticas de escalado de seguimiento de destino en la Guía del usuario de Application Auto Scaling.

Creación de una política de escalado de seguimiento de destino para el escalado automático de servicios de Amazon ECS

Cree una política de escalado de seguimiento de destino para que Amazon ECS aumente o disminuya automáticamente el número de tareas deseado en el servicio. El seguimiento de destino funciona a partir de un valor de métricas de destino.

Consola

1. Además de los permisos estándar de IAM para crear y actualizar servicios, necesita permisos adicionales. Para obtener más información, consulte Permisos de IAM necesarios para el escalado automático del servicio de Amazon ECS.
2. Determine las métricas que quiere utilizar para la política. Están disponibles las siguientes métricas:
   - ECSServiceAverageCPUUtilization: uso medio de la CPU que debe utilizar el servicio.
   - ECSServiceAverageMemoryUtilization: uso medio de la memoria que debe utilizar el servicio.
   - ALBRequestCountPerTarget: número medio de solicitudes por minuto que debe recibir de manera ideal.
3. Abra la consola en https://console.aws.amazon.com/ecs/v2.
4. En la página Clusters (Clústeres), elija el clúster.
5. En la página de detalles del clúster, en la sección Servicios, elija el servicio.

   Se abrirá la página de detalles del servicio.
6. Elija Establecer el número de tareas.
7. En Recuento de tareas de servicio de Amazon ECS, elija Usar escalado automático.

   Se abrirá la sección Recuento de tareas.

a. En Cantidad mínima de tareas, ingrese el límite mínimo del número de tareas que se va a utilizar para el escalado automático del servicio. El recuento deseado no será inferior a este recuento.
b. En Máximo, ingrese el límite máximo del número de tareas que se va a utilizar para el escalado automático del servicio. El recuento deseado no será superior a este recuento.
c. Seleccione Save.

   Se abrirá la página de políticas.

8. Elija Crear política de escalado.

   Se abrirá la página Crear política.

9. Para Scaling policy type (Tipo de política de escalado), elija Target tracking (Seguimiento de destino).
10. En Policy name (Nombre de la política), ingrese el nombre de la política.
11. En Tipo de métricas, elija las métricas de la lista de opciones.
12. En Utilización objetivo, introduzca el valor objetivo del porcentaje de tareas que debe mantener Amazon ECS. El escalado automático del servicio escala horizontalmente la capacidad hasta que la utilización media se encuentre en la utilización objetivo, o hasta que alcance el número máximo de tareas que haya especificado.
13. En Configuración adicional, haga lo siguiente:

    a. En Periodo de recuperación de reducción horizontal, escriba la cantidad de tiempo, en segundos, tras completarse una actividad de reducción horizontal antes de que pueda comenzar otra actividad de reducción horizontal.
    b. En Periodo de recuperación de escalado horizontal, ingrese la cantidad de tiempo, en segundos, que debe esperarse para que surta efecto una actividad de reducción horizontal anterior.
    c. Seleccione Deshabilitar la reducción horizontal para crear solo una política de escalado horizontal.

14. Elija Crear política de escalado.

AWS CLI

1. Registre su servicio de Amazon ECS como un destino escalable mediante el comando register-scalable-target.

2. Cree una política de escalado mediante el comando put-scaling-policy.

Uso de incrementos predefinidos en función de las alarmas de CloudWatch para escalar los servicios de Amazon ECS

Con las políticas de escalado por pasos, crea y administra las alarmas de CloudWatch que invocan el proceso de escalado. Cuando se infringe una alarma, Amazon ECS inicia la política de escalado asociada a esa alarma. La política de escalado por pasos escala las tareas mediante un conjunto de ajustes, conocidos como ajustes escalonados. La magnitud del ajuste varía en función del tamaño de la interrupción de la alarma.

- Si la infracción supera el primer umbral, Amazon ECS aplicará el ajuste del primer paso.
- Si la infracción supera el segundo umbral, Amazon ECS aplicará el ajuste del segundo paso, y así sucesivamente.

Le recomendamos que utilice políticas de escalado de seguimiento de objetivos 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 reducir o escalar horizontalmente de forma proporcional el número de tareas que utilizan el seguimiento de destino. Esto ayuda a garantizar que Amazon ECS siga de cerca la curva de demanda de sus aplicaciones.

Creación de una política de escalado por pasos para el escalado automático de servicios de Amazon ECS

Cree una política de escalado por pasos para que Amazon ECS aumente o disminuya automáticamente el número deseado de tareas en el servicio. El escalado por pasos se ejecuta según una serie de ajustes de escalado, denominados ajustes por pasos, que pueden variar en función del tamaño de la interrupción de alarma.

Consola

1. Además de los permisos estándar de IAM para crear y actualizar servicios, necesita permisos adicionales. Para obtener más información, consulte Permisos de IAM necesarios para el escalado automático del servicio de Amazon ECS.
2. Determine las métricas que quiere utilizar para la política. Están disponibles las siguientes métricas:

- ECSServiceAverageCPUUtilization: uso medio de la CPU que debe utilizar el servicio.
- ECSServiceAverageMemoryUtilization: uso medio de la memoria que debe utilizar el servicio.
- ALBRequestCountPerTarget: número medio de solicitudes por minuto que debe recibir de manera ideal.

3. Cree las alarmas de CloudWatch para las métricas. Para obtener más información, consulte Create a CloudWatch alarm based on a static threshold (Creación de una alarma de CloudWatch basada en un umbral estático) en la Guía del usuario de Amazon CloudWatch.
4. Abra la consola en https://console.aws.amazon.com/ecs/v2.
5. En la página Clusters (Clústeres), elija el clúster.
6. En la página de detalles del clúster, en la sección Servicios, elija el servicio.

   Se abrirá la página de detalles del servicio.
7. Elija Establecer el número de tareas.
8. En Recuento de tareas de servicio de Amazon ECS, elija Usar escalado automático.

   Se abrirá la sección Recuento de tareas.

   a. En Cantidad mínima de tareas, ingrese el límite mínimo del número de tareas que se va a utilizar para el escalado automático del servicio. El recuento deseado no será inferior a este recuento.
   b. En Máximo, ingrese el límite máximo del número de tareas que se va a utilizar para el escalado automático del servicio. El recuento deseado no será superior a este recuento.
   c. Seleccione Save.

      Se abrirá la página de políticas.
9. Elija Crear política de escalado.

   Se abrirá la página Crear política.
10. Para Tipo de política de escalado, elija Escalado de pasos.
11. Configure las propiedades de escalado horizontal. En Pasos para agregar tareas, haga lo siguiente:

    a. En Policy name (Nombre de la política), ingrese el nombre de la política.
    b. En Nombre de la alarma de CloudWatch, elija la alarma de CloudWatch.

c. En Tipo de agregación de métricas, elija cómo comparar la métrica seleccionada con el umbral definido.

d. En Tipos de ajuste, elija si el ajuste se basa en un cambio en el número de tareas o en un cambio en el porcentaje de tareas.

e. En Acciones que ejecutar, introduzca los valores de la acción que se va a realizar.

   Seleccione Agregar paso para agregar otras acciones.

12. Configure las propiedades de la reducción horizontal. En Pasos para quitar tareas, haga lo siguiente:

    a. En Policy name (Nombre de la política), ingrese el nombre de la política.

    b. En Nombre de la alarma de CloudWatch, elija la alarma de CloudWatch.

    c. En Tipo de agregación de métricas, elija cómo comparar la métrica seleccionada con el umbral definido.

    d. En Tipos de ajuste, elija si el ajuste se basa en un cambio en el número de tareas o en un cambio en el porcentaje de tareas.

    e. En Acciones que ejecutar, introduzca los valores de la acción que se va a realizar.

       Seleccione Agregar paso para agregar otras acciones.

13. En Periodo de recuperación, ingrese la cantidad de tiempo, en segundos, que debe esperarse para que surta efecto una actividad de reducción horizontal anterior. En el caso de una política de ampliación, este es el momento en el que, después de una actividad de escalado horizontal, la política de escalado bloquea las actividades de reducción horizontal y limita el número de tareas que se pueden escalar horizontalmente a la vez. En el caso de una política de reducción horizontal, este es el tiempo que debe transcurrir tras completarse una actividad de reducción horizontal antes de que pueda comenzar otra actividad de reducción horizontal.

14. Elija Crear política de escalado.

AWS CLI

1. Registre su servicio de Amazon ECS como un destino escalable mediante el comando register-scalable-target.

2. Cree una política de escalado mediante el comando put-scaling-policy.

Uso de acciones programadas para escalar los servicios de Amazon ECS

Con el escalado programado, puede configurar el escalado automático para su aplicación en función de cambios de carga predecibles mediante la creación de acciones programadas que aumenten o disminuyan el número de tareas en momentos específicos. Esto le permite escalar su aplicación de forma proactiva para adaptarla a los cambios de carga predecibles.

Estas acciones de escalado programadas le permiten optimizar los costes y el rendimiento. Su aplicación tiene un número de tareas suficiente para gestionar los picos de tráfico a mitad de semana, pero no aprovisiona en exceso el número de tareas en otros momentos.

Puede combinar el escalado programado y las políticas de escalado para obtener los beneficios de enfoques tanto proactivos como reactivos al escalado. Después de ejecutar una acción de escalado programado, la política de escalado puede seguir tomando decisiones sobre si desea ampliar el número de tareas. Esto ayuda a garantizar que tiene un número de tareas suficiente para gestionar la carga de su aplicación. Mientras la aplicación se escala para adaptarse a la demanda, la capacidad actual debe estar dentro del número de tareas mínimo y máximo que establece la acción programada.

Puede configurar el escalado de la programación mediante la AWS CLI. Para obtener más información sobre el escalado programado, consulte Scheduled Scaling en la Guía del usuario de Application Auto Scaling.

Creación de una acción programada para el escalado automático de servicios de Amazon ECS

Creación de una acción programada para que Amazon ECS aumento o disminuya el número de tareas que ejecuta el servicio en función de la fecha y la hora.

Consola

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), elija el clúster.
3. En la página de detalles del clúster, en la sección Servicios, elija el servicio.

   Se abrirá la página de detalles del servicio.
4. Elija Escalado automático de servicio.

   Aparece la página de escalado automático del servicio.
5. Si no ha configurado el escalado automático del servicio, elija Establecer la cantidad de tareas.

Aparece la sección Recuento de tareas del servicio de Amazon ECS.

En Recuento de tareas del servicio de Amazon ECS, seleccione Utilizar el escalado automático del servicio para ajustar el recuento de tareas deseado del servicio.

Se abrirá la sección Recuento de tareas.

a. En Cantidad mínima de tareas, ingrese el límite mínimo del número de tareas que se va a utilizar para el escalado automático del servicio. El recuento deseado no será inferior a este recuento.
b. En Máximo, ingrese el límite máximo del número de tareas que se va a utilizar para el escalado automático del servicio. El recuento deseado no será superior a este recuento.
c. Elija Elegir Guardar.

   Se abrirá la página de políticas.

6. Elija Acciones programadas y, a continuación, elija Crear.

   Se abrirá la página Crear acción programada.

7. En Nombre de acción, escriba un nombre único.
8. Para Zona horaria, elija una zona horaria.

   Todas las zonas horarias enumeradas provienen de la base de datos de zona horaria de IANA. Para obtener más información, consulte List of tz database time zones.

9. En Hora de inicio, introduzca la fecha y la hora en que comienza la acción.

   Si elige una programación recurrente, la hora de inicio define cuándo se ejecuta la primera acción programada de la serie recurrente.

10. En Recurrence (Recurrencia), elija una de las opciones disponibles.

    - Para escalar según una programación recurrente, elija la frecuencia con la que Amazon ECS ejecuta la acción programada.
      - Si elige una opción que comienza por Frecuencia, la expresión cron se crea automáticamente.
      - Si elige Cron, escriba una expresión cron que especifique cuándo se debe realizar la acción.
    - Para escalar una sola vez, elija Una vez.

11. En Ajustes de tareas, haga lo siguiente:
    - En Mínimo, introduzca el número mínimo de tareas que debe ejecutar el servicio.
    - En Máximo, introduzca el número máximo de tareas que debe ejecutar el servicio.
12. Elija Crear acción programada.

CLI

Use la AWS CLI como se indica a continuación para configurar políticas de escalado programado para el servicio. Reemplace cada *marcador de posición de entrada del usuario* con información propia.

Ejemplo: Escalar solo una vez

Utilice el siguiente comando put-scheduled-action con la `--start-time "YYYY-MM-DDThh:mm:ssZ"` y una o ambas opciones `--MinCapacity` y `--MaxCapacity`.

```
aws application-autoscaling put-scheduled-action --service-namespace ecs \
  --resource-id service/my-cluster/my-service \
  --scheduled-action-name my-one-time-schedule \
  --start-time 2021-01-30T12:00:00 \
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

Ejemplo: Para programar el escalado de forma periódica

Utilice el siguiente comando put-scheduled-action. Sustituya las *entradas del usuario* por sus valores.

```
aws application-autoscaling put-scheduled-action --service-namespace ecs \
  --resource-id service/my-cluster/my-service \
  --scheduled-action-name my-recurring-action \
  --schedule "rate(5 hours)" \
  --start-time 2021-01-30T12:00:00 \
  --end-time 2021-01-31T22:00:00 \
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

El programa de recurrencia especificado se ejecuta según 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 List of tz database time zones.

Uso de patrones históricos para escalar los servicios de Amazon ECS con escalado predictivo

El escalado predictivo analiza los datos de cargas anteriores de los flujos de tráfico para analizar los patrones diarios o semanales. Luego, utiliza este análisis para anticipar las necesidades futuras y aumentar proactivamente las tareas de su servicio según sea necesario.

El escalado automático predictivo es más útil en las siguientes situaciones.

- Tráfico cíclico: 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 el procesamiento por lotes, pruebas o análisis periódicos de datos.
- Aplicaciones que tardan mucho tiempo en inicializarse: esto puede repercutir en el rendimiento de las aplicaciones durante eventos de escalado horizontal, lo que provoca una latencia notable.

En general, si sus aplicaciones tardan mucho tiempo en inicializarse y el tráfico aumenta en un patrón regular, debe considerar el uso del escalado predictivo. Esto ayuda a escalar más rápido, ya que aumenta de forma proactiva el número de tareas para las cargas previstas en lugar de utilizar políticas de escalado dinámicas, como de seguimiento de destino o de escalado por pasos únicamente. Al ayudar a evitar la posibilidad de aprovisionar en exceso el número de tareas, con el escalado predictivo también puede ahorrar dinero.

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 escalar horizontalmente tareas 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 tareas mediante el escalado programado.

El escalado predictivo es una capacidad de nivel de servicio que escala la tarea del servicio independientemente del escalado de la capacidad de computación subyacente (por ejemplo, EC2 o Fargate). Para Fargate, AWS administra y escala de forma automática la capacidad subyacente en

función de los requisitos de la tarea. Para la capacidad de EC2, puede utilizar los proveedores de capacidad de grupo de escalado automático para escalar automáticamente las instancias de EC2 subyacentes en función de los requisitos de escalado de las tareas.

Contenido

- Cómo funciona el escalado predictivo en Amazon ECS
- Creación de una política de escalado predictivo para el escalado automático de servicios de Amazon ECS
- Evaluación de las políticas de escalado predictivo para Amazon ECS
- Uso de acciones programadas para anular los valores de previsión de Amazon ECS
- Políticas avanzadas de escalado predictivo mediante métricas personalizadas para Amazon ECS

Cómo funciona el escalado predictivo en Amazon ECS

Aquí puede obtener información sobre las consideraciones relacionadas con el uso del escalado predictivo, cómo funciona y cuáles son los límites.

Consideraciones sobre el uso del escalado predictivo

- Desea asegurarse de que el escalado predictivo sea adecuado para la carga de trabajo. Para comprobarlo, configure las políticas de escalado en el modo de solo previsión y consulte las recomendaciones de la consola. Debe evaluar la previsión y las recomendaciones antes de empezar a utilizar el escalado predictivo.
- Antes de que el escalado predictivo pueda iniciar el pronóstico, este necesita al menos 24 horas de datos históricos. Cuantos más datos históricos estén disponibles, más eficaz será la previsión. Lo ideal es dos semanas. También tendrá que esperar 24 horas para que el escalado predictivo pueda generar nuevas previsiones cuando elimine un servicio de Amazon ECS y cree uno nuevo. Una forma de acelerar este proceso es usar métricas personalizadas para agregar métricas de servicios de Amazon ECS nuevos y antiguos.
- 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 escalado dinámico con el escalado predictivo ayuda a seguir de cerca la demanda de la aplicación, de modo que puede reducir horizontalmente durante los periodos de menor demanda y escalar horizontalmente durante aumentos inesperados de tráfico. Cuando hay activas varias políticas de escalado, cada política determina el número de tareas deseado de forma independiente, y el número de tareas deseado se establece en el máximo de las políticas.

- Puede utilizar el escalado predictivo junto con sus políticas de escalado dinámico, como el seguimiento de destino o el escalado gradual, para que las aplicaciones se escalen en función de patrones históricos y en tiempo real. Por sí solo, el escalado predictivo no reduce horizontalmente las tareas.
- Si utiliza un rol personalizado al llamar a la API `register-scalable-target`, es posible que aparezca un error que indique que la política de escalado predictivo solo funciona con el SLR habilitado. En este caso, debe volver a llamar a `register-scalable-target`, pero sin role-arn. Utilice el SLR al registrar el objetivo escalable y llame a la API `put-scaling-policy`.

Funcionamiento del escalado predictivo

Utiliza el escalado predictivo al crear una política de escalado predictivo que especifique la métrica de CloudWatch que se va a supervisar y analizar. El escalado predictivo debe tener datos de, por lo menos, las últimas 24 horas, para comenzar a pronosticar los valores futuros.

Tras crear la política, el escalado predictivo comienza a analizar los datos de las métricas de los últimos 14 días para identificar patrones. Este análisis se utiliza para generar previsiones por hora de requisitos de las próximas 48 horas. Los datos más recientes de CloudWatch se utilizan para actualizar la previsión cada 6 horas. 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. Genera pronósticos en este modo, pero no escala el servicio de Amazon ECS en función de esos pronósticos. Esto significa que puede evaluar la precisión e idoneidad del pronóstico. Ve los datos del pronóstico mediante la operación de la API `GetPredictiveScalingForecast` o la AWS Management Console.

Cuando decida empezar a utilizar el escalado predictivo, cambie la política de escalado al modo de pronóstico y escalado. En este modo, ocurre lo siguiente.

De forma predeterminada, su servicio de Amazon ECS se escala al principio de cada hora en función del pronóstico de esa hora. Puede optar por empezar antes mediante el uso de la propiedad `SchedulingBufferTime` de la operación de la API `PutScalingPolicy`. Esto permite que las nuevas tareas se inicien antes de la demanda pronosticada y les da tiempo para arrancar y prepararse para gestionar el tráfico.

Límite máximo de tareas

Cuando registra los servicios de Amazon ECS para el escalado, define un número máximo de tareas que se pueden lanzar por servicio. De manera predeterminada, cuando se establecen políticas de escalado, estas no pueden aumentar el número de tareas por encima del límite máximo.

Como alternativa, puede permitir que el número máximo de tareas del servicio se incremente automáticamente si la previsión se acerca al número máximo de tareas del servicio de Amazon ECS o lo supera.

> **Warning**
>
> Tenga cuidado al permitir que el número máximo de tareas aumente automáticamente. Esto puede provocar que se lancen más tareas de las previstas si no se supervisa ni gestiona el aumento del número máximo de tareas. El aumento del número máximo de tareas se convierte en el nuevo número máximo normal de tareas del servicio de Amazon ECS hasta que lo actualice manualmente. El número máximo de tareas no disminuye automáticamente hasta el máximo original.

Regiones de 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 (Yakarta)
- Asia-Pacífico (Mumbai)
- Asia-Pacífico (Osaka)
- Asia-Pacífico (Seúl)
- Asia-Pacífico (Singapur)
- Asia-Pacífico (Sídney)
- Asia-Pacífico (Tokio)
- Canadá (centro)

- China (Pekín)
- China (Ningxia)
- Europa (Fráncfort)
- Europa (Irlanda)
- Europa (Londres)
- Europa (Milán)
- Europa (París)
- Europa (Estocolmo)
- Medio Oriente (Baréin)
- América del Sur (São Paulo)
- AWS GovCloud (Este de EE. UU.)
- AWS GovCloud (Oeste de EE. UU.)

Creación de una política de escalado predictivo para el escalado automático de servicios de Amazon ECS

Cree una política de escalado predictiva de modo que Amazon ECS aumente o disminuya la cantidad de tareas que el servicio ejecuta en función de datos históricos.

> **Note**
>
> Un servicio nuevo debe proporcionar datos de, por lo menos, las últimas 24 horas para poder generar un pronóstico.

Consola

1. Además de los permisos estándar de IAM para crear y actualizar servicios, necesita permisos adicionales. Para obtener más información, consulte Permisos de IAM necesarios para el escalado automático del servicio de Amazon ECS.
2. Determine las métricas que quiere utilizar para la política. Están disponibles las siguientes métricas:
   - ECSServiceAverageCPUUtilization: uso medio de la CPU que debe utilizar el servicio.
   - ECSServiceAverageMemoryUtilization: uso medio de la memoria que debe utilizar el servicio.

- ALBRequestCountPerTarget: número medio de solicitudes por minuto que debe recibir de manera ideal.

También puede utilizar una métrica personalizada. Debe definir los siguientes valores:

- Carga: una métrica de carga que representa con precisión la carga total de la aplicación y que es el aspecto de la aplicación que más le interesa escalar.

- Métrica de escalado: el mejor indicador del grado de utilización ideal para su aplicación.

3. Abra la consola en https://console.aws.amazon.com/ecs/v2.
4. En la página Clusters (Clústeres), elija el clúster.
5. En la página de detalles del clúster, en la sección Servicios, elija el servicio.

   Se abrirá la página de detalles del servicio.
6. Elija Escalado automático de servicio y, a continuación, elija Establecer la cantidad de tareas.
7. En Recuento de tareas de servicio de Amazon ECS, elija Usar escalado automático.

   Se abrirá la sección Recuento de tareas.

   a. En Cantidad mínima de tareas, ingrese el límite mínimo del número de tareas que se va a utilizar para el escalado automático del servicio. El recuento deseado no será inferior a este recuento.
   b. En Máximo, ingrese el límite máximo del número de tareas que se va a utilizar para el escalado automático del servicio. El recuento deseado no será superior a este recuento.
   c. Seleccione Save.

      Se abrirá la página de políticas.
8. Elija Crear política de escalado.

   Se abrirá la página Crear política.
9. Para Tipo de política de escalado, elija Escalado predictivo.
10. En Policy name (Nombre de la política), ingrese el nombre de la política.
11. En Par de métricas, elija las métricas de la lista de opciones.

    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). Recuento de solicitudes del equilibrador de carga de aplicación por destino solo se admite si ha adjuntado un grupo de destino del equilibrador de carga de aplicación para el servicio.

Si eligió Par de métricas personalizadas, elija métricas individuales en las listas de Métrica de carga y Métrica de escalado.

12. En Utilización objetivo, introduzca el valor objetivo del porcentaje de tareas que debe mantener Amazon ECS. El escalado automático del servicio escala horizontalmente la capacidad hasta que la utilización media se encuentre en la utilización objetivo, o hasta que alcance el número máximo de tareas que haya especificado.
13. Elija Crear política de escalado.

AWS CLI

Use la AWS CLI como se indica a continuación para configurar políticas de escalado predictivo para el servicio de Amazon ECS. Reemplace cada *marcador de posición de entrada del usuario* con información propia.

Para obtener más información acerca de las métricas de CloudWatch que puede especificar, consulte PredictivEscalingMetricSpecification en la Referencia de la API de Amazon EC2 Auto Scaling.

Ejemplo 1: política de escalado predictivo con memoria predefinida.

A continuación, se muestra un ejemplo de política con una configuración de memoria predefinida.

```
cat policy.json
{
    "MetricSpecifications": [
        {
            "TargetValue": 40,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ECSServiceMemoryUtilization"
            }
        }
    ],
    "SchedulingBufferTime": 3600,
    "MaxCapacityBreachBehavior": "HonorMaxCapacity",
    "Mode": "ForecastOnly"
}
```

En el ejemplo siguiente se muestra cómo crear la política mediante la ejecución del comando put-scaling-policy con el archivo de configuración especificado.

```
aws application-autoscaling put-scaling-policy \
--service-namespace ecs \
--region us-east-1 \
--policy-name predictive-scaling-policy-example \
--resource-id service/MyCluster/test \
--policy-type PredictiveScaling \
--scalable-dimension ecs:service:DesiredCount \
--predictive-scaling-policy-configuration file://policy.json
```

Si se ejecuta correctamente, este comando devolverá el ARN de la política.

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
east-1:012345678912:scalingPolicy:d1d72dfe-5fd3-464f-83cf-824f16cb88b7:resource/ecs/
service/MyCluster/test:policyName/predictive-scaling-policy-example",
    "Alarms": []
}
```

Ejemplo 2: política de escalado predictivo con CPU predefinida.

A continuación, se muestra un ejemplo de política con una configuración de CPU predefinida.

```
cat policy.json
{
    "MetricSpecifications": [
        {
            "TargetValue": 0.00000004,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ECSServiceCPUUtilization"
            }
        }
    ],
    "SchedulingBufferTime": 3600,
    "MaxCapacityBreachBehavior": "HonorMaxCapacity",
    "Mode": "ForecastOnly"
}
```

En el ejemplo siguiente se muestra cómo crear la política mediante la ejecución del comando put-scaling-policy con el archivo de configuración especificado.

```
aws aas put-scaling-policy \
--service-namespace ecs \
--region us-east-1 \
--policy-name predictive-scaling-policy-example \
--resource-id service/MyCluster/test \
--policy-type PredictiveScaling \
--scalable-dimension ecs:service:DesiredCount \
--predictive-scaling-policy-configuration file://policy.json
```

Si se ejecuta correctamente, este comando devolverá el ARN de la política.

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
east-1:012345678912:scalingPolicy:d1d72dfe-5fd3-464f-83cf-824f16cb88b7:resource/ecs/
service/MyCluster/test:policyName/predictive-scaling-policy-example",
    "Alarms": []
}
```

Evaluación de las políticas de escalado predictivo para Amazon ECS

Antes de utilizar una política de escalado predictivo para escalar automáticamente los servicios, consulte las recomendaciones y otros datos de la política en la consola de Amazon ECS. 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 servicio es nuevo, espere 24 horas para crear el primer pronóstico.

Cuando AWS crea un pronóstico, utiliza datos históricos. Si el servicio aún no cuenta con muchos datos históricos recientes, el escalado predictivo podría rellenar temporalmente el pronóstico 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.

Visualización de las recomendaciones de escalado predictivo

Para poder llevar a cabo un análisis eficaz, el escalado automático del servicio 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 ECS 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 ECS

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), elija el clúster.
3. En la página de detalles del clúster, en la sección Servicios, elija el servicio.

   Se abrirá la página de detalles del servicio.
4. Elija Escalado automático de servicio.
5. Elija la política de escalado predictivo y, a continuación, elija Acciones, Escalado predictivo y Ver recomendación.

   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 tareas 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 tareas, en comparación con no utilizar la política. Al aprovisionar demasiado, los servicios 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.
6. (Opcional) Para seleccionar el periodo deseado para los resultados de la recomendación, elija el valor que prefiera en el menú desplegable Periodo de evaluación: 2 días, 1 semana o 2 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, puede revisar el pronóstico 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 escale el número de tareas real.

Revisión de los gráficos de supervisión del escalado predictivo en la consola de Amazon ECS

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), elija el clúster.
3. En la página de detalles del clúster, en la sección Servicios, elija el servicio.

   Se abrirá la página de detalles del servicio.
4. Elija Escalado automático de servicio.
5. Elija la política de escalado predictivo y, a continuación, elija Acciones, Escalamiento predictivo y Ver gráfico.
6. 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 tareas pronosticado por la política. También se incluye el número real de tareas lanzadas. La línea vertical separa los valores históricos de las previsiones futuras. Estos gráficos estarán disponibles poco después de que se cree la política.
7. (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.

Comparación de datos en el gráfico Carga

Cada línea horizontal representa un conjunto diferente de puntos de datos de los que se ha informado en intervalos de una hora:

1. 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.

Comparación de los datos en el gráfico Capacidad

Cada línea horizontal representa un conjunto diferente de puntos de datos de los que se ha informado en intervalos de una hora:

1. En Número de tareas real observado se muestra la capacidad real del servicio de Amazon ECS en el pasado, lo que depende de las demás políticas de escalado y del tamaño mínimo del grupo en vigor durante el periodo 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. En Número de tareas necesario inferido se muestra el número ideal de tareas de su servicio para mantener la métrica de escalado en el valor de destino que haya elegido.
4. En Cantidad mínima de tareas se muestra el número mínimo de tareas del servicio.
5. En Capacidad máxima se muestra el número máximo de tareas del servicio.

Para calcular la capacidad necesaria inferida, primero suponemos que cada tarea se utiliza por igual en un valor objetivo específico. En la práctica, el número de tareas no se utiliza por igual. Sin embargo, si suponemos que la utilización se distribuye de manera uniforme entre las tareas, podemos hacer una estimación probabilística de la cantidad de capacidad que se necesita. A continuación, se calcula el requisito del número de tareas 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 el número de tareas, la métrica de escalado disminuye al mismo ritmo. Por ejemplo, si el número de tareas se duplica, la métrica de escalado debe reducirse a la mitad.

La fórmula para la capacidad necesaria inferida:

```
sum of (actualServiceUnits*scalingMetricValue)/(targetUtilization)
```

Por ejemplo, tomamos los valores de `actualServiceUnits` (`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 objetivo 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 ECS calcula el número recomendado de tareas para cada política de escalado no programado. A continuación, se escala en función de la política que proporciona la mayor cantidad de tareas.
> - Para permitir que se reduzca horizontalmente cuando la carga disminuya, el servicio 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 tareas que no se encuentre dentro del rango de capacidad mínima y máxima se reduzca o escale horizontalmente.

Supervisión de métricas de escalado predictivo para Amazon ECS con CloudWatch

Puede utilizar Amazon CloudWatch para supervisar los datos para el escalado predictivo. Una política de escalado predictivo recopila datos que se utilizan para pronosticar la carga futura. Los datos recopilados se almacenan automáticamente en CloudWatch a intervalos regulares y se pueden utilizar para visualizar el rendimiento de la política a lo largo del tiempo. También puede crear alarmas de CloudWatch para que le notifiquen cuando los indicadores de rendimiento cambien más allá de los límites que defina.

Visualización de los datos de las previsiones

Los datos de pronósticos de carga para una política de escalado predictivo se pueden ver en CloudWatch y pueden ser útiles al visualizar pronósticos con respecto a otras métricas de CloudWatch en un solo gráfico. También puede consultar un intervalo de tiempo mayor para ver 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 consultar los datos de las previsiones históricas mediante la consola de CloudWatch

1. Abra la consola de CloudWatch 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. Seleccione el espacio de nombres de la métrica de Escalado automático de aplicaciones.
4. Elija Pronósticos de carga de escalado predictivo.
5. En el campo de búsqueda, ingrese el nombre de la política de escalado predictivo o el nombre del grupo del servicio de Amazon ECS 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, consulte Graphing a metric (Representación gráfica de métricas) en Amazon CloudWatch User Guide (Guía del usuario de Amazon CloudWatch).
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. Aunque puede elegir cualquier estadística en cada métrica, tenga en cuenta que no todas las estadísticas son útiles para las métricas PredictiveScalingLoadForecast. Por ejemplo, las estadísticas Average (Media), Minimum (Mínimo) y Maximum (Máximo) son útiles para el uso de la CPU, pero no así la estadística Sum (Suma).
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.
9. (Opcional) Para agregar este gráfico a un panel CloudWatch, elija Acciones y después Agregar al panel.

Creación de métricas de precisión mediante la matemática métrica

Las matemáticas en las métricas le permiten consultar varias métricas de CloudWatch y usar expresiones matemáticas para crear nuevas series temporales basadas en estas métricas. Puede visualizar las series temporales resultantes en la consola de CloudWatch y agregarlas a los paneles. Para obtener más información, consulte Using metric math (Uso de cálculo de métricas) en Amazon CloudWatch User Guide (Guía del usuario de Amazon CloudWatch).

Con la matemática métrica, puede representar gráficamente los datos que genera el escalado automático del servicio para el escalado predictivo de distinas formas. 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.

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 métrica de CloudWatch que contiene los valores de carga reales (si suponemos que el uso de CPU es la métrica de carga que se especificó originalmente para la política denominada `my-predictive-scaling-policy`). `m2` es la métrica de CloudWatch 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 la métrica de precisión, seleccione la pestaña Metrics (Métricas) en la consola de CloudWatch. Puede hacer una representación gráfica de los datos desde allí. Para obtener más información, consulte Adding a math expression to a CloudWatch graph (Adición de una expresión matemática a un gráfico de CloudWatch) en Amazon CloudWatch User Guide (Guía del usuario de Amazon CloudWatch).

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 notificaciones, consulte Creación de una alarma de CloudWatch basada en una expresión matemática de métrica y Notificaciones de cambios de alarma para usuarios en la Guía del usuario de Amazon CloudWatch.

También puede utilizar GetMetricData y PutMetricAlarm para realizar los cálculos mediante matemáticas métricas y crear alarmas basadas en los resultados.

Uso de acciones programadas para anular los valores de previsión de Amazon ECS

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 previsiones podrían subestimar las tareas necesarias 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 cantidad de tareas superior a la pronosticada. En el tiempo de ejecución, Amazon ECS actualiza el número mínimo de tareas del servicio. Dado que el escalado predictivo optimiza en función de la cantidad de tareas, se cumple una acción programada con una cantidad mínima de tareas superior a los valores pronosticados. De este modo se evita que la cantidad de tareas sea inferior a la prevista. Para dejar de anular la previsión, utilice una segunda acción programada para devolver la cantidad mínima de tareas 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 cantidad de tareas sin que interfiera una política de escalado predictivo, utilice en su lugar el modo de solo previsión. Mientras esté en el modo de solo previsión, el escalado predictivo seguirá generando previsiones, pero no aumentará automáticamente la cantidad de tareas. De esta manera, puede supervisar la utilización de los recursos y reducir manualmente el número de tareas según sea necesario.

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 el pronóstico mediante el comando get-predictive-scaling-forecast de la AWS CLI, proporcione los siguientes parámetros en el comando:

   - Indique el nombre del clúster en el parámetro `resource-id`.
   - 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 application-autoscaling get-predictive-scaling-forecast \
    --service-namespace ecs \
    --resource-id service/MyCluster/test \
    --policy-name cpu40-predictive-scaling-policy \
    --scalable-dimension ecs:service:DesiredCount \
    --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.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), elija el clúster.

3. En la página de detalles del clúster, en la sección Servicios, elija el servicio.

   Se abrirá la página de detalles del servicio.

4. Elija Escalado automático de servicio.

   Se abrirá la página de políticas.

5. Elija Acciones programadas y, a continuación, elija Crear.

   Se abrirá la página Crear acción de programación.

6. En Nombre de acción, escriba un nombre único.
7. Para Zona horaria, elija una zona horaria.

   Todas las zonas horarias enumeradas provienen de la base de datos de zona horaria de IANA. Para obtener más información, consulte List of tz database time zones.

8. En Hora de inicio, introduzca la fecha y la hora en que comienza la acción.
9. En Recurrencia, elija Una vez.
10. En Ajustes de tareas, en Mínimo, introduzca un valor inferior o igual a la cantidad máxima de tareas.
11. Elija Crear acción programada.

    Se abrirá la página de políticas.

12. Configure una segunda acción programada para devolver la cantidad mínima de tareas a la configuración original al final del evento. El escalado predictivo puede escalar la cantidad de tareas únicamente cuando el valor establecido para Mínimo es menor que los valores de la previsión.

Para crear dos acciones programadas para eventos únicos (AWS CLI)

Para utilizar la 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 una expresión cron consta de cinco campos separados por espacios: [minuto] [hora] [día_del_mes] [mes_del_año] [día_de_la_semana]. 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 sobre cómo administrar las acciones programadas, consulte Uso de acciones programadas para escalar los servicios de Amazon ECS.

Políticas avanzadas de escalado predictivo mediante métricas personalizadas para Amazon ECS

Puede utilizar métricas predefinidas o personalizadas en una política de escalado predictivo. Las métricas personalizadas son prácticas cuando las métricas predefinidas (como CPU, memoria, etc.) no son suficientes para describir adecuadamente la carga de la aplicación.

Al crear una política de escalado predictivo con métricas personalizadas, puede especificar otras métricas de CloudWatch que proporciona AWS. Si lo desea, también puede especificar las métricas que defina y publique usted. También puede usar las matemáticas métricas para agregar y transformar las métricas existentes en una nueva serie temporal que AWS no realice un seguimiento automático. Por ejemplo, cuando combina valores en los datos al calcular nuevas sumas o promedios, se denomina agregación. 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.

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 proporciona AWS, debe, en primer lugar, publicar las métricas en CloudWatch. Para obtener más información, consulte Publicación de métricas personalizadas en la Guía del usuario de Amazon CloudWatch.

Si publica sus propias métricas, asegúrese de publicar los puntos de datos con una frecuencia mínima de cinco minutos. Los puntos de datos se recuperan de CloudWatch en función de la duración del periodo que necesita. Por ejemplo, la especificación de la métrica de carga utiliza métricas por horas que permiten medir la carga de su aplicación. CloudWatch utiliza sus datos métricos publicados para proporcionar un único valor de datos en cualquier periodo de una hora mediante la agrupación de todos los puntos de datos con marcas temporales que entran en cada periodo de una hora.

Prácticas recomendadas

Las siguientes prácticas recomendadas pueden ayudarlo a utilizar las métricas personalizadas de manera más eficaz:

- La métrica más útil para la especificación de las métricas de carga es una métrica que representa la carga en un grupo de escalado automático en su conjunto.
- La métrica más útil para el escalado por parte la especificación de la métrica de escalado es una métrica de rendimiento o uso promedio por tarea.
- 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, por ejemplo.
- 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. Como alternativa, una vez creada la política de escalado predictivo, examine los objetos `LoadForecast` que se devuelven mediante una llamada a la API GetPredictiveScalingForecast.
- Se recomienda configurar el escalado predictivo en modo Solo pronóstico para poder evaluar el pronóstico antes de que el escalado predictivo comience a escalar de forma activa.

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.

Solución de problemas de una política de escalado predictivo con métricas personalizadas

Si se produce un problema durante el uso de las métricas personalizadas, se recomienda seguir los siguientes pasos:

- Si encuentra un problema en una implementación azul/verde al usar una expresión de búsqueda, asegúrese de haber creado una expresión de búsqueda que busque una coincidencia parcial y no una coincidencia exacta. También debe comprobar que en la consulta solo se busquen los grupos de escalado automático que se ejecutan en la aplicación específica. Para obtener más información sobre la sintaxis de expresiones de búsqueda, consulte Sintaxis de la expresión de búsqueda de CloudWatch en la Guía del usuario de Amazon CloudWatch.
- El comando put-scaling-policy valida una expresión 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 problemas de la expresión desde la consola de CloudWatch.

- 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.

Creación del archivo JSON para las métricas personalizadas de escalado predictivo con Amazon ECS

En la siguiente sección hay ejemplos de cómo configurar el escalado predictivo para consultar datos de 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 métricas de CloudWatch proporcionadas por AWS o de las métricas que publique en CloudWatch, consulte Ejemplo de política de escalado predictivo con métricas personalizadas de escalado y de carga mediante la AWS CLI.

Ejemplo de política de escalado predictivo con métricas personalizadas de escalado y de carga mediante la AWS CLI

Para crear una política de escalado predictivo con métricas de carga y escalado personalizadas con AWS CLI, almacene los argumentos de `--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 MetricDataQuery en la Referencia de la API de Amazon EC2 Auto Scaling.

> **Note**
>
> Los siguientes son algunos de los recursos adicionales que pueden ayudarle a encontrar nombres de métricas, espacios de nombres, dimensiones y estadísticas para las métricas de CloudWatch:
>
> - Para obtener más información sobre las métricas disponibles en los servicios de AWS, consulte Servicios de AWS que publican métricas de CloudWatch en la Guía del usuario de Amazon CloudWatch.
> - Para obtener el nombre exacto de la métrica, el espacio de nombres y las dimensiones (si corresponde) de una métrica de CloudWatch con la AWS CLI, consulte 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 application-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 proporciona información acerca del uso de las matemáticas de métricas con políticas de escalado predictivo en su política.

Descripción del cálculo de métricas

Si todo lo que desea hacer es agrupar datos de métricas existentes, los cálculos de métricas de CloudWatch permiten ahorrar tiempo y costos en la publicación de otras métricas en CloudWatch. Puede utilizar cualquier métrica que proporcione AWS y, además, puede usar las métricas que defina como parte de sus aplicaciones.

Para obtener más información, consulte Uso de cálculo de métricas en la Guía del usuario de Amazon CloudWatch.

Si decide utilizar una expresión de cálculo de métricas en su política de escalado predictivo, tenga en cuenta los siguientes aspectos:

- Las operaciones de cálculo de métricas utilizan los puntos de datos de la combinación única de nombre de métrica, espacio de nombres y pares de claves/valores de la dimensión de las métricas.
- Puede utilizar cualquier operador aritmético (+ - * / ^), función estadística (como PROM o SUM) u otra función que sea compatible con 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 de CloudWatch. A continuación, cree la expresión. Para obtener más información, consulte Uso de cálculo de métricas en la Guía del usuario de Amazon CloudWatch.
- Compruebe que la expresión del cálculo de métricas sea válida mediante el uso de la consola de CloudWatch o la API GetMetricData de CloudWatch.

Ejemplo de política de escalado predictivo que combina las métricas mediante el uso del cálculo de métricas (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.

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 application-autoscaling put-scaling-policy --policy-name my-sqs-custom-metrics-
policy \
  --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
  --service-namespace ecs \
  --resource-id service/MyCluster/test \
  "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": []
}
```

Interconexión de los servicios de Amazon ECS

Las aplicaciones que se ejecutan en las tareas de Amazon ECS a menudo necesitan recibir conexiones de Internet o conectarse a otras aplicaciones que se ejecutan en los servicios de Amazon ECS. Si necesita conexiones externas desde Internet, le recomendamos usar Elastic Load Balancing. Para obtener más información sobre el equilibrador de carga integrado, consulte the section called "Uso del equilibrador de carga para distribuir el tráfico de servicio".

Si necesita una aplicación para conectarse a otras aplicaciones que se ejecutan en los servicios de Amazon ECS, Amazon ECS proporciona las siguientes formas de hacerlo sin un equilibrador de carga:

- Amazon ECS Service Connect

  Recomendamos Service Connect, que proporciona la configuración de Amazon ECS para la detección de servicios, la conectividad y la supervisión del tráfico. Con Service Connect, sus aplicaciones pueden usar nombres abreviados y puertos estándar para conectarse a los servicios de Amazon ECS del mismo clúster o a otros clústeres, incluso a través de VPC de la misma Región de AWS.

  Cuando utiliza Service Connect, Amazon ECS administra todas las partes de la detección de servicios: crea los nombres que se pueden descubrir, administra dinámicamente las entradas de cada tarea a medida que se inician y se detienen las tareas y ejecuta un agente en cada tarea que está configurado para descubrir los nombres. La aplicación puede buscar los nombres mediante la

funcionalidad estándar para los nombres de DNS y establecer conexiones. Si su aplicación ya lo hace, no tendrá que modificarla para usar Service Connect.

Usted proporciona la configuración completa dentro de cada definición de servicio y tarea. Amazon ECS administra los cambios en esta configuración en cada implementación de servicio para garantizar que todas las tareas de una implementación se comporten de la misma manera. Por ejemplo, un problema común con el DNS en la detección de servicios es el control de una migración. Si cambia un nombre de DNS para que apunte a las nuevas direcciones IP de reemplazo, es posible que pase el tiempo máximo de TTL antes de que todos los clientes comiencen a usar el nuevo servicio. Con Service Connect, la implementación del cliente actualiza la configuración sustituyendo las tareas del cliente. Puede configurar el disyuntor de implementación y otras configuraciones de implementación para que afecten a los cambios de Service Connect de la misma manera que cualquier otra implementación.

Para obtener más información, consulte Uso de Service Connect para conectar los servicios de Amazon ECS con nombres abreviados.

- Detección de servicios de Amazon ECS

  Otro enfoque para la comunicación de servicio a servicio es la comunicación directa mediante la detección de servicios. En este enfoque, puede utilizar la integración de detección de servicios AWS Cloud Map con Amazon ECS. Mediante la detección de servicios, Amazon ECS sincroniza la lista de tareas iniciadas en AWS Cloud Map, que mantiene un nombre de host DNS que se resuelve en las direcciones IP internas de una o más tareas de ese servicio en particular. Otros servicios de Amazon VPC pueden usar este nombre de host DNS para enviar tráfico directamente a otro contenedor mediante su dirección IP interna.

  Este enfoque de comunicación de servicio a servicio proporciona una baja latencia. No hay componentes adicionales entre los contenedores. El tráfico viaja directamente de un contenedor al otro.

  Este enfoque es adecuado cuando se utiliza el modo de red `awsvpc`, en el que cada tarea tiene su propia dirección IP única. La mayoría de los programas solo admiten el uso de registros `A` de DNS, que se resuelven directamente en las direcciones IP. Cuando se utiliza el modo de red `awsvpc`, la dirección IP de cada tarea es un registro `A`. Sin embargo, si utiliza el modo de red `bridge`, es posible que varios contenedores compartan la misma dirección IP. Además, las asignaciones dinámicas de puertos hacen que a los contenedores se les asignen números de puerto de forma aleatoria en esa única dirección IP. En este punto, un registro `A` ya no es suficiente para la detección de servicios. También debe usar un registro `SRV`. Este tipo de registro

puede hacer un seguimiento de las direcciones IP y los números de puerto, pero requiere que configure las aplicaciones de forma adecuada. Es posible que algunas aplicaciones prediseñadas que utilice no admitan registros `SRV`.

Otra ventaja del modo de red `awsvpc` es que tiene un grupo de seguridad único para cada servicio. Puede configurar este grupo de seguridad para permitir las conexiones entrantes únicamente desde los servicios ascendentes específicos que necesitan comunicarse con ese servicio.

La principal desventaja de la comunicación directa entre servicios mediante la detección de servicios es que se debe implementar una lógica adicional para llevar a cabo reintentos y solucionar los errores de conexión. Los registros de DNS tienen un período de vida (TTL) que controla el periodo durante el que se almacenan en caché. El registro DNS tarda algún tiempo en actualizarse y la caché en caducar para que las aplicaciones puedan recoger la última versión del registro DNS. Por lo tanto, su aplicación podría terminar resolviendo el registro DNS para que apunte a otro contenedor que ya no está allí. Su aplicación debe gestionar los reintentos y tener una lógica para ignorar los backends defectuosos.

Para obtener más información, consulte Uso de la detección de servicios para conectar los servicios de Amazon ECS con nombres de DNS

- Amazon VPC Lattice

  Amazon VPC Lattice es un servicio de redes de aplicaciones administrado que utilizan los clientes de Amazon ECS para observar, proteger y supervisar las aplicaciones creadas en servicios de computación, VPC y cuentas de AWS sin tener que modificar su código.

  VPC Lattice usa grupos de destino, que son un conjunto de recursos de computación. Estos destinos ejecutan la aplicación o servicio y pueden ser instancias de Amazon EC2, direcciones IP, funciones de Lambda y equilibradores de carga de aplicación. Al asociar sus servicios de Amazon ECS a un grupo de destino de VPC Lattice, los clientes ahora pueden habilitar las tareas de Amazon ECS como destinos de IP en VPC Lattice. Amazon ECS registra automáticamente las tareas en el grupo de destino de VPC Lattice cuando se lanzan las tareas del servicio registrado.

  Para obtener más información, consulte Uso de Amazon VPC Lattice para conectar, observar y proteger los servicios de Amazon ECS.

Tabla de compatibilidad del modo de red

La siguiente tabla describe la compatibilidad entre estas opciones y los modos de red de tareas. En la tabla, “cliente” hace referencia a la aplicación que hace las conexiones desde una tarea de Amazon ECS.

Opciones de interconexión	Puenteado	**`awsvpc`**	Host
Detección de servicios	sí, pero requiere que los clientes conozcan los registros SRV en DNS sin `hostPort`.	yes	sí, pero requiere que los clientes conozcan los registros SRV en DNS sin `hostPort`.
Service Connect	yes	sí	no
VPC Lattice	yes	sí	sí

Uso de Service Connect para conectar los servicios de Amazon ECS con nombres abreviados

Amazon ECS Service Connect proporciona la administración de la comunicación de servicio a servicio como configuración de Amazon ECS. Crea una detección de servicios y una malla de servicios en Amazon ECS. Esto proporciona la configuración completa de cada servicio que administra usted mediante implementaciones de servicios, una manera unificada de hacer referencia a los servicios dentro de los espacios de nombres que no depende de la configuración de DNS de VPC, y métricas y registros estandarizados para supervisar todas las aplicaciones. Service Connect solo interconecta servicios.

En el siguiente diagrama se muestra un ejemplo de red Service Connect con 2 subredes en la VPC y 2 servicios. Un servicio de cliente que ejecuta WordPress con 1 tarea en cada subred. Un servicio de servidor que ejecuta MySQL con 1 tarea en cada subred. Ambos servicios tienen una alta disponibilidad y son resilientes ante los problemas de tareas y zonas de disponibilidad, ya que cada servicio ejecuta varias tareas distribuidas en 2 subredes. Las flechas continuas muestran una conexión de WordPress a MySQL. Por ejemplo, un comando de la CLI de `mysql --host=mysql` que se ejecuta desde el interior del contenedor de WordPress en la tarea con la dirección IP

`172.31.16.1`. El comando utiliza el nombre corto `mysql` en el puerto predeterminado para MySQL. Este nombre y puerto se conectan al proxy de Service Connect en la misma tarea. El proxy de la tarea de WordPress utiliza el equilibrador de carga de distribución equilibrada y cualquier información de error anterior en la detección de valores atípicos para elegir a qué tarea de MySQL conectarse. Como muestran las flechas continuas del diagrama, el proxy se conecta al segundo proxy de la tarea MySQL con la dirección IP `172.31.16.2`. El segundo proxy se conecta al servidor MySQL local en la misma tarea. Ambos proxys informan del rendimiento de la conexión, que se puede ver en los gráficos de las consolas de Amazon ECS y de Amazon CloudWatch, de modo que puede obtener métricas de rendimiento de todo tipo de aplicaciones de la misma manera.

Los siguientes términos se utilizan con Service Connect:

nombre del puerto

La configuración de definición de tareas de Amazon ECS que asigna un nombre a una asignación de puertos determinada. Esta configuración solo la usa Amazon ECS Service Connect.

alias de cliente

La configuración del servicio de Amazon ECS que asigna el número de puerto que se usa en el punto de conexión. Además, el alias del cliente puede asignar el nombre DNS del punto de conexión y anular el nombre de detección. Si no se proporciona un nombre de detección en el servicio de Amazon ECS, el nombre del alias del cliente sustituye al nombre del puerto como nombre del punto de conexión. Para ver ejemplos de puntos de conexión, consulte la definición de punto de conexión. Se pueden asignar varios alias de cliente a un servicio de Amazon ECS. Esta configuración solo la usa Amazon ECS Service Connect.

nombre de detección

El nombre intermedio opcional que puede crear para un puerto especificado de la definición de la tarea. Este nombre se utiliza para crear un servicio de AWS Cloud Map. Si no se proporciona este nombre, se utiliza el nombre del puerto de la definición de la tarea. Se pueden asignar varios nombres de detección a un puerto específico de un servicio de Amazon ECS. Esta configuración solo la usa Amazon ECS Service Connect.

Los nombres de servicio de AWS Cloud Map deben ser únicos en un espacio de nombres. Debido a esta limitación, solo puede tener una configuración de Service Connect sin un nombre de detección para una definición de tarea determinada en cada espacio de nombres.

punto de conexión

La URL para conectarse a una API o un sitio web. La URL contiene el protocolo, un nombre DNS y el puerto. Para obtener más información sobre los puntos de conexión en general, consulte punto de conexión en el glosario de AWS en la Referencia general de Amazon Web Services.

Service Connect crea puntos de conexión que se conectan a los servicios de Amazon ECS y configura las tareas de los servicios de Amazon ECS para conectarse a los puntos de conexión. La URL contiene el protocolo, un nombre DNS y el puerto. Seleccione el protocolo y el nombre del puerto en la definición de la tarea, ya que el puerto debe coincidir con la aplicación que se encuentra dentro de la imagen del contenedor. En el servicio, selecciona cada puerto por nombre y puede asignar el nombre DNS. Si no especifica un nombre DNS en la configuración del servicio de Amazon ECS, se utiliza de forma predeterminada el nombre del puerto de la definición de la tarea. Por ejemplo, un punto de conexión de Service Connect podría ser `http://blog:80`, `grpc://checkout:8080` o `http://_db.production.internal:99`.

Servicio de Service Connect

La configuración de un punto de conexión único en un servicio de Amazon ECS. Forma parte de la configuración de Service Connect y consiste en una sola fila en la Service Connect and

discovery name configuration (Configuración de Service Connect y nombre de detección) de la consola, o un objeto de la lista de `services` de la configuración JSON de un servicio de Amazon ECS. Esta configuración solo la usa Amazon ECS Service Connect.

Para obtener más información, consulte ServiceConnectService en la Referencia de la API de Amazon Elastic Container Service.

namespace

El nombre corto o el nombre de recurso de Amazon (ARN) completo del espacio de nombres de AWS Cloud Map para su uso con Service Connect. El espacio de nombres debe estar en la misma Región de AWS que el servicio y el clúster de Amazon ECS. El tipo de espacio de nombres en AWS Cloud Map no afecta a Service Connect.

Service Connect utiliza el espacio de nombres de AWS Cloud Map como una agrupación lógica de tareas de Amazon ECS que se comunican entre sí. Cada servicio de Amazon ECS solo puede pertenecer a un espacio de nombres. Los servicios de un espacio de nombres se pueden distribuir en diferentes clústeres de Amazon ECS dentro de la misma Región de AWS en la misma Cuenta de AWS. Puede organizar con toda libertad los servicios según cualquier criterio.

servicio de cliente

Un servicio que ejecuta una aplicación cliente de red. Este servicio debe tener un espacio de nombres configurado. Cada tarea del servicio puede detectar y conectarse a todos los puntos de conexión del espacio de nombres a través de un contenedor del proxy de Service Connect.

Si alguno de los contenedores de la tarea necesita conectarse a un punto de conexión desde un servicio de un espacio de nombres, elija un servicio de cliente. Si una aplicación de frontend, proxy inverso o equilibrador de carga recibe tráfico externo mediante otros métodos, como desde Elastic Load Balancing, podría utilizar este tipo de configuración de Service Connect.

servicio cliente-servidor

Un servicio de Amazon ECS que ejecuta una aplicación de red o servicio web. Este servicio debe tener un espacio de nombres y al menos un punto de conexión configurado. Se puede acceder a cada tarea del servicio mediante los puntos de conexión. El contenedor del proxy de Service Connect escucha el nombre y el puerto del punto de conexión para dirigir el tráfico a los contenedores de aplicaciones de la tarea.

Si alguno de los contenedores expone y escucha el tráfico de red en un puerto, elija un servicio cliente-servidor. Estas aplicaciones no necesitan conectarse a otros servicios cliente-servidor

en el mismo espacio de nombres, pero la configuración del cliente es necesaria. Un backend, un middleware, un nivel empresarial o la mayoría de los microservicios pueden utilizar este tipo de configuración de Service Connect. Si desea que una aplicación de frontend, proxy inverso o equilibrador de carga reciba tráfico de otros servicios configurados con Service Connect en el mismo espacio de nombres, estos servicios deben usar este tipo de configuración de Service Connect.

La característica Service Connect crea una red virtual de servicios relacionados. Se puede usar la misma configuración de servicio en varios espacios de nombres diferentes para ejecutar conjuntos de aplicaciones independientes, pero idénticos. Service Connect define el contenedor del proxy en el servicio de Amazon ECS. De esta forma, se puede utilizar la misma definición de tarea para ejecutar aplicaciones idénticas en diferentes espacios de nombres con diferentes configuraciones de Service Connect. Cada tarea que lleva a cabo el servicio ejecuta un contenedor del proxy en la tarea.

Service Connect es adecuado para conexiones entre servicios de Amazon ECS dentro del mismo espacio de nombres. Para las siguientes aplicaciones, debe utilizar un método de interconexión adicional para conectarse a un servicio de Amazon ECS configurado con Service Connect:

- Tareas configuradas en otros espacios de nombres
- Tareas que no están configuradas para Service Connect
- Otras aplicaciones fuera de Amazon ECS

Estas aplicaciones pueden conectarse a través del proxy de Service Connect, pero no pueden resolver los nombres de los puntos de conexión de Service Connect.

Para que estas aplicaciones resuelvan las direcciones IP de las tareas de Amazon ECS, debe utilizar otro método de interconexión.

Temas

- Precios
- Componentes de Amazon ECS Service Connect
- Información general de la configuración de Amazon ECS Service Connect
- Cifrado del tráfico de Amazon ECS Service Connect
- Configuración de Amazon ECS Service Connect con la AWS CLI

Precios

- Los precios de Amazon ECS Service Connect dependerán de si utiliza AWS Fargate o la infraestructura de Amazon EC2 para alojar sus cargas de trabajo en contenedores. Cuando se utiliza Amazon ECS en AWS Outposts, los precios siguen el mismo modelo que cuando se utiliza Amazon EC2 directamente. Para obtener más información, consulte Precios de Amazon ECS.
- El uso de Amazon ECS Service Connect no supone ningún cargo adicional.
- El uso de AWS Cloud Map es completamente gratuito cuando Service Connect lo utiliza.
- Los clientes pagan por los recursos de computación que utiliza Amazon ECS Service Connect, incluidas la vCPU y la memoria. Como el agente de Amazon ECS Service Connect se ejecuta dentro de una tarea del cliente, ejecutarlo no conlleva ningún cargo adicional. Los recursos de la tarea se comparten entre la carga de trabajo del cliente y el agente de Amazon ECS Service Connect.
- Al utilizar la funcionalidad de cifrado de tráfico de Amazon ECS Service Connect con AWS Private CA, los clientes pagan por la autoridad de certificación privada que crean y por cada certificado TLS emitido. Para obtener más información, consulte Precios de AWS Private Certificate Authority. Para calcular el costo mensual de los certificados TLS, los clientes deben saber el número de servicios de Amazon ECS que tienen activado el TLS, multiplicarlo por el costo del certificado y, a continuación, multiplicarlo por seis. Como Amazon ECS Service Connect rota automáticamente los certificados TLS cada cinco días, se emiten seis certificados por servicio de Amazon ECS, por mes, de media.

Componentes de Amazon ECS Service Connect

Cuando utiliza Amazon ECS Service Connect, configura cada servicio de Amazon ECS para ejecutar una aplicación de servidor que recibe solicitudes de red (servicio cliente-servidor) o para ejecutar una aplicación cliente que hace las solicitudes (servicio de cliente).

Cuando se prepare para empezar a utilizar Service Connect, comience con un servicio cliente-servidor. Puede agregar una configuración de Service Connect a un servicio nuevo o existente. Amazon ECS crea un punto de conexión de Service Connect en el espacio de nombres. Además, Amazon ECS crea una nueva implementación en el servicio para reemplazar las tareas que se están ejecutando actualmente.

Las tareas y otras aplicaciones existentes pueden seguir conectándose a los puntos de conexión existentes y a las aplicaciones externas. Si un servicio cliente-servidor agrega tareas mediante el escalado horizontal, las nuevas conexiones de los clientes se equilibrarán entre todas las tareas. Si

se actualiza un servicio cliente-servidor, las nuevas conexiones de los clientes se equilibrarán entre las tareas de la nueva versión.

Las tareas existentes no pueden resolverse ni conectarse al nuevo punto de conexión. Solo las tareas nuevas con una configuración de Service Connect en el mismo espacio de nombres y que comiencen a ejecutarse después de esta implementación pueden resolverse y conectarse a este punto de conexión.

Esto significa que el operador de la aplicación cliente determina cuándo cambia la configuración de su aplicación, aunque el operador de la aplicación de servidor puede cambiar su configuración en cualquier momento. La lista de puntos de conexión del espacio de nombres puede cambiar cada vez que se implemente cualquier servicio del espacio de nombres. Las tareas existentes y las tareas de reemplazo siguen comportándose del mismo modo que después de la implementación más reciente.

Considere los siguientes ejemplos:

En primer lugar, suponga que está creando una aplicación que está disponible para el Internet público en una sola plantilla de AWS CloudFormation y una sola pila de AWS CloudFormation. AWS CloudFormation debe crear la detección pública y la accesibilidad en último lugar, incluido el servicio de cliente frontend. El servicio debe crearse en este orden para evitar un período en el que el servicio de cliente frontend se esté ejecutando y esté disponible para el público, pero el backend no lo esté. Esto evita que los mensajes de error se envíen al público durante ese periodo. En AWS CloudFormation, debe utilizar `dependsOn` para indicar a AWS CloudFormation que no se pueden crear varios servicios de Amazon ECS en paralelo o simultáneamente. Debe agregar `dependsOn` al servicio de cliente frontend para cada servicio cliente-servidor backend al que se conecten las tareas del cliente.

En segundo lugar, suponga que existe un servicio frontend sin la configuración de Service Connect. Las tareas se conectan a un servicio de backend existente. Agregue primero una configuración de Service Connect de cliente-servidor al servicio backend, con el mismo nombre en el DNS o el `clientAlias` que usa el frontend. Esto crea una nueva implementación, es decir, toda la detección de restauración de la implementación o la AWS Management Console, la AWS CLI, los SDK de AWS y otros métodos para revertir y restaurar el servicio de backend a la implementación y la configuración anteriores. Si está satisfecho con el rendimiento y el comportamiento del servicio de backend, agregue una configuración de Service Connect de cliente o cliente-servidor al servicio frontend. Solo las tareas de la nueva implementación utilizan el proxy de Service Connect que se agrega a esas tareas nuevas. Si tiene problemas con esta configuración, puede revertir y volver a su configuración anterior mediante la detección de restauración de la implementación o la AWS

Management Console, la AWS CLI, los SDK de AWS y otros métodos para revertir y restaurar el servicio de backend a la implementación y la configuración anteriores. Si utiliza otro sistema de detección de servicios basado en DNS en lugar de Service Connect, cualquier aplicación frontend o de cliente empezará a utilizar nuevos puntos de conexión y una configuración de punto de conexión modificada una vez que caduque la caché de DNS local, lo que suele tardar varias horas.

Red

De manera predeterminada, el proxy de Service Connect escucha en `containerPort` desde la asignación de puertos de la definición de la tarea. Las reglas del grupo de seguridad deben permitir el tráfico entrante a este puerto desde las subredes en las que se ejecutarán los clientes.

Incluso si establece un número de puerto en la configuración del servicio de Service Connect, esto no cambia el puerto del servicio cliente-servidor que escucha el proxy de Service Connect. Al configurar este número de puerto, Amazon ECS cambia el puerto del punto de conexión al que se conectan los servicios del cliente, en el proxy de Service Connect dentro de esas tareas. El proxy del servicio de cliente se conecta al proxy del servicio cliente-servidor mediante el `containerPort`.

Si desea cambiar el puerto que escucha el proxy de Service Connect, cambie `ingressPortOverride` en la configuración de Service Connect del servicio cliente-servidor. Si cambia este número de puerto, debe permitir el tráfico entrante en este puerto que utiliza el tráfico hacia este servicio.

El tráfico que sus aplicaciones envían a los servicios de Amazon ECS configurados para Service Connect requiere que Amazon VPC y las subredes cuenten con reglas de tabla de enrutamiento y reglas de ACL de red que permitan los números de puerto `containerPort` y `ingressPortOverride` que está utilizando.

Puede utilizar Service Connect para enviar tráfico entre las VPC. Los mismos requisitos para las reglas de la tabla de enrutamiento, las ACL de red y los grupos de seguridad se aplican a ambas VPC.

Por ejemplo, dos clústeres crean tareas en diferentes VPC. Un servicio de cada clúster está configurado para usar el mismo espacio de nombres. Las aplicaciones de estos dos servicios pueden resolver todos los puntos de conexión del espacio de nombres sin ninguna configuración de DNS de VPC. Sin embargo, los proxies no se pueden conectar a menos que el emparejamiento de VPC, las tablas de enrutamiento de subredes o VPC, y las ACL de red de VPC permitan el tráfico en los números de puerto `containerPort` y `ingressPortOverride`.

Para las tareas que utilizan el modo de red `bridge`, debe crear un grupo de seguridad con una regla de entrada que permita el tráfico en el rango superior de puertos dinámicos. A continuación, asigne el grupo de seguridad a todas las instancias EC2 del clúster de Service Connect.

Proxy de Service Connect

Si crea o actualiza un servicio con la configuración de Service Connect, Amazon ECS agrega un contenedor nuevo a cada tarea nueva a medida que se inicia. Este patrón de uso de un contenedor independiente se denomina `sidecar`. Este contenedor no está presente en la definición de la tarea y no puede configurarlo. Amazon ECS administra la configuración de este contenedor en el servicio. Esto le permite reutilizar las mismas definiciones de tareas entre varios servicios, espacios de nombres y tareas sin Service Connect.

Recursos de proxy

- Para las definiciones de tareas, debe establecer los parámetros de la CPU y la memoria.

  Recomendamos agregar 256 unidades de CPU adicionales y al menos 64 MiB de memoria a la memoria y la CPU de la tarea del contenedor del proxy de Service Connect. En AWS Fargate, la cantidad mínima de memoria que puede configurar es de 512 MiB. En Amazon EC2, es necesaria la memoria de definición de tareas.
- Para el servicio, se establece la configuración del registro en la configuración de Service Connect.
- Si espera que las tareas de este servicio reciban más de 500 solicitudes por segundo en su carga máxima, le recomendamos agregar 512 unidades de CPU a la CPU de tareas en esta definición de tareas para el contenedor del proxy de Service Connect.
- Si espera crear más de 100 servicios de Service Connect en el espacio de nombres o 2000 tareas en total en todos los servicios de Amazon ECS dentro del espacio de nombres, le recomendamos agregar 128 MiB de memoria a la memoria de tareas para el contenedor del proxy de Service Connect. Debe hacerlo en todas las definiciones de tareas que utilicen todos los servicios de Amazon ECS del espacio de nombres.

Configuración del proxy

Sus aplicaciones se conectan al proxy del contenedor sidecar en la misma tarea en la que se encuentra la aplicación. Amazon ECS configura la tarea y los contenedores para que las aplicaciones solo se conecten al proxy cuando la aplicación se conecta a los nombres de los puntos de conexión en el mismo espacio de nombres. El resto del tráfico no utiliza el proxy. El resto del tráfico incluye direcciones IP en la misma VPC, puntos de conexión de servicios de AWS y tráfico externo.

Equilibrio de carga

Service Connect configura el proxy para que utilice la estrategia de distribución equilibrada para equilibrar la carga entre las tareas en un punto de conexión de Service Connect. El proxy local que se encuentra en la tarea desde la que proviene la conexión selecciona una de las tareas del servicio cliente-servidor que proporciona el punto de conexión.

Por ejemplo, consideremos una tarea que ejecuta WordPress en un servicio configurado como servicio de cliente en un espacio de nombres llamado local. Hay otro servicio con 2 tareas que ejecuta la base de datos MySQL. Este servicio está configurado para proporcionar un punto de conexión llamado `mysql` a través de Service Connect en el mismo espacio de nombres. En la tarea de WordPress, la aplicación de WordPress se conecta a la base de datos mediante el nombre del punto de conexión. Las conexiones con este nombre van al proxy que se ejecuta en un contenedor asociado en la misma tarea. A continuación, el proxy puede conectarse a cualquiera de las tareas de MySQL mediante la estrategia de distribución equilibrada.

Estrategias de equilibrio de carga: distribución equilibrada

Detección de valores atípicos

Esta característica utiliza los datos que el proxy tiene sobre conexiones con errores anteriores para evitar enviar nuevas conexiones a los hosts que tenían las conexiones con errores. Service Connect configura la característica de detección de valores atípicos del proxy para proporcionar comprobaciones de estado pasivas.

Con el ejemplo anterior, el proxy puede conectarse a cualquiera de las tareas de MySQL. Si el proxy realizó varias conexiones a una tarea de MySQL específica y 5 o más de las conexiones fallaron en los últimos 30 segundos, el proxy evita esa tarea de MySQL durante 30 a 300 segundos.

Reintentos

Service Connect configura el proxy para volver a intentar la conexión que pasa por el proxy y falla, y el segundo intento evita usar el host de la conexión anterior. Esto garantiza que cada conexión a través de Service Connect no tenga errores por motivos puntuales.

Número de reintentos: 2

Tiempo de espera

Service Connect configura el proxy para que espere un tiempo máximo a que respondan las aplicaciones cliente-servidor. El valor de tiempo de espera predeterminado es de 15 segundos, pero se puede actualizar.

Parámetros opcionales:

idleTimeout: el tiempo en segundos que una conexión permanece activa mientras está inactiva. Un valor de `0` deshabilita `idleTimeout`.

El `idleTimeout` predeterminado de `HTTP/HTTP2/GRPC` es 5 minutos.

El `idleTimeout` predeterminado de `TCP` es una hora.

perRequestTimeout: el tiempo que se tarda en esperar a que el remitente responda con una respuesta completa por solicitud. El valor `0` desactiva `perRequestTimeout`. Esto solo se puede configurar cuando `appProtocol` del contenedor de la aplicación es `HTTP`, `HTTP2` o `GRPC`. El valor predeterminado es de 15 segundos.

> **Note**
>
> Si `idleTimeout` se establece en un tiempo inferior a `perRequestTimeout`, la conexión se cerrará cuando `idleTimeout` se alcance y no el `perRequestTimeout`.

Consideraciones

Cuando utilice Service Connect, tenga en cuenta lo siguiente:

- Las tareas que se ejecutan en Fargate deben utilizar la versión `1.4.0` o superior de la plataforma de Fargate Linux para utilizar Service Connect.
- La versión del agente de Amazon ECS en la instancia de contenedor debe ser `1.67.2` o una superior.
- Las instancias de contenedor deben ejecutar la versión AMI de Amazon Linux 2023 optimizada para Amazon ECS `20230428` o una posterior o la versión AMI de Amazon Linux optimizada para Amazon ECS `2.0.20221115` para utilizar Service Connect. Estas versiones tienen el agente de Service Connect además del agente de contenedor de Amazon ECS. Para obtener más información sobre el agente de ECS, consulte Agente de Service Connect de Amazon ECS en GitHub.
- Las instancias de contenedor deben tener el permiso `ecs:Poll` para el recurso `arn:aws:ecs:`*`region`*`:`*`0123456789012`*`:task-set/`*`cluster`*`/*`. Si utiliza `ecsInstanceRole`, no es necesario que agregue permisos adicionales. La política administrada `AmazonEC2ContainerServiceforEC2Role` tiene los permisos necesarios. Para obtener más información, consulte Rol de IAM de instancia de contenedor de Amazon ECS.

- Las tareas que usan el modo de red `bridge` y utilizan Service Connect no admiten el parámetro de definición de contenedor `hostname`.
- Las definiciones de tareas deben establecer el límite de memoria de tareas para usar Service Connect. Para obtener más información, consulte Proxy de Service Connect.
- No se admiten las definiciones de tareas que establezcan límites de memoria de contenedores.

  Puede establecer límites de memoria de contenedores en sus contenedores, pero debe establecer el límite de memoria de tareas en un número mayor que la suma de los límites de memoria del contenedor. El contenedor del proxy de Service Connect y otros contenedores que no establecen límites de contenedores utilizan la CPU y la memoria adicionales de los límites de tareas que no están asignadas en los límites de contenedores. Para obtener más información, consulte Proxy de Service Connect.
- Puede configurar Service Connect para utilizar cualquier espacio de nombres de AWS Cloud Map de la misma región en la misma Cuenta de AWS.
- Cada servicio puede pertenecer a un solo espacio de nombres.
- Solo se admiten las tareas que crean los servicios.
- Todos los puntos de conexión deben ser únicos dentro de un espacio de nombres.
- Todos los nombres de detección deben ser únicos dentro de un espacio de nombres.
- Debe volver a implementar los servicios existentes para que las aplicaciones puedan resolver nuevos puntos de conexión. Los puntos de conexión nuevos que se agreguen al espacio de nombres después de la implementación más reciente no se agregarán a la configuración de la tarea. Para obtener más información, consulte the section called “Componentes de Service Connect”.
- Service Connect no elimina los espacios de nombres cuando se eliminan los clústeres. Debe eliminar los espacios de nombres en AWS Cloud Map.
- El tráfico del Equilibrador de carga de aplicación se dirige de forma predeterminada a través del agente Service Connect en el modo de red `awsvpc`. Si desea que el tráfico no relacionado con el servicio omita el agente de Service Connect, utilice el parámetro `ingressPortOverride` en la configuración del servicio de Service Connect.
- Service Connect con TLS no admite el modo de red Bridge. Solo se admite el modo de red `awsvpc`.

Service Connect no admite lo siguiente:

- Contenedores de Windows

- HTTP 1.0
- Tareas independientes
- Servicios que utilizan los tipos de implementación azul/verde impulsada por CodeDeploy y la implementación externa
- Service Connect no admite la instancia de contenedor `External` para Amazon ECS Anywhere.
- PPv2

Información general de la configuración de Amazon ECS Service Connect

Cuando se utiliza Service Connect, hay parámetros que se deben configurar en los recursos.

En la siguiente tabla se describen los parámetros de configuración de los recursos de Amazon ECS.

Ubicación de parámetros	Tipo de aplicación	Descripción	Obligatorio
Definición de tarea	Cliente	No hay cambios disponibles para Service Connect en las definiciones de tareas del cliente.	N/A
Definición de tarea	Cliente-servidor	Los servidores deben agregar campos `name` a los puertos en las `portMappings` de los contenedores. Para obtener más información, consulte `portMappings`	Sí
Definición de tarea	Cliente-servidor	De manera opcional, los servidores pueden proporcionar un protocolo de aplicación (por ejemplo, HTTP) para recibir métricas específicas del protocolo para sus aplicaciones de servidor (por ejemplo, `HTTP 5xx`).	No
Definición de servicio	Cliente	Los servicios de cliente deben agregar una `serviceConnectConfiguration` para configurar el espacio de nombres al cual unirse. Este	Sí

Ubicación de parámetros	Tipo de aplicación	Descripción	Obligatorio
		espacio de nombres debe contener todos los servicios de servidor que este servicio debe detectar. Para obtener más información, consulte `serviceConnectConfiguration`.	
Definición de servicio	Cliente-servidor	Los servicios del servidor deben agregar una `serviceConnectConfiguration` para configurar los nombres DNS, los números de puerto y el espacio de nombres desde los que está disponible el servicio. Para obtener más información, consulte `serviceConnectConfiguration`.	Sí
Clúster	Cliente	Los clústeres pueden agregar un espacio de nombres de Service Connect predeterminado. Los nuevos servicios del clúster heredan el espacio de nombres cuando Service Connect se configura en un servicio.	No
Clúster	Cliente-servidor	No hay cambios disponibles para Service Connect en los clústeres que aplican a los servicios del servidor. Las definiciones y los servicios de las tareas del servidor deben establecer la configuración correspondiente.	N/A

Descripción general de los pasos para configurar Service Connect

En los siguientes pasos se proporciona información general sobre cómo configurar Service Connect.

> **Important**
>
> - Service Connect crea servicios de AWS Cloud Map en la cuenta. La modificación de estos recursos AWS Cloud Map mediante el registro o la anulación del registro manual de las instancias, el cambio de los atributos de la instancia o la eliminación de un servicio puede provocar un comportamiento inesperado en el tráfico de las aplicaciones o en las implementaciones posteriores.
> - Service Connect no admite enlaces en la definición de la tarea.

1. Agregue los nombres de los puertos a las asignaciones de puertos en las definiciones de las tareas. Además, puede identificar el protocolo de capa 7 de la aplicación para obtener métricas adicionales.
2. Cree un clúster con un espacio de nombres de AWS Cloud Map o cree el espacio de nombres por separado. Para una organización sencilla, cree un clúster con el nombre que desea para el espacio de nombres y especifique un nombre idéntico para el espacio de nombres. En este caso, Amazon ECS crea un nuevo espacio de nombres HTTP con la configuración necesaria. Service Connect no utiliza ni crea zonas alojadas de DNS en Amazon Route 53.
3. Configure los servicios para crear puntos de conexión de Service Connect dentro del espacio de nombres.
4. Implemente los servicios para crear los puntos de conexión. Amazon ECS agrega un contenedor del proxy de Service Connect a cada tarea y crea los puntos de conexión de Service Connect en AWS Cloud Map. Este contenedor no está configurado en la definición de la tarea y la definición de la tarea se puede reutilizar sin modificaciones para crear varios servicios en el mismo espacio de nombres o en varios espacios de nombres.
5. Implemente aplicaciones cliente como servicios para conectarse a los puntos de conexión. Amazon ECS los conecta a los puntos de conexión de Service Connect a través del proxy de Service Connect en cada tarea.

   Las aplicaciones solo utilizan el proxy para conectarse a los puntos de conexión de Service Connect. No hay ninguna configuración adicional para utilizar el proxy. El proxy realiza el equilibrio de cargas de distribución equilibrada, la detección de valores atípicos y los reintentos. Para obtener más información sobre el proxy, consulte Proxy de Service Connect.
6. Monitoree el tráfico a través del proxy de Service Connect en Amazon CloudWatch.

Configuración del clúster

Puede establecer un espacio de nombres predeterminado para Service Connect cuando crea o actualiza el clúster. Si especifica un nombre de espacio de nombres que no existe en la misma Región de AWS y cuenta, se crea un nuevo espacio de nombres HTTP.

Si crea un clúster y especifica un espacio de nombres de Service Connect predeterminado, el clúster espera en estado `PROVISIONING` mientras Amazon ECS crea el espacio de nombres. Puede ver una `attachment` en el estado del clúster que muestra el estado del espacio de nombres. Las conexiones no se muestran de forma predeterminada en la AWS CLI, debe agregar `--include ATTACHMENTS` para verlos.

Configuración del servicio

Service Connect está diseñado para requerir la configuración mínima. Debe establecer un nombre para cada asignación de puertos que desee utilizar con Service Connect en la definición de la tarea. En el servicio, debe activar Service Connect y seleccionar un espacio de nombres para crear un servicio de cliente. Para crear un servicio cliente-servidor, debe agregar una configuración del servicio Service Connect única que coincida con el nombre de una de las asignaciones de puertos. Amazon ECS reutiliza el número de puerto y el nombre del puerto de la definición de la tarea para definir el servicio y el punto de conexión de Service Connect. Para anular esos valores, puede utilizar los demás parámetros Discovery, DNS y Port en la consola o `discoveryName` y `clientAliases`, respectivamente, en la API de Amazon ECS.

Cifrado del tráfico de Amazon ECS Service Connect

Amazon ECS Service Connect admite el cifrado automático del tráfico con certificados de seguridad de la capa de transporte (TLS) para los servicios de Amazon ECS. Cuando dirige sus servicios de Amazon ECS hacia un AWS Private Certificate Authority(AWS Private CA), Amazon ECS aprovisiona automáticamente certificados TLS para cifrar el tráfico entre sus servicios de Amazon ECS Service Connect. Amazon ECS genera, rota y distribuye los certificados TLS que se utilizan para el cifrado del tráfico.

El cifrado automático del tráfico con Service Connect utiliza funcionalidades de cifrado líderes del sector para proteger la comunicación entre servicios, lo que ayuda a cumplir sus requisitos de seguridad. Es compatible con los certificados TLS AWS Private Certificate Authority con el cifrado `256-bit ECDSA` y `2048-bit RSA`. También tiene el control total sobre los certificados privados y las claves de firma para ayudarle a cumplir los requisitos de conformidad. De forma predeterminada, se admite TLS 1.3, pero no se admite TLS 1.0 - 1.2. Service Connect admite TLS 1.3 con los siguientes cifrados:

- `TLS_AES_128_GCM_SHA256`
- `TLS_AES_256_GCM_SHA384`
- `TLS_CHACHA20_POLY1305_SHA256`

> **Note**
>
> Para poder utilizar TLS 1.3, debe habilitarlo en el oyente del destino.
> Solo se cifra el tráfico entrante y saliente que pasa por el agente de Amazon ECS.

Service Connect y comprobación del estado del equilibrador de carga de aplicación

Puede usar Service Connect con las comprobaciones de estado del equilibrador de carga de aplicación y el cifrado TLS 1.3.

Configuración del equilibrador de carga de aplicación

Configure el equilibrador de carga de aplicación con la siguiente configuración:

- Configure un oyente TLS con una política de seguridad TLS 1.3 (como `ELBSecurityPolicy-TLS13-1-2-2021-06`). Para obtener más información, consulte Políticas de seguridad para el Equilibrador de carga de aplicación.
- Cree un grupo de destinos con la siguiente configuración:
  - Establezca el protocolo en HTTPS
  - Adjunte el grupo de destinos al oyente de TLS
  - Configure el puerto de comprobación de estado para que coincida con el puerto del contenedor del servicio Service Connect

Configuración de Service Connect

Configure un servicio con la siguiente configuración:

- Configure el servicio para que utilice el modo de red `awsvpc`, ya que el modo de red `bridge` no es compatible.
- Habilite Service Connect para el servicio.
- Establezca la configuración del equilibrador de carga con los siguientes ajustes:

- Especifique el grupo de destino que configuró para el rquilibrador de carga de aplicación
- Configure el puerto del contenedor para que coincida con el puerto del contenedor del servicio TLS de Service Connect
- Evite configurar `ingressPortOverride` para el servicio. Para obtener más información, consulte ServiceConnectService en la Referencia de la API de Amazon Elastic Container Service.

Consideraciones

Cuando utilice el equilibrador de carga de aplicación, TLS y Service Connect, tenga en cuenta lo siguiente:

- Utilice el modo de red `awsvpc` en lugar del modo de red `bridge` para las comprobaciones de estado de HTTPS cuando utilice Service Connect con cifrado TLS. Las comprobaciones de estado de HTTP seguirán funcionando con el modo `bridge`.
- Configure el puerto de comprobación de estado del grupo objetivo para que coincida con el puerto del contenedor del servicio Service Connect, no con el puerto HTTPS predeterminado (443).

Certificados AWS Private Certificate Authority y Service Connect

Debe tener el rol de IAM de infraestructura. Para obtener más información sobre este rol, consulte Rol de IAM para la infraestructura de Amazon ECS.

Modos de AWS Private Certificate Authority de Service Connect

AWS Private Certificate Authority puede funcionar en dos modos: de uso general y de corta duración.

- Uso general: certifica que se pueden configurar con cualquier fecha de caducidad.
- De corta duración: emite certificados con un periodo de validez máximo de siete días.

Si bien Amazon ECS admite ambos modos, recomendamos utilizar certificados de corta duración. De forma predeterminada, los certificados se renuevan cada cinco días y, si se ejecutan en el modo de corta duración, se obtienen importantes ahorros de costos en comparación con el uso general.

Service Connect no admite la revocación de certificados, pero aprovecha los certificados de corta duración con una rotación frecuente de certificados. Tiene la autoridad para modificar la frecuencia de rotación, deshabilitar o eliminar los secretos mediante la rotación administrada en Secrets Manager, pero hacerlo podría acarrear las siguientes consecuencias.

- Frecuencia de rotación más corta: una frecuencia de rotación más corta implica costos más altos debido a que AWS Private CA, AWS KMS, Secrets Manager y Auto Scaling experimentan una mayor carga de trabajo para la rotación.
- Frecuencia de rotación más larga: las comunicaciones de sus aplicaciones fallan si la frecuencia de rotación supera los siete días.
- Eliminación del secreto: la eliminación del secreto provoca un error de rotación y afecta a las comunicaciones de las aplicaciones con los clientes.

En caso de que la rotación secreta no funcione, se publicará un evento `RotationFailed` en AWS CloudTrail. También puede configurar una alarma de CloudWatch para `RotationFailed`.

> **Important**
>
> No agregue regiones de réplica a los secretos. Si las agrega, se evita que Amazon ECS elimine el secreto, ya que Amazon ECS no tiene permiso para eliminar regiones de la replicación. Si ya agregó la replicación, ejecute el siguiente comando.
>
> ```
> aws secretsmanager remove-regions-from-replication \
>  --secret-id SecretId \
>  --remove-replica-regions region-name
> ```

Entidades de certificación subordinadas

Puede incorporar cualquier AWS Private CA, raíz o subordinado a TLS de Service Connect para emitir certificados de entidad final para los servicios. El emisor proporcionado se considera el firmante y la raíz de la confianza en todas partes. Puede emitir certificados de entidad final para distintas partes de la solicitud desde distintas entidades de certificación subordinadas. Al utilizar la AWS CLI, proporcione el Nombre de recurso de Amazon (ARN) de la CA para establecer la cadena de confianza.

Autoridades de certificación en las instalaciones

Para usar su CA en las instalaciones, debe crear y configurar una CA subordinada en AWS Private Certificate Authority. Esto garantiza que todos los certificados TLS emitidos para sus cargas de trabajo de Amazon ECS compartan la cadena de confianza con las cargas de trabajo que ejecuta en las instalaciones y puedan conectarse de forma segura.

> **Important**
>
> Agregue la etiqueta obligatoria `AmazonECSManaged : true` en su AWS Private CA.

Infraestructura como código

Al utilizar TLS de Service Connect con las herramientas de infraestructura como código (IaC), es importante configurar las dependencias correctamente para evitar problemas, como el agotamiento de los servicios. La clave de AWS KMS (si la ha proporcionado), el rol de IAM y las dependencias de AWS Private CA se deben eliminar después de utilizar el servicio de Amazon ECS.

Service Connect y Secrets Manager

Cuando se utiliza Amazon ECS Service Connect con el cifrado TLS, el servicio interactúa con Secrets Manager de las siguientes maneras:

Service Connect utiliza el rol de infraestructura proporcionado para crear secretos en Secrets Manager. Estos secretos se utilizan para almacenar las claves privadas asociadas para los certificados TLS para cifrar el tráfico entre los servicios de Service Connect.

> **Warning**
>
> La creación y administración automáticas de estos secretos mediante Service Connect agiliza el proceso de implementación del cifrado TLS para los servicios. Sin embargo, es importante tener en cuenta las posibles implicaciones de seguridad. Es posible que otros roles de IAM que tengan acceso de lectura a Secrets Manager puedan acceder a estos secretos creados automáticamente. Esto podría exponer el material criptográfico confidencial a terceros no autorizados si los controles de acceso no están configurados correctamente. Para mitigar este riesgo, siga estas prácticas recomendadas:
>
> - Administre y restrinja con cautela el acceso a Secrets Manager, sobre todo a los secretos creados por Service Connect.
> - Audite periódicamente los roles de IAM y sus permisos para garantizar que se mantenga el principio del privilegio mínimo.

Al conceder acceso de lectura a Secrets Manager, considere la posibilidad de excluir las claves privadas TLS creadas por Service Connect. Para ello, utilice una condición en las políticas de IAM para excluir los secretos con ARN que coincidan con el patrón:

```
"arn:aws:secretsmanager:::secret:ecs-sc!"
```

Un ejemplo de política de IAM que niega la acción `GetSecretValue` a todos los secretos con el prefijo `ecs-sc!`:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "arn:aws:secretsmanager:*:*:secret:ecs-sc!*"
        }
    ]
}
```

> **Note**
>
> Este es un ejemplo general y puede que sea necesario ajustarlo en función del caso de uso específico y de la configuración de la cuenta de AWS. Pruebe siempre minuciosamente las políticas de IAM para garantizar que proporcionan el acceso previsto y, al mismo tiempo, mantienen la seguridad.

Al comprender cómo interactúa Service Connect con Secrets Manager, podrá administrar mejor la seguridad de los servicios de Amazon ECS y, al mismo tiempo, aprovechar las ventajas del cifrado TLS automático.

Service Connect y AWS Key Management Service

Puede usar AWS Key Management Service para cifrar y descifrar los recursos de Service Connect. AWS KMS es un servicio administrado por AWS con el que puede crear y administrar claves criptográficas que protejan sus datos.

Al usar AWS KMS con Service Connect, puede elegir usar una clave AWS propia que AWS administre por usted o puede elegir una clave AWS KMS existente. También puede crear una clave de AWS KMS nueva para utilizarla.

Proporcionar su propia clave de cifrado

Puede proporcionar sus propios materiales clave o puede utilizar un almacén de claves externo mediante AWS Key Management Service Import your own key into AWS KMS y, a continuación, especificar el nombre de recurso de Amazon (ARN) de esa clave en Amazon ECS Service Connect.

A continuación, se muestra una política AWS KMS de ejemplo. Sustituya las *entradas del usuario* por valores propios.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "id",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/role-name"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey",
        "kms:GenerateDataKeyPair"
      ],
      "Resource": "*"
    }
  ]
}
```

Para obtener información sobre las políticas de claves, consulte Creating a key policy en la Guía para desarrolladores de AWS Key Management Service.

> **Note**
>
> Service Connect solo es compatible con claves de cifrado de AWS KMS simétricas. No puede utilizar ningún otro tipo de clave de AWS KMS para cifrar los recursos de Service Connect. Para obtener ayuda a fin de determinar si una clave AWS KMS es una clave de cifrado simétrica, consulte Identificación de claves KMS asimétricas.

Para obtener más información sobre la clave de cifrado simétrica de AWS Key Management Service, consulte Symmetric encryption AWS KMS keys en la Guía para desarrolladores de AWS Key Management Service.

Habilitación de TLS para Amazon ECS Service Connect

El cifrado de tráfico se activa al crear o actualizar un servicio de Service Connect.

Para habilitar el cifrado del tráfico de un servicio en un espacio de nombres existente mediante la AWS Management Console

1. Debe tener el rol de IAM de infraestructura. Para obtener más información sobre este rol, consulte Rol de IAM para la infraestructura de Amazon ECS.
2. Abra la consola en https://console.aws.amazon.com/ecs/v2.
3. En el panel de navegación, seleccione Namespaces (Espacios de nombres).
4. Elija el Espacio de nombres con el Servicio para el que quiera habilitar el cifrado del tráfico.
5. Elija el Servicio para el que quiere habilitar el cifrado del tráfico.
6. Elija Actualizar servicio en la esquina superior derecha y desplácese hacia abajo hasta la sección Service Connect.
7. Elija Activar el cifrado de tráfico en la información del servicio para activar el TLS.
8. En Rol de TLS de Service Connect, elija entre crear un nuevo rol de IAM de infraestructura o usar uno que ya exista.
9. En Autoridad de certificación firmante, elija una entidad de certificación existente o cree una nueva.

Para obtener más información, consulte Certificados AWS Private Certificate Authority y Service Connect.

10. En Elegir una AWS KMS key, elija una clave propia y administrada por AWS o puede elegir una clave diferente. También tiene la opción de crear una nueva.

Si quiere ver un ejemplo del uso de la AWS CLI para configurar TLS para su servicio, consulte Configuración de Amazon ECS Service Connect con la AWS CLI.

Comprobación de la habilitación de TLS para Amazon ECS Service Connect

Service Connect inicia el TLS en el agente de Service Connect y lo termina en el agente de destino. Como resultado, el código de la aplicación nunca ve las interacciones de TLS. Para comprobar que TLS esté habilitado, haga lo siguiente.

1. Incluya la CLI de `openssl` en la imagen de la aplicación.
2. Habilite ECS Exec en sus servicios para que se conecten a sus tareas a través de SSM. Como alternativa, puede lanzar una instancia de Amazon EC2 en la misma VPC de Amazon que el servicio.
3. Recupere la IP y el puerto de una tarea de un servicio que desee verificar. Puede recuperar la dirección IP de la consola de AWS Cloud Map. La información se encuentra en la página de detalles del servicio dentro del espacio de nombres.
4. Inicie sesión en cualquiera de las tareas mediante `execute-command`, tal como se muestra en el ejemplo siguiente. Como alternativa, inicie sesión en la instancia de Amazon EC2 creada en el paso 2.

```
$ aws ecs execute-command --cluster cluster-name \
    --task task-id  \
    --container container-name \
    --interactive \
    --command "/bin/sh"
```

> **Note**
>
> Al llamar directamente al nombre DNS no se revela el certificado.

5. En el shell conectado, utilice la CLI de `openssl` para comprobar y ver el certificado adjunto a la tarea.

Ejemplo:

```
openssl s_client -connect 10.0.147.43:6379 < /dev/null 2> /dev/null \
| openssl x509 -noout -text
```

Ejemplo de respuesta:

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            <serial-number>
        Signature Algorithm: ecdsa-with-SHA256
        Issuer: <issuer>
        Validity
            Not Before: Jan 23 21:38:12 2024 GMT
            Not After : Jan 30 22:38:12 2024 GMT
        Subject: <subject>
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    <pub>
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
            X509v3 Subject Alternative Name:
                DNS:redis.yelb-cftc
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Authority Key Identifier:
                keyid:<key-id>

            X509v3 Subject Key Identifier:
                1D:<id>
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication
    Signature Algorithm: ecdsa-with-SHA256
        <hash>
```

Configuración de Amazon ECS Service Connect con la AWS CLI

Puede crear un servicio de Amazon ECS para una tarea de Fargate que utilice Service Connect a través de la AWS CLI.

> **Note**
>
> Puede utilizar puntos de conexión de servicio de doble pila para interactuar con Amazon ECS desde la AWS CLI, los SDK y la API de Amazon ECS a través de IPv4 e IPv6. Para obtener más información, consulte Uso de puntos de conexión de doble pila en Amazon ECS.

Requisitos previos

A continuación, se indican los requisitos previos de Service Connect:

- Compruebe que la última versión de la AWS CLI esté instalada y configurada. Para obtener más información, consulte Instalación o actualización de la versión más reciente de la AWS CLI.
- Su usuario de IAM dispone de los permisos requeridos que se especifican en la política de IAM AmazonECS_FullAccess de ejemplo.
- Tiene una VPC, una subred, una tabla de enrutamiento y un grupo de seguridad creados para utilizarlos. Para obtener más información, consulte the section called “Creación de una nube virtual privada”.
- Tiene un rol de ejecución de tareas con el nombre `ecsTaskExecutionRole` y la política administrada `AmazonECSTaskExecutionRolePolicy` está asociada al rol. Este rol permite a Fargate escribir los registros de aplicaciones de NGINX y los registros de proxy de Service Connect en los registros de Amazon CloudWatch. Para obtener más información, consulte Creación del rol de de ejecución de tareas.

Paso 1: Crear el clúster

Siga los pasos a continuación para crear el clúster y el espacio de nombres de Amazon ECS.

Para crear un clúster de Amazon ECS y espacio de nombres de AWS Cloud Map

1. Cree un clúster de Amazon ECS con el nombre `tutorial` para su utilización. El parámetro `--service-connect-defaults` establece el espacio de nombres predeterminado del clúster. En el resultado del ejemplo, no existe un espacio de nombres de AWS Cloud Map con

nombre `service-connect` en esta cuenta y Región de AWS, por lo tanto, Amazon ECS crea el espacio de nombres. El espacio de nombres se crea en AWS Cloud Map en la cuenta y es visible con todos los demás espacios de nombres, así que debe usar un nombre que indique el propósito.

```
aws ecs create-cluster --cluster-name tutorial --service-connect-defaults
 namespace=service-connect
```

Salida:

```
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
        "clusterName": "tutorial",
        "serviceConnectDefaults": {
            "namespace": "arn:aws:servicediscovery:us-
west-2:123456789012:namespace/ns-EXAMPLE"
        },
        "status": "PROVISIONING",
        "registeredContainerInstancesCount": 0,
        "runningTasksCount": 0,
        "pendingTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [],
        "settings": [
            {
                "name": "containerInsights",
                "value": "disabled"
            }
        ],
        "capacityProviders": [],
        "defaultCapacityProviderStrategy": [],
        "attachments": [
            {
                "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                "type": "sc",
                "status": "ATTACHING",
                "details": []
            }
        ],
        "attachmentsStatus": "UPDATE_IN_PROGRESS"
```

```
    }
}
}
```

2. Compruebe que se haya creado el clúster:

```
aws ecs describe-clusters --clusters tutorial
```

Salida:

```
{
    "clusters": [
        {
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
            "clusterName": "tutorial",
            "serviceConnectDefaults": {
                "namespace": "arn:aws:servicediscovery:us-
west-2:123456789012:namespace/ns-EXAMPLE"
            },
            "status": "ACTIVE",
            "registeredContainerInstancesCount": 0,
            "runningTasksCount": 0,
            "pendingTasksCount": 0,
            "activeServicesCount": 0,
            "statistics": [],
            "tags": [],
            "settings": [],
            "capacityProviders": [],
            "defaultCapacityProviderStrategy": []
        }
    ],
    "failures": []
}
```

3. (Opcional) Compruebe que el espacio de nombres se haya creado en AWS Cloud Map. Puede usar la AWS Management Console o la configuración normal de la AWS CLI, ya que esto se crea en AWS Cloud Map.

   Por ejemplo, use la AWS CLI:

```
aws servicediscovery get-namespace --id ns-EXAMPLE
```

Salida:

```
{
    "Namespace": {
        "Id": "ns-EXAMPLE",
        "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
EXAMPLE",
        "Name": "service-connect",
        "Type": "HTTP",
        "Properties": {
            "DnsProperties": {
                "SOA": {}
            },
            "HttpProperties": {
                "HttpName": "service-connect"
            }
        },
        "CreateDate": 1661749852.422,
        "CreatorRequestId": "service-connect"
    }
}
```

Paso 2: crear el servicio para el servidor

La característica de Service Connect está diseñada para interconectar varias aplicaciones en Amazon ECS. Al menos una de esas aplicaciones debe proporcionar un servicio web al cual conectarse. En este paso, creará:

- La definición de tarea que utiliza la imagen del contenedor oficial de NGINX sin modificar e incluye la configuración de Service Connect.
- La definición de servicio de Amazon ECS que configura Service Connect para proporcionar la detección de servicios y el proxy de malla de servicios para el tráfico a este servicio. La configuración reutiliza el espacio de nombres predeterminado de la configuración del clúster para reducir la cantidad de configuración que se realiza para cada servicio.
- El servicio de Amazon ECS. Ejecuta una tarea mediante la definición de tarea e inserta un contenedor adicional para el proxy de Service Connect. El proxy escucha en el puerto desde la asignación de puertos del contenedor de la definición de la tarea. En una aplicación de cliente que se ejecuta en Amazon ECS, el proxy de la tarea del cliente escucha las conexiones salientes al

nombre del puerto de definición de la tarea, el nombre de detección de servicios o el nombre de alias del cliente de servicio y el número de puerto del alias de cliente.

Para crear el servicio web con Service Connect

1. Registre una definición de tarea que sea compatible con Fargate y utilice el `awsvpc` en modo de red. Siga estos pasos:

   a. Cree un archivo denominado `service-connect-nginx.json` con los contenidos de la siguiente definición de tareas.

      Esta definición de tarea configura Service Connect al agregar los parámetros `name` y `appProtocol` a la asignación de puertos. El nombre del puerto hace que este sea más identificable en la configuración del servicio cuando se utilizan varios puertos. El nombre del puerto también se usa de forma predeterminada como nombre detectable para que lo usen otras aplicaciones en el espacio de nombres.

      La definición de la tarea contiene el rol de IAM de la tarea porque el servicio tiene habilitado ECS Exec.

      > **Important**
      >
      > Esta definición de tarea usa una `logConfiguration` para enviar la salida de nginx desde `stdout` y `stderr` hacia los registros de Amazon CloudWatch. Este rol de ejecución de tareas no tiene los permisos adicionales necesarios para crear el grupo de registro de registros de CloudWatch. Cree el grupo de registro de registros de CloudWatch mediante la AWS Management Console o la AWS CLI. Si no desea enviar los registros de nginx a registros de CloudWatch, puede eliminar la `logConfiguration`.
      > Sustituya el ID de la Cuenta de AWS en el rol de ejecución de la tarea por el ID de su Cuenta de AWS.

```
{
    "family": "service-connect-nginx",
    "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam::123456789012:role/ecsTaskRole",
```

```
    "networkMode": "awsvpc",
    "containerDefinitions": [
        {
        "name": "webserver",
        "image": "public.ecr.aws/docker/library/nginx:latest",
        "cpu": 100,
        "portMappings": [
            {
                "name": "nginx",
                "containerPort": 80,
                "protocol": "tcp",
                "appProtocol": "http"
            }
        ],
        "essential": true,
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-nginx",
                "awslogs-region": "region",
                "awslogs-stream-prefix": "nginx"
            }
        }
        }
    ],
    "cpu": "256",
    "memory": "512"
}
```

b. Registre la definición de tareas mediante el archivo `service-connect-nginx.json`:

```
aws ecs register-task-definition --cli-input-json file://service-connect-
nginx.json
```

2. Cree un servicio:

   a. Cree un archivo llamado `service-connect-nginx-service.json` con el contenido del servicio de Amazon ECS que va a crear. En este ejemplo se utiliza la definición de tareas creada en el paso anterior. Es necesario un `awsvpcConfiguration` porque el ejemplo de definición de tareas utiliza el modo de red `awsvpc`.

Cuando cree el servicio de ECS, especifique el tipo de lanzamiento de Fargate y la versión `LATEST` de la plataforma que admite Service Connect. Los `securityGroups` y las `subnets` deben pertenecer a una VPC que cumpla los requisitos de uso de Amazon ECS. Puede obtener los ID de subred y grupos de seguridad en la consola de Amazon VPC.

Este servicio configura Service Connect al agregar el parámetro `serviceConnectConfiguration`. El espacio de nombres no es obligatorio porque el clúster tiene configurado un espacio de nombres predeterminado. Las aplicaciones de cliente que se ejecutan en ECS en el espacio de nombres se conectan a este servicio mediante `portName` y el puerto de `clientAliases`. Por ejemplo, se puede acceder a este servicio con `http://nginx:80/`, ya que nginx proporciona una página de bienvenida en la ubicación raíz `/`. Las aplicaciones externas que no se ejecutan en Amazon ECS o que no están en el mismo espacio de nombres pueden acceder a esta aplicación a través del proxy de Service Connect mediante la dirección IP de la tarea y el número de puerto de la definición de la tarea. Para su configuración de `tls`, agregue el certificado `arn` para su `awsPcaAuthorityArn` su `kmsKey`, y `roleArn` de su rol de IAM.

Este servicio utiliza una `logConfiguration` para enviar la salida del proxy de Service Connect desde `stdout` y `stderr` hacia los registros de Amazon CloudWatch. Este rol de ejecución de tareas no tiene los permisos adicionales necesarios para crear el grupo de registro de registros de CloudWatch. Cree el grupo de registro de registros de CloudWatch mediante la AWS Management Console o la AWS CLI. Le recomendamos que cree este grupo de registro y almacene los registros de proxy en registros de CloudWatch. Si no desea enviar los registros de proxy a registros de CloudWatch, puede eliminar la `logConfiguration`.

```
{
    "cluster": "tutorial",
    "deploymentConfiguration": {
        "maximumPercent": 200,
        "minimumHealthyPercent": 0
    },
    "deploymentController": {
        "type": "ECS"
    },
    "desiredCount": 1,
    "enableECSManagedTags": true,
    "enableExecuteCommand": true,
```

```
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "assignPublicIp": "ENABLED",
            "securityGroups": [
                "sg-EXAMPLE"
            ],
            "subnets": [
                "subnet-EXAMPLE",
                "subnet-EXAMPLE",
                "subnet-EXAMPLE"
            ]
           }
    },
    "platformVersion": "LATEST",
    "propagateTags": "SERVICE",
    "serviceName": "service-connect-nginx-service",
    "serviceConnectConfiguration": {
        "enabled": true,
        "services": [
            {
                "portName": "nginx",
                "clientAliases": [
                    {
                        "port": 80
                    }
                ],
                "tls": {
                   "issuerCertificateAuthority": {
                      "awsPcaAuthorityArn": "certificateArn"
                   },
                   "kmsKey": "kmsKey",
                   "roleArn": "iamRoleArn"
                }
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "region",
                "awslogs-stream-prefix": "service-connect-proxy"
            }
        }
```

```
    },
    "taskDefinition": "service-connect-nginx"
}
```

b. Cree un servicio mediante el archivo `service-connect-nginx-service.json`:

```
aws ecs create-service --cluster tutorial --cli-input-json file://service-
connect-nginx-service.json
```

Salida:

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/tutorial/
service-connect-nginx-service",
        "serviceName": "service-connect-nginx-service",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
        "loadBalancers": [],
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 1,
        "runningCount": 0,
        "pendingCount": 0,
        "launchType": "FARGATE",
        "platformVersion": "LATEST",
        "platformFamily": "Linux",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
service-connect-nginx:1",
        "deploymentConfiguration": {
            "deploymentCircuitBreaker": {
                "enable": false,
                "rollback": false
            },
            "maximumPercent": 200,
            "minimumHealthyPercent": 0
        },
        "deployments": [
            {
                "id": "ecs-svc/3763308422771520962",
                "status": "PRIMARY",
```

```
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/service-connect-nginx:1",
                "desiredCount": 1,
                "pendingCount": 0,
                "runningCount": 0,
                "failedTasks": 0,
                "createdAt": 1661210032.602,
                "updatedAt": 1661210032.602,
                "launchType": "FARGATE",
                "platformVersion": "1.4.0",
                "platformFamily": "Linux",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "assignPublicIp": "ENABLED",
                        "securityGroups": [
                            "sg-EXAMPLE"
                        ],
                        "subnets": [
                            "subnet-EXAMPLEf",
                            "subnet-EXAMPLE",
                            "subnet-EXAMPLE"
                        ]
                    }
                },
                "rolloutState": "IN_PROGRESS",
                "rolloutStateReason": "ECS deployment ecs-
svc/3763308422771520962 in progress.",
                "failedLaunchTaskCount": 0,
                "replacedTaskCount": 0,
                "serviceConnectConfiguration": {
                    "enabled": true,
                    "namespace": "service-connect",
                    "services": [
                        {
                            "portName": "nginx",
                            "clientAliases": [
                                {
                                    "port": 80
                                }
                            ]
                        }
                    ],
                    "logConfiguration": {
                        "logDriver": "awslogs",
```

```
                        "options": {
                            "awslogs-group": "/ecs/service-connect-proxy",
                            "awslogs-region": "us-west-2",
                            "awslogs-stream-prefix": "service-connect-proxy"
                        },
                        "secretOptions": []
                    }
                },
                "serviceConnectResources": [
                    {
                        "discoveryName": "nginx",
                        "discoveryArn": "arn:aws:servicediscovery:us-
west-2:123456789012:service/srv-EXAMPLE"
                    }
                ]
            }
        ],
        "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
        "version": 0,
        "events": [],
        "createdAt": 1661210032.602,
        "placementConstraints": [],
        "placementStrategy": [],
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "assignPublicIp": "ENABLED",
                "securityGroups": [
                    "sg-EXAMPLE"
                ],
                "subnets": [
                    "subnet-EXAMPLE",
                    "subnet-EXAMPLE",
                    "subnet-EXAMPLE"
                ]
            }
        },
        "schedulingStrategy": "REPLICA",
        "enableECSManagedTags": true,
        "propagateTags": "SERVICE",
        "enableExecuteCommand": true
    }
}
```

La `serviceConnectConfiguration` que ha proporcionado aparece dentro de la primera implementación de la salida. A medida que hace cambios en el servicio de ECS de manera que es necesario realizar cambios en las tareas, Amazon ECS crea una nueva implementación.

Paso 3: Comprobar que puede conectarse

Para comprobar que Service Connect está configurado y funciona, siga estos pasos para conectarse al servicio web desde una aplicación externa. A continuación, consulte las métricas adicionales en CloudWatch que crea el proxy de Service Connect.

Para conectarse al servicio web desde una aplicación externa

- Conéctese a la dirección IP de la tarea y al puerto del contenedor mediante la dirección IP de la tarea

  Utilice la AWS CLI para obtener el ID de la tarea, mediante el `aws ecs list-tasks --cluster tutorial`.

  Si las subredes y el grupo de seguridad permiten el tráfico de la Internet pública en el puerto de la definición de la tarea, puede conectarse a la IP pública desde su equipo. Sin embargo, la IP pública no está disponible en “describe-tasks”, por lo que los pasos incluyen ir a la AWS Management Console o la AWS CLI de Amazon EC2 para obtener los detalles de la interfaz de red elástica.

  En este ejemplo, una instancia de Amazon EC2 de la misma VPC usa la IP privada de la tarea. La aplicación es nginx, pero el encabezado `server: envoy` muestra que se utiliza el proxy de Service Connect. El proxy de Service Connect escucha el puerto del contenedor de la definición de la tarea.

```
$ curl -v 10.0.19.50:80/
*   Trying 10.0.19.50:80...
* Connected to 10.0.19.50 (10.0.19.50) port 80 (#0)
> GET / HTTP/1.1
> Host: 10.0.19.50
> User-Agent: curl/7.79.1
> Accept: */*
```

```
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< server: envoy
< date: Tue, 23 Aug 2022 03:53:06 GMT
< content-type: text/html
< content-length: 612
< last-modified: Tue, 16 Apr 2019 13:08:19 GMT
< etag: "5cb5d3c3-264"
< accept-ranges: bytes
< x-envoy-upstream-service-time: 0
<
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

Para ver las métricas de Service Connect

El proxy de Service Connect crea métricas de aplicaciones (conexión HTTP, HTTP2, gRPC o TCP) en las métricas de CloudWatch. Cuando utilice la consola de CloudWatch, consulte las dimensiones de métricas adicionales de DiscoveryName, (DiscoveryName, ServiceName, ClusterName), TargetDiscoveryName y (TargetDiscoveryName, ServiceName, ClusterName) en el espacio de nombres de Amazon ECS. Para obtener más información acerca de estas métricas y sus dimensiones, consulte View Available Metrics en la Guía del usuario de Registros de Amazon CloudWatch.

Uso de la detección de servicios para conectar los servicios de Amazon ECS con nombres de DNS

Su servicio de Amazon ECS también puede configurarse para utilizar la detección de servicios de Amazon ECS. La detección de servicios utiliza acciones de la API de AWS Cloud Map para administrar los espacios de nombres de DNS y HTTP para sus servicios de Amazon ECS. Para obtener más información, consulte ¿Qué es AWS Cloud Map? en la Guía para desarrolladores de AWS Cloud Map.

La detección de servicios se encuentra disponible en las siguientes regiones de AWS:

Nombre de la región	Región
Este de EE. UU. (Norte de Virginia)	us-east-1
Este de EE. UU. (Ohio)	us-east-2
Oeste de EE. UU. (Norte de California)	us-west-1
Oeste de EE. UU. (Oregón)	us-west-2
África (Ciudad del Cabo)	af-south-1
Asia-Pacífico (Hong Kong)	ap-east-1
Asia-Pacífico (Bombay)	ap-south-1
Asia-Pacífico (Hyderabad)	ap-south-2
Asia-Pacífico (Tokio)	ap-northeast-1
Asia-Pacífico (Seúl)	ap-northeast-2

Nombre de la región	Región
Asia-Pacífico (Osaka)	ap-northeast-3
Asia-Pacífico (Singapur)	ap-southeast-1
Asia-Pacífico (Sídney)	ap-southeast-2
Asia-Pacífico (Yakarta)	ap-southeast-3
Asia-Pacífico (Melbourne)	ap-southeast-4
Canadá (centro)	ca-central-1
Oeste de Canadá (Calgary)	ca-west-1
China (Pekín)	cn-north-1
China (Ningxia)	cn-northwest-1
Europe (Frankfurt)	eu-central-1
Europa (Zúrich)	eu-central-2
Europa (Irlanda)	eu-west-1
Europa (Londres)	eu-west-2
Europa (París)	eu-west-3
Europa (Milán)	eu-south-1
Europa (Estocolmo)	eu-north-1
Israel (Tel Aviv)	il-central-1
Europa (España)	eu-south-2
Medio Oriente (EAU)	me-central-1
Medio Oriente (Baréin)	me-south-1

Nombre de la región	Región
América del Sur (São Paulo)	sa-east-1
AWS GovCloud (Este de EE. UU.)	us-gov-east-1
AWS GovCloud (Oeste de EE.UU.)	us-gov-west-1

Conceptos sobre la detección de servicios

La detección de servicios consta de los siguientes componentes:

- Espacio de nombres de detección de servicios: grupo lógico de servicios de detección de servicios que comparten el mismo nombre de dominio, como `example.com`, que es adónde quiere dirigir el tráfico. Puede crear un espacio de nombres con una llamada al comando `aws servicediscovery create-private-dns-namespace` o en la consola de Amazon ECS. Puede utilizar el comando `aws servicediscovery list-namespaces` para ver la información de resumen de los espacios de nombres creados por la cuenta corriente. Para obtener más información acerca de los comandos de detección de servicios, consulte `create-private-dns-namespace` y `list-namespaces` en la Guía de referencia de AWS Cloud Map (detección de servicios) de la AWS CLI.
- Service discovery service (Servicio de detección de servicios): existe dentro del espacio de nombres de detección de servicios y consta del nombre del servicio y la configuración de DNS para el espacio de nombres. Proporciona los siguientes componentes principales:
  - Registro de servicios: permite buscar un servicio mediante DNS o con las acciones de la API de AWS Cloud Map y recuperar uno o varios puntos de enlace disponibles que se pueden utilizar para conectarse al servicio.
- Service discovery instance (Instancia de detección de servicios): existe dentro del servicio de detección de servicios y consiste en atributos asociados a cada servicio de Amazon ECS del directorio de servicios.
  - Instance attributes (Atributos de instancia): se agregan los siguientes metadatos como atributos personalizados para cada servicio de Amazon ECS que se configura para utilizar la detección de servicios:
    - **`AWS_INSTANCE_IPV4`**: en el caso de un registro `A`, la dirección IPv4 que Route 53 devuelve en respuesta a consultas de DNS y AWS Cloud Map devuelve al detectar detalles de la instancia, por ejemplo, `192.0.2.44`.

  - **`AWS_INSTANCE_PORT`**: el valor del puerto asociado al servicio de detección de servicios.
  - **`AVAILABILITY_ZONE`**: la zona de disponibilidad en la que se lanzó la tarea. Para tareas que utilizan el tipo de lanzamiento de EC2, esta es la zona de disponibilidad en la que existe la instancia de contenedor. Para tareas que utilizan el tipo de lanzamiento de Fargate, esta es la zona de disponibilidad en la que existe la interfaz de red elástica.
  - **`REGION`**: la región en la que existe la tarea.
  - **`ECS_SERVICE_NAME`**: el nombre del servicio de Amazon ECS al que pertenece la tarea.
  - **`ECS_CLUSTER_NAME`**: el nombre del clúster de Amazon ECS al que pertenece la tarea.
  - **`EC2_INSTANCE_ID`**: el ID de la instancia de contenedor en el que se colocó la tarea. Este atributo personalizado no se agrega si la tarea utiliza el tipo de lanzamiento de Fargate.
  - **`ECS_TASK_DEFINITION_FAMILY`**: la familia de definición de tareas que utiliza la tarea.
  - **`ECS_TASK_SET_EXTERNAL_ID`**: si se crea un conjunto de tareas para una implementación externa y se asocia a un registro de detección de servicios, el atributo `ECS_TASK_SET_EXTERNAL_ID` contendrá el ID externo del conjunto de tareas.
- Amazon ECS health checks (Comprobaciones de estado de Amazon ECS): Amazon ECS realiza comprobaciones de estado periódicas en el nivel de contenedor. Si un punto de enlace no supera la comprobación de estado, se elimina del direccionamiento de DNS y se marca como en mal estado.

Consideraciones sobre la detección de servicios

Al utilizar la detección de servicios, se debe tener en cuenta lo siguiente:

- La detección de servicios es compatible con tareas alojadas en Fargate que utilizan la versión 1.1.0 de la plataforma o una posterior. Para obtener más información, consulte Versiones de la plataforma Fargate para Amazon ECS.
- Los servicios configurados para utilizar la detección de servicios tienen un límite de 1000 tareas por servicio. Esto se debe a una cuota de servicio de Route 53.
- El flujo de trabajo de creación de servicio en la consola de Amazon ECS solo es admite el registro de servicios en espacios de nombres de DNS privado. Cuando se crea un espacio de nombres de DNS privado de AWS Cloud Map, se creará automáticamente una zona alojada privada de Route 53.

- Los atributos de DNS de la VPC deben estar configurados para una resolución de DNS correcta. Para obtener información acerca de cómo configurar los atributos, consulte Compatibilidad de la VPC con DNS en la Guía del usuario de Amazon VPC.
- Los registros de DNS creados para un servicio de detección de servicios se registran siempre con la dirección IP privada de la tarea, en lugar de la dirección IP pública, incluso cuando se utilizan espacios de nombres públicos.
- La detección de servicios requiere que las tareas especifiquen el modo de red `awsvpc`, `bridge` o `host` (`none` no se admite).
- Si la definición de tarea de servicio usa el modo de red `awsvpc`, puede crear cualquier combinación de registros `A` o SRV para cada tarea de servicio. Si utiliza los registros SRV, se necesita un puerto.
- Si la definición de tarea de servicio usa el modo de red `bridge` o `host`, el único tipo de registro de DNS admitido es un registro SRV. Cree un registro SRV para cada tarea de servicio. El registro SRV debe especificar una combinación de nombre y puerto de contenedor en la definición de tarea.
- Los registros de DNS de un servicio de detección de servicios se pueden consultar dentro de la VPC. Utilizan el siguiente formato `<service-discovery-service-name>.<service-discovery-namespace>`.
- Al realizar una consulta de DNS en el nombre del servicio, los registros `A` devuelven un conjunto de direcciones IP que corresponden a las tareas. Los registros SRV devuelven un conjunto de direcciones IP y puertos por cada tarea.
- Si tiene ocho registros o menos en buen estado, Route 53 responde a todas las consultas de DNS con todos los registros en buen estado.
- Cuando todos los registros están en mal estado, Route 53 responde a las consultas de DNS con hasta ocho registros en mal estado.
- Puede configurar la detección de servicios para un servicio que se encuentre detrás de un equilibrador de carga, pero el tráfico de la detección de servicios siempre se dirige a la tarea, no al equilibrador de carga.
- La detección de servicios no admite el uso del equilibrador de carga clásico.
- Se recomienda utilizar las comprobaciones de estado de nivel de contenedor administradas por Amazon ECS para el servicio de detección de servicios.
  - HealthCheckCustomConfig: Amazon ECS administra las comprobaciones de estado en su nombre. Amazon ECS utiliza información del contenedor y de las comprobaciones de estado, además del estado de la tarea, para actualizar el estado mediante AWS Cloud Map.

Esto se especifica a través del parámetro `--health-check-custom-config` cuando se crea el servicio de detección de servicios. Para obtener más información, consulte HealthCheckCustomConfig en la Referencia de la API de AWS Cloud Map.

- Los recursos de AWS Cloud Map que se crean cuando se utiliza la detección de servicios se deben limpiar manualmente.
- Las tareas y las instancias se registran como `UNHEALTHY` hasta que las comprobaciones de estado del contenedor devuelvan un valor. Si se pasan las comprobaciones de estado, el estado se actualiza a `HEALTHY`. Si las comprobaciones de estado del contenedor fallan, se anula el registro de la instancia de detección de servicios.

Precios de la detección de servicios

Los clientes que utilizan la detección de servicios de Amazon ECS deben pagar cargos por los recursos de Route 53 y las operaciones de la API de detección de AWS Cloud Map. Se cobran costos por crear zonas alojadas en Route 53 y por las consultas al registro de servicios. Para obtener más información, consulte Precios de AWS Cloud Map en la Guía para desarrolladores de AWS Cloud Map.

Amazon ECS realiza comprobaciones de estado en el nivel de contenedor y las expone a operaciones de la API de comprobación de estado personalizada de AWS Cloud Map. Actualmente, esto está a disposición de los clientes sin ningún costo adicional. Si configura las comprobaciones de estado de red adicionales para tareas expuestas de forma pública, se le cobrará por dichas comprobaciones de estado.

Creación de un servicio de Amazon ECS que utilice la detección de servicios

Obtenga información sobre cómo crear un servicio que contenga una tarea de Fargate que utilice la detección de servicios a través de la AWS CLI.

Para obtener una lista de Regiones de AWS que admiten la detección de servicios, consulte Uso de la detección de servicios para conectar los servicios de Amazon ECS con nombres de DNS.

Para obtener información acerca de las regiones que admiten Fargate, consulte the section called “Regiones de AWS Fargate”.

> **Note**
>
> Puede utilizar puntos de conexión de servicio de doble pila para interactuar con Amazon ECS desde la AWS CLI, los SDK y la API de Amazon ECS a través de IPv4 e IPv6. Para obtener más información, consulte Uso de puntos de conexión de doble pila en Amazon ECS.

Requisitos previos

Antes de empezar este tutorial, asegúrese de que se cumplen los siguientes requisitos previos:

- La última versión de la AWS CLI está instalada y configurada. Para obtener más información, consulte Instalación o actualización de la versión más reciente de la AWS CLI.
- Los pasos descritos en Configuración para utilizar Amazon ECS están completos.
- Su usuario de IAM dispone de los permisos requeridos que se especifican en la política de IAM AmazonECS_FullAccess de ejemplo.
- Ha creado al menos una VPC y un grupo de seguridad. Para obtener más información, consulte the section called “Creación de una nube virtual privada”.

Paso 1: crear los recursos para la detección de servicios en AWS Cloud Map

Siga estos pasos para crear el espacio de nombres para la detección de servicios y el servicio de detección de servicios:

1. Cree un espacio de nombres de detección de servicios de Cloud Map privado. Este ejemplo crea un espacio de nombres denominado `tutorial`. Reemplace *`vpc-abcd1234`* con el ID de una de las VPC existentes.

   ```
   aws servicediscovery create-private-dns-namespace \
         --name tutorial \
         --vpc vpc-abcd1234
   ```

   A continuación se muestra la salida de este comando.

   ```
   {
       "OperationId": "h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e"
   }
   ```

2. Mediante el `OperationId` de la salida del paso anterior, compruebe que el espacio de nombres privado se haya creado correctamente. Anote el ID del espacio de nombres porque lo utilizará en los comandos posteriores.

```
aws servicediscovery get-operation \
      --operation-id h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e
```

El resultado es el siguiente.

```
{
    "Operation": {
        "Id": "h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e",
        "Type": "CREATE_NAMESPACE",
        "Status": "SUCCESS",
        "CreateDate": 1519777852.502,
        "UpdateDate": 1519777856.086,
        "Targets": {
           "NAMESPACE": "ns-uejictsjen2i4eeg"
        }
    }
}
```

3. Mediante el ID de `NAMESPACE` de la salida del paso anterior, cree un servicio de detección de servicios. En este ejemplo, se crea un servicio denominado `myapplication`. Anote el ID de servicio y el ARN porque los utilizará en comandos posteriores.

```
aws servicediscovery create-service \
      --name myapplication \
      --dns-config "NamespaceId="ns-
uejictsjen2i4eeg",DnsRecords=[{Type="A",TTL="300"}]" \
      --health-check-custom-config FailureThreshold=1
```

El resultado es el siguiente.

```
{
    "Service": {
       "Id": "srv-utcrh6wavdkggqtk",
        "Arn": "arn:aws:servicediscovery:region:aws_account_id:service/srv-
utcrh6wavdkggqtk",
        "Name": "myapplication",
        "DnsConfig": {
```

```
            "NamespaceId": "ns-uejictsjen2i4eeg",
            "DnsRecords": [
                {
                    "Type": "A",
                    "TTL": 300
                }
            ]
        },
        "HealthCheckCustomConfig": {
            "FailureThreshold": 1
        },
        "CreatorRequestId": "e49a8797-b735-481b-a657-b74d1d6734eb"
    }
}
```

Paso 2: Crear los recursos de Amazon ECS

Siga estos pasos para crear el clúster, la definición de tareas y el servicio de Amazon ECS:

1. Cree un clúster de Amazon ECS. Este ejemplo crea un clúster denominado `tutorial`.

```
aws ecs create-cluster \
      --cluster-name tutorial
```

2. Registre una definición de tarea que sea compatible con Fargate y utilice el `awsvpc` en modo de red. Siga estos pasos:

   a. Cree un archivo denominado `fargate-task.json` con los contenidos de la siguiente definición de tareas.

```
{
    "family": "tutorial-task-def",
        "networkMode": "awsvpc",
        "containerDefinitions": [
            {
                "name": "sample-app",
                "image": "public.ecr.aws/docker/library/httpd:2.4",
                "portMappings": [
                    {
                        "containerPort": 80,
                        "hostPort": 80,
                        "protocol": "tcp"
```

```
                    }
                ],
                "essential": true,
                "entryPoint": [
                    "sh",
                    "-c"
                ],
                "command": [
                    "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style>
 </head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample
 App</h1> <h2>Congratulations!</h2> <p>Your application is now running on a
 container in Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/
htdocs/index.html && httpd-foreground\""
                ]
            }
        ],
        "requiresCompatibilities": [
            "FARGATE"
        ],
        "cpu": "256",
        "memory": "512"
}
```

b. Registre la definición de tareas mediante `fargate-task.json`.

```
aws ecs register-task-definition \
      --cli-input-json file://fargate-task.json
```

3. Cree un servicio de ECS siguiendo estos pasos:

   a. Cree un archivo llamado `ecs-service-discovery.json` con el contenido del servicio de ECS que va a crear. En este ejemplo se utiliza la definición de tareas creada en el paso anterior. Es necesario un `awsvpcConfiguration` porque el ejemplo de definición de tareas utiliza el modo de red `awsvpc`.

      Cree el servicio de ECS, especifique el tipo de lanzamiento de Fargate y la versión `LATEST` de la plataforma, que admite la detección de servicios. Cuando se crea el servicio de detección de servicios en AWS Cloud Map, `registryArn` es el ARN devuelto. Los `securityGroups` y las `subnets` deben pertenecer a la VPC que se usa para crear el espacio de nombres de Cloud Map. Puede obtener los ID de subred y grupos de seguridad en la consola de Amazon VPC.

```
{
    "cluster": "tutorial",
    "serviceName": "ecs-service-discovery",
    "taskDefinition": "tutorial-task-def",
    "serviceRegistries": [
       {
          "registryArn":
 "arn:aws:servicediscovery:region:aws_account_id:service/srv-utcrh6wavdkggqtk"
       }
    ],
    "launchType": "FARGATE",
    "platformVersion": "LATEST",
    "networkConfiguration": {
       "awsvpcConfiguration": {
          "assignPublicIp": "ENABLED",
          "securityGroups": [ "sg-abcd1234" ],
          "subnets": [ "subnet-abcd1234" ]
       }
    },
    "desiredCount": 1
}
```

b. Cree el servicio de ECS mediante `ecs-service-discovery.json`.

```
aws ecs create-service \
      --cli-input-json file://ecs-service-discovery.json
```

Paso 3: verificar la detección de servicios en AWS Cloud Map

Puede comprobar que todo se haya creado de forma correcta al consultar la información de detección de servicios. Una vez configurada la detección de servicios, puede usar las operaciones de la API de AWS Cloud Map o llamar a `dig` desde una instancia de la VPC. Siga estos pasos:

1. Mediante el ID del servicio de detección de servicios, enumere las instancias de detección de servicios. Anote el ID de la instancia (marcado en negrita) para la limpieza de recursos.

```
aws servicediscovery list-instances \
      --service-id srv-utcrh6wavdkggqtk
```

El resultado es el siguiente.

```
{
    "Instances": [
        {
            "Id": "16becc26-8558-4af1-9fbd-f81be062a266",
            "Attributes": {
                "AWS_INSTANCE_IPV4": "172.31.87.2"
                "AWS_INSTANCE_PORT": "80",
                "AVAILABILITY_ZONE": "us-east-1a",
                "REGION": "us-east-1",
                "ECS_SERVICE_NAME": "ecs-service-discovery",
                "ECS_CLUSTER_NAME": "tutorial",
                "ECS_TASK_DEFINITION_FAMILY": "tutorial-task-def"
            }
        }
    ]
}
```

2. Utilice el espacio de nombres de detección de servicios, el servicio y los parámetros adicionales, como el nombre del clúster ECS, para consultar los detalles de las instancias de detección de servicios.

```
aws servicediscovery discover-instances \
      --namespace-name tutorial \
      --service-name myapplication \
      --query-parameters ECS_CLUSTER_NAME=tutorial
```

3. Los registros de DNS que se crearon en la zona alojada de Route 53 para el servicio de detección de servicios se pueden consultar mediante los siguientes comandos de la AWS CLI:

   a. Mediante el ID de espacio de nombres, obtenga información acerca del espacio de nombres, lo que incluye el ID de la zona alojada de Route 53.

   ```
   aws servicediscovery \
         get-namespace --id ns-uejictsjen2i4eeg
   ```

   El resultado es el siguiente.

   ```
   {
       "Namespace": {
           "Id": "ns-uejictsjen2i4eeg",
   ```

```
        "Arn": "arn:aws:servicediscovery:region:aws_account_id:namespace/ns-uejictsjen2i4eeg",
        "Name": "tutorial",
        "Type": "DNS_PRIVATE",
        "Properties": {
             "DnsProperties": {
                "HostedZoneId": "Z35JQ4ZFDRYPLV"
            }
        },
        "CreateDate": 1519777852.502,
        "CreatorRequestId": "9049a1d5-25e4-4115-8625-96dbda9a6093"
    }
}
```

b. Mediante el ID de la zona alojada de Route 53 del paso anterior, obtenga el conjunto de registros de recursos de la zona alojada.

```
aws route53 list-resource-record-sets \
      --hosted-zone-id Z35JQ4ZFDRYPLV
```

4. También puede consultar el DNS desde una instancia de la VPC con `dig`.

```
dig +short myapplication.tutorial
```

Paso 4: Limpiar

Cuando termine este tutorial, debe limpiar los recursos asociados para evitar incurrir en cargos generados por recursos sin utilizar. Siga estos pasos:

1. Anule el registro de las instancias del servicio de descubrimiento de servicios con el ID de servicio y el ID de instancia que anotó con anterioridad.

```
aws servicediscovery deregister-instance \
      --service-id srv-utcrh6wavdkggqtk \
      --instance-id 16becc26-8558-4af1-9fbd-f81be062a266
```

El resultado es el siguiente.

```
{
    "OperationId": "xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv"
```

```
}
```

2. Con el `OperationId` del resultado del paso anterior, verifique que las instancias de servicio de detección de servicios fueron dadas de baja con éxito.

```
aws servicediscovery get-operation \
      --operation-id xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv
```

```
{
  "Operation": {
        "Id": "xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv",
        "Type": "DEREGISTER_INSTANCE",
        "Status": "SUCCESS",
        "CreateDate": 1525984073.707,
        "UpdateDate": 1525984076.426,
        "Targets": {
            "INSTANCE": "16becc26-8558-4af1-9fbd-f81be062a266",
            "ROUTE_53_CHANGE_ID": "C5NSRG1J4I1FH",
            "SERVICE": "srv-utcrh6wavdkggqtk"
        }
    }
}
```

3. Elimine el servicio de detección de servicios mediante el ID de servicio.

```
aws servicediscovery delete-service \
      --id srv-utcrh6wavdkggqtk
```

4. Elimine el espacio de nombres de detección de servicios mediante el ID del espacio de nombres.

```
aws servicediscovery delete-namespace \
      --id ns-uejictsjen2i4eeg
```

El resultado es el siguiente.

```
{
    "OperationId": "c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj"
}
```

5. Con el `OperationId` del resultado del paso anterior, compruebe que el espacio de nombres de la detección de servicios se haya eliminado correctamente.

```
aws servicediscovery get-operation \
      --operation-id c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj
```

El resultado es el siguiente.

```
{
    "Operation": {
        "Id": "c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj",
        "Type": "DELETE_NAMESPACE",
        "Status": "SUCCESS",
        "CreateDate": 1525984602.211,
        "UpdateDate": 1525984602.558,
        "Targets": {
            "NAMESPACE": "ns-rymlehshst7hhukh",
            "ROUTE_53_CHANGE_ID": "CJP2A2M86XW3O"
        }
    }
}
```

6. Actualice el recuento deseado para el servicio Amazon ECS para`0`. Debe hacerlo para eliminar el servicio en el siguiente paso.

```
aws ecs update-service \
      --cluster tutorial \
      --service ecs-service-discovery \
      --desired-count 0
```

7. Elimine el servicio Amazon ECS.

```
aws ecs delete-service \
      --cluster tutorial \
      --service ecs-service-discovery
```

8. Elimine el clúster de Amazon ECS.

```
aws ecs delete-cluster \
      --cluster tutorial
```

Uso de Amazon VPC Lattice para conectar, observar y proteger los servicios de Amazon ECS

Amazon VPC Lattice es un servicio de redes de aplicaciones completamente administrado que utilizan los clientes de Amazon ECS para observar, proteger y monitorear las aplicaciones creadas en servicios de computación, VPC y cuentas de AWS sin tener que modificar su código.

VPC Lattice usa grupos de destino, que son un conjunto de recursos de computación. Estos destinos ejecutan la aplicación o servicio y pueden ser instancias de Amazon EC2, direcciones IP, funciones de Lambda y equilibradores de carga de aplicación. Al asociar sus servicios de Amazon ECS a un grupo de destino de VPC Lattice, los clientes ahora pueden habilitar las tareas de Amazon ECS como destinos de IP en VPC Lattice. Amazon ECS registra automáticamente las tareas en el grupo de destino de VPC Lattice cuando se lanzan las tareas del servicio registrado.

> **Note**
>
> Cuando se utilizan cinco configuraciones de VPC Lattice, el tiempo de implementación puede ser un poco más que cuando se utilizan menos configuraciones.

Se utiliza una regla de oyente para reenviar el tráfico a un grupo de destino específico cuando se cumplen las condiciones. Un oyente comprueba las solicitudes de conexión mediante el protocolo en el puerto que haya configurado. Un servicio dirige las solicitudes a sus destinos registrados en función de las reglas que haya definido al configurar el oyente.

Amazon ECS también reemplaza automáticamente una tarea si no funciona correctamente según las comprobaciones de estado de VPC Lattice. Una vez asociados a VPC Lattice, los clientes de Amazon ECS también pueden aprovechar muchas otras características de conectividad, seguridad y observabilidad entre cómputos de VPC Lattice, como la conexión a servicios entre clústeres, VPC y cuentas con AWS Resource Access Manager, integración de IAM para autorización y autenticación y características avanzadas de administración del tráfico.

Los clientes de Amazon ECS pueden beneficiarse de VPC Lattice de las siguientes formas.

- Aumento de la productividad de los desarrolladores: VPC Lattice aumenta la productividad de los desarrolladores al permitir centrarse en la creación de características, mientras que VPC Lattice gestiona los desafíos de redes, seguridad y observabilidad de manera uniforme en todas las plataformas de computación.

- Mejor postura de seguridad: VPC Lattice permite a los desarrolladores autenticar y proteger fácilmente la comunicación entre aplicaciones y plataformas de computación, aplicar el cifrado en tránsito y aplicar controles de acceso detallados con políticas de autenticación de VPC Lattice. Esto le permite adoptar una postura de seguridad más sólida que cumpla con los requisitos normativos y de cumplimiento líderes del sector.
- Mejora de la escalabilidad y resiliencia de las aplicaciones: VPC Lattice le permite crear una red de aplicaciones implementadas con característica como enrutamiento, autenticación, autorización y monitoreo basados en rutas, encabezados y métodos. Estas ventajas se proporcionan sin sobrecargar los recursos de las cargas de trabajo y permiten hacer implementaciones de varios clústeres que generan millones de solicitudes por segundo sin agregar una latencia significativa.
- Flexibilidad de implementación con una infraestructura heterogénea: VPC Lattice proporciona características uniformes en todos los servicios de computación, como Amazon ECS, Fargate, Amazon EC2, Amazon EKS y Lambda, y le da a su organización la flexibilidad de elegir la infraestructura adecuada para cada aplicación.

Funcionamiento de VPC Lattice con otros servicios de Amazon ECS

El uso de VPC Lattice con Amazon ECS puede cambiar la forma en que utiliza otros servicios de Amazon ECS, mientras que otros seguirán siendo los mismos.

Equilibrador de carga de aplicación

Ya no es necesario crear un equilibrador de carga de aplicación específico para usarlo con el tipo de grupo de destino del equilibrador de carga de aplicación en VPC Lattice, que luego se vincula al servicio de Amazon ECS. En su lugar, solo tiene que configurar el servicio de Amazon ECS con un grupo de destino de VPC Lattice. También puede optar por utilizar el equilibrador de carga de aplicación con Amazon ECS al mismo tiempo.

Implementaciones continuas de Amazon ECS

Solo las implementaciones continuas de Amazon ECS funcionan con VPC Lattice, y Amazon ECS incorpora las tareas a los servicios y las elimina de forma segura durante la implementación. No se admiten la implementación de código ni las implementaciones azul/verde.

Para obtener más información sobre VPC Lattice, consulte la Guía del usuario de Amazon VPC Lattice.

Creación de un servicio que utilice VPC Lattice

Puede usar la AWS Management Console o la AWS CLI para crear un servicio con VPC Lattice.

Requisitos previos

Antes de empezar este tutorial, asegúrese de que se cumplen los siguientes requisitos previos:

- La última versión de la AWS CLI está instalada y configurada. Para obtener más información, consulte Instalación de la AWS Command Line Interface.

> **Note**
>
> Puede utilizar puntos de conexión de servicio de doble pila para interactuar con Amazon ECS desde la AWS CLI, los SDK y la API de Amazon ECS a través de IPv4 e IPv6. Para obtener más información, consulte Uso de puntos de conexión de doble pila en Amazon ECS.

- Los pasos descritos en Configuración para utilizar Amazon ECS están completos.
- Su usuario de IAM dispone de los permisos requeridos que se especifican en la política de IAM AmazonECS_FullAccess de ejemplo.

Creación de un servicio que utiliza VPC Lattice con la AWS Management Console

Siga estos pasos para crear un servicio con VPC Lattice mediante la AWS Management Console.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, elija Clústeres.
3. En la página Clústeres, seleccione el clúster en el que va a crear el servicio.
4. En la pestaña Services (Servicios), elija Create (Crear).

   Si nunca antes ha creado un servicio, siga los pasos que se indican en Creación de un servicio de Amazon ECS mediante la consola y, a continuación, continúe con estos pasos cuando llegue a la sección de VPC Lattice.
5. Marque el botón para elegir Activar VPC Lattice.
6. A fin de utilizar un rol existente para el rol de infraestructura de ECS para Amazon ECS, elija uno que ya haya creado para usarlo al crear el grupo de destino de VPC Lattice. Para crear un nuevo rol, cree un rol de infraestructura de ECS.

7. Elija la VPC.

   La VPC depende del modo de red que haya seleccionado al registrar la definición de la tarea. Si utiliza el modo `host` o `network` con el tipo de lanzamiento de EC2, elija su VPC.

   Si utiliza el modo `awsvpc`, la VPC se selecciona automáticamente en función de la VPC que haya elegido en Redes y no se podrá cambiar.

8. En Grupos de destino, elija el grupo o grupos de destino. Debe elegir un mínimo de un grupo de destino y un máximo de cinco. Elija Agregar grupo de destino para agregar grupos de destino adicionales. Elija el nombre del puerto, el protocolo y el puerto para cada grupo de destino que elija. Para eliminar un grupo de destino, elija Eliminar.

   > **Note**
   >
   > - Si quiere agregar grupos de destino existentes, debe utilizar la AWS CLI. Para obtener instrucciones sobre cómo agregar grupos de destino mediante la AWS CLI, consulte register-targets en la Referencia de AWS Command Line Interface.
   > - Si bien un servicio de VPC Lattice puede tener varios grupos de destino, cada grupo de destino solo se puede agregar a un servicio.

9. En este punto, vaya a la consola de VPC Lattice para continuar con la configuración. Aquí es donde se incluyen los nuevos grupos de destino en la acción predeterminada del oyente o en las reglas de un servicio de VPC Lattice existente.

   Para obtener más información, consulte Reglas de oyente para el servicio de VPC Lattice.

> **Important**
>
> Debe permitir el prefijo `vpc-lattice` de la regla entrante en su grupo de seguridad o tareas y se puede producir un error en las comprobaciones de estado.

Creación de un servicio que utiliza VPC Lattice con la AWS CLI

Utilice la AWS CLI para crear un servicio con VPC Lattice. Reemplace cada *`marcador de posición de entrada del usuario`* con información propia.

1. Cree un archivo de configuración de grupo de destino. El siguiente ejemplo se denomina `tg-config.json`

```
{
    "ipAddressType": "IPV4",
    "port": 443,
    "protocol": "HTTPS",
    "protocolVersion": "HTTP1",
    "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
}
```

2. Utilice el siguiente comando para crear un grupo de destino de VPC Lattice.

```
aws vpc-lattice create-target-group \
    --name my-lattice-target-group-ip \
    --type IP \
    --config file://tg-config.json
```

Ejemplo de salida:

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-0eaa4b9ab4EXAMPLE",
    "config": {
        "healthCheck": {
            "enabled": true,
            "healthCheckIntervalSeconds": 30,
            "healthCheckTimeoutSeconds": 5,
            "healthyThresholdCount": 5,
            "matcher": {
                "httpCode": "200"
            },
            "path": "/",
            "protocol": "HTTPS",
            "protocolVersion": "HTTP1",
            "unhealthyThresholdCount": 2
        },
        "ipAddressType": "IPV4",
        "port": 443,
        "protocol": "HTTPS",
        "protocolVersion": "HTTP1",
        "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
```

```
    },
    "id": "tg-0eaa4b9ab4EXAMPLE",
    "name": "my-lattice-target-group-ip",
    "status": "CREATE_IN_PROGRESS",
    "type": "IP"
}
```

3. El siguiente archivo JSON denominado *ecs-service-vpc-lattice.json* es un ejemplo que se utiliza para adjuntar un servicio de Amazon ECS a un grupo de destino de VPC Lattice. El `portName` en el ejemplo que aparece a continuación es el mismo que definió en el campo `name` de la propiedad `portMappings` de la definición de la tarea.

```
{
    "serviceName": "ecs-service-vpc-lattice",
    "taskDefinition": "ecs-task-def",
        "vpcLatticeConfigurations": [
        {
            "targetGroupArn": "arn:aws:vpc-lattice:us-
west-2:123456789012:targetgroup/tg-0eaa4b9ab4EXAMPLE",
            "portName": "testvpclattice",
            "roleArn": "arn:aws:iam::123456789012:role/
ecsInfrastructureRoleVpcLattice"
        }
    ],
    "desiredCount": 5,
    "role": "ecsServiceRole"
}
```

Utilice el siguiente comando para crear un servicio de Amazon ECS y adjuntarlo al grupo de destino de VPC Lattice mediante el ejemplo de json anterior.

```
aws ecs create-service \
    --cluster clusterName \
    --serviceName ecs-service-vpc-lattice \
    --cli-input-json file://ecs-service-vpc-lattice.json
```

> **Note**
>
> VPC Lattice no es compatible con Amazon ECS Anywhere.

Proteja sus tareas de Amazon ECS para que no se den por terminadas por eventos de reducción horizontal

Puede usar la protección de reducción horizontal de tareas de Amazon ECS para proteger sus tareas y evitar que finalicen debido a eventos de reducción horizontal derivados del escalado automático o las implementaciones de un servicio.

Algunas aplicaciones requieren un mecanismo para evitar que las tareas de misión crítica finalicen por eventos de reducción horizontal en momentos de baja utilización o durante las implementaciones de servicios. Por ejemplo:

- Tiene una aplicación asíncrona de procesamiento de colas, como un trabajo de transcodificación de video, en el que algunas tareas deben ejecutarse durante horas, incluso cuando la utilización acumulada del servicio es baja.
- Tiene una aplicación de juegos que ejecuta servidores de juegos como tareas de Amazon ECS que deben seguir ejecutándose incluso si todos los usuarios han cerrado sesión para reducir la latencia de inicio del reinicio del servidor.
- Al implementar una nueva versión de código, es necesario que las tareas sigan ejecutándose, ya que volver a procesarlas sería costoso.

Para proteger las tareas que pertenecen a su servicio y evitar que se terminen en un evento de reducción horizontal, establezca el atributo `ProtectionEnabled` en `true`. Si se establece `ProtectionEnabled` como verdadero, las tareas quedan protegidas durante 2 horas de forma predeterminada. Puede personalizar el periodo de protección mediante el atributo `ExpiresInMinutes`. Puede proteger sus tareas durante un mínimo de 1 minuto y hasta un máximo de 2880 minutos (48 horas). Si utiliza la AWS CLI, puede especificar la opción `--protection-enabled`.

Cuando una tarea termine su trabajo requerido, puede establecer el atributo `ProtectionEnabled` en `false`, lo que permite que la tarea finalice mediante eventos de reducción horizontal posteriores. Si utiliza la AWS CLI, puede especificar la opción `--no-protection-enabled`.

Mecanismos de protección de reducción horizontal de tareas

Puede configurar y obtener una protección de reducción horizontal de tareas mediante el punto de conexión del agente de contenedores de Amazon ECS o la API de Amazon ECS.

- Punto de conexión del agente de contenedor de Amazon ECS

Recomendamos utilizar el punto de conexión del agente de contenedores de Amazon ECS para tareas que puedan determinar por sí mismas la necesidad de protección. Utilice este enfoque para cargas de trabajo basadas en colas o de procesamiento de trabajos.

Cuando un contenedor comienza a procesar el trabajo, por ejemplo, al consumir un mensaje SQS, puede configurar el atributo `ProtectionEnabled` a través de la ruta del punto de conexión de protección de reducción horizontal de tareas `$ECS_AGENT_URI/task-protection/v1/state` desde el contenedor. Amazon ECS no finalizará esta tarea durante los eventos de reducción horizontal. Cuando la tarea termine su trabajo, puede borrar el atributo `ProtectionEnabled` con el mismo punto de conexión, lo que permite que la tarea pueda finalizarse durante eventos de reducción horizontal posteriores.

Para obtener más información sobre cómo usar el punto de conexión del agente de contenedores de Amazon ECS, consulte Punto de conexión de protección de reducción horizontal de tareas de Amazon ECS.

- API de Amazon ECS

  Puede usar la API de Amazon ECS para configurar y recuperar la protección de reducción horizontal de tareas si su aplicación tiene un componente que rastrea el estado de las tareas activas. Utilice `UpdateTaskProtection` para marcar una o más tareas como protegidas. Use `GetTaskProtection` para recuperar el estado de protección.

  Un ejemplo de este enfoque sería si su aplicación aloja sesiones de servidor de juegos como tareas de Amazon ECS. Cuando un usuario inicia sesión en el servidor (tarea), puede marcar la tarea como protegida. Cuando el usuario cierre la sesión, puede desactivar la protección específicamente para esta tarea o desactivar de forma periódica la protección para tareas similares que ya no tengan sesiones activas, según sus necesidades de mantener los servidores inactivos.

  Para obtener más información, consulte UpdateTaskProtection y GetTaskProtection en la Referencia de la API de Amazon Elastic Container Service.

Puede combinar ambos enfoques. Por ejemplo, utilice el punto de conexión del agente de Amazon ECS para configurar la protección de tareas desde un contenedor y utilice la API de Amazon ECS para quitar la protección de tareas de su servicio de controlador externo.

Consideraciones

Tenga en cuenta los siguientes puntos antes de utilizar la protección de reducción horizontal de tareas:

- La protección de tareas frente a la reducción horizontal solo se admite con las tareas implementadas desde un servicio.
- Recomendamos utilizar el punto de conexión del agente de contenedor de Amazon ECS porque el agente de Amazon ECS tiene mecanismos de reintento incorporados y una interfaz más sencilla.
- Para restablecer el periodo de caducidad de la protección de reducción horizontal de tareas, invoque `UpdateTaskProtection` en una tarea que ya tenga activada la protección.
- Determine cuánto tiempo necesitaría una tarea para completar el trabajo requerido y configure la propiedad `expiresInMinutes` en consecuencia. Si establece que la caducidad de la protección sea más larga de lo necesario, incurrirá en costos y se retrasará la implementación de nuevas tareas.
- La protección de reducción horizontal es compatible con el agente de contenedor de Amazon ECS `1.65.0` o una posterior.

  Puede agregar compatibilidad con esta característica en instancias de Amazon EC2 que utilizan versiones anteriores del agente de contenedor de Amazon ECS si actualiza el agente a la versión más reciente. Para obtener más información, consulte Actualización del agente de contenedor de Amazon ECS.
- Consideraciones sobre la implementación:
  - Si el servicio utiliza una actualización continua, se crearán nuevas tareas, pero las tareas que ejecuten una versión anterior no se terminarán hasta que `protectionEnabled` se desactive o caduque. Puede ajustar el parámetro `maximumPercentage` en la configuración de implementación a un valor que permita crear nuevas tareas cuando las tareas antiguas estén protegidas.
  - Si se aplica una actualización azul/verde, la implementación azul con tareas protegidas no se eliminará si las tareas tienen `protectionEnabled`. El tráfico se desviará a las nuevas tareas que surjan y las tareas más antiguas solo se eliminarán cuando `protectionEnabled` se desactive o caduque. Según el tiempo de espera de las actualizaciones de CodeDeploy o CloudFormation, es posible que se agote el tiempo de espera de la implementación y que las tareas azules más antiguas sigan presentes.

- Si usa CloudFormation, la pila de actualizaciones tiene un tiempo de espera de 3 horas. Por lo tanto, si configura la protección de sus tareas durante más de 3 horas, la implementación de CloudFormation puede provocar un error y una restauración.

  Durante el tiempo en que sus tareas antiguas están protegidas, en la pila de CloudFormation se muestra `UPDATE_IN_PROGRESS`. Si se elimina la protección de reducción horizontal de tareas o esta caduca dentro del período de 3 horas, su implementación se realizará correctamente y pasará al estado `UPDATE_COMPLETE`. Si la implementación se detiene en `UPDATE_IN_PROGRESS` durante más de 3 horas, se producirá un error, se mostrará el estado `UPDATE_FAILED` y, a continuación, se llevará a cabo una restauración al conjunto de tareas anterior.
- Amazon ECS vende eventos de servicio cuando las tareas protegidas impiden que una implementación (continua o azul/verde) alcance un estado estable, para que pueda tomar medidas correctivas. Al intentar actualizar el estado de protección de una tarea, si recibe un mensaje de error `DEPLOYMENT_BLOCKED`, significa que el servicio tiene más tareas protegidas que el recuento deseado de tareas para el servicio. Para corregir este error, realice alguna de las siguientes acciones:
  - Espere a que caduque la protección de tareas actual. A continuación, establezca la protección de tareas.
  - Determine qué tareas se pueden detener. A continuación, utilice `UpdateTaskProtection` con la opción `protectionEnabled` configurada en `false` para estas tareas.
  - Aumente el recuento de tareas deseado del servicio a un número mayor al de tareas protegidas.

Permisos de IAM requeridos para la protección de reducción horizontal de tareas

La tarea debe tener el rol de tarea de Amazon ECS con los siguientes permisos:

- `ecs:GetTaskProtection`: permite que el agente de contenedor de Amazon ECS llame a `GetTaskProtection`.
- `ecs:UpdateTaskProtection`: permite que el agente de contenedor de Amazon ECS llame a `UpdateTaskProtection`.

Punto de conexión de protección de reducción horizontal de tareas de Amazon ECS

El agente del contenedor de Amazon ECS inyecta automáticamente la variable de entorno `ECS_AGENT_URI` en los contenedores de las tareas de Amazon ECS para proporcionar un método que permita interactuar con el punto de conexión de la API del agente de contenedor.

Recomendamos utilizar el punto de conexión del agente de contenedores de Amazon ECS para tareas que puedan determinar por sí mismas la necesidad de protección.

Cuando un contenedor comienza a procesar el trabajo, puede configurar el atributo `protectionEnabled` a través de la ruta del punto de conexión de protección de reducción horizontal de tareas `$ECS_AGENT_URI/task-protection/v1/state` desde el contenedor.

Utilice una solicitud PUT a este URI desde dentro de un contenedor para establecer la protección de reducción horizontal. Una solicitud GET a este URI devolverá el estado de protección actual de una tarea.

Parámetros de solicitud de protección de reducción horizontal de tareas

Puede configurar la protección de reducción horizontal de tareas mediante el punto de conexión `${ECS_AGENT_URI}/task-protection/v1/state` con los siguientes parámetros de solicitud.

`ProtectionEnabled`

Especifique `true` para marcar una tarea para su protección. Especifique `false` si desea eliminar la protección y hacer que la tarea pueda cancelarse.

Tipo: Booleano

Obligatorio: sí

`ExpiresInMinutes`

Número de minutos que debe transcurrir para proteger la tarea. Puede especificar un mínimo de 1 minuto y un máximo de 2880 minutos (48 horas). Durante este periodo, la tarea no finalizará con eventos de reducción horizontal del escalado automático del servicio o implementaciones. Una vez transcurrido este período, el parámetro `protectionEnabled` se restablecerá a `false`.

Si no especifica el tiempo, la tarea se protege automáticamente durante 120 minutos (2 horas).

Tipo: entero

Obligatorio: no

En los siguientes ejemplos se muestra cómo configurar la protección de tareas con diferentes duraciones.

Ejemplo de cómo proteger una tarea con el periodo predeterminado

En este ejemplo se muestra cómo proteger una tarea con el periodo de tiempo predeterminado de 2 horas.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true}'
```

Ejemplo de cómo proteger una tarea durante 60 minutos

En este ejemplo se muestra cómo proteger una tarea durante 60 minutos mediante el parámetro `expiresInMinutes`.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true,"ExpiresInMinutes":60}'
```

Ejemplo de cómo proteger una tarea durante 24 horas

En este ejemplo se muestra cómo proteger una tarea durante 24 horas mediante el parámetro `expiresInMinutes`.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true,"ExpiresInMinutes":1440}'
```

Ejemplos de contenedores de Windows

Para los contenedores de Windows, puede usar el cmdlet `Invoke-RestMethod` de PowerShell en lugar de curl. En los ejemplos siguientes se muestran los equivalentes en PowerShell de los comandos curl anteriores.

Ejemplo de cómo proteger una tarea de contenedor de Windows con el periodo predeterminado

En este ejemplo se muestra cómo proteger una tarea con el periodo predeterminado de 2 horas mediante PowerShell.

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Put -Body
 '{"ProtectionEnabled":true}' -ContentType 'application/json'
```

Ejemplo de cómo proteger una tarea de contenedor de Windows durante 60 minutos

En este ejemplo se muestra cómo proteger una tarea durante 60 minutos mediante el parámetro `expiresInMinutes` con PowerShell.

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Put -Body
 '{"ProtectionEnabled":true,"ExpiresInMinutes":60}' -ContentType 'application/json'
```

Ejemplo de cómo proteger una tarea de contenedor de Windows durante 24 horas

En este ejemplo se muestra cómo proteger una tarea durante 24 horas mediante el parámetro `expiresInMinutes` con PowerShell.

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Put -Body
 '{"ProtectionEnabled":true,"ExpiresInMinutes":1440}' -ContentType 'application/json'
```

La solicitud PUT devolverá la siguiente respuesta.

```
{
  "protection": {
    "ExpirationDate": "2023-12-20T21:57:44.837Z",
    "ProtectionEnabled": true,
    "TaskArn": "arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0"
  }
}
```

Parámetros de respuesta de protección de la reducción horizontal de tareas

La siguiente información se devuelve desde el punto de conexión de reducción horizontal de tareas `${ECS_AGENT_URI}/task-protection/v1/state` en la respuesta de JSON.

`ExpirationDate`

La época en la que caducará la protección de la tarea. Si la tarea no está protegida, este valor será nulo.

`ProtectionEnabled`

El estado de protección de la tarea. Si la protección de reducción horizontal está habilitada para una tarea, el valor es `true`. De lo contrario, es `false`.

`TaskArn`

Nombre de recurso de Amazon (ARN) de la tarea a la que pertenece el contenedor.

En el ejemplo siguiente se muestran los detalles que se devuelven de una tarea protegida.

```
curl --request GET ${ECS_AGENT_URI}/task-protection/v1/state
```

Para los contenedores de Windows, utilice el siguiente comando de PowerShell para obtener el estado de protección:

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Get
```

```
{
    "protection":{
        "ExpirationDate":"2023-12-20T21:57:44Z",
        "ProtectionEnabled":true,
        "TaskArn":"arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0"
    }
}
```

Cuando se produce un error, se devuelve la siguiente información.

`Arn`

El nombre de recurso de Amazon (ARN) de la tarea.

`Detail`

Los detalles relacionados con el error.

`Reason`

El motivo del error.

En el ejemplo siguiente se muestran los detalles que se devuelven de una tarea que no está protegida.

```
{
    "failure":{
        "Arn":"arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0",
```

```
        "Detail":null,
        "Reason":"TASK_NOT_VALID"
    }
}
```

Cuando se produce una excepción, se devuelve la siguiente información.

`requestID`

El ID de solicitud de AWS para la llamada a la API de Amazon ECS que produce una excepción.

`Arn`

El nombre de recurso de Amazon (ARN) completo del servicio o la tarea.

`Code`

Código de error.

`Message`

Mensaje de error.

> **Note**
>
> Si aparece un error `RequestError` o `RequestTimeout`, es probable que se trate de un problema de red. Intente utilizar puntos de conexión de VPC para Amazon ECS.

En el ejemplo siguiente se muestran los detalles que se devuelven cuando se produce un error.

```
{
    "requestID":"12345-abc-6789-0123-abc",
    "error":{
        "Arn":"arn:aws:ecs:us-west-2:555555555555:task/my-cluster-
name/1234567890abcdef0",
        "Code":"AccessDeniedException",
        "Message":"User: arn:aws:sts::444455556666:assumed-role/my-ecs-task-
role/1234567890abcdef0 is not authorized to perform: ecs:GetTaskProtection on resource:
 arn:aws:ecs:us-west-2:555555555555:task/test/1234567890abcdef0 because no identity-
based policy allows the ecs:GetTaskProtection action"
    }
}
```

El siguiente error aparece si el agente de Amazon ECS no puede obtener una respuesta del punto de conexión de Amazon ECS por motivos como problemas de red o si el plano de control de Amazon ECS no funciona.

```
{
  "error": {
    "Arn": "arn:aws:ecs:us-west-2:555555555555:task/my-cluster-name/1234567890abcdef0",
    "Code": "RequestCanceled",
    "Message": "Timed out calling Amazon ECS Task Protection API"
  }
}
```

El siguiente error aparece cuando el agente de Amazon ECS recibe una excepción de limitación de Amazon ECS.

```
{
  "requestID": "12345-abc-6789-0123-abc",
  "error": {
    "Arn": "arn:aws:ecs:us-west-2:555555555555:task/my-cluster-name/1234567890abcdef0",
    "Code": "ThrottlingException",
    "Message": "Rate exceeded"
  }
}
```

Utilice la inyección de errores con las cargas de trabajo de Amazon ECS y Fargate

Los clientes pueden utilizar la inyección de errores con Amazon ECS tanto en Amazon EC2 como en Fargate para probar de qué manera la aplicación responde a determinados escenarios de deterioro. Estas pruebas proporcionan información útil para optimizar el rendimiento y la resiliencia de la aplicación.

Cuando la inyección de errores está habilitada, el agente de contenedores de Amazon ECS permite que las tareas accedan a nuevos puntos de conexión de inyección de errores. Debe optar por utilizar la inyección de errores. Para ello, configure el valor del parámetro de definición de la tarea `enableFaultInjection` en `true`. El valor predeterminado es `false`.

```
{
    ...
   "enableFaultInjection": true
```

```
}
```

> **Note**
>
> La inyección de errores solo funciona con tareas que utilicen los modos de red `awsvpc` o `host`.
> La inyección de errores no está disponible en Windows.

Para obtener información sobre cómo habilitar la inyección de errores en la AWS Management Console, consulte Creación de una definición de tarea de Amazon ECS mediante la consola.

Tendrá que habilitar la característica para las pruebas en AWS Fault Injection Service. Para obtener más información, consulte Uso de las acciones aws:ecs:task de AWS FIS.

> **Note**
>
> Si no utiliza las nuevas AMI optimizadas para Amazon ECS o tiene una AMI personalizada, instale las siguientes dependencias:
>
> - `tc`
> - Módulo de kernel de `sch_netem`

Puntos de conexión de inyección de errores de Amazon ECS

El agente del contenedor de Amazon ECS inyecta automáticamente la variable de entorno `ECS_AGENT_URI` en los contenedores de las tareas de Amazon ECS para proporcionar un método que permita interactuar con el punto de conexión de la API del agente de contenedor. Cada punto de conexión incluye un punto de conexión de `/start`, `/stop` y `/status` Los puntos de conexión solo aceptan solicitudes de tareas que tengan habilitada la inyección de errores, y cada punto de conexión tiene un límite de velocidad de 1 solicitud cada 5 segundos por contenedor. Si se supera el límite, se produce un error.

> **Note**
>
> Se requiere el agente de Amazon ECS `version 1.88.0+` para utilizar los puntos de conexión de inyección de errores.

Los tres puntos de conexión que se utilizan con la inyección de errores son:

- Punto de conexión del puerto blackhole de la red
- Punto de conexión de pérdida de paquetes de red
- Punto de conexión de latencia de red

Si la solicitud se realiza correctamente, se obtiene un código de respuesta `200` con un mensaje de `running` cuando se llama al punto de conexión de `/start`, `stopped` para el punto de conexión de `/stop` y `running` o `not-running` para el punto de conexión de `/status`.

```
{
    "Status": <string>
}
```

Si la solicitud no se realiza correctamente, se devuelve uno de los siguientes códigos de error:

- `400`: solicitud errónea
- `409`: la solicitud de inyección de errores entra en conflicto con otro error de ejecución
- `429`: la solicitud se ha limitado
- `500`: se presentó un error inesperado en el servidor

```
{
 "Error":  <string message>
}
```

> **Note**
>
> Se puede inyectar un error de latencia de red o un error de pérdida de paquetes de red cada vez. Si se intenta inyectar más de uno, se rechazará la solicitud.

Punto de conexión del puerto blackhole de la red

El punto de conexión de `{ECS_AGENT_URI}/fault/v1/network-blackhole-port` descarta el tráfico entrante o saliente para un puerto y protocolo específicos en el espacio de nombres de red de una tarea y es compatible con dos modos:

- awsvp: los cambios se aplican al espacio de nombres de la red de la tarea
- host: los cambios se aplican a la instancia de contenedor de espacio de nombres de red predeterminada

{ECS_AGENT_URI}/fault/v1/network-blackhole-port/start

Este punto de conexión inicia las inyecciones de errores en el puerto blackhole de la red y tiene los siguientes parámetros:

Puerto

El puerto especificado que se utilizará para la inyección de errores del puerto blackhole.

Tipo: entero

Obligatorio: sí

Protocolo

El protocolo que se utilizará para la inyección de errores en el puerto blackhole.

Tipo: cadena

Valores válidos: `tcp | udp`

Obligatorio: sí

TrafficType

El tipo de tráfico que la inyección de errores utiliza.

Tipo: cadena

Valores válidos: `ingress | egress`

Obligatorio: sí

SourcesToFilter

Una matriz JSON de direcciones IPv4 o bloques CIDR que están protegidos frente al error.

Tipo: matriz de cadenas

Obligatorio: no

A continuación, se muestra un ejemplo de solicitud para utilizar el punto de conexión de `start` (sustituya los valores en *`rojo`* por valores propios):

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-blackhole-port/start

Http method:POST

Request payload:
{
    "Port": 1234,
    "Protocol": "tcp|udp",
    "TrafficType": "ingress|egress"
    "SourcesToFilter": ["${IP1}", "${IP2}", ...],
}
```

{ECS_AGENT_URI}/fault/v1/network-blackhole-port/stop

Este punto de conexión detiene el error especificado en la solicitud. Este punto de conexión tiene los siguientes parámetros:

Puerto

El puerto afectado por el error que se debe detener.

Tipo: entero

Obligatorio: sí

Protocolo

El protocolo que se utilizará para detener el error.

Tipo: cadena

Valores válidos: `tcp | udp`

Obligatorio: sí

TrafficType

El tipo de tráfico que la inyección de errores utiliza.

Tipo: cadena

Valores válidos: `ingress | egress`

Obligatorio: sí

A continuación, se muestra un ejemplo de solicitud para utilizar el punto de conexión de `stop` (sustituya los valores en *`rojo`* por valores propios):

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-blackhole-port/stop

Http method: POST

Request payload:
{
    "Port": 1234,
    "Protocol": "tcp|udp",
    "TrafficType": "ingress|egress",
}
```

{ECS_AGENT_URI}/fault/v1/network-blackhole-port/status

Este punto de conexión se utiliza para comprobar el estado de la inyección de errores. Este punto de conexión tiene los siguientes parámetros:

Puerto

El puerto afectado para comprobar el estado del error.

Tipo: entero

Obligatorio: sí

Protocolo

El protocolo que se utilizará al comprobar el estado del error.

Tipo: cadena

Valores válidos: `tcp | udp`

Obligatorio: sí

TrafficType

El tipo de tráfico que la inyección de errores utiliza.

Tipo: cadena

Valores válidos: `ingress | egress`

Obligatorio: sí

A continuación, se muestra un ejemplo de solicitud para utilizar el punto de conexión de `status` (sustituya los valores en *`rojo`* por valores propios):

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-blackhole-port/status

Http method: POST

Request payload:
{
   "Port": 1234,
   "Protocol": "tcp|udp",
   "TrafficType": "ingress|egress",
}
```

Punto de conexión de latencia de red

El punto de conexión de `{ECS_AGENT_URI}/fault/v1/network-latency` agrega retardo y fluctuación a la interfaz de red de la tarea para el tráfico a determinados orígenes. El punto de conexión es compatible con dos modos:

- awsvpc: los cambios se aplican a la interfaz de red de la tarea
- host: los cambios se aplican a la interfaz de red predeterminada

{ECS_AGENT_URI}/fault/v1/network-latency/start

Este punto de conexión de `/start` inicia la inyección de errores de latencia de la red y tiene los siguientes parámetros:

DelayMilliseconds

La cantidad de milisegundos de retardo que se agregarán a la interfaz de red que se utilizará para la inyección de errores.

Tipo: entero

Obligatorio: sí

JitterMilliseconds

La cantidad de milisegundos de fluctuación que se agregarán a la interfaz de red que se utilizará para la inyección de errores.

Tipo: entero

Obligatorio: sí

Orígenes

Una matriz JSON de direcciones IPv4 o bloques de CIDR que son destino para su uso con inyección de errores.

Tipo: matriz de cadenas

Obligatorio: sí

SourcesToFilter

Una matriz JSON de direcciones IPv4 o bloques CIDR protegidos frente al error. `SourcesToFilter` tiene prioridad sobre `Sources`.

Tipo: matriz de cadenas

Obligatorio: no

A continuación, se muestra un ejemplo de solicitud para utilizar el punto de conexión de `/start` (sustituya los valores en *`rojo`* por valores propios):

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-latency/start

Http method: POST

Request payload:
{
    "DelayMilliseconds": 123,
    "JitterMilliseconds": 123,
```

```
    "Sources": ["${IP1}", "${IP2}", ...],
    "SourcesToFilter": ["${IP1}", "${IP2}", ...],
}
```

{ECS_AGENT_URI}/fault/v1/network-latency/stop and /status

El punto de conexión de `{ECS_AGENT_URI}/fault/v1/network-latency/stop` detiene el error, mientras que `{ECS_AGENT_URI}/fault/v1/network-latency/status` comprueba el estado del error.

Los siguientes son dos ejemplos de solicitudes para utilizar los puntos de conexión de `/stop` y `/status`. Ambos utilizan el método `POST HTTP`.

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-latency/stop
```

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-latency/status
```

Punto de conexión de pérdida de paquetes de red

El punto de conexión de `{ECS_AGENT_URI}/fault/v1/network-packet-loss` agrega la pérdida de paquetes a la interfaz de red dada. Este punto de conexión es compatible con dos modos:

- awsvpc: los cambios se aplican a la interfaz de red de la tarea
- host: los cambios se aplican a la interfaz de red predeterminada

{ECS_AGENT_URI}/fault/v1/network-packet-loss/start

Este punto de conexión de `/start` inicia la inyección de errores de pérdida de paquetes de red y tiene los siguientes parámetros:

LossPercent

El porcentaje de pérdida de paquetes

Tipo: entero

Obligatorio: sí

Orígenes

Una matriz JSON de direcciones IPv4 o bloques CIDR a utilizar para las pruebas de inyección de errores.

Tipo: matriz de cadenas

Obligatorio: sí

SourcesToFilter

Una matriz JSON de direcciones IPv4 o bloques CIDR protegidos frente al error. `SourcesToFilter` tiene prioridad sobre `Sources`.

Tipo: matriz de cadenas

Obligatorio: no

A continuación, se muestra un ejemplo de solicitud para utilizar el punto de conexión de `start` (sustituya los valores en *`rojo`* por valores propios):

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-packet-loss/start

Http method: POST

{
    "LossPercent": 6,
    "Sources": ["${IP1}", "${IP2}", ...],
    "SourcesToFilter": ["${IP1}", "${IP2}", ...],
}
```

{ECS_AGENT_URI}/fault/v1/network-packet-loss/stop y /status

El punto de conexión de `{ECS_AGENT_URI}/fault/v1/network-packet-loss/stop` detiene el error, mientras que `{ECS_AGENT_URI}/fault/v1/network-packet-loss/status` comprueba el estado del error. Solo se admite uno de cada tipo de error a la vez.

Los siguientes son dos ejemplos de solicitudes para utilizar los puntos de conexión de `/stop` y `/status`. Ambos utilizan el método `POST HTTP`.

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-packet-loss/stop
```

```
Endpoint: ${{ECS_AGENT_URI}/fault/v1/network-packet-loss/status
```

Actualización de los parámetros de servicio de Amazon ECS

Después de crear un servicio, hay ocasiones en las que puede que necesite actualizar sus parámetros, por ejemplo, el número de tareas.

Cuando el programador de servicios lanza nuevas tareas, determina la ubicación de ellas en el clúster con la siguiente lógica.

- Determine cuál de las instancias de contenedor de su clúster puede admitir la definición de tarea de su servicio. Por ejemplo, si tienen la CPU, la memoria, los puertos y los atributos de instancia de contenedor requeridos.
- De forma predeterminada, el programador de servicios intenta equilibrar las tareas entre las zonas de disponibilidad de esta manera, aunque puede elegir una estrategia de ubicación diferente.
  - Ordene las instancias de contenedor válidas por el número menor de tareas en ejecución para este servicio en la misma zona de disponibilidad que la instancia. Por ejemplo, si la zona A tiene una tarea de servicio en ejecución y las zonas B y C tienen cero cada una, las instancias de contenedor válidas en la zona B o C se consideran óptimas para colocación.
  - Coloque la nueva tarea de servicio en una instancia de contenedor válida en una zona de disponibilidad óptima (basada en los pasos anteriores), favoreciendo las instancias de contenedor con el número más bajo de tareas en ejecución para este servicio.

Cuando el programador de servicios detiene las tareas en ejecución, intenta mantener un balance entre las zonas de disponibilidad del clúster usando la siguiente lógica:

- Ordene las instancias de contenedor por el número mayor de tareas en ejecución para este servicio en la misma zona de disponibilidad que la instancia. Por ejemplo, si la zona A tiene una tarea de servicio en ejecución y las zonas B y C tienen dos cada una, las instancias de contenedor en la zona B o C se consideran óptimas para terminación.
- Pare la tarea en una instancia de contenedor en una zona de disponibilidad óptima (basada en los pasos anteriores), favoreciendo las instancias de contenedor con el número mayor de tareas en ejecución para este servicio.

Use la lista para determinar si puede cambiar el parámetro de servicio.

Reequilibrio de zona de disponibilidad

Indica si se debe utilizar el reequilibrio de zonas de disponibilidad para el servicio.

Puede cambiar este parámetro para las implementaciones continuas.

Estrategia de proveedores de capacidad

Los detalles de una estrategia de proveedor de capacidad. Puede establecer un proveedor de capacidad cuando crea un clúster, ejecuta una tarea o actualiza un servicio.

Cuando usa Fargate, los proveedores de capacidad son `FARGATE` o `FARGATE_SPOT`.

Cuando utiliza Amazon EC2, los proveedores de capacidad son grupos de escalado automático.

Puede cambiar los proveedores de capacidad para implementaciones continuas e implementaciones azul/verde.

En la siguiente lista, se indican las transiciones válidas:

- Actualización del tipo de lanzamiento de Fargate a un proveedor de capacidad de grupos de escalado automático.
- Actualización del tipo de lanzamiento de Amazon EC2 a un proveedor de capacidad de Fargate.
- Actualización del proveedor de capacidad de Fargate a un proveedor de capacidad de grupo de escalado automático.
- Actualización del proveedor de capacidad de Amazon EC2 a un proveedor de capacidad de Fargate.
- Actualización del grupo de escalado automático o el proveedor de capacidad de Fargate de vuelta al tipo de lanzamiento. Cuando usa la CLI o la API, pasa una lista vacía en el parámetro `capacityProviderStrategy`.

Clúster

No puede modificar el nombre del clúster.

Configuración de implementación

La configuración de implementación incluye las alarmas de CloudWatch y el disyuntor utilizados para detectar fallos, así como la configuración requerida.

El disyuntor de implementación determina si una implementación de servicio fallará cuando el servicio no pueda alcanzar un estado estable. Si usa el disyuntor de implementación, una implementación de servicio pasará a un estado fallido y dejará de iniciar nuevas tareas. Si usa la

opción de restauración, cuando se produce un error en la implementación de un servicio, este se restaura a la última implementación que se haya completado de forma correcta.

Cuando actualiza un servicio que utiliza un interruptor de Amazon ECS, Amazon ECS crea una implementación y una revisión de servicios. Estos recursos le permiten ver información detallada sobre el historial del servicio. Para obtener más información, consulte Visualización del historial de servicios mediante las implementaciones de servicios de Amazon ECS.

El programador de servicio utiliza los parámetros de porcentaje máximo y porcentaje mínimo en buen estado (en la configuración de implementación del servicio) para determinar la estrategia de implementación.

Si un servicio utiliza el tipo de implementación de actualizaciones continuas (`ECS`), el porcentaje mínimo en buen estado representa un límite inferior en el número de las tareas en un servicio que deben permanecer en el estado `RUNNING` durante una implementación, como un porcentaje del número de tareas deseado (redondeado al entero superior más próximo). El parámetro también se aplica mientras haya instancias de contendedor en el estado `DRAINING` si el servicio contiene tareas que utilizan el tipo de lanzamiento de EC2. Utilice este parámetro para efectuar implementaciones sin utilizar capacidad de clúster adicional. Por ejemplo, si su servicio tiene un número deseado de cuatro tareas y un porcentaje mínimo de estado del 50 %, el programador puede detener dos tareas existentes para liberar la capacidad del clúster antes de iniciar dos nuevas tareas. El servicio considera que las tareas están en buen estado para servicios que no utilizan un equilibrador de carga si están en el estado `RUNNING`. El servicio considera que las tareas están en buen estado para servicios que no usan un equilibrador de carga si están en el estado `RUNNING` y el equilibrador de carga notifica que están en buen estado. El valor predeterminado del porcentaje mínimo de estado es el 100 %.

Si un servicio utiliza el tipo de implementación de actualizaciones continuas (`ECS`), el parámetro porcentaje máximo representa un límite superior en el número de las tareas en un servicio que pueden permanecer en el estado `PENDING`, `RUNNING` o `STOPPING` durante una implementación, como un porcentaje del número de tareas deseado (redondeado al entero inferior más próximo). El parámetro también se aplica mientras haya instancias de contendedor en el estado `DRAINING` si el servicio contiene tareas que utilizan el tipo de lanzamiento de EC2. Utilice este parámetro para definir el tamaño del lote de implementación. Por ejemplo, si su servicio tiene un número deseado de cuatro tareas y un valor porcentual máximo del 200 %, el programador puede iniciar cuatro nuevas tareas antes de detener las cuatro más antiguas. Esto es siempre que los recursos del clúster requeridos para hacer esto estén disponibles. El valor predeterminado para el porcentaje máximo es el 200 %.

Cuando el programador de servicio sustituye una tarea durante una actualización, el servicio elimina primero la tarea del balanceador de carga (si se usa) y espera a que las conexiones se vacíen. A continuación, se emite el equivalente de docker stop a los contenedores que ejecutan la tarea. Esto da lugar a una señal `SIGTERM` y a un tiempo de espera de 30 segundos, tras el cual se envía `SIGKILL` y los contenedores se paran por la fuerza. Si el contenedor gestiona la señal `SIGTERM` correctamente y sale antes de los 30 segundos de haberla recibido, no se envía la señal `SIGKILL`. El programador de servicio inicia y para tareas definidas por la configuración de porcentaje en buen estado mínimo y porcentaje máximo.

El programador de servicios también reemplaza las tareas que se determina que están en mal estado después de que se produzca un error en una comprobación de estado del contenedor o en una comprobación de estado del grupo objetivo del equilibrador de cargas. Este reemplazo depende de los parámetros de definición del servicio `maximumPercent` y `desiredCount`. Si una tarea está marcada como en mal estado, el programador de servicios iniciará primero una tarea de reemplazo. Luego, ocurrirá lo siguiente.

- Si la tarea de reemplazo tiene un estado de `HEALTHY`, el programador de servicios detiene la tarea en mal estado.
- Si la tarea de reemplazo tiene un estado de `UNHEALTHY`, el programador detendrá la tarea de reemplazo en mal estado o la tarea existente en mal estado para igualar el recuento total de tareas en `desiredCount`.

Si el parámetro `maximumPercent` impide que el programador inicie primero una tarea de reemplazo, detendrá las tareas en mal estado de forma aleatoria de una en una para liberar capacidad y, a continuación, iniciará una tarea de reemplazo. El proceso de inicio y parada continúa hasta que todas las tareas en mal estado se sustituyan por tareas en buen estado. Una vez que se hayan reemplazado todas las tareas en mal estado y solo se estén ejecutando las tareas en buen estado, si el recuento total de tareas supera el límite de `desiredCount`, las tareas en buen estado se detienen aleatoriamente hasta que el recuento total de tareas sea igual a `desiredCount`. Para obtener más información sobre `maximumPercent` y `desiredCount`, consulte Parámetros de definición de servicios.

Controlador de implementación

El controlador de implementación que utilizar para el servicio. Existen tres tipos de controlador de implementación disponibles:

- `ECS`
- `EXTERNAL`
- `CODE_DEPLOY`

Al actualizar un servicio, puede actualizar el controlador de implementación que utiliza. En la siguiente lista, se indican las transiciones válidas:

- Actualice las implementaciones azul/verde de CodeDeploy (`CODE_DEPLOY`) a las implementaciones continuas o implementaciones azul/verde de ECS (`ECS`).
- Actualice las implementaciones azul/verde de CodeDeploy (`CODE_DEPLOY`) a las implementaciones externas (`EXTERNAL`).
- Actualice desde implementaciones continuas de ECS o implementaciones azul/verde (`ECS`) a implementaciones externas (`EXTERNAL`).
- Actualice desde implementaciones externas (`EXTERNAL`) a implementaciones continuas de ECS o azul/verde (`ECS`).

Tenga en cuenta lo siguiente al actualizar el controlador de implementación de un servicio:

- No puede actualizar el controlador de implementación de un servicio desde el controlador de implementación `ECS` a ningún otro controlador si utiliza VPC Lattice o Amazon ECS Service Connect.
- No puede actualizar el controlador de implementación de un servicio durante una implementación de servicio en curso.
- No puede actualizar el controlador de implementación de un servicio a `CODE_DEPLOY` si no hay equilibradores de carga en el servicio.
- No puede actualizar el controlador de implementación de un servicio de `ECS` a ningún otro controlador si la `deploymentConfiguration` incluye alarmas, un interruptor de circuito de implementación o una estrategia de implementación `BLUE_GREEN`. Para obtener más información, consulte Estrategias y controladores de implementación de servicios de Amazon ECS.
- Amazon ECS no utilizará el valor que especifique para `versionConsistency` en la definición del contenedor si actualiza el controlador de implementación del servicio de `ECS` a cualquiera de los demás controladores.
- Si actualiza el controlador de implementación de un servicio desde `ECS` a uno de los demás controladores, las respuestas `UpdateService` y `DescribeService` de la API seguirán devolviendo `deployments` en lugar de `taskSets`. Para más información acerca de `UpdateService` y `CreateService`, consulte UpdateService y CreateService en la Referencia de la API de Amazon ECS.
- Si un servicio utiliza una estrategia de implementación de actualizaciones continuas, la actualización del controlador de implementación desde `ECS` a cualquiera de los demás

controladores cambiará la forma en que el valor `maximumPercent` se utiliza en el `deploymentConfiguration`. En lugar de únicamente limitar el total de tareas en una implementación de actualizaciones continuas, `maximumPercent` se utiliza para reemplazar las tareas que no funcionan correctamente. Para más información sobre cómo el programador sustituye las tareas en mal estado, consulte Servicios de Amazon ECS.

- Si actualiza el controlador de implementación de un servicio desde `ECS` a uno de los demás controladores de implementación, se ignorará cualquier `advancedConfiguration` que especifique en la configuración del equilibrador de carga. Para más información, consulte LoadBalancer y AdvancedConfiguration en la Referencia de la API de Amazon ECS.

Al actualizar el controlador de implementación de un servicio mediante CloudFormation, tenga en cuenta lo siguiente en función del tipo de migración que vaya a realizar.

- Si tiene una plantilla de CloudFormation que contiene la información de controlador de implementación `EXTERNAL` y los recursos `TaskSet` y `PrimaryTaskSet`, y elimina los recursos del conjunto de tareas de la plantilla al actualizar de `EXTERNAL` a `ECS`, las llamadas `DescribeTaskSet` y `DeleteTaskSet` de la API devolverán un error 400 una vez que se actualice el controlador de implementación a `ECS`. Esto provoca un error al eliminar CloudFormation en los recursos del conjunto de tareas, aunque la pila de CloudFormation pase al estado `UPDATE_COMPLETE`. Para más información, consulte Recurso eliminado de la pila pero no borrado en la Guía del usuario de AWS CloudFormation. Para solucionar este problema, elimine los conjuntos de tareas directamente mediante la API `DeleteTaskSet` de Amazon ECS. Para más información sobre cómo eliminar un conjunto de tareas, consulte DeleteTaskSet en la Referencia de la API de Amazon Elastic Container Service.
- Si está migrando de `CODE_DEPLOY` a `ECS` con una nueva definición de tarea y CloudFormation realiza una operación de reversión, la solicitud `UpdateService` de Amazon ECS falla y muestra el siguiente error:

```
Resource handler returned message: "Invalid request provided: Unable to update
task definition on services with a CODE_DEPLOY deployment controller. Use AWS
CodeDeploy to trigger a new deployment. (Service: Ecs, Status Code: 400,
Request ID: 0abda1e2-f7b3-4e96-b6e9-c8bc585181ac) (SDK Attempt Count: 1)"
(RequestToken: ba8767eb-c99e-efed-6ec8-25011d9473f0, HandlerErrorCode:
 InvalidRequest)
```

- Tras una migración correcta de `ECS` a un controlador de implementación `EXTERNAL`, debe eliminar manualmente el conjunto de tareas `ACTIVE`, ya que Amazon ECS ya no administra la implementación. Para obtener información acerca de cómo eliminar un conjunto de tareas, consulte DeleteTaskSet en la Referencia de la API de Amazon Elastic Container Service.

Recuento deseado de tareas

El número de instancias de la tarea para ubicar y seguir ejecutando en el servicio.

Si desea detener temporalmente el servicio, establezca este valor en 0. A continuación, cuando lo tenga todo listo para iniciar el servicio, actualícelo con el valor original.

Puede cambiar este parámetro para las implementaciones continuas y las implementaciones azul/verde.

Habilitación de las etiquetas administradas

Determina si se activan las etiquetas administradas por Amazon ECS para las tareas del servicio.

Solo reflejarán la actualización las tareas iniciadas después de que se realice. Para actualizar las etiquetas de todas las tareas, use la opción de implementación forzada.

Puede cambiar este parámetro para las implementaciones continuas y las implementaciones azul/verde.

Habilitación de ECS Exec

Determina si se utiliza Amazon ECS Exec.

Si no desea anular el valor que se estableció cuando se creó el servicio, puede establecerlo en “null” al realizar esta acción.

Puede cambiar este parámetro para las implementaciones continuas.

Periodo de gracia de la comprobación de estado

El periodo de tiempo, en segundos, que el programador de servicio de Amazon ECS ignora las comprobaciones de estado en mal estado de Elastic Load Balancing, VPC Lattice y del contenedor después de que se haya iniciado una tarea por primera vez. Si no se especifica ningún valor para el periodo de gracia de comprobación de estado, se utiliza el valor predeterminado `0`. Si no se utiliza ninguna de las comprobaciones de estado, entonces `healthCheckGracePeriodSeconds` no se utilizará.

Si las tareas del servicio tardan mucho en iniciarse y en responder a las comprobaciones de estado, se puede especificar un periodo de gracia de hasta 2 147 483 647 segundos (aproximadamente 69 años). Durante este periodo, el programador de servicio de Amazon ECS hará caso omiso del resultado de las comprobaciones de estado. Este periodo de gracia puede

evitar que el programador de servicio interprete que las tareas están en mal estado y las detenga antes de que tengan tiempo de iniciarse.

Puede cambiar este parámetro para las implementaciones continuas y las implementaciones azul/verde.

Equilibradores de carga

Debe utilizar un rol vinculado a un servicio al actualizar un equilibrador de carga.

Una lista de objetos de equilibrador de carga de Elastic Load Balancing. Contiene el nombre del equilibrador de carga, el nombre del contenedor y el puerto del contenedor para acceder desde el equilibrador de carga. El nombre del contenedor es el que aparece en una definición de contenedor.

Amazon ECS no actualiza automáticamente los grupos de seguridad asociados a los balanceadores de carga de Elastic Load Balancing ni a las instancias de contenedor de Amazon ECS.

Cuando agrega, actualiza o elimina una configuración de equilibrador de carga, Amazon ECS inicia nuevas tareas con la configuración actualizada de Elastic Load Balancing y detiene las tareas antiguas cuando se ejecutan las nuevas.

Para los servicios que utilizan actualizaciones continuas, puede agregar, actualizar o eliminar grupos de destino de Elastic Load Balancing. Puede actualizar de un único grupo de destino a varios grupos de destino y de varios grupos de destino a un único grupo de destino.

Para los servicios que utilizan implementaciones azul/verde, puede actualizar los grupos de destino de Elastic Load Balancing mediante `CreateDeployment` a través de CodeDeploy. Tenga en cuenta que no se admiten varios grupos de destino en las implementaciones azul/verde. Para obtener más información, consulte Registrar varios grupos de destino con un servicio.

Para los servicios que utilizan el controlador de implementación externo, puede agregar, actualizar o eliminar equilibradores de carga mediante CreateTaskSet. Tenga en cuenta que no se admiten varios grupos de destino en las implementaciones externas. Para obtener más información, consulte Registrar varios grupos de destino con un servicio.

Pase una lista vacía para eliminar los equilibradores de carga.

Puede cambiar este parámetro para las implementaciones continuas.

Configuración de red

La configuración de red del servicio.

Puede cambiar este parámetro para las implementaciones continuas.

Restricciones para la ubicación

Una matriz de objetos de restricción de ubicación de la tarea para actualizar el servicio que se usará. Si no se especifica ningún valor, las restricciones de ubicación existentes para el servicio permanecerán sin cambios. Si se especifica este valor, anulará cualquier restricción de ubicación existente definida para el servicio. Para eliminar todas las restricciones de ubicación existentes, especifique una matriz vacía.

Puede especificar 10 restricciones como máximo para cada tarea. Este límite incluye las restricciones en la definición de tareas y las especificadas en el tiempo de puesta en marcha.

Puede cambiar este parámetro para las implementaciones continuas y las implementaciones azul/verde.

Estrategias de ubicación

Los objetos de la estrategia de colocación de tareas para actualizar el servicio que se usará. Si no se especifica ningún valor, la estrategia de ubicación existente para el servicio permanecerá sin cambios. Si se especifica este valor, anulará la estrategia de ubicación existente definida para el servicio. Para eliminar una estrategia de ubicación existente, especifique un objeto vacío.

Puede cambiar este parámetro para las implementaciones continuas y las implementaciones azul/verde.

Versión de la plataforma

La versión de la plataforma Fargate en la que se ejecuta su servicio.

Un servicio que utiliza una versión de la plataforma Linux no se puede actualizar para utilizar una versión de la plataforma Windows y viceversa.

Puede cambiar este parámetro para las implementaciones continuas.

Propagación de etiquetas

Determina si se deben propagar las etiquetas de la definición de tareas o el servicio para la tarea. Si no se especifica ningún valor, las etiquetas no se propagan.

Solo reflejarán la actualización las tareas iniciadas después de que se realice. Para actualizar las etiquetas de todas las tareas, configure `forceNewDeployment` en `true`, de modo que Amazon ECS inicie nuevas tareas con las etiquetas actualizadas.

Puede cambiar este parámetro para las implementaciones continuas y las implementaciones azul/verde.

Configuración de Service Connect

La configuración de Amazon ECS Service Connect. Este parámetro determina cómo se conecta el servicio a otros servicios de la aplicación.

Puede cambiar este parámetro para las implementaciones continuas.

Registros de servicios

Debe utilizar un rol vinculado a un servicio al actualizar los registros de servicios.

Los detalles de los registros de detección de servicios que se deben asignar a este servicio. Para obtener más información, consulte Detección de servicios.

Cuando agrega, actualiza o elimina la configuración de los registros de servicio, Amazon ECS inicia nuevas tareas con la configuración actualizada de los registros de servicio y detiene las tareas antiguas cuando se ejecutan las nuevas.

Pase una lista vacía para eliminar los registros de servicios.

Puede cambiar este parámetro para las implementaciones continuas.

Definición de tarea

La definición y la revisión de la tarea que se va a utilizar en el servicio.

Si cambia los puertos utilizados por los contenedores en una definición de tarea, es posible que tenga que actualizar los grupos de seguridad de las instancias del contenedor para que funcionen con los puertos actualizados.

Si actualiza la definición de la tarea del servicio, el nombre y el puerto del contenedor que se especificaron en la configuración del equilibrador de carga deben permanecer en la definición de la tarea.

El comportamiento de extracción de imágenes del contenedor varía según el tipo de lanzamiento. Para obtener más información, consulte una de las siguientes:

- Tipo de lanzamiento de Fargate para Amazon ECS

- Tipo de lanzamiento de EC2 para Amazon ECS
- Tipo de lanzamiento externo (Amazon ECS Anywhere) para Amazon ECS

Puede cambiar este parámetro para las implementaciones continuas.

Configuraciones de volúmenes

Los detalles del volumen que era `configuredAtLaunch`. Cuando `configuredAtLaunch` se establece como `true` en la definición de la tarea, este parámetro de servicio configura un volumen de Amazon EBS para cada tarea del servicio que se va a crear y adjuntar durante la implementación. Puede configurar el tamaño, el volumeType, el IOPS, el rendimiento, las instantáneas y el cifrado en ServiceManagedEBSVolumeConfiguration. El `name` del volumen debe coincidir con el `name` de la definición de la tarea. Si se establece como nulo, no se activa ninguna nueva implementación. De lo contrario, si esta configuración difiere de la existente, se activa una nueva implementación.

Puede cambiar este parámetro para las implementaciones continuas.

Configuración de VPC Lattice

La configuración de VPC Lattice para su servicio. Esto define la forma en que su servicio se integra con VPC Lattice para la comunicación de servicio a servicio.

Puede cambiar este parámetro para las implementaciones continuas.

Consideraciones sobre AWS CDK

El AWS CDK no rastrea los estados de los recursos. No sabe si está creando o actualizando un servicio. Los clientes deben usar la escotilla de escape para acceder directamente al constructo ecs `Service` L1.

Para obtener información sobre las escotillas de escape, consulte Personalice los constructos desde la Biblioteca de constructos de AWS en la Guía para desarrolladores de AWS Cloud Development Kit (AWS CDK) v2.

Para migrar su servicio existente al constructo `ecs.Service`, haga lo siguiente:

1. Use la escotilla de escape para acceder al constructo `Service` L1.
2. Establezca manualmente las siguientes propiedades en el constructo `Service` L1.

   Si su servicio utiliza capacidad de Amazon EC2:

- `daemon?`
- `placementConstraints?`
- `placementStrategies?`
- Si utiliza el modo de red `awsvpc`, debe configurar los constructos `vpcSubnets?` y `securityGroups?`.

Si su servicio usa Fargate:

- `FargatePlatformVersion`
- Los constructos `vpcSubnets?` y `securityGroups?`.

3. Configure el `launchType` como se indica a continuación:

```
const cfnEcsService = service.node.findChild('Service') as ecs.CfnService;
cfnEcsService.launchType = "FARGATE";
```

Para migrar de un tipo de lanzamiento a un proveedor de capacidad, haga lo siguiente:

1. Use la escotilla de escape para acceder al constructo `Service` L1.
2. Agregue el constructo `capacityProviderStrategies?`.
3. Implemente el servicio.

Actualización de un servicio de Amazon ECS

Después de crear un servicio, hay ocasiones en las que puede que necesite actualizar sus parámetros, por ejemplo, el número de tareas.

Cuando actualiza un servicio que utiliza un interruptor de Amazon ECS, Amazon ECS crea una implementación y una revisión de servicios. Estos recursos le permiten ver información detallada sobre el historial del servicio. Para obtener más información, consulte Visualización del historial de servicios mediante las implementaciones de servicios de Amazon ECS.

Requisitos previos

Antes de actualizar un servicio, compruebe qué parámetros del servicio se pueden cambiar para su tipo de implementación. Para obtener una lista completa de parámetros que se pueden modificar, consulte Actualización de los parámetros de servicio de Amazon ECS.

Procedimiento

Console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), elija el clúster.
3. En la página de detalles del clúster, en la sección Servicios, seleccione la casilla de verificación situada junto al servicio y, a continuación, seleccione Actualizar.
4. Para que su servicio inicie una nueva implementación, seleccione Force new deployment (Forzar una nueva implementación).
5. En Definición de tareas, elija la familia y la revisión de definiciones de tareas.

   > **Important**
   >
   > La consola valida que la familia y la revisión de definiciones de tareas seleccionadas sean compatibles con la configuración de cómputos definida. Si recibe una advertencia, compruebe la compatibilidad de la definición de tarea y la configuración de cómputos seleccionada.

6. Si eligió Replica (Réplica), en Desired tasks (Tareas deseadas), ingrese el número de tareas que se lanzarán y mantendrán en el servicio.
7. Si eligió Réplica para que Amazon ECS supervise la distribución de las tareas entre las zonas de disponibilidad y las redistribuya cuando haya un desequilibrio, en Reequilibrio del servicio de zonas de disponibilidad, seleccione Reequilibrio del servicio de zonas de disponibilidad.
8. En Max running tasks (Máximo de tareas en ejecución), ingrese el límite máximo del número de tareas del servicio que se permiten en el estado `RUNNING` durante una implementación, como porcentaje del número de tareas deseado del servicio (redondeado al entero inferior más próximo). Para obtener más información, consulte Configuración de la implementación.
9. En Max running tasks (Máximo de tareas en ejecución), ingrese el límite máximo del número de tareas del servicio que se permiten en el estado `RUNNING` o `PENDING` durante una implementación, como porcentaje del número de tareas deseado del servicio (redondeado al entero inferior más próximo).
10. Para configurar cómo se implementan las tareas para su servicio, expanda Opciones de implementación y, a continuación, configure sus opciones.

a. En Tipo de controlador de implementación, especifique el controlador de implementación del servicio. La consola de Amazon ECS admite los siguientes tipos de controladores: `ECS`.

b. En Estrategia de implementación, elija la estrategia que haya utilizado Amazon ECS para implementar nuevas versiones del servicio.

c. Dependiendo de la estrategia de implementación que elija, haga lo siguiente:

Estrategia de implementación	Pasos	
Actualización continua	a. En Porcentaje mínimo de tareas en ejecución, especifique un porcentaje mínimo de tareas que deben ejecutarse durante la implementación de un servicio. Para obtener más información, consulte Implementación de los servicios de Amazon ECS mediante el reemplazo de tareas. b. En Porcentaje máximo de tareas en ejecución, especifique un porcentaje máximo de tareas que deben ejecutarse durante la implementación de un servicio. Para obtener más información, consulte Implementación de los servicios de Amazon ECS mediante el reemplazo de tareas.	

Estrategia de implementación	Pasos	
Azul/verde	En Tiempo de incorporación, especifique una duración, en minutos, en la cual las revisiones de servicio azul y verde deben ejecutarse simultáneamente. Para obtener más información, consulte Implementaciones “blue/green” de Amazon ECS.	

d. Para ejecutar las funciones de Lambda en una etapa del ciclo de vida, en Enlaces de ciclo de vida de la implementación, haga lo siguiente para cada función de Lambda única:

   i. Elija Agregar.

      Repita este procedimiento para cada función única que desee ejecutar.

   ii. En Función de Lambda, ingrese el nombre de la función.

   iii. En Rol, elija el rol que creó en los requisitos previos con los permisos azul/verde.

      Para obtener más información, consulte Permisos necesarios para las funciones de Lambda en las implementaciones azul/verde de Amazon ECS.

   iv. En Etapas del ciclo de vida, seleccione las etapas que ejecuta la función de Lambda.

   v. (Opcional) En Detalles del enlace, introduzca un par clave-valor que proporcione información sobre el enlace.

11. Para configurar el modo en que Amazon ECS detecta y gestiona los errores de implementación, expanda Deployment failure detection (Detección de errores de implementación) y, a continuación, elija sus opciones.

a. Para detener una implementación cuando las tareas no puedan iniciarse, seleccione Use the Amazon ECS deployment circuit breaker (Utilizar el interruptor de circuito de implementación de Amazon ECS).

   Para que el software restaure automáticamente la implementación a su último estado completado cuando el disyuntor de implementación establezca un estado con error, seleccione Restauración en caso de error.

b. Para detener una implementación en función de las métricas de la aplicación, seleccione Use CloudWatch alarms. A continuación, elija las alarmas en Nombre de la alarma de CloudWatch. Para crear una alarma nueva, vaya a la consola de CloudWatch.

   Para que el software restaure automáticamente la implementación a su último estado de implementación completada cuando una alarma de CloudWatch establezca un estado con error, seleccione Restauración en caso de error.

12. Para cambiar las opciones de computación, expanda Configuración de computación y, a continuación, haga lo siguiente:

    a. Para los servicios en AWS Fargate, en Platform version (Versión de la plataforma), elija la nueva versión.

    b. Para los servicios que utilizan una estrategia de proveedor de capacidad, en Estrategia de proveedor de capacidad, haga lo siguiente:

       - Para agregar un proveedor de capacidad adicional, seleccione Agregar más. A continuación, en Proveedor de capacidad, seleccione el proveedor de capacidad.
       - Para eliminar un proveedor de capacidad, a la derecha del proveedor de capacidad, seleccione Eliminar.

       Un servicio que utiliza un proveedor de capacidad de grupos de escalado automático no se puede actualizar para que utilice un proveedor de capacidad de Fargate. Un servicio que utiliza un proveedor de capacidad de Fargate no puede actualizarse para utilizar un proveedor de capacidad de grupo de escalado automático.

13. (Opcional) Para configurar el escalado automático de servicios, expanda Escalado automático de servicios y, a continuación, especifique los siguientes parámetros. Para utilizar el escalado automático predictivo, que analiza los datos de cargas anteriores de los flujos de tráfico, configúrelo después de crear el servicio. Para obtener más información, consulte Uso de patrones históricos para escalar los servicios de Amazon ECS con escalado predictivo.

a. Para utilizar el escalado automático de servicios, seleccione Service auto scaling (Escalado automático de servicios).

b. En Cantidad mínima de tareas, ingrese el límite mínimo del número de tareas que se va a utilizar para el escalado automático del servicio. El recuento deseado no será inferior a este recuento.

c. En Cantidad máxima de tareas, ingrese el límite máximo del número de tareas que se va a utilizar para el escalado automático del servicio. El recuento deseado no será superior a este recuento.

d. Elija el tipo de política. En Tipo de política de escalado, elija una de las siguientes opciones.

Para utilizar este tipo de política	Haga lo siguiente	
Seguimiento de destino	a. Para Scaling policy type (Tipo de política de escalado), elija Target tracking (Seguimiento de destino). b. En Policy name (Nombre de la política), ingrese el nombre de la política. c. En Métrica de servicio de ECS, seleccione una de las siguientes métricas. • ECSServiceAverageCPUUtilization: uso medio de la CPU del servicio. • ECSServiceAverageMemoryUtilization: uso medio de la memoria del servicio. • ALBRequestCountPerTarget: número de peticiones completadas por destino en un grupo de destinos del equilibrador de carga de aplicación.	

Para utilizar este tipo de política	Haga lo siguiente	
	d. En Target value (Valor de destino), ingrese el valor que el servicio mantiene para la métrica seleccionada. e. En Periodo de recuperación de escalado horizontal, ingrese la cantidad de tiempo, en segundos, después de una actividad de escalado horizontal (agregar tareas) que debe transcurrir antes de que pueda iniciarse otra actividad de escalado horizontal. f. En Periodo de recuperación de desescalado horizonta l, ingrese la cantidad de tiempo, en segundos, después de una actividad de reducción horizontal (eliminar tareas) que debe transcurrir antes de que pueda iniciarse otra actividad de reducción horizontal. g. Para evitar que la política lleve a cabo	

Para utilizar este tipo de política	Haga lo siguiente	
	una actividad de reducción horizonta l, seleccione Turn off scale-in (Desactivar la reducción horizontal). h. • (Opcional) Seleccion e Desactivar la acción de desescalar horizonta lmente si desea que la política de escalado se escale horizonta lmente para adaptarse al aumento del tráfico, pero no necesita que se reduzca horizonta lmente cuando el tráfico disminuya.	

Para utilizar este tipo de política	Haga lo siguiente	
Escalado por pasos	a. Para Scaling policy type (Tipo de política de escalado), elija Step scaling (Escalado de pasos). b. En Nombre de política, ingrese el nombre de la política. c. En Alarm name, escriba un nombre único para la alarma. d. En Métrica de servicio de Amazon ECS, elija la métrica de servicio que desea utilizar para su alarma. e. En Estadística, elija la estadística de la alarma. f. En Período, elija el período de evaluación de la alarma. g. En Estado de alarma, elija cómo comparar la métrica seleccionada con el umbral definido. h. En Umbral para comparar las métricas y Período de evaluació n para iniciar la alarma, introduzca el umbral utilizado para la alarma	

Para utilizar este tipo de política	Haga lo siguiente	
	y durante cuánto tiempo se evaluará el umbral. i. En Acciones de escalado, haga lo siguiente: • Elija Acción si desea agregar, quitar o establecer un recuento específico para su servicio. • Si decide agregar o quitar las tareas, en Valor, ingrese el número de tareas (o porcentaje de las tareas existentes) que desea agregar o quitar cuando se inicie la acción de escalado. Si opta por establecer el recuento deseado, introduzca el número de tareas. En Tipo, seleccione si el Valor es un número entero o un porcentaje del recuento deseado existente. • En Límite inferior y Límite superior, introduzca el límite	

Para utilizar este tipo de política	Haga lo siguiente	
	inferior y el límite superior del ajuste de escalado de pasos. De forma predeterminada, el límite inferior para Agregar política es el umbral de la alarma y el límite superior es infinito positivo (+). De forma predeterminada, el límite superior de Quitar política es el límite de la alarma y el límite inferior es infinito negativo (-). • (Opcional) Agregue opciones de escalado adicional es. Seleccione Agregar nueva acción de escalado y, a continuación, repita los pasos de Acciones de escalado. • En Periodo de recuperación, ingrese la cantidad de tiempo, en segundos, que debe esperarse para que surta efecto	

Para utilizar este tipo de política	Haga lo siguiente	
	una actividad de reducción horizontal anterior. En el caso de una política de ampliación, este es el momento en el que, después de una actividad de escalado horizontal, la política de escalado bloquea las actividades de reducción horizontal y limita el número de tareas que se pueden escalar horizontalmente a la vez. En el caso de una política de reducción horizontal, este es el tiempo que debe transcurrir tras completarse una actividad de reducción horizontal antes de que pueda comenzar otra actividad de reducción horizontal.	

14. (Opcional) Para usar Service Connect, seleccione Turn on Service Connect (Activar Service Connect) y, a continuación, especifique lo siguiente:

    a. En Service Connect configuration (Configuración de Service Connect), especifique el modo cliente.

- Si su servicio ejecuta una aplicación cliente de red que solo necesita conectarse a otros servicios del espacio de nombres, elija Solo en el lado del cliente.
- Si su servicio ejecuta una aplicación de servicio web o red y necesita proporcionar puntos de conexión para este servicio y se conecta a otros servicios del espacio de nombres, elija Client and server (Cliente y servidor).

b. Para usar un espacio de nombres que no sea el espacio de nombres predeterminado del clúster, en Namespace (Espacio de nombres), elija el espacio de nombres del servicio.

15. Si su tarea usa un volumen de datos compatible con la configuración en el momento de la implementación, puede expandir Volume para configurar el volumen.

    El nombre y el tipo de volumen se configuran cuando crea una revisión de la definición de la tarea y no se pueden cambiar cuando se actualiza un servicio. Para actualizar el nombre y el tipo de volumen, debe crear una nueva revisión de la definición de la tarea y actualizar el servicio con la nueva revisión.

Para configurar este tipo de volumen	Haga lo siguiente	
Amazon EBS	a. En Tipo de volumen de EBS, elija el tipo de volumen de EBS que desee adjuntar a la tarea. b. En Tamaño (GiB), ingrese un valor válido para el tamaño de volumen en gibibytes (GiB). Puede especificar un tamaño de volumen mínimo de 1 GiB y máximo de 16 384 GiB. Este valor es obligatorio a menos que proporcione un ID de instantánea. c. En IOPS, ingrese el número máximo de	

Para configurar este tipo de volumen	Haga lo siguiente	
	operaciones de entrada/salida (IOPS) que debe proporcionar el volumen. Este valor solo puede configurarse para los tipos de volumen `io1`, `io2` y `gp3`. d. En Rendimiento (MiB/s), ingrese el rendimiento que debe proporcionar el volumen, en mebibytes por segundo (MiBps o MiB/s). Este valor solo puede configurarse para el tipo de volumen `gp3`. e. En ID de instantánea, elija una instantánea de volumen de Amazon EBS existente o ingrese el ARN de una instantán ea si desea crear un volumen a partir de una instantánea. También puede crear un volumen nuevo y vacío sin elegir ni ingresar ningún ID de instantánea. f. Si especifica un ID de instantánea, puede especificar una velocidad de inicialización del volumen (MiB/s). Ingrese un valor entre 100 y 300,	

Para configurar este tipo de volumen	Haga lo siguiente	
	en MiB/s, que determina rá la rapidez con la que se cargan los datos de la instantánea especific ada mediante el ID de instantánea para la creación del volumen. g. En Tipo de sistema de archivos, elija el tipo de sistema de archivos que se utilizará para almacenar y recuperar datos en el volumen. Puede elegir el sistema operativo predeterm inado o un tipo de sistema de archivos específico. El valor predeterminado para Linux es `XFS`. En el caso de los volúmenes creados a partir de una instantánea, debe especificar el mismo tipo de sistema de archivos que utilizaba el volumen cuando se creó la instantánea. Si hay un error de coinciden cia con el tipo de sistema de archivos, la tarea no podrá iniciarse.	

Para configurar este tipo de volumen	Haga lo siguiente	
	h. En Rol de infraestructura, elija un rol de IAM con los permisos necesarios que permitan a Amazon ECS administrar los volúmenes de Amazon EBS para las tareas. Puede adjuntar la política de `AmazonECSInfrastructureRolePolicyForVolumes` administrada al rol, o puede utilizar la política como guía para crear y adjuntar su propia política con los permisos que cumplan sus necesidades específicas. Para obtener información sobre los permisos de necesarios, consulte Rol de IAM de infraestructura de Amazon ECS. i. En Cifrado, elija Predeterminado si quiere usar el cifrado de Amazon EBS como configuración predeterminada. Si su cuenta tiene configurado Cifrado predeterminado, el volumen se cifrará	

Para configurar este tipo de volumen	Haga lo siguiente	
	con la clave AWS Key Management Service (AWS KMS) especificada en la configuración. Si selecciona Predeterminado y el cifrado predeterminado de Amazon EBS no está activado, el volumen se descifrará. Si elige Personalizado, puede especificar una AWS KMS key de su preferencia para el cifrado por volumen. Si selecciona Ninguno, el volumen no se cifrará a menos que tenga el cifrado configurado de forma predeterminada o si crea un volumen a partir de una instantánea cifrada. j. Si ha elegido Personalizado en Cifrado, debe especificar la AWS KMS key que desee utilizar. En Clave de KMS, elija una AWS KMS key o escriba un ARN. Si decide cifrar su volumen mediante una clave	

Para configurar este tipo de volumen	Haga lo siguiente	
	simétrica administrada por el cliente, asegúrese de tener los permisos correctos definidos en su política de AWS KMS key. Para obtener más información, consulte Data encryption for Amazon EBS volumes. k. (Opcional) En Etiquetas , puede propagar las etiquetas de la definición de la tarea o del servicio o proporcionar sus propias etiquetas para agregar etiquetas a su volumen de Amazon EBS. Si desea propagar etiquetas desde la definición de la tarea, seleccione Definición de tarea en Propagar etiquetas desde. Si desea propagar etiquetas desde el servicio, elija Servicio en Propagar etiquetas desde. Si elige No propagar o si no elige un valor, las etiquetas no se propagarán.	

Para configurar este tipo de volumen	Haga lo siguiente	
	Si quiere proporcionar sus propias etiquetas , seleccione Agregar etiqueta y, a continuac ión, proporcione la clave y el valor de cada etiqueta que agregue. Para obtener más información acerca del etiquetado de volúmenes de Amazon EBS, consulte Tagging Amazon EBS volumes.	

16. (Opcional) Para ayudar a identificar su servicio, expanda la sección Tags (Etiquetas) y, a continuación, configure sus etiquetas.
    - [Agregar una etiqueta] Elija Agregar etiqueta y haga lo siguiente:
        - En Clave, escriba el nombre de la clave.
        - En Valor, escriba el valor de la clave.
    - [Eliminar una etiqueta] Junto a la etiqueta, elija Remove tag (Quitar etiqueta).
17. Elija Actualizar.

AWS CLI

- Ejecute `update-service`. Para obtener información sobre la ejecución del comando, consulte update-service en la Referencia de AWS Command Line Interface.

  En el siguiente ejemplo de `update-service`, se actualiza el recuento de tareas deseado del servicio `my-http-service` a 2.

  Sustituya las *`entradas del usuario`* por sus valores.

```
aws ecs update-service \
    --cluster MyCluster \
    --service my-http-service \
    --desired-count 2
```

Pasos a seguir a continuación

Haga un seguimiento de la implementación y consulte el historial de los servicios que utiliza el interruptor de Amazon ECS. Para obtener más información, consulte Visualización del historial de servicios mediante las implementaciones de servicios de Amazon ECS.

Eliminación de un servicio de Amazon ECS mediante la consola

Los siguientes son algunos de los motivos por los que eliminaría un servicio:

- Ya no se necesita la aplicación
- Va a migrar el servicio a un nuevo entorno
- La aplicación no se utiliza activamente
- La aplicación utiliza más recursos de los necesarios y usted está intentando optimizar sus costos

El servicio se reduce verticalmente a cero de forma automática antes de eliminarse. Los recursos del equilibrador de carga o la detección del servicio con recursos asociados al servicio no se ven afectados por la eliminación del servicio. Para eliminar los recursos de Elastic Load Balancing, consulte uno de los siguientes temas, según el tipo de balanceador de carga: Eliminación de un Application Load Balancer o Eliminación de un Network Load Balancer.

Al eliminar un servicio, Amazon ECS elimina todas las implementaciones y revisiones del servicio.

Cuando se elimina un servicio, si todavía hay tareas en ejecución que requieren limpieza, el estado del servicio pasa de `ACTIVE` a `DRAINING` y deja de estar visible en la consola o en la operación de la API de `ListServices`. Unas vez que las tareas tengan el estado `STOPPING` o `STOPPED`, el estado del servicio cambia de `DRAINING` a `INACTIVE`. Los servicios en el estado `DRAINING` o `INACTIVE` se pueden seguir viendo la operación `DescribeServices` de la API.

> **Important**
>
> Si intenta crear un nuevo servicio con el mismo nombre que un servicio existente en cualquiera de los estados `ACTIVE` o `DRAINING`, recibirá un mensaje de error.

Procedimiento

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clusters (Clústeres), seleccione el clúster para el servicio.
3. En la página Clusters (Clústeres), elija el clúster.
4. En la página de Cluster: ***name*** (Clúster: nombre), elija la pestaña de Services (Servicios).
5. Seleccione los servicios y, a continuación, elija Delete (Eliminar).
6. Para eliminar un servicio aunque no se haya reducido a cero tareas, seleccione Force delete service (Forzar la eliminación del servicio).
7. En la pregunta de confirmación, escriba delete (eliminar) y, a continuación, elija Eliminar.

Migración de un ARN de servicio corto de Amazon ECS a un ARN largo

Amazon ECS asigna un nombre de recurso de Amazon (ARN) único a cada servicio. Los servicios que se crearon antes de 2021 tienen un formato de ARN corto:

```
arn:aws:ecs:region:aws_account_id:service/service-name
```

Amazon ECS cambió el formato del ARN para incluir el nombre del clúster. Este es un formato de ARN largo:

```
arn:aws:ecs:region:aws_account_id:service/cluster-name/service-name
```

Su servicio debe tener el formato de ARN largo para poder etiquetarlo.

Puede migrar un servicio con un formato de ARN corto al formato de ARN largo sin tener que volver a crear el servicio. Puede utilizar la API, la CLI o la consola. No puede deshacer una operación de migración.

El proceso de migración es fluido y garantiza que su servicio no tenga ningún tiempo de inactividad. Durante la migración:

- Disponibilidad del servicio: su servicio sigue funcionando con normalidad sin interrumpir el tráfico ni la funcionalidad.
- Tareas en ejecución: las tareas existentes siguen ejecutándose sin interrupciones. Las nuevas tareas que se inicien después de la migración utilizarán el formato de ARN largo si la configuración de la cuenta `taskLongArnFormat` está habilitada.
- Instancias de contenedor: las instancias de contenedor no se ven afectadas por la migración del ARN del servicio y siguen funcionando con normalidad.
- Configuración del servicio: todos los ajustes del servicio, incluidas las configuraciones de definición de tareas, redes y equilibrador de carga permanecen sin cambios.

Si desea utilizar AWS CloudFormation para etiquetar un servicio con un formato de ARN corto, debe migrar el servicio mediante la API, la CLI o la consola. Una vez completada la migración, puede usar AWS CloudFormation para etiquetar el servicio.

Si desea utilizar Terraform para etiquetar un servicio con un formato de ARN corto, debe migrar el servicio mediante la API, la CLI o la consola. Una vez completada la migración, puede usar Terraform para etiquetar el servicio.

Después de completar la migración, el servicio presenta los siguientes cambios:

- Formato de ARN largo

  `arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:service/`*`cluster-name`*`/`*`service-name`*

- Al migrar mediante la consola, Amazon ECS agrega una etiqueta al servicio con la clave establecida en “ecs:serviceArnMigratedAt” y el valor establecido en la marca de tiempo de la migración (formato UTC).

  Esta etiqueta se tiene en cuenta para su cuota de etiquetas.
- Cuando el valor de `PhysicalResourceId` en una pila de AWS CloudFormation representa un ARN de servicio, el valor no cambia y seguirá siendo el ARN de servicio corto.

Requisitos previos

Realice las siguientes operaciones antes de migrar el ARN de servicio.

1. Para comprobar si tiene un ARN de servicio corto, consulte los detalles del servicio en la consola de Amazon ECS (verá una advertencia cuando el servicio tenga el formato de ARN corto) o

el parámetro de devolución `serviceARN` de `describe-services`. Si el ARN no incluye el nombre del clúster, tiene un ARN corto. A continuación se muestra el formato de un ARN corto:

```
arn:aws:ecs:region:aws_account_id:service/service-name
```

2. Tenga en cuenta la fecha de creación.
3. Si tiene políticas de IAM que utilizan el formato de ARN corto, actualícelas al formato de ARN largo.

   Reemplace cada *marcador de posición de entrada del usuario* con información propia.

   ```
   arn:aws:ecs:region:aws_account_id:service/cluster-name/service-name
   ```

   Para obtener más información, consulta Políticas de IAM de edición en la Guía del usuario de AWS Identity and Access Management.
4. Si tiene herramientas que utilizan el formato de ARN corto, actualícelas al formato de ARN largo.

   Reemplace cada *marcador de posición de entrada del usuario* con información propia.

   ```
   arn:aws:ecs:region:aws_account_id:service/cluster-name/service-name
   ```
5. Active el formato de ARN largo del servicio. Ejecute `put-account-setting` con la opción `serviceLongArnFormat` establecida en `enabled`. Para obtener más información, consulte put-account-setting en la Referencia de la API de Amazon Elastic Container Service.

   Ejecute el comando como usuario root cuando el servicio tenga un valor de fecha desconocido para `createdAt`.

   ```
   aws ecs put-account-setting --name serviceLongArnFormat --value enabled
   ```

   Ejemplo de resultado

   ```
   {
       "setting": {
           "name": "serviceLongArnFormat",
           "value": "enabled",
           "principalArn": "arn:aws:iam::123456789012:role/your-role",
           "type": user
       }
   ```

```
}
```

6. Active el formato de ARN largo de la tarea. Esta configuración de cuenta controla el formato de ARN de las nuevas tareas que se inician una vez finalizada la migración del servicio. Ejecute `put-account-setting` con la opción `taskLongArnFormat` establecida en `enabled`. Para obtener más información, consulte put-account-setting en la Referencia de la API de Amazon Elastic Container Service.

   Ejecute el comando como usuario root cuando el servicio tenga un valor de fecha desconocido para `createdAt`.

```
aws ecs put-account-setting --name taskLongArnFormat --value enabled
```

   Ejemplo de resultado

```
{
    "setting": {
        "name": "taskLongArnFormat",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:role/your-role",
        "type": user
    }
}
```

> **Note**
>
> La configuración `taskLongArnFormat` no migra directamente las tareas existentes. Solo afecta al formato del ARN de las nuevas tareas que se crean después de habilitar la configuración. Las tareas en ejecución existentes retienen su formato de ARN actual hasta que se sustituyan mediante las operaciones de servicio normales (como las implementaciones o las actividades de escalado).

Procedimiento

Utilice lo siguiente para migrar el ARN de su servicio.

Consola

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.

2. En la página Clusters (Clústeres), elija el clúster.
3. En la sección Servicios, elija un servicio que tenga una advertencia en la columna ARN.

   Se abrirá la página de detalles del servicio.
4. Elija Migrar a un ARN largo.

   Aparecerá el cuadro de diálogo del servicio de migración.
5. Seleccione Migrar.

CLI

Después de completar los requisitos previos, puede etiquetar su servicio. Ejecuta el siguiente comando:

Amazon ECS considera la posibilidad de transferir el formato de ARN largo a una solicitud de API `tag-resource` para un servicio con un ARN corto como señal para migrar el servicio y usar el formato de ARN largo.

```
aws ecs tag-resource \
    --resource-arn arn:aws:ecs:region:aws_account_id:service/cluster-name/service-name
    --tags key=key1,value=value1
```

El siguiente ejemplo etiqueta MyService con una etiqueta que tiene una clave establecida en “TestService” y un valor establecido en “WebServers:

```
aws ecs tag-resource \
    --resource-arn arn:aws:ecs:us-east-1:123456789012:service/MyCluster/MyService
    --tags key=TestService1,value=WebServers
```

Terraform

Después de completar los requisitos previos, puede etiquetar su servicio. Cree un recurso `aws_ecs_service` y establezca la referencia `tags`. Para obtener más información, consulte Resource: aws_ecs_service en la documentación de Terraform.

```
resource "aws_ecs_service" "MyService" {
  name    = "example"
  cluster = aws_ecs_cluster.MyService.id
```

```
 tags = {
 "Name"  =  "MyService"
 "Environment"  =  "Production"
 "Department"  =  "QualityAssurance"
  }
}
```

Pasos a seguir a continuación

Puede agregar etiquetas al servicio. Para obtener más información, consulte Adición de etiquetas a los recursos de Amazon ECS.

Si desea que Amazon ECS propague las etiquetas desde la definición de tareas o el servicio a la tarea, ejecute `update-service` con el parámetro `propagateTags`. Para obtener más información, consulte update-service en Referencia de la AWS Command Line Interface.

Solución de problemas

Algunos usuarios pueden encontrar el siguiente error al migrar del formato ARN corto al formato ARN largo.

```
There was an error while migrating the ARN of service service-name. The
specified account does not have serviceLongArnFormat or taskLongArnFormat
account settings enabled. Add account settings in order to enable tagging.
```

Si ya ha activado la configuración de la cuenta de `serviceLongArnFormat`, pero sigue apareciendo este error, puede que se deba a que la configuración de la cuenta para el formato ARN largo no esté activada para la entidad principal de IAM específica que creó originalmente el servicio.

1. Identifica la entidad principal que creó el servicio.

   1. En la consola, esta información está disponible en el campo Creado por de la pestaña Configuración y redes, dentro de la página de detalles del servicio de la consola de Amazon ECS.
   2. En el caso de la AWS CLI, ejecute el siguiente comando:

      Sustituya las *entradas del usuario* por sus valores.

      ```
      aws ecs describe-services --cluster cluster-name --services service-name --query
       'services[0].{createdBy: createdBy}'
      ```

2. Active la configuración de cuenta requerida para esa entidad principal específica. Puedes hacerlo de una de las siguientes formas:

   a. Asuma el usuario o el rol de IAM para esa entidad principal. A continuación, ejecute `put-account-setting`.

   b. Utilice el usuario raíz para ejecutar el comando y, al mismo tiempo, especifique la creación de la entidad principal con `principal-arn`.

      Ejemplo.

      Reemplace el valor de *`principal-arn`* por el valor del paso 1.

```
aws ecs put-account-setting --name serviceLongArnFormat --value enabled --
principal-arn arn:aws:iam::123456789012:role/jdoe
```

Ambos métodos permiten la configuración de cuenta de `serviceLongArnFormat` requerida en la entidad principal que creó el servicio, de modo que se puede continuar con la migración del ARN.

Lógica de limitación controlada de servicios de Amazon ECS

El programador de servicios de Amazon ECS incluye una lógica de protección que limita la frecuencia con la que se lanzan las tareas del servicio si no pueden iniciarse tras varios intentos. Esto ayuda a evitar el consumo innecesario de recursos y reduce los costos.

Cuando las tareas de un servicio no pasan del estado `PENDING` a `RUNNING` y, en lugar de ello, pasan directamente a `STOPPED`, el programador:

- Aumenta gradualmente el tiempo entre los intentos de reinicio
- Sigue aumentando los retrasos hasta un máximo de 27 minutos entre intentos
- Genera un mensaje de evento de servicio para notificarle el problema

> **Note**
>
> El periodo máximo de retraso de 27 minutos puede cambiar en futuras actualizaciones.

Cuando se active la limitación, recibirá este mensaje de evento de servicio:

```
(service service-name) is unable to consistently start tasks successfully.
```

Características importantes de la lógica de limitación:

- Los servicios continúan reintentándolo indefinidamente
- La única modificación es el aumento del tiempo entre reinicios
- No hay parámetros configurables por el usuario

Resolución de problemas de limitación

Para resolver un problema de limitación, puede hacer lo siguiente:

- Actualice el servicio para utilizar una nueva definición de tarea, que inmediatamente devuelve el servicio a una operación normal, sin limitaciones. Para obtener más información, consulte Actualización de un servicio de Amazon ECS.
- Aborde la causa subyacente de los errores en las tareas.

Entre las causas más comunes de los errores en las tareas que provocan la limitación se incluyen las siguientes:

- Recursos de clúster insuficientes (puertos, memoria o CPU)
  - Lo indica un mensaje de evento de servicio de recursos insuficientes
- Fallos al extraer la imagen del contenedor
  - Puede deberse a nombres o etiquetas de imagen no válidos o a permisos insuficientes
  - Da como resultado `CannotPullContainerError` en Visualización de los errores de las tareas detenidas de Amazon ECS
- Espacio en disco insuficiente
  - Da como resultado `CannotCreateContainerError` en errores de tareas detenidas
  - Para ver los pasos de resolución, consulte Solución del problema de Docker `API error (500): devmapper` en Amazon ECS

> **Important**
>
> Los siguientes escenarios NO activan la lógica de limitación:

- Tareas que se detienen después de alcanzar el estado `RUNNING`
- Se detuvieron tareas debido a comprobaciones de estado de Elastic Load Balancing con errores
- Tareas en las que el comando de contenedor se cierra con un código distinto de cero después de alcanzar el estado `RUNNING`

Parámetros de definición de servicio de Amazon ECS

Una definición de servicio define cómo ejecutar el servicio de Amazon ECS. Los siguientes parámetros se pueden especificar en una definición de servicio.

Tipo de lanzamiento

`launchType`

Tipo: cadena

Valores válidos: `EC2` | `FARGATE` | `EXTERNAL`

Obligatorio: no

El tipo de lanzamiento en el que ejecutar su servicio. Si no se especifica ningún tipo de lanzamiento, se usará `capacityProviderStrategy` de forma predeterminada. Para obtener más información, consulte Tipos de lanzamiento de Amazon ECS.

Si se especifica `launchType`, se debe omitir el parámetro `capacityProviderStrategy`.

Estrategia de proveedores de capacidad

`capacityProviderStrategy`

Tipo: Matriz de objetos

Obligatorio: no

La estrategia de proveedores de capacidad que se utilizará para el servicio.

Una estrategia de proveedores de capacidad consiste en uno o más proveedores de capacidad junto con la `base` y `weight` que se les asigne. Un proveedor de capacidad debe estar

asociado con el clúster que se utilizará en una estrategia de proveedores de capacidad. La API PutClusterCapacityProviders se utiliza para asociar un proveedor de capacidad a un clúster. Solo se pueden utilizar los proveedores de capacidad con un estado `UPDATING` o `ACTIVE`.

Si se especifica `capacityProviderStrategy`, se debe omitir el parámetro `launchType`. Si no se especifica `capacityProviderStrategy` o `launchType`, se utiliza `defaultCapacityProviderStrategy` para el clúster.

Si desea especificar un proveedor de capacidad que utiliza un grupo de escalado automático, el proveedor de capacidad debe estar ya creado. Se pueden crear nuevos proveedores de capacidad con la operación CreateCapacityProvider de la API.

Para utilizar un proveedor de capacidad de Fargate de AWS, especifique los proveedores de capacidad `FARGATE` o `FARGATE_SPOT`. Los proveedores de capacidad de AWS Fargate están disponibles para todas las cuentas y solo necesitan estar asociados al clúster que se va a utilizar.

La operación PutClusterCapacityProviders de la API se utiliza para actualizar la lista de proveedores de capacidad disponibles para un clúster después de que se haya creado el clúster.

`capacityProvider`

Tipo: cadena

Obligatorio: sí

El apodo o el nombre de recurso de Amazon (ARN) del proveedor de capacidad.

`weight`

Tipo: entero

Rango válido: números enteros entre 0 y 1000.

Obligatorio: no

El valor peso designa el porcentaje relativo del número total de tareas lanzadas que utiliza el proveedor de capacidad especificado.

Por ejemplo, suponga que tiene una estrategia que contiene dos proveedores de capacidad y ambos tienen una ponderación de uno. Cuando se completa la base, las tareas se dividen en partes iguales entre los dos proveedores de capacidad. Con la misma lógica, suponga que especifica un peso de 1 para capacityProviderA y un peso de 4 para capacityProviderB.

Luego, para cada tarea que se ejecute con capacityProviderA, cuatro tareas utilizan capacityProviderB.

`base`

Tipo: entero

Rango válido: números enteros entre 0 y 100 000.

Obligatorio: no

El valor de base designa cuántas tareas, como mínimo, se ejecutarán en el proveedor de capacidad especificado. Solo un proveedor de capacidad en una estrategia de proveedor de capacidad puede tener una base definida.

Definición de tarea

`taskDefinition`

Tipo: cadena

Requerido: no

La `family` y `revision` (`family:revision`) o el nombre de recurso de Amazon (ARN) completo de la definición de tareas que se va a ejecutar en el servicio. Si no se especifica una `revision`, se utiliza la última revisión `ACTIVE` de la familia especificada.

Debe especificarse una definición de tarea cuando se utiliza el controlador de implementación de actualización continua (`ECS`).

Sistema operativo de la plataforma

`platformFamily`

Tipo: String

Obligatorio: condicional

Predeterminado: Linux

Este parámetro es necesario para los servicios de Amazon ECS alojados en Fargate.

Este parámetro se ignora para los servicios de Amazon ECS alojados en Amazon EC2.

El sistema operativo de los contenedores que ejecuta el servicio. Los valores válidos son `LINUX`, `WINDOWS_SERVER_2019_FULL`, `WINDOWS_SERVER_2019_CORE`, `WINDOWS_SERVER_2022_FULL` y `WINDOWS_SERVER_2022_CORE`.

El valor `platformFamily` para cada tarea que especifique para el servicio debe coincidir con el servicio del valor `platformFamily`. Por ejemplo, si configuró el `platformFamily` a `WINDOWS_SERVER_2019_FULL`, el valor `platformFamily` para todas las tareas debe ser `WINDOWS_SERVER_2019_FULL`.

Versión de la plataforma

`platformVersion`

Tipo: cadena

Requerido: no

La versión de la plataforma en la que se ejecutan sus tareas en el servicio. La versión de la plataforma solo se especifica para las tareas que utilizan el tipo de lanzamiento de Fargate. Si no se especifica ninguna, se usará la versión más reciente (`LATEST`) de forma predeterminada.

Las versiones de la plataforma de AWS Fargate se utilizan para hacer referencia a un entorno en tiempo de ejecución específico para la infraestructura de tareas de Fargate. Cuando se especifica la versión de la plataforma `LATEST` al ejecutar una tarea o crear un servicio, se obtiene la versión más actual de la plataforma disponible para las tareas. Cuando se escala un servicio, esas tareas reciben la versión de la plataforma especificada en la implementación actual del servicio. Para obtener más información, consulte Versiones de la plataforma Fargate para Amazon ECS.

> **Note**
>
> No se especifican las versiones de la plataforma para las tareas que utilizan el tipo de lanzamiento de EC2.

Clúster

`cluster`

Tipo: cadena

Requerido: no

El nombre abreviado o nombre de recurso de Amazon (ARN) completo del clúster en el que ejecutar el servicio. Si no especifica un clúster, se supone el clúster `default`.

Nombre del servicio

`serviceName`

Tipo: cadena

Obligatorio: sí

El nombre de su servicio. Se admiten hasta 255 letras (mayúsculas y minúsculas), números, guiones y caracteres de subrayado. Los nombres de servicio deben ser únicos dentro de un clúster, pero puede tener servicios con el mismo nombre en varios clústeres dentro de una región o en varias regiones.

Estrategia de programación

`schedulingStrategy`

Tipo: cadena

Valores válidos: `REPLICA` | `DAEMON`

Obligatorio: no

La estrategia de programación que se va a utilizar. Si no se especifica ninguna estrategia de programación, se utiliza la estrategia de `REPLICA`. Para obtener más información, consulte Servicios de Amazon ECS.

Existen dos estrategias del programador de servicio:

- `REPLICA`: la estrategia de programación de réplicas sitúa y mantiene en el clúster el número de tareas deseado. De forma predeterminada, el programador de servicio distribuye las tareas en zonas de disponibilidad. Puede utilizar estrategias y restricciones de ubicación de tareas para personalizar las decisiones de ubicación de las tareas. Para obtener más información, consulte Estrategia de programación de réplicas.
- `DAEMON`: la estrategia de programación del daemon implementa exactamente una tarea en cada instancia de contenedor activa que cumpla todas las restricciones de ubicación de

tareas que se especifiquen para el clúster. Cuando se utiliza esta estrategia, no es necesario especificar un número deseado de tareas, ni una estrategia de ubicación de tareas ni utilizar políticas de Auto Scaling de servicios. Para obtener más información, consulte Estrategia de programación de daemon.

> **Note**
>
> Las tareas de Fargate no admiten la estrategia de programación de `DAEMON`.

Recuento deseado

`desiredCount`

Tipo: entero

Obligatorio: no

El número de instancias de la definición de tarea especificada para ubicar y seguir ejecutando en el servicio.

Este parámetro es necesario si se utiliza la estrategia de programación de `REPLICA`. Si el servicio utiliza la estrategia de programación de `DAEMON`, este parámetro es opcional.

Cuando usa el escalado automático de servicios, si actualiza un servicio que se está ejecutando actualmente con un valor de `desiredCount` inferior al número de tareas que se están ejecutando actualmente, el servicio se reduce verticalmente al valor de `desiredCount` especificado.

Configuración de implementación

`deploymentConfiguration`

Tipo: objeto

Obligatorio: no

Parámetros de implementación opcionales que controlan cuántas tareas se ejecutan durante la implementación y la ordenación de tareas de parada e inicio.

`maximumPercent`

Tipo: entero

Obligatorio: no

Si un servicio utiliza el tipo de implementación de actualización continua (`ECS`), el parámetro `maximumPercent` representa un límite superior en el número de tareas de servicio que se permiten en el estado `RUNNING`, `STOPPING` o `PENDING` durante una implementación. Se expresa como un porcentaje de `desiredCount` que se redondea al número entero más cercano. Puede utilizar este parámetro para definir el tamaño del lote de implementación. Por ejemplo, si el servicio utiliza el programador de servicios `REPLICA` y tiene un `desiredCount` de cuatro tareas y un valor `maximumPercent` de 200 %, el programador inicia cuatro nuevas tareas antes de detener las cuatro tareas más antiguas. Esto es siempre que los recursos del clúster requeridos para hacer esto estén disponibles. El valor `maximumPercent` predeterminado para un servicio que utiliza el programador de servicio `REPLICA` es 200 %.

El programador de Amazon ECS usa este parámetro para reemplazar las tareas en mal estado. Para ello, inicia primero las tareas de reemplazo y, luego, las detiene, siempre que haya recursos del clúster disponibles para iniciar las tareas de reemplazo. Para obtener más información sobre cómo el programador reemplaza las tareas en mal estado, consulte Servicios de Amazon ECS.

Si el servicio utiliza el tipo de programador de servicio `DAEMON`, el `maximumPercent` debería permanecer al 100 %. Este es el valor predeterminado.

El número máximo de tareas durante una implementación es el `desiredCount` multiplicado por el `maximumPercent`/100, redondeado al valor del entero inferior más próximo.

Si un servicio está utilizando el tipo de implementación blue/green (`CODE_DEPLOY`) o `EXTERNAL` y tareas que usan el tipo de lanzamiento EC2, el valor de porcentaje máximo se establece en el valor predeterminado. El valor se usa para definir el límite superior del número de las tareas en el servicio que permanecen en el estado `RUNNING` al mismo tiempo que las instancias de contenedor se encuentran en el estado `DRAINING`.

> **Note**
>
> No puede especificar un valor `maximumPercent` personalizado para un servicio que utilice los tipos azul/verde (`CODE_DEPLOY`) o de implementación `EXTERNAL` y que tenga tareas que utilicen el tipo de lanzamiento de EC2.

Si el servicio utiliza los tipos de implementación azul/verde (`CODE_DEPLOY`) o `EXTERNAL`, y las tareas del servicio utilizan el tipo de lanzamiento Fargate, no se utilizará el valor porcentual máximo. El valor aún se devuelve al describir el servicio.

`minimumHealthyPercent`

Tipo: entero

Obligatorio: no

Si un servicio utiliza el tipo de implementación de actualización continua (`ECS`), el parámetro `minimumHealthyPercent` representa un límite inferior en el número de tareas de servicio que permanecen en el estado `RUNNING` durante una implementación. Se expresa como un porcentaje de `desiredCount` que se redondea al número entero más cercano. Puede utilizar este parámetro para implementar sin utilizar capacidad de clúster adicional.

Por ejemplo, si el servicio tiene un parámetro `desiredCount` de cuatro tareas, un parámetro `minimumHealthyPercent` del 50 % y un parámetro `maximumPercent` del 100 %, el programador de servicio detiene dos tareas existentes para liberar capacidad de clúster antes de iniciar dos nuevas tareas.

Si alguna tarea se encuentra en mal estado y `maximumPercent` no permite que el programador de Amazon ECS inicie tareas de reemplazo, el programador detiene las tareas en mal estado una por una (y utiliza `minimumHealthyPercent` como una restricción) para liberar capacidad y lanzar tareas de reemplazo. Para obtener más información sobre cómo el programador reemplaza las tareas en mal estado, consulte Servicios de Amazon ECS.

Para los servicios que no utilizan un equilibrador de carga, tenga en cuenta lo siguiente:

- Se considera que un servicio está en buen estado si todos los contenedores esenciales dentro de las tareas del servicio superan sus comprobaciones de estado.
- Si una tarea no tiene contenedores esenciales con una comprobación de estado definida, el programador de servicios esperará 40 segundos después de que una tarea alcance un

estado de `RUNNING` antes de que la tarea se cuente hacia el porcentaje total mínimo de buen estado.

- Si una tarea tiene uno o más contenedores esenciales con una comprobación de estado definida, el programador de servicios esperará a que la tarea alcance un buen estado antes de contarla hacia el porcentaje total mínimo de buen estado. Se considera que una tarea está en buen estado cuando todos los contenedores esenciales de la tarea han superado sus comprobaciones de estado. El tiempo que puede esperar el programador de servicios viene determinado por la configuración de comprobación de estado del contenedor. Para obtener más información, consulte Comprobación de estado.

Para los servicios que sí utilizan un equilibrador de carga, tenga en cuenta lo siguiente:

- Si una tarea no tiene contenedores esenciales con una comprobación de estado definida, el programador de servicios esperará a que la comprobación de estado del grupo de destino del equilibrador de carga devuelva un “buen estado” antes de contar la tarea hacia el porcentaje total mínimo del estado correcto.
- Si una tarea tiene un contenedor esencial con una comprobación de estado definida, el programador de servicios esperará a que la tarea alcance un buen estado y a que la comprobación de estado del grupo de destino del equilibrador de carga devuelva un “buen estado” antes de contar la tarea hacia el porcentaje total mínimo del estado correcto.

El valor predeterminado para un servicio de réplica de `minimumHealthyPercent` es del 100%. El valor `minimumHealthyPercent` predeterminado para un servicio que utiliza el programador de servicios `DAEMON` es del 0 % para la AWS CLI, los SDK de AWS y las API, y del 50 % para la AWS Management Console.

El número mínimo de tareas en buen estado durante una implementación es el `desiredCount` multiplicado por el `minimumHealthyPercent`/100, redondeado al valor del entero superior más próximo.

Si un servicio utiliza el tipo de implementación “blue/green” (`CODE_DEPLOY`) o `EXTERNAL` y ejecuta tareas que usan el tipo de lanzamiento de EC2, los valores de porcentaje mínimo en buen estado y porcentaje máximo se establecen en los valores predeterminados. El valor se usa para definir el límite inferior del número de las tareas en el servicio que permanecen en el estado `RUNNING` al mismo tiempo que las instancias de contenedor se encuentran en el estado `DRAINING`.

> **Note**
>
> No puede especificar un valor `maximumPercent` personalizado para un servicio que utilice los tipos azul/verde (`CODE_DEPLOY`) o de implementación `EXTERNAL` y que tenga tareas que utilicen el tipo de lanzamiento de EC2.

Si un servicio utiliza los tipos de implementación “blue/green” (`CODE_DEPLOY`) o `EXTERNAL` y ejecuta tareas que utilizan el tipo de lanzamiento de Fargate, el valor de porcentaje mínimo de estado correcto no se utiliza, aunque se devuelva al describir el servicio.

Controlador de implementación

`deploymentController`

Tipo: objeto

Obligatorio: no

El controlador de implementación que utilizar para el servicio. Si no se especifica ningún controlador de implementación, se utiliza el controlador `ECS`. Para obtener más información, consulte Servicios de Amazon ECS.

`type`

Tipo: cadena

Valores válidos: `ECS` | `CODE_DEPLOY` | `EXTERNAL`

Obligatorio: sí

El tipo de controlador de implementación que se va a utilizar. Existen tres tipos de controlador de implementación disponibles:

`ECS`

El tipo de implementación de actualización acumulativa (`ECS`) implica la sustitución de la versión de ejecución actual del contenedor por la versión más reciente. Para controlar el número de contenedores que Amazon ECS agrega o elimina del servicio durante una actualización acumulativa, se ajusta el número mínimo y máximo de tareas en estado

correcto permitidas durante una implementación de servicio, tal y como se especifica en deploymentConfiguration.

`CODE_DEPLOY`

El tipo de implementación “blue/green” (`CODE_DEPLOY`) utiliza el modelo de implementación “blue/green” (azul/verde) con tecnología de CodeDeploy, que le permite verificar una nueva implementación de un servicio antes de enviarle tráfico de producción.

`EXTERNAL`

Utilice el tipo de implementación externa cuando quiera usar cualquier controlador de implementación de terceros para tener un control completo del proceso de implementación de un servicio de Amazon ECS.

Ubicación de tareas

`placementConstraints`

Tipo: Matriz de objetos

Obligatorio: no

Una matriz de objetos de restricción de colocación que utilizar para tareas en su servicio. Puede especificar 10 restricciones como máximo por tarea. Este límite incluye restricciones en la definición de tareas y las especificadas en tiempo de ejecución. Si utiliza el tipo de lanzamiento de Fargate, no se admiten las restricciones de ubicación de tareas.

`type`

Tipo: cadena

Requerido: no

El tipo de restricción. Para garantizar que cada tarea de un determinado grupo se ejecute en una instancia de contenedor diferente, utilice `distinctInstance`. Utilice `memberOf` para restringir la selección a un grupo de candidatos válidos. El valor `distinctInstance` no se admite en las definiciones de tareas.

`expression`

Tipo: cadena

Requerido: no

Una expresión de lenguaje de consulte de clúster que aplicar a la restricción. No puede especificar una expresión si el tipo de restricción es `distinctInstance`. Para obtener más información, consulte Creación de expresiones para definir instancias de contenedor para las tareas de Amazon ECS.

`placementStrategy`

Tipo: Matriz de objetos

Obligatorio: no

Los objetos de estrategia colocación que utilizar para tareas en su servicio. Puede especificar un máximo de cuatro reglas de estrategia por servicio.

`type`

Tipo: cadena

Valores válidos: `random` | `spread` | `binpack`

Obligatorio: no

Es el tipo de estrategia de colocación. La estrategia de colocación `random` coloca las tareas aleatoriamente en los candidatos disponibles. La estrategia de ubicación `spread` distribuye la ubicación entre los candidatos disponibles de manera uniforme en función del parámetro `field`. La estrategia `binpack` ubica las tareas en los candidatos disponibles que tengan la menor cantidad disponible del recurso que se especifica con el parámetro `field`. Por ejemplo, si aplica dicha estrategia en la memoria, se coloca una tarea en la instancia con la menor cantidad de memoria restante, pero suficiente para ejecutar la tarea.

`field`

Tipo: cadena

Requerido: no

El campo en el que aplicar la estrategia de ubicación. Para la estrategia de ubicación `spread`, los valores válidos son `instanceId` (o `host`, que tiene el mismo efecto) o cualquier plataforma o atributo personalizado que se aplique a una instancia de contenedor, como por ejemplo `attribute:ecs.availability-zone`. Para la estrategia de colocación `binpack`,

los valores válidos son `cpu` y `memory`. Para la estrategia de colocación `random`, este campo no se utiliza.

Etiquetas

`tags`

Tipo: Matriz de objetos

Obligatorio: no

Los metadatos que se aplican al servicio para ayudarle a categorizarlas y organizarlas. Cada etiqueta está formada por una clave y un valor opcional, ambos definidos por el usuario. Cuando se elimina un servicio, también se eliminan las etiquetas. Se puede aplicar un máximo de 50 etiquetas al servicio. Para obtener más información, consulte Etiquetado de los recursos de Amazon ECS.

`key`

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Obligatorio: no

Una parte de un par clave-valor que compone una etiqueta. Un clave es una etiqueta general que actúa como una categoría para valores de etiqueta más específicos.

`value`

Tipo: cadena

Limitaciones de longitud: longitud mínima de 0. La longitud máxima es de 256 caracteres.

Obligatorio: no

La parte opcional de un par clave-valor que compone una etiqueta. Un valor actúa como un descriptor en una categoría de etiquetas (clave).

`enableECSManagedTags`

Tipo: Booleano

Valores válidos: `true` | `false`

Obligatorio: no

Especifica si se deben usar etiquetas administradas por Amazon ECS para las tareas del servicio. El valor predeterminado es `false` si no se especifica ningún valor. Para obtener más información, consulte Uso de etiquetas para facturación.

`propagateTags`

Tipo: cadena

Valores válidos: `TASK_DEFINITION` | `SERVICE`

Obligatorio: no

Especifica si se deben copiar las etiquetas de la definición de tareas o el servicio en las tareas del servicio. Si no se especifica ningún valor, las etiquetas no se copian. Solo se pueden copiar las etiquetas en las tareas del servicio durante la creación del servicio. Para agregar etiquetas a una tarea tras la creación del servicio, utilice la acción de la API `TagResource`.

Configuración de red

`networkConfiguration`

Tipo: objeto

Obligatorio: no

La configuración de red del servicio. Este parámetro es necesario para definiciones de tareas que usan el modo de red `awsvpc` para recibir su propia interfaz de red elástica y no se admite para otros modos de red. Si se usa el tipo de lanzamiento Fargate, es necesario el modo de red `awsvpc`. Para obtener más información acerca de las redes para el tipo de lanzamiento de Amazon EC2, consulte Opciones de redes de tareas de Amazon ECS para el tipo de lanzamiento de EC2. Para obtener más información acerca de las redes para el tipo de lanzamiento de Fargate, consulte Opciones de redes de tareas de Amazon ECS para el tipo de lanzamiento de Fargate.

`awsvpcConfiguration`

Tipo: objeto

Obligatorio: no

Un objeto que representa las subredes y los grupos de seguridad para una tarea o servicio.

`subnets`

Tipo: matriz de cadenas

Obligatorio: sí

Las subredes asociadas a la tarea o servicio. Existe un límite de 16 subredes que se pueden especificar según `awsvpcConfiguration`.

`securityGroups`

Tipo: matriz de cadenas

Obligatorio: no

Los grupos de seguridad asociados a la tarea o servicio. Si no se especifica un grupo de seguridad, se usará el grupo de seguridad predeterminado para la VPC. Existe un límite de cinco grupos de seguridad que se puede especificar en función de `awsvpcConfiguration`.

`assignPublicIP`

Tipo: cadena

Valores válidos: `ENABLED` | `DISABLED`

Obligatorio: no

Indica si la interfaz de red elástica de la tarea recibe una dirección IP pública. Si no se especifica ningún valor, se utiliza el valor predeterminado `DISABLED`.

`healthCheckGracePeriodSeconds`

Tipo: entero

Obligatorio: no

El periodo de tiempo, en segundos, que el programador de servicio de Amazon ECS ignora las comprobaciones de estado en mal estado de Elastic Load Balancing, VPC Lattice y del contenedor después de que se haya iniciado una tarea por primera vez. Si no se especifica ningún valor para el período de gracia de comprobación de estado, se utiliza el valor predeterminado: 0. Si no se utiliza ninguna de las comprobaciones de estado, entonces `healthCheckGracePeriodSeconds` no se utilizará.

Si las tareas del servicio tardan mucho en iniciarse y en responder, se puede especificar un periodo de gracia para las comprobaciones de estado de hasta 2 147 483 647 segundos (aproximadamente 69 años). Durante este periodo, el programador de servicio de Amazon ECS hará caso omiso del estado de las comprobaciones de estado. Este periodo de gracia puede evitar que el programador de servicio interprete que las tareas están en mal estado y las detenga antes de que tengan tiempo de iniciarse.

`loadBalancers`

Tipo: Matriz de objetos

Obligatorio: no

Un objeto de balanceador de carga que representa los balanceadores de carga que utilizar con su servicio. En el caso de los servicios que utilizan un equilibrador de carga de aplicación o un equilibrador de carga de red, existe un límite de cinco grupos de destino que puede asociar a un servicio.

Después de crear un servicio, la configuración del equilibrador de carga no se puede cambiar desde la AWS Management Console. Puede usar el Copiloto de AWS, AWS CloudFormation, AWS CLI o SDK para modificar la configuración del equilibrador de carga solo del controlador de implementación progresiva `ECS`, no AWS CodeDeploy azul/verde o exterior. Cuando agrega, actualiza o elimina una configuración de equilibrador de carga, Amazon ECS inicia una nueva implementación con la configuración actualizada de Elastic Load Balancing. Esto hace que las tareas se registren y eliminen el registro de los equilibradores de carga. Le recomendamos que lo verifique en un entorno de prueba antes de actualizar la configuración de Elastic Load Balancing. Para obtener más información acerca de cómo modificar la configuración, consulte UpdateService en la Referencia de la API de Amazon Elastic Container Service.

En el caso de los balanceadores de carga de aplicacione y los balanceadores de carga de red, este objeto debe contener el ARN del grupo de destino del balanceador de carga, el nombre del contenedor (tal como aparece en la definición de contenedor) y el puerto del contenedor para obtener acceso desde el balanceador de carga. Cuando una tarea de este servicio se ubica en la instancia de contenedor, la combinación de instancia y puerto se registran como un destino en el grupo de destino especificado.

`targetGroupArn`

Tipo: cadena

Requerido: no

El nombre de recurso de Amazon (ARN) completo del grupo o grupos de destino del equilibrador de carga elástico asociados a un servicio.

Un ARN del grupo de destino solo se especifica cuando se utiliza un Application Load Balancer o un Network Load Balancer.

`loadBalancerName`

Tipo: cadena

Requerido: no

El nombre del balanceador de carga que se va a asociar con el servicio.

Si utiliza un equilibrador de carga de aplicación o un equilibrador de carga de red, se debe omitir el parámetro del nombre del equilibrador de carga.

`containerName`

Tipo: cadena

Requerido: no

El nombre del contenedor (tal como aparece en una definición de contenedor) para asociar al balanceador de carga.

`containerPort`

Tipo: entero

Obligatorio: no

El puerto en el contenedor para asociar al balanceador de carga. Este puerto debe corresponderse con un `containerPort` en la definición de tarea que utilizan las tareas del servicio. En el caso de las tareas que utilizan el tipo de lanzamiento de EC2, la instancia de contenedor debe permitir el tráfico de entrada en el `hostPort` de asignación de puertos.

`role`

Tipo: cadena

Requerido: no

El nombre abreviado o el ARN completo del rol de IAM que permite a Amazon ECS realizar llamadas al balanceador de carga en su nombre. Este parámetro solo se permite si utiliza un balanceador de carga con un solo grupo de destino para el servicio y su definición de tareas no

utiliza el modo de red `awsvpc`. Si especifica el parámetro `role`, también debe especificar un objeto de balanceador de carga con el parámetro `loadBalancers`.

Si el rol especificado tiene una ruta distinta de `/`, entonces debe especificar el ARN de rol completo (se recomienda esto) o prefijar el nombre del rol con la ruta. Por ejemplo, si un rol con el nombre `bar` tiene una ruta `/foo/` debería especificar `/foo/bar` como nombre del rol. Para obtener más información, consulte Nombres fáciles de recordar y rutas en la Guía del usuario de IAM.

> **Important**
>
> Si su cuenta ya ha creado el rol vinculado al servicio Amazon ECS, se usa ese rol de forma predeterminada para su servicio, a menos que especifique un rol aquí. El rol vinculado al servicio es necesario si su definición de tarea usa el modo de red awsvpc, en cuyo caso no debe especificar un rol aquí. Para obtener más información, consulte Uso de roles vinculados al servicio para Amazon ECS.

`serviceConnectConfiguration`

Tipo: objeto

Obligatorio: no

La configuración para que este servicio detecte otros servicios y se conecte a ellos, y para que otros servicios dentro de un espacio de nombres lo detecten y se conecten a él.

Para obtener más información, consulte Uso de Service Connect para conectar los servicios de Amazon ECS con nombres abreviados.

`enabled`

Tipo: Booleano

Obligatorio: sí

Especifica si se debe utilizar Service Connect con este servicio.

`namespace`

Tipo: cadena

Requerido: no

El nombre corto o el nombre de recurso de Amazon (ARN) completo del espacio de nombres de AWS Cloud Map para su uso con Service Connect. El espacio de nombres debe estar en la misma Región de AWS que el servicio y el clúster de Amazon ECS. El tipo de espacio de nombres no afecta a Service Connect. Para obtener más información acerca de AWS Cloud Map, consulte Trabajo con los servicios en la Guía para desarrolladores de AWS Cloud Map.

`services`

Tipo: Matriz de objetos

Obligatorio: no

Una serie de objetos de servicio de Service Connect. Se trata de nombres y alias (también conocidos como puntos de conexión) que utilizan otros servicios de Amazon ECS para conectarse a este servicio.

Este campo no es obligatorio para que un servicio de “cliente” de Amazon ECS, miembro de un espacio de nombres, solo se conecte a otros servicios dentro del espacio de nombres. Un ejemplo es una aplicación frontend que acepta solicitudes entrantes de un equilibrador de carga que está asociado al servicio o por otros medios.

Un objeto selecciona un puerto de la definición de tarea, asigna un nombre para el servicio de AWS Cloud Map y una serie de alias (también conocidos como puntos de conexión) y puertos para que las aplicaciones de cliente hagan referencia a este servicio.

`portName`

Tipo: cadena

Obligatorio: sí

El `portName` debe coincidir con el `name` de una de las `portMappings` de todos los contenedores en la definición de tareas de este servicio de Amazon ECS.

`discoveryName`

Tipo: cadena

Requerido: no

`discoveryName` es el nombre del servicio nuevo de AWS Cloud Map que Amazon ECS crea para este servicio de Amazon ECS. Debe ser único dentro del espacio de nombres de AWS Cloud Map.

Si este campo no está especificado, se utiliza `portName`.

`clientAliases`

Tipo: Matriz de objetos

Obligatorio: no

La lista de alias de clientes para este servicio de Service Connect. Se utilizan para asignar nombres que pueden utilizar las aplicaciones de cliente. El número máximo de alias de cliente que puede tener en esta lista es 1.

Cada alias (“punto de conexión”) es un nombre de DNS y un número de puerto que otros servicios de Amazon ECS (“clientes”) pueden usar para conectarse a este servicio.

Cada combinación de nombre y de puerto debe ser única en el espacio de nombres.

Estos nombres se configuran dentro de cada tarea del servicio de cliente, no en AWS Cloud Map. Las solicitudes de DNS para resolver estos nombres no abandonan la tarea y no se cuentan para la cuota de solicitudes de DNS por segundo por interfaz de red elástica.

`port`

Tipo: entero

Obligatorio: sí

El número de puerto de escucha para el proxy de Service Connect. Este puerto está disponible dentro de todas las tareas del mismo espacio de nombres.

Para evitar cambiar las aplicaciones en los servicios de cliente de Amazon ECS, configúrelo con el mismo puerto que la aplicación de cliente utiliza de forma predeterminada.

`dnsName`

Tipo: cadena

Requerido: no

`dnsName` es el nombre que se utiliza en las aplicaciones de tareas del cliente para conectarse al servicio. El nombre debe ser una etiqueta de DNS válida.

Si no se especifica este campo, el valor se establece de forma predeterminada en `discoveryName.namespace`. Si el `discoveryName` no se especifica, se utiliza el `portName` de la definición de la tarea.

Para evitar cambiar las aplicaciones en los servicios de cliente de Amazon ECS, configúrelo con el mismo nombre que la aplicación de cliente utiliza de forma predeterminada. Por ejemplo, algunos nombres comunes son `database`, `db` o el nombre en minúsculas de una base de datos, como `mysql` o `redis`.

`ingressPortOverride`

Tipo: entero

Obligatorio: no

(Opcional) El número de puerto en el que se escuchará el proxy de Service Connect.

Utilice el valor de este campo para omitir el proxy para el tráfico en el número de puerto especificado en la `portMapping` denominada en la definición de tarea de esta aplicación y luego utilícelo en los grupos de seguridad de Amazon VPC para permitir el tráfico en el proxy para este servicio de Amazon ECS.

En el modo `awsvpc`, el valor predeterminado es el número de puerto del contenedor que se especifica en la `portMapping` denominada de la definición de la tarea de esta aplicación. En el modo `bridge`, el valor predeterminado es el puerto efímero dinámico del proxy de Service Connect.

`logConfiguration`

Tipo: objeto LogConfiguration

Obligatorio: no

Esto define dónde se publican los registros de proxy de Service Connect. Utilice los registros para depurar errores durante eventos inesperados. Esta configuración establece el parámetro `logConfiguration` en el contenedor del proxy de Service Connect en cada tarea de este servicio de Amazon ECS. El contenedor del proxy no se especifica en la definición de tarea.

Le recomendamos que utilice la misma configuración de registro que los contenedores de aplicaciones de la definición de tareas para este servicio de Amazon ECS. Para FireLens,

esta es la configuración de registro del contenedor de la aplicación. No es el contenedor del enrutador de registro de FireLens el que usa la imagen del contenedor `fluent-bit` o `fluentd` .

`serviceRegistries`

Tipo: Matriz de objetos

Obligatorio: no

Los detalles de la configuración de detección de servicios para el servicio. Para obtener más información, consulte Uso de la detección de servicios para conectar los servicios de Amazon ECS con nombres de DNS.

`registryArn`

Tipo: string

Requerido: no

El nombre de recurso de Amazon (ARN) del registro de servicios. El registro de servicios compatible actualmente es AWS Cloud Map. Para obtener más información, consulte Utilización de servicios en la Guía para desarrolladores de AWS Cloud Map.

`port`

Tipo: entero

Obligatorio: no

El valor del puerto utilizado si el servicio de detección de servicios se especifica en un registro de SRV. Este campo es obligatorio si se utilizan el modo de red `awsvpc` y los registros SRV.

`containerName`

Tipo: cadena

Requerido: no

El valor del nombre del contenedor que se va a utilizar para el servicio de detección de servicios. Este valor se especifica en la definición de tarea. Si la definición de tarea especificada por la tarea de servicio utiliza el modo de red `bridge` o `host`, se debe especificar una combinación de `containerName` y `containerPort` a partir de la definición

de tarea. Si la definición de tarea especificada por la tarea de servicio usa el modo de red `awsvpc` y se utiliza un registro DNS de tipo SRV, se debe especificar una combinación de `containerName` y `containerPort` o un valor `port`, pero no ambos.

`containerPort`

Tipo: entero

Obligatorio: no

El valor del puerto que se va a utilizar para el servicio de detección de servicios. Este valor se especifica en la definición de tarea. Si la definición de tarea especificada por la tarea de servicio utiliza el modo de red `bridge` o `host`, se debe especificar una combinación de `containerName` y `containerPort` a partir de la definición de tarea. Si la definición de especificada por la tarea de servicio usa el modo de red `awsvpc` y se utiliza un registro DNS de tipo SRV, se debe especificar una combinación de `containerName` y `containerPort` o un valor de `port`, pero no ambos.

Token de cliente

`clientToken`

Tipo: cadena

Requerido: no

Identificador único con distinción entre mayúsculas y minúsculas que se proporciona para garantizar la idempotencia de la solicitud. Puede tener hasta 32 caracteres ASCII.

Reequilibrio de la zona de disponibilidad

`availabilityZoneRebalancing`

Tipo: cadena

Requerido: no

Indica si el servicio utiliza el reequilibrio de la zona de disponibilidad. Los valores válidos son `ENABLED` y `DISABLED`. Para obtener más información acerca del reequilibrio de la zona de disponibilidad, consulte Equilibrio de un servicio de Amazon ECS entre zonas de disponibilidad.

Configuraciones de volúmenes

`volumeConfigurations`

Tipo: objeto

Obligatorio: no

Configuración que se utilizará para crear volúmenes para las tareas que administra el servicio. Con este objeto, solo se pueden configurar los volúmenes marcados como `configuredAtLaunch` en la definición de la tarea.

Este objeto es necesario para adjuntar volúmenes de Amazon EBS a las tareas que administra un servicio. Para obtener más información, consulte Uso de volúmenes de Amazon EBS con Amazon ECS.

`name`

Tipo: cadena

Obligatorio: sí

Nombre de un volumen que se configura al crear o actualizar un servicio. Se permiten hasta 255 letras (mayúsculas y minúsculas), números, símbolos de subrayado (_) y guiones (-). Este valor debe coincidir con el nombre del volumen que se especifica en la definición de la tarea.

`managedEBSVolume`

Tipo: objeto

Obligatorio: no

Configuración de volúmenes utilizada para la creación de volúmenes de Amazon EBS que se adjuntan a las tareas que mantiene un servicio cuando se crea o actualiza un servicio. Se adjunta un volumen por tarea.

`encrypted`

Tipo: Booleano

Obligatorio: no

Valores válidos: `true|false`

Especifica si se debe cifrar cada volumen de Amazon EBS que se crea. Si activó el cifrado de Amazon EBS de manera predeterminada para una Región de AWS particular para su Cuenta de AWS, pero configuró este parámetro como `false`, este parámetro se anulará y los volúmenes se cifrarán con la clave de KMS especificada para el cifrado de manera predeterminada. Para obtener más información acerca del cifrado de Amazon EBS de manera predeterminada, consulte Activación del cifrado de Amazon EBS de manera predeterminada en la Guía del usuario de Amazon EBS. Para obtener más información sobre el cifrado de los volúmenes de Amazon EBS adjuntos a tareas de Amazon ECS, consulte Datos cifrados almacenados en volúmenes de Amazon EBS adjuntos a tareas de Amazon ECS.

`kmsKeyId`

Tipo: cadena

Requerido: no

El identificador de AWS Key Management Service (AWS KMS) para utilizar el cifrado de Amazon EBS. Si se especifica `kmsKeyId`, el estado de cifrado debe ser `true`.

La clave especificada con este parámetro anula la clave de KMS predeterminada de Amazon EBS o cualquier clave de KMS a nivel de clúster para el cifrado de almacenamiento administrado por Amazon ECS que haya especificado. Para obtener más información, consulte Datos cifrados almacenados en volúmenes de Amazon EBS adjuntos a tareas de Amazon ECS.

Puede indicar la clave de KMS mediante alguno de los métodos siguientes:

- Id. de la clave: por ejemplo, `1234abcd-12ab-34cd-56ef-1234567890ab`.
- Alias de la clave: por ejemplo, `alias/ExampleAlias`.
- ARN de la clave: por ejemplo, `arn:aws:kms:us-east-1:012345678910:key/1234abcd-12ab-34cd-56ef-1234567890ab`.
- ARN del alias: por ejemplo, `arn:aws:kms:us-east-1:012345678910:alias/ExampleAlias`.

> **Important**
>
> AWS autentica la clave de KMS de manera asíncrona. Por lo tanto, si indica un id., alias o ARN que no es válido, puede parecer que la acción es correcta, pero

eventualmente produce un error. Para más información, consulte Troubleshooting Amazon EBS volume attachment issues.

`volumeType`

Tipo: cadena

Requerido: no

Valores válidos: `gp2|gp3|io1|io2|sc1|st1|standard`

El tipo de volumen de Amazon EBS. Para obtener más información acerca de los tipos de volúmenes, consulte Tipos de volúmenes de Amazon EBS en la Guía del usuario de Amazon EBS. El tipo de volumen predeterminado es `gp3`.

> **Note**
>
> El tipo de volumen `standard` no se admite para las tareas de Fargate.

`sizeInGiB`

Tipo: entero

Obligatorio: no

Rango válido: números enteros entre 1 y 16 384

El tamaño del volumen de EBS en gibibytes (GiB). Si no proporciona un id. de instantánea para configurar un volumen para adjuntarlo, debe proporcionar un valor de tamaño. Si configura un volumen para adjuntarlo mediante una instantánea, el valor predeterminado es el tamaño de la instantánea. Puede indicar un tamaño superior o igual al tamaño de la instantánea.

Para los tipos de volúmenes `gp2` y `gp3`, el rango válido es de 1 a 16 384.

Para los tipos de volúmenes `io1` y `io2`, el rango válido es de 4 a 16 384.

Para los tipos de volúmenes `st1` y `sc1`, el rango válido es de 125 a 16 384.

Para el tipo de volúmenes `standard`, el rango válido es de 1 a 1024.

`snapshotId`

Tipo: cadena

Requerido: no

El ID de la instantánea de un volumen de Amazon EBS existente que Amazon ECS utiliza para crear nuevos volúmenes que se adjuntarán. Debe especificar un valor `snapshotId` o `sizeInGiB`.

`volumeInitializationRate`

Tipo: entero

Obligatorio: no

La velocidad, en MiB/s, a la que se obtienen los datos de una instantánea de un volumen de Amazon EBS existente para crear nuevos volúmenes que se adjuntarán. Esta propiedad solo se puede especificar si especifica un valor de `snapshotId`. Para obtener más información sobre la tasa de inicialización de volúmenes, incluidos consulte el rango de tasas compatibles con la inicialización, consulte Inicialización de volúmenes de Amazon EBS en la Guía del usuario de Amazon EBS.

`iops`

Tipo: entero

Obligatorio: no

Número de operaciones de E/S por segundo (IOPS). Para los volúmenes `gp3`, `io1` y `io2`, esto representa el número de IOPS aprovisionadas para el volumen. Para los volúmenes `gp2`, este valor representa el rendimiento de referencia del volumen y la velocidad a la que el volumen acumula créditos de E/S para ráfaga. Este parámetro es obligatorio para los volúmenes `io1` y `io2`. Este parámetro no es compatible con los volúmenes de `gp2`, `st1`, `sc1` o `standard`.

Para los volúmenes de `gp3`, el rango de valores válido es de 3000 a 16 000.

Para los volúmenes de `io1`, el rango de valores válido es de 100 a 64 000.

Para los volúmenes de `io2`, el rango de valores válido es de 100 a 64 000.

`throughput`

Tipo: entero

Obligatorio: no

El rendimiento necesario para aprovisionar los volúmenes adjuntos a tareas que mantiene un servicio.

> **Important**
>
> Este parámetro solo es compatible solo con los volúmenes de `gp3`.

`roleArn`

Tipo: cadena

Obligatorio: sí

El ARN de recursos de Amazon (ARN) del rol de AWS Identity and Access Management (IAM) de la infraestructura que proporciona los permisos de Amazon ECS para administrar los recursos de Amazon EBS para las tareas. Para obtener más información, consulte Rol de IAM de infraestructura de Amazon ECS.

`tagSpecifications`

Tipo: objeto

Obligatorio: no

Indicación para que las etiquetas se apliquen a cada volumen de Amazon EBS.

`resourceType`

Tipo: cadena

Obligatorio: sí

Valores válidos: `volume`

El tipo de recurso que se debe etiquetar en la creación.

`tags`

Tipo: Matriz de objetos

Obligatorio: no

Los metadatos que se aplican a los volúmenes para categorizarlos y organizarlos. Cada etiqueta está formada por una clave y un valor opcional, ambos definidos por el usuario. `AmazonECSCreated` y `AmazonECSManaged` son etiquetas reservadas que Amazon ECS agrega en su nombre, por lo que puede indicar un máximo de 48 etiquetas propias. Cuando se elimina un volumen, también se eliminan las etiquetas. Para obtener más información, consulte Etiquetado de los recursos de Amazon ECS.

`key`

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Obligatorio: no

Una parte de un par clave-valor que compone una etiqueta. Un clave es una etiqueta general que actúa como una categoría para valores de etiqueta más específicos.

`value`

Tipo: cadena

Limitaciones de longitud: longitud mínima de 0. La longitud máxima es de 256 caracteres.

Obligatorio: no

La parte opcional de un par clave-valor que compone una etiqueta. Un valor actúa como un descriptor en una categoría de etiquetas (clave).

`propagateTags`

Tipo: cadena

Valores válidos: `TASK_DEFINITION` | `SERVICE` | `NONE`

Obligatorio: no

Indica si se deben copiar las etiquetas de la definición de tareas o el servicio en un volumen. Si se indica `NONE` o no se indica ningún valor, las etiquetas no se copian.

`fileSystemType`

Tipo: cadena

Requerido: no

Valores válidos: `xfs|ext3|ext4|NTFS`

Tipo de sistema de archivos de un volumen. El tipo de sistema de archivos del volumen determina cómo se almacenan y recuperan los datos en el volumen. En el caso de los volúmenes creados a partir de una instantánea, debe especificar el mismo tipo de sistema de archivos que utilizaba el volumen cuando se creó la instantánea. Si hay un error de coincidencia con el tipo de sistema de archivos, la tarea no podrá iniciarse.

Los valores válidos para Linux son `xfs`, ext3`, and ext4`. El valor predeterminado para los volúmenes adjuntos a las tareas de Linux es `XFS`.

Los valores válidos para Windows son `NTFS`. El valor predeterminado para los volúmenes adjuntos a las tareas de Windows es `NTFS`.

Plantilla de definición de servicio

A continuación, se muestra la representación JSON de una definición de servicio de Amazon ECS.

Tipo de lanzamiento de Amazon EC2

```
{
    "cluster": "",
    "serviceName": "",
    "taskDefinition": "",
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "loadBalancerName": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
```

```
    "desiredCount": 0,
    "clientToken": "",
    "launchType": "EC2",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "role": "",
    "deploymentConfiguration": {
        "deploymentCircuitBreaker": {
            "enable": true,
            "rollback": true
        },
        "maximumPercent": 0,
        "minimumHealthyPercent": 0,
        "alarms": {
            "alarmNames": [
                ""
            ],
            "enable": true,
            "rollback": true
        }
    },
    "placementConstraints": [
        {
            "type": "distinctInstance",
            "expression": ""
        }
    ],
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
```

```
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "healthCheckGracePeriodSeconds": 0,
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION",
    "enableExecuteCommand": true,
    "availabilityZoneRebalancing": "ENABLED",
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "",
        "services": [
            {
                "portName": "",
                "discoveryName": "",
                "clientAliases": [
                    {
                        "port": 0,
                        "dnsName": ""
                    }
                ],
                "ingressPortOverride": 0
            }
        ],
        "logConfiguration": {
            "logDriver": "journald",
            "options": {
                "KeyName": ""
            },
            "secretOptions": [
                {
```

```
                    "name": "",
                    "valueFrom": ""
                }
            ]
        }
    },
    "volumeConfigurations": [
        {
            "name": "",
            "managedEBSVolume": {
                "encrypted": true,
                "kmsKeyId": "",
                "volumeType": "",
                "sizeInGiB": 0,
                "snapshotId": "",
                "volumeInitializationRate": 0,
                "iops": 0,
                "throughput": 0,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "",
                                "value": ""
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "",
                "filesystemType": ""
            }
        }
    ]
}
```

Tipo de lanzamiento de Fargate

```
{
    "cluster": "",
    "serviceName": "",
    "taskDefinition": "",
```

```
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "loadBalancerName": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "desiredCount": 0,
    "clientToken": "",
    "launchType": "FARGATE",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "platformFamily": "",
    "role": "",
    "deploymentConfiguration": {
        "deploymentCircuitBreaker": {
            "enable": true,
            "rollback": true
        },
        "maximumPercent": 0,
        "minimumHealthyPercent": 0,
        "alarms": {
            "alarmNames": [
                ""
            ],
            "enable": true,
            "rollback": true
        }
    },
```

```
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "healthCheckGracePeriodSeconds": 0,
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION",
    "enableExecuteCommand": true,
    "availabilityZoneRebalancing": "ENABLED",
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "",
        "services": [
            {
                "portName": "",
                "discoveryName": "",
                "clientAliases": [
                    {
                        "port": 0,
                        "dnsName": ""
                    }
```

```
                ],
                "ingressPortOverride": 0
            }
        ],
        "logConfiguration": {
            "logDriver": "journald",
            "options": {
                "KeyName": ""
            },
            "secretOptions": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ]
        }
    },
    "volumeConfigurations": [
        {
            "name": "",
            "managedEBSVolume": {
                "encrypted": true,
                "kmsKeyId": "",
                "volumeType": "",
                "sizeInGiB": 0,
                "snapshotId": "",
                "volumeInitializationRate": 0,
                "iops": 0,
                "throughput": 0,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "",
                                "value": ""
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "",
                "filesystemType": ""
            }
```

```
        }
    ]
}
```

Puede crear esta plantilla de definición de servicio mediante el siguiente comando de la AWS CLI.

```
aws ecs create-service --generate-cli-skeleton
```

Etiquetado de los recursos de Amazon ECS

Como ayuda para administrar los recursos de Amazon ECS, puede asignar sus propios metadatos a cada recurso mediante etiquetas. Cada etiqueta consta de una clave y un valor opcional.

Puede utilizar etiquetas para clasificar los recursos de Amazon ECS de diversas maneras, por ejemplo, según su finalidad, propietario o entorno. Esto es útil cuando se tienen muchos recursos del mismo tipo. Puede identificar rápidamente un recurso específico según las etiquetas que le haya asignado. Por ejemplo, podría definir un conjunto de etiquetas para las instancias de contenedor de Amazon ECS de su cuenta. Esto le permite realizar un seguimiento del propietario y el nivel de la pila de cada instancia.

Puede utilizar etiquetas para sus informes de costo y uso. Puede utilizar estos informes para analizar el costo y el uso de los recursos de Amazon ECS. Para obtener más información, consulte the section called “Informes de uso de”.

> **Warning**
>
> Hay muchas API que devuelven las claves de las etiquetas y sus valores. Denegar el acceso a `DescribeTags` no deniega automáticamente el acceso a etiquetas devueltas por otras API. Como práctica recomendada, no debe incluir datos confidenciales en las etiquetas.

Recomendamos que idee un conjunto de claves de etiqueta que cumpla sus necesidades para cada tipo de recurso. Puede utilizar un conjunto coherente de claves de etiquetas para administrar los recursos más fácilmente. Puede buscar y filtrar los recursos en función de las etiquetas que agregue.

Las etiquetas no tienen ningún significado semántico para Amazon ECS, por lo que se interpretan estrictamente como cadenas de caracteres. Puede editar las claves y los valores de las etiquetas y también puede eliminar etiquetas de un recurso en cualquier momento. Puede establecer el valor de una etiqueta como una cadena vacía, pero no puede asignarle un valor nulo. Si agrega una etiqueta con la misma clave que una etiqueta existente en ese recurso, el nuevo valor sobrescribirá al anterior. Cuando elimina un recurso, también se eliminarán las etiquetas asignadas a dicho recurso.

Si utiliza AWS Identity and Access Management (IAM), puede controlar qué usuarios de su cuenta de AWS tienen permiso para administrar etiquetas.

Cómo se etiquetan los recursos

Hay varias formas de etiquetar las tareas, los servicios, las definiciones de tareas y los clústeres de Amazon ECS:

- El usuario etiqueta manualmente un recurso con la AWS Management Console, la API de Amazon ECS, la AWS CLI o un AWS SDK.
- El usuario crea un servicio o ejecuta una tarea independiente y selecciona la opción de etiquetas administradas por Amazon ECS.

  Amazon ECS etiqueta automáticamente todas las tareas recién lanzadas. Para obtener más información, consulte the section called “Etiquetas administradas por Amazon ECS”.
- El usuario crea un recurso con la consola. La consola etiqueta automáticamente los recursos.

  Estas etiquetas se devuelven en la AWS CLI y las respuestas del SDK de AWS se muestran en la consola. No se pueden modificar ni eliminar.

  Para obtener información sobre las etiquetas agregadas, consulte la columna Etiquetas agregadas automáticamente por la consola en la tabla Compatibilidad de etiquetado para los recursos de Amazon ECS.

Si especifica etiquetas al crear un recurso y las etiquetas no se pueden aplicar, Amazon ECS revierte el proceso de creación. Esto garantiza que los recursos se creen con etiquetas o, de lo contrario, no se creen y que ningún recurso se quede jamás sin etiquetar. Al etiquetar los recursos en el momento de su creación, ya no es necesario ejecutar scripts de etiquetado personalizados después de la creación del recurso.

En la siguiente tabla se describen los recursos de Amazon ECS que admiten etiquetas.

Recurso	Admite etiquetas	Admite la propagación de etiquetas	Etiquetas agregadas automáticamente por la consola
Tareas de Amazon ECS	Sí	Sí, desde la definición de tarea.	Clave: `aws:ecs:clusterName` Valor: `cluster-name`

Recurso	Admite etiquetas	Admite la propagación de etiquetas	Etiquetas agregadas automáticamente por la consola
Servicios de Amazon ECS	Sí	Sí, ya sea desde la definición de tareas o el servicio a las tareas del servicio.	Clave: `ecs:service:stackId` Valor: `arn:aws:cloudformation:` *`arn`*
Conjuntos de tareas de Amazon ECS	Sí	No	N/A
Definiciones de tareas de Amazon ECS	Sí	No	Clave: `ecs:taskDefinition:createdFrom` Valor: `ecs-console-v2`

Recurso	Admite etiquetas	Admite la propagación de etiquetas	Etiquetas agregadas automáticamente por la consola
Clústeres de Amazon ECS	Sí	No	Clave: `aws:cloudformation:logical-id` Valor: `ECSCluster` Clave: `aws:cloudformation:stack-id` Valor: `arn:aws:cloudformation:` *`arn`* Clave: `aws:cloudformation:stack-name` Valor: `ECS-Console-V2-Cluster-` *`EXAMPLE`*
Instancias de contenedor de Amazon ECS	Sí	Sí, desde la instancia de Amazon EC2. Para obtener más información, consulte Adición de etiquetas a una instancia de contenedor de Amazon ECS.	N/A
Instancias externas de Amazon ECS	Sí	No	N/A

Recurso	Admite etiquetas	Admite la propagación de etiquetas	Etiquetas agregadas automáticamente por la consola
Proveedor de capacidad de Amazon ECS	Sí. No se pueden etiquetar los proveedores de capacidad predefinidos `FARGATE` ni `FARGATE_SPOT` .	No	N/A

Etiquetado de recursos durante la creación

Los recursos siguientes admiten el etiquetado en el momento de la creación mediante la API de Amazon ECS, la AWS CLI o el AWS SDK:

- Tareas de Amazon ECS
- Servicios de Amazon ECS
- Definición de tarea de Amazon ECS
- Conjuntos de tareas de Amazon ECS
- Clústeres de Amazon ECS
- Instancias de contenedor de Amazon ECS
- Proveedores de capacidad de Amazon ECS

Amazon ECS tiene la opción de utilizar la autorización de etiquetado para la creación de recursos. Cuando la Cuenta de AWS se configura para autorizar el etiquetado, los usuarios deben tener permisos para llevar a cabo las acciones que crean el recurso, como `ecsCreateCluster`. Si indica las etiquetas en la acción de creación de recursos, AWS hace otra autorización para verificar que los usuarios o roles tengan permisos para crear etiquetas. Por lo tanto, usted debe conceder permisos explícitos para utilizar la acción `ecs:TagResource`. Para obtener más información, consulte the section called “Recursos de etiquetas durante la creación”. Para obtener información acerca de cómo configurar la opción, consulte the section called “Autorización de etiquetado”.

Restricciones

Se aplican las siguientes restricciones a las etiquetas:

- Se puede asociar un máximo de 50 etiquetas a un recurso.
- Las claves de etiquetas no se pueden repetir para un recurso. Cada clave de etiqueta debe ser única y solo puede tener un valor.
- Las claves pueden tener hasta 128 caracteres en UTF-8.
- Los valores pueden tener hasta 256 caracteres en UTF-8.
- Si hay múltiples Servicios de AWS y los recursos usan su esquema de etiquetado, limite los tipos de caracteres que usa. Algunos servicios pueden tener restricciones en cuanto a los caracteres permitidos. En general, los caracteres permitidos son letras, números, espacios y los siguientes caracteres: **+ - = . _ : / @**.
- Las claves y los valores de las etiquetas distinguen entre mayúsculas y minúsculas.
- No puede utilizar `aws:`, `AWS:`, ni ninguna combinación de mayúsculas o minúsculas de dichas letras como prefijo para claves ni valores. Estos están reservados solo para la utilización de AWS. Las claves y valores de etiquetas que tienen este prefijo no se pueden editar. Las etiquetas con este prefijo no cuentan para el límite de etiquetas por recurso.

Etiquetas administradas por Amazon ECS

Cuando se utilizan las etiquetas administradas por Amazon ECS, dicho servicio etiqueta automáticamente todas las tareas que se acaban de lanzar y los volúmenes de Amazon EBS adjuntos a las tareas con la información del clúster y las etiquetas de definición de tareas agregadas por el usuario o las etiquetas de servicio. A continuación, se describen las etiquetas agregadas:

- Tareas independientes: una etiqueta con una clave como `aws:ecs:clusterName` y un valor establecidos en el nombre del clúster. Todas las etiquetas de definición de tareas que agregaron los usuarios. Un volumen de Amazon EBS adjunto a una tarea independiente recibirá la etiqueta con una clave como `aws:ecs:clusterName` y un valor establecidos en el nombre del clúster. Para obtener más información sobre el etiquetado de volúmenes de Amazon EBS, consulte Tagging Amazon EBS volumes.
- Tareas que forman parte de un servicio: una etiqueta con una clave como `aws:ecs:clusterName` y un valor establecidos en el nombre del clúster. Una etiqueta con una

clave como `aws:ecs:serviceName` y un valor establecidos en el nombre del servicio. Etiquetas de alguno de los recursos siguientes:

- Definiciones de tareas: todas las etiquetas de definiciones de tareas que agregaron los usuarios.
- Servicios: todas las etiquetas de servicios que agregaron los usuarios.

  Un volumen de Amazon EBS adjunto a una tarea que forma parte de un servicio recibirá una etiqueta con una clave como `aws:ecs:clusterName` y un valor establecidos en el nombre del clúster, y una etiqueta con una clave como `aws:ecs:serviceName` y un valor establecidos en el nombre del servicio. Para obtener más información sobre el etiquetado de volúmenes de Amazon EBS, consulte Tagging Amazon EBS volumes.

Se requieren las siguientes opciones para esta función:

- Debe optar incluir los nuevos formatos de nombre de recurso de Amazon (ARN) e identificador (ID) de recursos. Para obtener más información, consulte Nombres de recursos de Amazon (ARN) e ID.
- Cuando usa las API para crear un servicio o ejecutar una tarea, debe configurar `enableECSManagedTags` a `true` para `run-task` y `create-service`. Para obtener más información, consulte create-service y run-task en la Referencia de la API AWS Command Line Interface.
- Amazon ECS utiliza etiquetas administradas para determinar cuándo están habilitadas algunas características, por ejemplo, el escalado automático de clústeres. Recomendamos no modificar las etiquetas manualmente para que Amazon ECS pueda administrar las características de forma eficaz.

Uso de etiquetas para facturación

AWS proporciona una herramienta de elaboración de informes llamada Cost Explorer, que puede utilizar para analizar el coste y la utilización de los recursos de Amazon ECS.

Puede utilizar Cost Explorer para ver gráficos sobre el uso y los costes. Puede ver los datos de los últimos 13 meses y predecir la cantidad que probablemente va a gastar durante los tres meses siguientes. Cost Explorer se puede utilizar para observar patrones de lo que se gasta en recursos de AWS a lo largo del tiempo. Por ejemplo, se puede utilizar para identificar aspectos que deben estudiarse más a fondo y observar tendencias que pueden ayudar a comprender los costos. También puede especificar intervalos de tiempo para los datos y ver los datos temporales por día o por mes.

Puede utilizar etiquetas administradas por Amazon ECS o agregadas por el usuario para su informe de costo y uso. Para obtener más información, consulte Informes de uso de Amazon ECS.

Para ver el costo de los recursos combinados, puede organizar la información de facturación basada en los recursos que tienen los mismos valores de clave de etiqueta. Por ejemplo, puede etiquetar varios recursos con un nombre de aplicación específico y luego organizar su información de facturación para ver los costos totales de la aplicación en distintos servicios. Para obtener más información acerca de la configuración de un informe de asignación de costos con etiquetas, consulte Informe de asignación de costos mensual en la Guía del usuario de AWS Billing.

Además, puede activar los Datos de asignación de costos divididos para incluir datos de uso de CPU y memoria a nivel de tarea en sus informes de costos y uso. Para obtener más información, consulte Informes sobre costo y uso de nivel de tarea.

> **Note**
>
> Si ha habilitado los informes, puede que los datos correspondientes al mes actual demoren hasta 24 horas en estar disponibles para su visualización.

Adición de etiquetas a los recursos de Amazon ECS

Puede etiquetar tareas, servicios, definiciones de tareas y clústeres nuevos o existentes. Para obtener más información sobre el etiquetado de instancias de contenedor, consulte Adición de etiquetas a una instancia de contenedor de Amazon ECS.

> **Warning**
>
> No agregue información de identificación personal (PII) ni otra información confidencial en las etiquetas. Las etiquetas son accesibles para muchos servicios de AWS, incluida la facturación. Las etiquetas no se han diseñado para usarse con información privada o confidencial.

Puede utilizar los siguientes recursos para especificar etiquetas al crear el recurso.

Tarea	Consola	AWS CLI	Acción API
Ejecute una o varias tareas.	Ejecución de una aplicación como tarea de Amazon ECS	run-task	RunTask
Cree un servicio.	Creación de una implementación de actualización continua de Amazon ECS	create-service	CreateService
Cree un conjunto de tareas.	Implementación de los servicios de Amazon ECS mediante un controlador de terceros	create-task-set	CreateTaskSet
Registre una definición de tareas.	the section called “Creación de una definición de tareas con la consola”	register-task-definition	RegisterTaskDefinition
Cree un clúster.	Creación de un clúster de Amazon ECS para el tipo de lanzamiento de Fargate	create-cluster	CreateCluster
Ejecute una o más instancias de contenedor.	Lanzamiento de una instancia de contenedo	run-instances	RunInstances

Tarea	Consola	AWS CLI	Acción API
	r de Linux de Amazon ECS		

Adición de etiquetas a los recursos existentes (consola de Amazon ECS)

Puede agregar o eliminar las etiquetas asociadas a los clústeres, servicios, tareas y definiciones de tareas directamente desde la página del recurso.

Para modificar una etiqueta de un recurso individual

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la barra de navegación, elija la Región de AWS a utilizar.
3. En el panel de navegación, seleccione un tipo de recurso (por ejemplo, Clusters [Clústeres]).
4. Seleccione el recurso de la lista de recursos, elija la pestaña Tags (Etiquetas) y, luego, elija Manage tags (Administrar etiquetas).
5. Configure sus etiquetas.

   [Agregar una etiqueta] Seleccione Add tag (Agregar etiqueta), y, a continuación, haga lo siguiente:

   - En Clave, escriba el nombre de la clave.
   - En Valor, escriba el valor de la clave.
6. Seleccione Save.

Adición de etiquetas a los recursos existentes (AWS CLI)

Puede agregar o sobrescribir una o varias etiquetas mediante la AWS CLI o una API.

> **Note**
>
> Puede utilizar puntos de conexión de servicio de doble pila para interactuar con Amazon ECS desde la AWS CLI, los SDK y la API de Amazon ECS a través de IPv4 e IPv6. Para obtener más información, consulte Uso de puntos de conexión de doble pila en Amazon ECS.

- AWS CLI: tag-resource
- Acción de API: TagResource

Adición de etiquetas a una instancia de contenedor de Amazon ECS

Puede asociar etiquetas con las instancias de contenedor utilizando uno de los siguientes métodos:

- Método 1: al crear su instancia de contenedor desde la API de Amazon EC2, la CLI o la consola, especifique las etiquetas pasando los datos del usuario a la instancia mediante el parámetro de configuración del agente de contenedor `ECS_CONTAINER_INSTANCE_TAGS`. Esto crea etiquetas que están asociadas solo a la instancia de contenedor de Amazon ECS, y no pueden enumerar mediante la API de Amazon EC2. Para obtener más información, consulte Arranque de instancias de contenedor de Linux de Amazon ECS para la transferencia de datos.

  > **Important**
  >
  > Si inicia las instancias de contenedor mediante un grupo de Amazon EC2 Auto Scaling, debe utilizar el parámetro de configuración del agente ECS_CONTAINER_INSTANCE_TAGS para agregar etiquetas. Esto se debe a la forma en que se agregan las etiquetas a las instancias de Amazon EC2 que se lanzan mediante grupos de Auto Scaling.

  A continuación, se muestra un ejemplo de un script de datos de usuario que asocia etiquetas con su instancia de contenedor:

  ```
  #!/bin/bash
  cat <<'EOF' >> /etc/ecs/ecs.config
  ECS_CLUSTER=MyCluster
  ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
  EOF
  ```

- Método 2: al crear su instancia de contenedor desde la API de Amazon EC2, la CLI o la consola, especifique primero las etiquetas mediante el parámetro `TagSpecification.N`. A continuación, transfiera los datos de usuario a la instancia mediante el parámetro de configuración del agente

de contenedor `ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM`. Al hacerlo, se propaga desde Amazon EC2 a Amazon ECS.

A continuación, se muestra un ejemplo de un script de datos de usuario que propaga las etiquetas que están asociadas a una instancia de Amazon EC2, además de registrar la instancia en un clúster con el nombre `MyCluster`.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM=ec2_instance
EOF
```

Para proporcionar acceso y permitir que las etiquetas de instancia de contenedor se propaguen de Amazon EC2 a Amazon ECS, agregue manualmente los siguientes permisos como una política insertada al rol de IAM de instancia de contenedor de Amazon ECS. Para obtener más información, consulte Adición y eliminación de políticas de IAM.

- `ec2:DescribeTags`

A continuación, se presenta una política de ejemplo mediante la cual se agregan estos permisos.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
             "ec2:DescribeTags"
            ],
            "Resource": "*"
        }
    ]
}
```

Instancias de contenedor externas

Para asociar etiquetas a las instancias de contenedor, puede utilizar uno de los siguientes métodos.

- Método 1: antes de ejecutar el script de instalación para registrar la instancia externa en el clúster, cree o edite el archivo de configuración del agente contenedor de Amazon ECS en `/etc/ecs/ecs.config` y agregue el parámetro de configuración del agente de contenedor `ECS_CONTAINER_INSTANCE_TAGS`. De esto modo, se crean etiquetas que están asociadas a la instancia externa.

  A continuación, se muestra un ejemplo sintaxis .

  ```
  ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
  ```

- Método 2: una vez que la instancia externa se registra en el clúster, puede usar AWS Management Console para agregar etiquetas. Para obtener más información, consulte Adición de etiquetas a los recursos existentes (consola de Amazon ECS).

Informes de uso de Amazon ECS

AWS proporciona una herramienta de elaboración de informes llamada Cost Explorer, que puede utilizar para analizar el coste y la utilización de los recursos de Amazon ECS.

Puede utilizar Cost Explorer para ver gráficos sobre el uso y los costes. Puede ver los datos de los últimos 13 meses y predecir la cantidad que probablemente va a gastar durante los tres meses siguientes. Cost Explorer se puede utilizar para observar patrones de lo que se gasta en recursos de AWS a lo largo del tiempo. Por ejemplo, se puede utilizar para identificar aspectos que deben estudiarse más a fondo y observar tendencias que pueden ayudar a comprender los costos. También puede especificar intervalos de tiempo para los datos y ver los datos temporales por día o por mes.

Los datos de medición del informe de uso y costos muestran el uso en todas las tareas de Amazon ECS. Los datos de medición incluyen el uso de CPU como `vCPU-Hours` y el uso de memoria como `GB-Hours` de cada tarea que se ejecutó. La forma de presentación de los datos depende del tipo de lanzamiento de la tarea.

Para las tareas que utilizan el tipo de lanzamiento de Fargate, la columna `lineItem/Operation` muestra `FargateTask`, y verá el costo asociado a cada tarea.

Para las tareas que utilizan el tipo de lanzamiento de EC2, la columna `lineItem/Operation` muestra `ECSTask-EC2`, y las tareas no tendrán un coste directo asociado a ellas. Los datos de medición que se muestran en el informe, como la utilización de memoria, representan el total de recursos que reservó la tarea durante el período de facturación indicado. Puede utilizar estos datos

para determinar el coste del clúster subyacente de instancias de Amazon EC2. Los datos de costo y uso de las instancias de Amazon EC2 se mostrarán por separado en el servicio de Amazon EC2.

También puede usar las etiquetas administradas por Amazon ECS para identificar el servicio o el clúster al que pertenece cada tarea. Para obtener más información, consulte Uso de etiquetas para facturación.

> **Important**
>
> Solo se pueden ver datos de medición de tareas lanzadas a partir del 16 de noviembre de 2018. Las tareas lanzadas antes de esta fecha no muestran datos de medición.

A continuación, se muestra un ejemplo con algunos de los campos que puede utilizar para ordenar los datos de asignación de costes mediante el Explorador de costes.

- Cluster name (Nombre del clúster)
- Nombre del servicio
- Etiquetas de recursos
- Tipo de lanzamiento
- Región de AWS
- Tipo de uso

Para obtener más información acerca de cómo crear un Informe de uso y costos de AWS, consulte Informe de uso y costos de AWS en la Guía del usuario de AWS Billing.

Informes sobre costo y uso de nivel de tarea

AWS Cost Management puede proporcionar datos de uso de CPU y memoria en AWS Cost and Usage Report para cada tarea de Amazon ECS, incluidas las tareas de Fargate y las de EC2. Estos datos se denominan datos de asignación de costos divididos. Puede utilizar estos datos para analizar los costos y el uso de las aplicaciones. Además, puede dividir y asignar los costos a unidades de negocio y equipos individuales con etiquetas de asignación de costos y categorías de costos. Para obtener más información acerca de los datos de asignación de costos divididos, consulte Understanding split cost allocation data en la Guía del usuario de AWS Cost and Usage Report.

Puede activar los datos de asignación de costos divididos de nivel de tarea para la cuenta en la AWS Cost Management Console. Si tiene una cuenta de administración (pagadora), puede optar por aplicar esta configuración desde la cuenta de pagador a todas las cuentas vinculadas.

Tras configurar los datos de asignación de costos divididos, habrá columnas adicionales bajo el encabezado SplitLineItem del informe. Para obtener más información, consulte Split line item details en la Guía del usuario de AWS Cost and Usage Report

En el caso de las tareas de EC2, estos datos dividen el costo de la instancia de EC2 en función del uso o las reservas de los recursos y de los recursos restantes de la instancia.

A continuación, se muestran los requisitos previos:

- Establezca el parámetro de configuración de agente `ECS_DISABLE_METRICS` de Amazon ECS en `false`.

  Cuando esta configuración es `false`, el agente de Amazon ECS envía métricas a Amazon CloudWatch. En Linux, esta configuración es `false` de manera predeterminada y las métricas se envían a CloudWatch. En Windows, esta configuración es `true` de manera predeterminada, por lo que debe cambiarla a `false` para enviar las métricas a CloudWatch para que AWS Cost Management las utilice. Para obtener más información sobre la configuración del agente de ECS, consulte Configuración del agente de contenedor de Amazon ECS.
- La versión mínima de Docker para obtener métricas fiables es la versión de Docker v20.10.13 y versiones posteriores, que se incluyen en la AMI 20220607 optimizada para Amazon ECS y versiones posteriores.

Para utilizar los datos de asignación de costos divididos, debe crear un informe y seleccionar los datos de asignación de costos divididos. Para obtener más información, consulte Creating Cost and Usage Reports en la Guía del usuario de AWS Cost and Usage Report.

AWS Cost Management calcula los datos de asignación de costos divididos con el uso de memoria y CPU de la tarea. AWS Cost Management puede utilizar la reserva de CPU y memoria de la tarea en lugar del uso, si este no está disponible. Si ve que el CUR utiliza las reservas, compruebe que las instancias de contenedor cumplen los requisitos previos y que las métricas de uso de los recursos de la tarea aparecen en CloudWatch.

Monitoreo de Amazon ECS

El monitoreo es una parte importante a la hora de mantener la fiabilidad, la disponibilidad y el rendimiento de Amazon ECS y de las soluciones de AWS. Debe recopilar datos de monitorización de todas las partes de su solución de AWS para que le resulte más sencillo depurar un error que se produce en distintas partes del código, en caso de que ocurra. Antes de comenzar a supervisar Amazon ECS, debe crear un plan de supervisión que incluya respuestas a las siguientes preguntas:

- ¿Cuáles son los objetivos de la supervisión?
- ¿Qué recursos va a supervisar?
- ¿Con qué frecuencia va a supervisar estos recursos?
- ¿Qué herramientas de supervisión va a utilizar?
- ¿Quién se encargará de realizar las tareas de supervisión?
- ¿Quién debería recibir una notificación cuando surjan problemas?

Las métricas disponibles dependen del tipo de lanzamiento de las tareas y de los servicios de los clústeres. Si utiliza el tipo de lanzamiento de Fargate para los servicios, se proporcionan métricas de utilización de CPU y de memoria que lo ayudarán a monitorear los servicios. Para el tipo de lanzamiento de Amazon EC2, usted es el propietario de las instancias EC2 que componen la infraestructura subyacente y deberá monitorearlas. Se ofrecen métricas adicionales de uso y reserva de CPU y memoria en el clúster, el servicio y la tarea.

El siguiente paso consiste en establecer un punto de referencia del rendimiento normal de Amazon ECS en su entorno. Para ello, se debe medir el rendimiento en distintos momentos y bajo distintas condiciones de carga. A medida que realice el monitoreo de Amazon ECS, vaya almacenando los datos de monitoreo históricos a fin de poder compararlos con los datos de rendimiento actual, identificar patrones de rendimiento normal y anomalías, y desarrollar métodos para la resolución de problemas.

Para establecer un punto de referencia debe, como mínimo, monitorizar los elementos siguientes:

- Métricas de utilización y reserva de CPU y de memoria de sus clústeres de Amazon ECS
- Métricas de utilización de CPU y de memoria de los servicios de Amazon ECS

  Para obtener más información, consulte Visualización de métricas de Amazon ECS.

Prácticas recomendadas de supervisión de Amazon ECS

Use las siguientes prácticas recomendadas para la supervisión de Amazon ECS.

- Haga de la supervisión una prioridad para solventar pequeñas cuestiones antes de que se conviertan en grandes problemas
- Creación de un plan de supervisión que incluya respuestas a las siguientes preguntas
  - ¿Cuáles son los objetivos de la supervisión?
  - ¿Qué recursos va a supervisar?
  - ¿Con qué frecuencia va a supervisar estos recursos?
  - ¿Qué herramientas de supervisión va a utilizar?
  - ¿Quién se encargará de realizar las tareas de supervisión?
  - ¿Quién debería recibir una notificación cuando surjan problemas?
- Automatice las tareas de supervisión en la medida de lo posible.
- Compruebe los archivos de registro de Amazon ECS. Para obtener más información, consulte Visualización de los registros del agente de contenedor de Amazon ECS.
- Utilice la Supervisión en tiempo de ejecución para ayudar a proteger las cuentas, los contenedores, las cargas de trabajo y los datos de su entorno de AWS. Para obtener más información, consulte Identificación de comportamientos no autorizados mediante la supervisión en tiempo de ejecución.

Herramientas de supervisión para Amazon ECS

AWS proporciona varias herramientas que se pueden utilizar para monitorear Amazon ECS. Puede configurar algunas de estas herramientas para que monitoricen por usted, pero otras herramientas requieren intervención manual. Le recomendamos que automatice las tareas de monitorización en la medida de lo posible.

Herramientas de monitoreo automatizadas

Puede utilizar las siguientes herramientas de monitoreo automatizadas para vigilar Amazon ECS e informar cuando haya algún problema:

- Alarmas de Amazon CloudWatch: vigile una sola métrica durante el período de tiempo que especifique y realice una o varias acciones según el valor que muestre la métrica en relación con

un determinado umbral durante una serie de períodos de tiempo. La acción es una notificación enviada a un tema de Amazon Simple Notification Service (Amazon SNS) o a una política de Amazon EC2 Auto Scaling. Las alarmas de CloudWatch no invocan acciones tan solo por tener un estado determinado; es necesario que el estado haya cambiado y se mantenga durante un número específico de periodos. Para obtener más información, consulte Supervisión de Amazon ECS con CloudWatch.

En el caso de servicios con tareas que utilizan el tipo de lanzamiento de Fargate, puede usar las alarmas de CloudWatch para la reducción y el escalado horizontal de las tareas de su servicio en función de métricas de CloudWatch como, por ejemplo, la utilización de CPU y memoria. Para obtener más información, consulte Escalado automático de su servicio de Amazon ECS.

En el caso de clústeres con tareas o servicios que utilizan el tipo de lanzamiento de EC2, puede usar las alarmas de CloudWatch para la reducción y el escalado horizontal de las instancias de contenedor en función de métricas de CloudWatch como, por ejemplo, la reserva de memoria del clúster.

Para las instancias de contenedor que se lanzaron mediante la AMI de Amazon Linux optimizada para Amazon ECS, puede utilizar CloudWatch Logs para ver diferentes registros desde sus instancias de contenedor en una ubicación cómoda. Debe instalar el agente de CloudWatch en las instancias de contenedor. Para obtener más información, consulte Descarga y configuración del agente de CloudWatch mediante la línea de comandos en la Guía del usuario de Amazon CloudWatch. También debe agregar la política de `ECS-CloudWatchLogs` al rol `ecsInstanceRole`. Para obtener más información, consulte Supervisión de los permisos de instancias de contenedores.

- Amazon CloudWatch Logs: para monitorear, almacenar y obtener acceso a los archivos de registro desde los contenedores de las tareas de Amazon ECS, especifique el controlador de registros `awslogs` en las definiciones de tareas. Para obtener más información, consulte Envío de registros de Amazon ECS a CloudWatch .

  También puede monitorear, almacenar y obtener acceso a los archivos de registro del sistema operativo y del agente de contenedor de Amazon ECS desde las instancias de contenedor de Amazon ECS. Este método de acceso a los registros se puede utilizar para los contenedores que usan el tipo de lanzamiento EC2.

- Eventos de Amazon CloudWatch: seleccione los eventos y diríjalos hacia uno o varios flujos o funciones de destino para realizar cambios, capturar información de estado y aplicar medidas correctivas. Para obtener más información, consulte Automaticzación de las respuestas a los

errores de Amazon ECS mediante EventBridge en esta guía y EventBridge es la evolución de los Eventos de Amazon CloudWatch en la Guía del usuario de Amazon EventBridge.

- Información de contenedores: recopile, agregue y resuma métricas y registros de sus aplicaciones en contenedores y microservicios. Información de contenedores recopila datos como Eventos de registro de rendimiento con embedded metric format (formato de métricas integradas). Estos eventos de registro de rendimiento son entradas que utilizan un esquema JSON estructurado que permite capturar y almacenar datos de cardinalidad alta a escala. A partir de estos datos, CloudWatch crea métricas agregadas por clúster, tarea y servicio como métricas de CloudWatch. Las métricas que recopila Información de contenedores están disponibles en los paneles automáticos de CloudWatch y también se pueden ver en la sección Métricas de la consola de CloudWatch.
- Monitoreo de registros de AWS CloudTrail: para compartir archivos de registro entre cuentas y monitorear los archivos de registro de CloudTrail en tiempo real, envíelos a CloudWatch Logs, escriba aplicaciones de procesamiento de registros en Java y compruebe que los archivos de registro no hayan cambiado después de que CloudTrail los entregara. Para obtener más información, consulte Registro de llamadas a la API de Amazon ECS mediante AWS CloudTrail en esta guía y Trabajar con archivos de registro de CloudTrail en la Guía del usuario de AWS CloudTrail.
- Supervisión en tiempo de ejecución: detecte amenazas para los clústeres y contenedores de su entorno de AWS. La supervisión en tiempo de ejecución utiliza un agente de seguridad de GuardDuty que agrega visibilidad en tiempo de ejecución a las cargas de trabajo individuales de Amazon ECS (por ejemplo, el acceso a los archivos, la ejecución de procesos y las conexiones de red).

Herramientas de supervisión manuales

Otra parte importante del monitoreo de Amazon ECS implica el monitoreo manual de los elementos que no cubren las alarmas de CloudWatch. Los paneles de las consolas de CloudWatch, Trusted Advisor y otras consolas de AWS proporcionan una vista rápida del entorno de AWS. Le recomendamos que también compruebe los archivos de registro en sus instancias de contenedor y los contenedores en sus tareas.

- Consola de Amazon ECS:
  - Métricas del clúster para el tipo de lanzamiento de EC2
  - Métricas de servicios

  - Estado de los servicios
  - Eventos de implementación de servicios
- Página de inicio de CloudWatch:
  - Alarmas y estado actual
  - Gráficos de alarmas y recursos
  - Estado de los servicios

  Además, puede utilizar CloudWatch para hacer lo siguiente:

  - Crear paneles personalizados para monitorear los servicios que le interesan.
  - Realizar un gráfico con los datos de las métricas para resolver problemas y descubrir tendencias.
  - Buscar y examinar todas sus métricas de recursos de AWS.
  - Crear y editar las alarmas de notificación de problemas.
- Comprobación de estado del contenedor: son comandos que se ejecutan localmente en un contenedor y validan el estado y la disponibilidad de la aplicación. Usted los configura por contenedor en la definición de tarea.
- AWS Trusted Advisor puede ayudarlo a monitorear los recursos de AWS para mejorar el rendimiento, la fiabilidad, la seguridad y la rentabilidad. Hay cuatro comprobaciones de Trusted Advisor disponibles para todos los usuarios y hay más de 50 comprobaciones disponibles para usuarios con un plan de soporte Business o Enterprise. Para obtener más información, consulte AWS Trusted Advisor.

  Trusted Advisor cuenta con este tipo de comprobaciones relacionadas con Amazon ECS:

  - Una tolerancia a errores que indica que tiene un servicio en ejecución en una única zona de disponibilidad.
  - Una tolerancia a errores que indica que no ha utilizado la estrategia de colocación distribuida para varias zonas de disponibilidad.
- AWS Compute Optimizer es un servicio que analiza las métricas de configuración y utilización de sus recursos de AWS. Informa si sus recursos son óptimos y genera recomendaciones de optimización para reducir el costo y mejorar el rendimiento de sus cargas de trabajo.

  Para obtener más información, consulte Recomendaciones de AWS Compute Optimizer para Amazon ECS.

Supervisión de Amazon ECS con CloudWatch

Puede monitorear los recursos de Amazon ECS mediante Amazon CloudWatch, que recopila y procesa los datos sin procesar de Amazon ECS y los convierte en métricas legibles prácticamente en tiempo real. Estas estadísticas se registran durante un período de dos semanas, de forma que pueda tener acceso a información histórica y obtener una mejor perspectiva sobre el rendimiento de loa clústeres o servicios. Los datos de las métricas de Amazon ECS se envían automáticamente a CloudWatch en períodos de 1 minuto. Para obtener más información acerca de CloudWatch, consulte la Guía del usuario de Amazon CloudWatch.

Amazon ECS ofrece métricas gratuitas para clústeres y servicios. Por un costo adicional, puede activar Información de contenedores de CloudWatch de Amazon ECS en su clúster para recopilar métricas por tarea, lo que incluye el uso de CPU, memoria y sistema de archivos de EBS. Para obtener más información sobre Información de contenedores, consulte Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada.

Consideraciones

Se debe tener en cuenta lo siguiente al utilizar métricas de CloudWatch para Amazon ECS.

- Cualquier servicio de Amazon ECS alojado en Fargate dispone de métricas de uso de CPU y de memoria de CloudWatch de forma automática, por lo que no es necesario hacer ningún paso manual.

- Para cualquier tarea o servicio de Amazon ECS alojado en instancias de Amazon EC2, la instancia de Amazon EC2 requiere la versión `1.4.0` o posterior (Linux) o `1.0.0` o posterior (Windows) del agente de contenedor para generar las métricas de CloudWatch. No obstante, recomendamos utilizar la versión del agente de contenedor más reciente. Para obtener información sobre la comprobación de la versión del agente y la actualización a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS.
- La versión mínima de Docker para obtener métricas fiables de CloudWatch es la versión de Docker `20.10.13` y posteriores.
- Las instancias de Amazon EC2 también requieren el permiso `ecs:StartTelemetrySession` en el rol de IAM con el que se lanzan las instancias de Amazon EC2. Si ha creado el rol de IAM para la instancia de contenedor de Amazon ECS antes de que las métricas de CloudWatch estuvieran disponibles para Amazon ECS, es posible que tenga que agregar este permiso. Para obtener información acerca del rol de IAM de la instancia de contenedor y cómo adjuntar la política de IAM

administrada para instancias de contenedor, consulte Rol de IAM de instancia de contenedor de Amazon ECS.

- Para desactivar la recopilación de métricas de CloudWatch en las instancias de Amazon EC2, establezca `ECS_DISABLE_METRICS=true` en la configuración del agente de contenedor de Amazon ECS. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS.

Métricas recomendadas

Amazon ECS ofrece métricas de CloudWatch gratuitas que puede usar para supervisar los recursos. Con estas métricas, se pueden medir la reserva de CPU y de memoria y el uso de la CPU, de la memoria y del sistema de archivos de EBS del clúster en su totalidad, y el uso de la CPU, de la memoria y del sistema de archivos de EBS en los servicios de sus clústeres. Para sus cargas de trabajo de GPU, puede medir la reserva de GPU de todo el clúster.

La infraestructura en la que se alojan las tareas de Amazon ECS en sus clústeres determina qué métricas están disponibles. Para las tareas alojadas en la infraestructura de Fargate, Amazon ECS proporciona métricas de uso de la CPU, de la memoria y del sistema de archivos de EBS para ayudar a supervisar los servicios. Para las tareas alojadas en instancias de EC2, Amazon ECS proporciona métricas de reserva de CPU, memoria y GPU, así como métricas de uso de la CPU y la memoria por clúster y servicio. Debe monitorear las instancias de Amazon EC2 que componen la infraestructura subyacente por separado. Para obtener más información acerca de la supervisión de instancias de EC2, consulte Monitorear Amazon EC2 en la Guía del usuario de Amazon EC2.

Para obtener información sobre las alarmas recomendadas para su uso con Amazon ECS, consulte una de las siguientes opciones en la Guía del usuario de Registros de Amazon CloudWatch:

- Amazon ECS
- Amazon ECS con Información de contenedores

Visualización de métricas de Amazon ECS

Una vez que tenga los recursos en ejecución en su clúster, puede consultar las métricas en las consolas de Amazon ECS y CloudWatch. La consola de Amazon ECS ofrece una vista máxima, mínima y promedio de 24 horas de las métricas del clúster y de servicio. La consola de CloudWatch ofrece una visualización precisa y personalizable de sus recursos, así como el número de tareas en ejecución de un servicio.

Consola de Amazon ECS

Las métricas de utilización de memoria y de CPU del servicio de Amazon ECS se encuentran disponibles en la consola de Amazon ECS. La vista proporcionada para las métricas del servicio muestra los valores medio, mínimo y máximo del periodo de 24 horas anterior, con puntos de datos disponibles en intervalos de 5 minutos. Para obtener más información, consulte Métricas de uso de los servicios de Amazon ECS.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. Seleccione el clúster del que desea ver las métricas.
3. Determine las métricas que desea ver.

Para ver las métricas de	Pasos
Clústeres	En la página de detalles del clúster, seleccione la pestaña Métricas. También se proporciona un enlace a la consola de CloudWatch para ver las métricas de Información de contenedores de CloudWatch, si las tiene activadas.
Servicios	En la página de detalles del clúster, en la pestaña Servicios, seleccione el servicio. A continuación, las métricas estarán disponibles en la pestaña Estado y métricas.

Consola de CloudWatch

Para el tipo de lanzamiento de Fargate, las métricas de servicio de Amazon ECS también se pueden consultar en la consola de CloudWatch. La consola proporciona la vista más detallada de métricas

de Amazon ECS y permite adaptar las vistas en función de las necesidades. Puede ver el uso del servicio y el recuento de tareas RUNNING (EN EJECUCIÓN) del servicio.

Para el tipo de lanzamiento de EC2, las métricas de clúster y servicio de Amazon ECS también se pueden consultar desde la consola de CloudWatch. La consola proporciona la vista más detallada de métricas de Amazon ECS y permite adaptar las vistas en función de las necesidades.

Para obtener información sobre la visualización de métricas, consulte Visualización de las métricas disponibles en la Guía del usuario de Amazon CloudWatch.

Métricas de Amazon ECS CloudWatch

Puede utilizar las métricas de uso de CloudWatch para proporcionar visibilidad sobre el uso de los recursos de su cuenta. Utilice estas métricas para visualizar el uso actual del servicio en paneles y gráficos de CloudWatch.

Amazon ECS envía métricas a CloudWatch a intervalos de un minuto. Estas métricas se recopilan para los recursos que tienen tareas en el estado `RUNNING`. Si un clúster, servicio u otro recurso no tiene tareas en ejecución, no se informarán métricas para ese recurso durante ese período. Por ejemplo, si tiene un clúster con un servicio, pero ese servicio no tiene tareas en un estado `RUNNING`, no se enviarán métricas a CloudWatch. De manera semejante, si tiene dos servicios y uno de ellos tiene tareas en ejecución y el otro no, solo se enviarán las métricas del servicio con tareas en ejecución.

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
CPUReservation	El porcentaje de unidades de CPU reservadas en el clúster o servicio. La reserva de CPU (filtrada por `ClusterName`) se mide como el total de unidades de	`ClusterName` .	Promedio, Mínimo, Máximo	Porcentaje

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
	CPU reservadas por tareas de Amazon ECS del clúster, dividido por el total de unidades de CPU de todas las instancias de Amazon EC2 registradas en el clúster. Solo las instancias de Amazon EC2 que tengan el estado `ACTIVE` o `DRAINING` afectarán a las métricas de reserva de CPU. Este método solo se admite para tareas alojadas en una instancia de Amazon EC2.			

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`CPUUtilization`	El porcentaje de unidades de CPU que se usa en el clúster o servicio. El uso de la CPU del clúster (filtrada por `ClusterName`) se mide como el total de unidades de CPU en uso por tareas de Amazon ECS del clúster, dividido por el total de unidades de CPU de todas las instancias de Amazon EC2 registradas en el clúster. Solo las instancias de Amazon EC2 que tengan el estado `ACTIVE` o `DRAINING` afectarán a las métricas de reserva de CPU. La métrica de clúster solo se admite para	`ClusterName` , `ServiceName`	Promedio, Mínimo, Máximo	Porcentaje

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
	tareas alojadas en una instancia de Amazon EC2. El uso de la CPU por servicio (filtrada por `ClusterName` y `ServiceName`) se mide como el total de unidades de CPU en uso por las tareas que pertenecen al servicio, dividido por el total de unidades de CPU reservadas por las tareas que pertenecen al servicio. La métrica de servicio se admite para tareas alojadas en un instancias de Amazon EC2 y Fargate.			

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`MemoryReservation`	El porcentaje de memoria reservada por las tareas en ejecución en el clúster. La reserva de memoria del clúster se mide como la memoria total reservada por las tareas de Amazon ECS del clúster, dividida por la cantidad total de memoria de todas las instancias de Amazon EC2 registradas en el clúster. Esta métrica solo se puede filtrar por `ClusterName`. Solo las instancias de Amazon EC2 que tengan el estado `ACTIVE` o `DRAINING` afectarán a las métricas de reserva	`ClusterName` .	Promedio, Mínimo, Máximo	Porcentaje

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
	de memoria. La métrica de reserva de memoria de clúster solo se admite para tareas alojadas en una instancia de Amazon EC2. **Note** Al calcular la utilización de la memoria, si se especifica `MemoryReservation`, se utiliza en el cálculo en lugar de la memoria total.			

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`MemoryUtilization`	El porcentaje de memoria que usa el clúster o servicio. El uso de memoria por clúster (filtrada por `ClusterName`) se mide como el total de memoria en uso por tareas de Amazon ECS del clúster, dividido por el total de memoria de todas las instancias de Amazon EC2 registradas en el clúster. Solo las instancias de Amazon EC2 que tengan el estado `ACTIVE` o `DRAINING` afectarán a las métricas de uso de memoria. La métrica de clúster solo se admite para tareas alojadas	`ClusterName` , `ServiceName`	Promedio, Mínimo, Máximo	Porcentaje

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
	en una instancia de Amazon EC2. El uso de memoria por servicio (filtrada por `ClusterName` y `ServiceName`) se mide como el total de memoria en uso por las tareas que pertenecen al servicio, dividido por el total de memoria reservada por las tareas que pertenecen al servicio. La métrica de servicio se admite para tareas alojadas en un instancias de Amazon EC2 y Fargate.			

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`EBSFilesystemUtilization`	El porcentaje del sistema de archivos de Amazon EBS que utilizan las tareas de un servicio. La métrica de uso del sistema de archivos de EBS a nivel de servicio (filtrada por `ClusterName` y `ServiceName`) se mide como la cantidad total del sistema de archivos de EBS que utilizan las tareas que pertenecen al servicio, dividida por la cantidad total de almacenamiento del sistema de archivos de EBS que se asigna a todas las tareas que pertenecen al servicio. La métrica de uso	`ClusterName`, `ServiceName`	Promedio, Mínimo, Máximo	Porcentaje

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
	del sistema de archivos de EBS por servicio solo está disponible para las tareas alojadas en las instancias de Amazon EC2 (con la versión de agente de contenedor `1.79.0`) y Fargate (con la versión de la plataforma `1.4.0`) que tienen un volumen de EBS adjunto. **Note** En el caso de las tareas alojadas en Fargate, hay espacio en el disco que solo			

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
	utiliza Fargate. El espacio que usa Fargate no tiene ningún costo, pero verá este almacenamiento adicional con herramientas como `df`.			

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`GPUReservation`	El porcentaje de unidades de GPU disponibles reservadas por las tareas en ejecución en el clúster. La reserva de GPU de clúster se mide como el número de GPU reservadas por tareas de Amazon ECS en el clúster, dividido por el número total de GPU disponibles en todas las instancias de Amazon EC2 con GPU registradas en el clúster. Solo las instancias de Amazon EC2 que tengan el estado `ACTIVE` o `DRAINING` afectarán a las métricas de reserva de GPU.	`ClusterName`	Promedio, Mínimo, Máximo	Porcentaje

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`ActiveConnectionCount`	El número total de conexiones simultáneas activas desde los clientes a los proxys de Amazon ECS Service Connect que se ejecutan en tareas que comparten el `DiscoveryName` seleccionado. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect. Dimensiones válidas: `DiscoveryName` y `DiscoveryName, ServiceName, ClusterName`.	`DiscoveryName` y `DiscoveryName, ServiceName, ClusterName`	Promedio, Mínimo, Máximo, Suma	Recuento

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`NewConnectionCount`	El número total de conexiones nuevas establecidas desde los clientes a los proxys de Amazon ECS Service Connect que se ejecutan en tareas que comparten el `Discovery Name` seleccionado. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect.	`Discovery Name` y `Discovery Name`, `ServiceName`, `ClusterName`	Promedio, Mínimo, Máximo, Suma	Recuento
`ProcessedBytes`	El número total de bytes de tráfico entrante procesados por los proxys de Service Connect. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect.	`Discovery Name` y `Discovery Name`, `ServiceName`, `ClusterName`	Promedio, Mínimo, Máximo, Suma	Bytes

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`RequestCount`	El número de solicitudes de tráfico entrante procesadas por los proxys de Service Connect. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect. También debe configurar `appProtocol` en la asignación de puertos en la definición de su tarea.	`DiscoveryName` y `DiscoveryName`, `ServiceName`, `ClusterName`	Promedio, Mínimo, Máximo, Suma	Recuento

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`GrpcRequestCount`	El número de solicitudes de tráfico entrante de gRPC procesadas por los proxys de Service Connect. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect y el `appProtocol` es `GRPC` en la asignación de puertos de la definición de la tarea.	`DiscoveryName` y `DiscoveryName`, `ServiceName`, `ClusterName`	Promedio, Mínimo, Máximo, Suma	Recuento

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`HTTPCode_Target_2XX_Count`	El número de códigos de respuesta HTTP con los números 200 a 299 generados por las aplicaciones en estas tareas. Estas tareas son los destinos. Esta métrica solo cuenta las respuestas enviadas a los proxys de Service Connect por las aplicaciones en estas tareas, no las respuestas enviadas directamente. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect y el `appProtocol` es `HTTP` o `HTTP2` en la asignación de puertos de la	`TargetDiscoveryName` y `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Promedio, Mínimo, Máximo, Suma	Recuento

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
	definición de la tarea. Dimensiones válidas:			

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`HTTPCode_Target_3XX_Count`	El número de códigos de respuesta HTTP con los números 300 a 399 generados por las aplicaciones en estas tareas. Estas tareas son los destinos. Esta métrica solo cuenta las respuestas enviadas a los proxys de Service Connect por las aplicaciones en estas tareas, no las respuestas enviadas directamente. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect y el `appProtocol` es `HTTP` o `HTTP2` en la asignación de puertos de la	`TargetDiscoveryName` y `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Promedio, Mínimo, Máximo, Suma	Recuento

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
	definición de la tarea.			

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`HTTPCode_Target_4XX_Count`	El número de códigos de respuesta HTTP con los números 400 a 499 generados por las aplicaciones en estas tareas. Estas tareas son los destinos. Esta métrica solo cuenta las respuestas enviadas a los proxys de Service Connect por las aplicaciones en estas tareas, no las respuestas enviadas directamente. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect y el `appProtocol` es `HTTP` o `HTTP2` en la asignación de puertos de la	`TargetDiscoveryName` y `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Promedio, Mínimo, Máximo, Suma	Recuento

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
	definición de la tarea.			

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`HTTPCode_Target_5XX_Count`	El número de códigos de respuesta HTTP con los números 500 a 599 generados por las aplicaciones en estas tareas. Estas tareas son los destinos. Esta métrica solo cuenta las respuestas enviadas a los proxys de Service Connect por las aplicaciones en estas tareas, no las respuestas enviadas directamente. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect y el `appProtocol` es `HTTP` o `HTTP2` en la asignación de puertos de la	`TargetDiscoveryName` y `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Promedio, Mínimo, Máximo, Suma	Recuento

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
	definición de la tarea.			
`RequestCountPerTarget`	El número promedio de solicitudes recibidas por cada destino que comparten los `Discovery Name` seleccionados. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect.	`TargetDiscoveryName` y `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Media	Recuento
`TargetProcessedBytes`	El número total de bytes procesados por los proxys de Service Connect. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect.	`TargetDiscoveryName` y `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Promedio, Mínimo, Máximo, Suma	Bytes

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`TargetResponseTime`	La latencia del procesamiento de solicitudes de aplicaciones. El tiempo transcurrido, en milisegundos, después de que la solicitud llegue al proxy de Service Connect de la tarea de destino hasta que el proxy recibe una respuesta de la aplicación de destino. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect.	`TargetDiscoveryName` y `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Promedio, Mínimo, Máximo	Milisegundos

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`ClientTLSNegotiationErrorCount`	El número total de veces que se produjo un error de conexión TLS. Esta métrica solo se usa cuando está activado TLS. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect.	`DiscoveryName` y `DiscoveryName`, `ServiceName`, `ClusterName`	Promedio, Mínimo, Máximo, Suma	Recuento

Métrica	Descripción	Dimensión válida	Estadísticas útiles	Unidad
`TargetTLSNegotiationErrorCount`	El número total de veces que se produjo un error en la conexión TLS debido a la falta de certificados de cliente, errores en las verificaciones de AWS Private CA o errores en las verificaciones del SAN. Esta métrica solo se usa cuando está activado TLS. Esta métrica solo está disponible si ha configurado Amazon ECS Service Connect.	`ServiceName`, `ClusterName`, `TargetDiscoveryName` y `TargetDiscoveryName`	Promedio, Mínimo, Máximo, Suma	Recuento

Dimensiones de las métricas de Amazon ECS

Las métricas de Amazon ECS utilizan el espacio de nombres `AWS/ECS` y proporcionan métricas para las siguientes dimensiones. Amazon ECS solo envía métricas para los recursos que tienen tareas en el estado `RUNNING`. Por ejemplo, si tiene un clúster con un servicio, pero ese servicio no tiene tareas en un estado `RUNNING`, no se enviarán métricas a CloudWatch. Si tiene dos servicios y uno de ellos tiene tareas en ejecución y el otro no, solo se enviarán las métricas del servicio con tareas en ejecución.

Dimensión	Definición
`ClusterName`	Esta dimensión filtra los datos solicitados de todos los recursos en el clúster especificado. Todas las métricas de Amazon ECS se filtran por `ClusterName` .
`ServiceName`	Esta dimensión filtra los datos solicitados de todos los recursos de un servicio especificado dentro de un clúster específico.
`DiscoveryName`	Esta dimensión filtra los datos que solicita para las métricas de tráfico a un nombre de detección de Service Connect específico en todos los clústeres de Amazon ECS. Tenga en cuenta que un puerto específico de un contenedor en ejecución puede tener varios nombres de detección.
`DiscoveryName, ServiceName, ClusterName`	Esta dimensión filtra los datos que solicita para las métricas de tráfico a un nombre de detección de Service Connect específico en todas las tareas que tienen este nombre de detección y que este servicio crea en este clúster. Utilice esta dimensión para ver las métricas del tráfico entrante de un servicio específico, si ha reutilizado el mismo nombre de detección en varios servicios de diferentes espacios de nombres. Tenga en cuenta que un puerto específico de un contenedor en ejecución puede tener varios nombres de detección.

Dimensión	Definición
`TargetDiscoveryName`	Esta dimensión filtra los datos que solicita para las métricas de tráfico a un nombre de detección de Service Connect específico en todos los clústeres de Amazon ECS. A diferencia de `DiscoveryName` , estas métricas de tráfico solo miden el tráfico entrante a este `DiscoveryName` que proviene de otras tareas de Amazon ECS que tienen una configuración de Service Connect en este espacio de nombres. Esto incluye las tareas que han realizado los servicios con una configuración de Service Connect de solo cliente o de cliente y servidor. Tenga en cuenta que un puerto específico de un contenedor en ejecución puede tener varios nombres de detección.

Dimensión	Definición
`TargetDiscoveryName, ServiceName, ClusterName`	Esta dimensión filtra los datos que solicita para las métricas de tráfico a un nombre de detección de Service Connect específico, pero solo cuenta el tráfico de las tareas que este servicio crea en este clúster. Utilice esta dimensión para ver las métricas de tráfico entrante que provienen de un cliente específico de otro servicio. A diferencia de `DiscoveryName, ServiceName, ClusterName` , estas métricas de tráfico solo miden el tráfico entrante a este `DiscoveryName` que proviene de otras tareas de Amazon ECS que tienen una configuración de Service Connect en este espacio de nombres. Esto incluye las tareas que han realizado los servicios con una configuración de Service Connect de solo cliente o de cliente y servidor. Tenga en cuenta que un puerto específico de un contenedor en ejecución puede tener varios nombres de detección.

Métricas de uso de AWS Fargate

Puede utilizar las métricas de uso de CloudWatch para proporcionar visibilidad sobre el uso de los recursos de su cuenta. Utilice estas métricas para visualizar el uso actual del servicio en paneles y gráficos de CloudWatch.

Las métricas de uso de AWS Fargate se corresponden con las cuotas de servicio de AWS. Puede configurar alarmas que le avisen cuando su uso se acerque a una Service Quota. Para obtener más información acerca de las Service Quotas de Fargate, consulte Puntos de conexión y cuotas de Amazon ECS en la Referencia general de Amazon Web Services.

AWS Fargate publica las siguientes métricas en el espacio de nombres `AWS/Usage`.

Métrica	Descripción
`ResourceCount`	El número total de los recursos especificados que se ejecutan en su cuenta. Los recursos se definen por las dimensiones asociadas a la métrica.

Las siguientes dimensiones se utilizan para ajustar las métricas de uso publicadas por AWS Fargate.

Dimensión	Descripción
`Service`	El nombre del servicio de AWS que contiene el recurso. Para las métricas de uso de AWS Fargate, el valor de esta dimensión es `Fargate`.
`Type`	El tipo de entidad que se registra. Actualmente, el único valor válido para las métricas de uso de AWS Fargate es `Resource`.
`Resource`	El tipo de recurso que se está ejecutando. El tipo de recurso que se está ejecutando. Actualmente, el único valor válido para las métricas de uso de AWS Fargate es `vCPU`, que devuelve información sobre las instancias en ejecución.
`Class`	La clase de recurso a la que se realiza el seguimiento. La clase de recurso a la que se realiza el seguimiento. En el caso de las métricas de uso de AWS Fargate con vCPU como valor de la dimensión Resource (Recurso), los valores válidos son `Standard/OnDemand` y `Standard/Spot` .

Puede usar la consola de Service Quotas para visualizar la utilización en gráficos y configurar alarmas que le avisen cuando su uso de AWS Fargate se aproxime a una cuota de servicio. Para obtener información sobre cómo crear una alarma de CloudWatch que le notifique cuando esté cerca del umbral de un valor de cuota, consulte Service Quotas and Amazon CloudWatch alarms en la Guía del usuario de Cuotas de servicio

.

Métricas de reserva del clúster de Amazon ECS

Las métricas de reserva de clúster se miden como el porcentaje de CPU, memoria y GPU reservado por todas las tareas de Amazon ECS de un clúster en comparación con la CPU, la memoria y las GPU totales que se registraron para cada instancia de contenedor activa en el clúster. Solo las instancias de contenedor que tengan el estado `ACTIVE` o `DRAINING` afectarán a las métricas de reserva de clúster. Esta métrica solo se emplea en clústeres con tareas o servicios alojados en instancias de EC2. No se admite en clústeres con tareas alojadas en AWS Fargate.

```
                                (Total CPU units reserved by tasks in cluster) x 100
Cluster CPU reservation =
 ---------------------------------------------------------------
                           (Total CPU units registered by container instances in
 cluster)
```

```
                                  (Total MiB of memory reserved by tasks in cluster x
 100)
Cluster memory reservation =
 ------------------------------------------------------------------
                              (Total MiB of memory registered by container instances in
 cluster)
```

```
                                 (Total GPUs reserved by tasks in cluster x 100)
Cluster GPU reservation =
 ------------------------------------------------------------------
                              (Total GPUs registered by container instances in cluster)
```

Cuando se ejecuta una tarea en un clúster, Amazon ECS analiza su definición de tarea y reserva las unidades de CPU, MiB de memoria y GPU totales especificadas en sus definiciones de contenedor. Cada minuto, Amazon ECS calcula el número de unidades de CPU, MiB de memoria y GPU reservadas actualmente para cada tarea que se está ejecutando en el clúster. Se calcula la cantidad total de CPU, memoria y GPU reservadas para todas las tareas que se ejecutan en el clúster, y dichas cifras se notifican a CloudWatch como un porcentaje de los recursos totales registrados para el clúster. Si especifica un límite flexible (`memoryReservation`) en la definición de tareas, se utiliza para calcular la cantidad de memoria reservada. De lo contrario, se utiliza el límite máximo (`memory`). El MiB total de memoria reservado por tareas de un clúster también incluye el tamaño del volumen del sistema de archivos temporal (`tmpfs`) y `sharedMemorySize` si se define en la definición de

la tarea. Para obtener más información sobre límites invariables y flexibles, el tamaño de memoria compartida y el tamaño de volumen tmpfs, consulte Parámetros de definición de tareas.

Por ejemplo, un clúster tiene dos instancias de contenedor registradas, una instancia `c4.4xlarge` y una instancia `c4.large`. La instancia `c4.4xlarge` se registra en el clúster con 16 384 unidades de CPU y 30 158 MiB de memoria. La instancia `c4.large` se registra con 2 048 unidades de CPU y 3 768 MiB de memoria. Los recursos totales de este clúster son 18 432 unidades de CPU y 33 926 MiB de memoria.

Si una definición de tarea reserva 1 024 unidades de CPU y 2 048 MiB de memoria y se inician diez tareas con esta definición de tarea en este clúster (y no se está ejecutando ninguna otra tarea), se reserva un total de 10 240 unidades de CPU y 20 480 MiB de memoria. Esto se registra en CloudWatch como una reserva de CPU del 55 % y una reserva de memoria del 60 % para el clúster.

La siguiente ilustración muestra las unidades de CPU totales registradas en un clúster y qué significa la reserva y la utilización para las tareas existentes y la nueva ubicación de tareas. Los bloques inferiores (reservados, utilizados) y centrales (reservados, no utilizados) representan el total de unidades de CPU reservadas para las tareas existentes que se están ejecutando en el clúster, o la métrica `CPUReservation` de CloudWatch. El bloque inferior representa las unidades de CPU reservadas que las tareas en ejecución están utilizando realmente en el clúster, o la métrica `CPUUtilization` de CloudWatch. El bloque superior representa unidades de CPU que no han reservado tareas existentes; estas unidades de CPU están disponibles para la nueva ubicación de tareas. Las tareas existentes pueden utilizar también estas unidades de CPU sin reservar, si aumenta su necesidad de recursos de CPU. Para obtener más información, consulte la documentación del parámetro `cpu` de definición de tarea.

Métricas de uso del clúster de Amazon ECS

Las métricas de uso del clúster están disponibles para la CPU, la memoria y, si hay un volumen de EBS asociado a sus tareas, para el uso del sistema de archivos de EBS. Estas métricas solo están disponibles para clústeres con tareas o servicios alojados en instancias de Amazon EC2. No se admiten en clústeres con tareas alojadas en AWS Fargate.

Métricas de uso de CPU y de memoria por clúster de Amazon ECS

El uso de CPU y memoria se mide como porcentaje de la CPU y la memoria usado por todas las tareas de un clúster en comparación con la CPU y la memoria total que se registró para cada instancia de Amazon EC2 activa registrada en el clúster. Solo las instancias de Amazon EC2 que tengan el estado `ACTIVE` o `DRAINING` afectarán a las métricas de uso del clúster.

```
                                  (Total CPU units used by tasks in cluster) x 100
Cluster CPU utilization =
 ---------------------------------------------------------------
                            (Total CPU units registered by container instances in
 cluster)
```

```
                                    (Total MiB of memory used by tasks in cluster x
 100)
Cluster memory utilization =
 ----------------------------------------------------------------
                              (Total MiB of memory registered by container instances in
 cluster)
```

Cada minuto, el agente de contenedor de Amazon ECS de cada instancia de Amazon EC2 calcula el número de unidades de CPU y MiB de memoria que se están utilizando actualmente para cada tarea que se ejecuta en dicha instancia de Amazon EC2, y esta información se reenvía a Amazon ECS. Se calcula la cantidad total de CPU y memoria utilizada para todas las tareas que se ejecutan en el clúster, y dichas cifras se notifican a CloudWatch como un porcentaje de los recursos totales registrados para el clúster.

Por ejemplo, un clúster tiene dos instancias de Amazon EC2 activas registradas, una instancia `c4.4xlarge` y una instancia `c4.large`. La instancia `c4.4xlarge` se registra en el clúster con `16,384` unidades de CPU y `30,158` MiB de memoria. La instancia `c4.large` se registra con `2,048` unidades de CPU y `3,768` MiB de memoria. Los recursos totales de este clúster son `18,432` unidades de CPU y `33,926` MiB de memoria.

Si se están ejecutando diez tareas en este clúster y cada tarea consume `1,024` unidades de CPU y `2,048` MiB de memoria, se utiliza un total de `10,240` unidades de CPU y `20,480` MiB de memoria en el clúster. Esto se registra en CloudWatch como una utilización de CPU del 55 de memoria del 60 % para el clúster.

Uso del sistema de archivos de Amazon EBS por clúster de Amazon ECS

La métrica de uso del sistema de archivos de EBS por clúster se mide como la cantidad total del sistema de archivos de EBS que usan las tareas que se ejecutan en el clúster, dividida por la cantidad total de almacenamiento del sistema de archivos de EBS que se asignó a todas las tareas del clúster.

```
                                      (Total GB of EBS filesystem used by tasks in
 cluster x 100)
Cluster EBS filesystem utilization =
 ----------------------------------------------------------------
                                      (Total GB of EBS filesystem allocated to tasks
 in cluster)
```

Métricas de uso de los servicios de Amazon ECS

Las métricas de uso del servicio están disponibles para la CPU, la memoria y, si hay un volumen de EBS asociado a sus tareas, para el uso del sistema de archivos de EBS. Las métricas de servicio se admiten para servicios con tareas alojadas en instancias de Amazon EC2 y Fargate.

Uso de CPU y memoria por servicio

El uso de CPU y memoria se mide como el porcentaje de CPU y de memoria utilizado por las tareas de Amazon ECS que pertenecen al servicio de un clúster, en comparación con la CPU y la memoria que se especifican en la definición de tarea del servicio.

Al ver estas métricas en CloudWatch, puede elegir diferentes estadísticas:

- Promedio: la utilización media de todas las tareas del servicio. Esto se calcula utilizando la fórmula siguiente.
- Mínimo: la utilización de la tarea con el menor uso de recursos del servicio. Representa el porcentaje de CPU o memoria utilizado por la tarea que consume menos recursos en comparación con lo que se especificó en la definición de la tarea.
- Máximo: la utilización de la tarea con el mayor uso de recursos del servicio. Representa el porcentaje de CPU o memoria utilizado por la tarea que consume más recursos en comparación con lo que se especificó en la definición de la tarea.

Las siguientes fórmulas muestran cómo se calcula la estadística Promedio:

```
                                  (Total CPU units used by tasks in service) x 100
Service CPU utilization =
 ---------------------------------------------------------------
                           (Total CPU units specified in the task definition) x (Number
 of tasks in the service)
```

```
                                     (Total MiB of memory used by tasks in service x
 100)
Service memory utilization =
 ------------------------------------------------------------------
                                (Total MiB of memory specified in the task definition)  x
 (Number of tasks in the service)
```

> **Note**
>
> Las fórmulas anteriores solo se aplican a la estadística Promedio. Las estadísticas Mínimo y Máximo representan la tarea individual con la utilización de recursos más baja y más alta, respectivamente, en lugar de un cálculo agregado de todas las tareas.

Amazon ECS recopila métricas cada 20 segundos. Cada minuto, el agente de contenedor de Amazon ECS calcula el número de unidades de CPU y MiB de memoria que se están utilizando actualmente para cada tarea del servicio en ejecución. Esta información se remite a Amazon ECS. Se calcula la cantidad total de CPU y memoria utilizada para todas las tareas del servicio que se ejecutan en el clúster y dichas cifras se notifican a CloudWatch como un porcentaje de los recursos totales que se han especificado para el servicio en la definición de tarea de este. Los valores mínimo y máximo son los más pequeños y los más grandes de las métricas de 20 segundos. El promedio es el agregado de los 3 valores.

Si especifica un límite flexible (`memoryReservation`), se utiliza para calcular la cantidad de memoria reservada. De lo contrario, se utiliza el límite máximo (`memory`). Para obtener más información acerca de los límites máximos y flexibles, consulte Tamaño de tarea.

Por ejemplo, la definición de tareas para un servicio especifica un total de 512 unidades de CPU y 1 024 MiB de memoria (con el parámetro `memory` de límite estricto) para todos sus contenedores. El servicio tiene un recuento deseado de una tarea en ejecución, el servicio se ejecuta en un clúster con una instancia de contenedor `c4.large` (con 2 048 unidades de CPU y 3 768 MiB de memoria total) y en el clúster no se está ejecutando ninguna otra tarea. Aunque la tarea especifica 512 unidades de CPU, porque es la única tarea en ejecución en una instancia de contenedor con 2 048 unidades de CPU, puede utilizar hasta cuatro veces la cantidad especificada (2 048 / 512). Sin embargo, la memoria especificada de 1 024 MiB es un límite fijo y no se puede superar, de modo que, en este caso, la utilización de memoria del servicio no puede superar el 100%.

Si el ejemplo anterior utilizara el parámetro de límite flexible `memoryReservation` en lugar del límite invariable `memory`, las tareas del servicio podrían utilizar más de 1024 MiB de memoria especificada si fuera necesario. En este caso, el uso de memoria del servicio podría superar el 100%.

Si la aplicación presenta un aumento repentino en la utilización de la memoria durante un breve período de tiempo, no verá que la utilización de la memoria del servicio aumente porque Amazon

ECS recopila varios puntos de datos cada minuto y, a continuación, los agrega a un punto de datos que se envía a CloudWatch.

Si esta tarea realiza un trabajo con uso intensivo de la CPU durante un período y utiliza todas las 2 048 unidades de CPU disponibles y 512 MiB de memoria, entonces el servicio registra una utilización de la CPU del 400% y una utilización de memoria del 50%. Si la tarea está inactiva y utiliza 128 unidades de CPU y 128 MiB de memoria, el servicio registra una utilización de la CPU del 25% y un uso de memoria del 12,5%.

> **Note**
>
> En este ejemplo, la utilización de la CPU solo superará el 100 % cuando las unidades de CPU se definan en el nivel de contenedor. Si define unidades de CPU en el nivel de tarea, la utilización no superará el límite definido de nivel de tarea.

Uso del sistema de archivos de EBS por servicio

El uso del sistema de archivos de EBS por servicio se mide como la cantidad total del sistema de archivos de EBS que utilizan las tareas que pertenecen al servicio, dividida por la cantidad total de almacenamiento del sistema de archivos de EBS que se asigna a todas las tareas que pertenecen al servicio.

Recuento de tareas **RUNNING** del servicio

Puede utilizar las métricas de CloudWatch para consultar el número de tareas de los servicios que tienen el estado `RUNNING`. Por ejemplo, puede definir una alarma de CloudWatch para esta métrica a fin de que le alerte si el número de tareas en ejecución en el servicio cae por debajo de un valor especificado.

Recuento de tareas **RUNNING** del servicio en Amazon ECS CloudWatch Container Insights

Un “número de tareas en ejecución” (`RunningTaskCount`) está disponible por clúster y servicio cuando utiliza Amazon ECS CloudWatch Container Insights. Puede usar Container Insights para todos los nuevos clústeres creados mediante la configuración de la cuenta de `containerInsights`, en clústeres individuales al habilitar la configuración del clúster durante la creación de este o en clústeres existentes mediante la API UpdateClusterSettings. Las métricas recopiladas por CloudWatch Container Insights se cobran como métricas personalizadas. Para obtener más información acerca de los precios de CloudWatch, consulte Precios de CloudWatch.

Para ver esta métrica, consulte Métricas de Amazon ECS Container Insights en la Guía del usuario de Amazon CloudWatch.

Casos de uso de métricas de utilización de los servicios de Amazon ECS

En los siguientes recursos se proporciona información adicional sobre cuándo utilizar las métricas de Amazon ECS.

- Supervisión de la utilización de los recursos: utilice estadísticas medias para supervisar los patrones de consumo de CPU y memoria, establecer valores de referencia de rendimiento y detectar tendencias de degradación gradual del rendimiento.
- Optimización de costos: utilice estadísticas medias para supervisar el uso de los recursos, dimensionar correctamente los contenedores, ajustar las reservas en función de los patrones de uso e implementar un escalado programado.
- Evaluación comparativa del rendimiento: utilice estadísticas medias para comparar las métricas entre las revisiones de los servicios, establecer los KPI de rendimiento y validar las mejoras de optimización.
- Detección del límite mínimo de recursos: utilice estadísticas medias para identificar las necesidades mínimas de recursos durante los periodos de inactividad, establecer las reservas adecuadas y detectar caídas anormales.
- Detección de anomalías: utilice estadísticas mínimas para detectar caídas inusuales en la utilización de los recursos que indiquen posibles problemas, como errores de inicialización o periodos de inactividad inesperados.
- Perfeccionamiento de la política de escalado: utilice estadísticas mínimas para establecer umbrales de reducción horizontal óptimos basados en una utilización mínima viable para evitar un escalado agresivo.
- Planificación de la capacidad: utilice las estadísticas máximas para establecer los tamaños de tareas adecuados y planifique la infraestructura con suficiente margen de maniobra para los picos de tráfico.
- Identificación de los cuellos de botella en el rendimiento: utilice las estadísticas máximas para detectar los puntos de saturación de los recursos, identificar los cuellos de botella y determinar cuándo aumentar el tamaño de las tareas.
- Configuración de la política de escalado: utilice las estadísticas máximas para establecer umbrales de escalado horizontal óptimos en función de los patrones de picos y configure la capacidad de ampliación.

- Supervisión del cumplimiento de los SLA: utilice las estadísticas máximas para hacer un seguimiento de los tiempos de respuesta máxima y las tasas de error a fin de garantizar que el rendimiento del servicio cumpla con los SLA definidos.

Información de métricas relacionadas

Para obtener información sobre Información de contenedores, consulte Casos de uso de Información de contenedores en la Guía del usuario de CloudWatch.

Automaticzación de las respuestas a los errores de Amazon ECS mediante EventBridge

Con Amazon EventBridge, puede automatizar sus servicios de AWS y responder automáticamente a eventos del sistema, como problemas de disponibilidad de aplicaciones o cambios de recursos. Los eventos de los servicios de AWS se envían a EventBridge casi en tiempo real. Puede crear reglas sencillas para indicar qué eventos le resultan de interés, así como qué acciones automatizadas se van a realizar cuando un evento cumple una de las reglas. Entre las acciones que se pueden activar automáticamente, se incluyen las siguientes:

- Adición de eventos a grupos de registros en CloudWatch Logs
- Invocar una función de AWS Lambda
- Invocar Ejecutar comando de Amazon EC2
- Desviar el evento a Amazon Kinesis Data Streams
- Activar una máquina de estado de AWS Step Functions
- Notificación de un tema de Amazon SNS o una cola de Amazon Simple Queue Service (Amazon SQS)

Para obtener más información, consulte Introducción a Amazon EventBridge en la Guía del usuario de Amazon EventBridge.

Puede utilizar eventos de Amazon ECS para EventBridge con el fin de recibir notificaciones casi en tiempo real sobre el estado actual de los clústeres de Amazon ECS. Si las tareas utilizan el tipo de lanzamiento de EC2, puede ver el estado actual de las instancias de contenedor y de todas las tareas que se ejecutan en ellas. Si las tareas utilizan el tipo de lanzamiento de Fargate, puede ver el estado de las instancias de contenedor.

Mediante EventBridge, puede crear programadores personalizados encima de Amazon ECS que son responsables de organizar tareas entre clústeres y de monitorear el estado de los clústeres casi en tiempo real. Puede eliminar el código de programación y monitoreo que sondea continuamente el servicio de Amazon ECS para detectar cambios de estado y controlar en su lugar los cambios de estado de Amazon ECS de forma asíncrona utilizando cualquier destino de EventBridge. Entre los objetivos, podemos mencionar AWS Lambda, Amazon Simple Queue Service, Amazon Simple Notification Service o Amazon Kinesis Data Streams.

Una secuencia de eventos de Amazon ECS garantiza que todos los eventos se entreguen al menos una vez. Si se envían eventos duplicados, el evento ofrece información suficiente para identificar duplicados. Para obtener más información, consulte Control de eventos de Amazon ECS.

Los eventos están ordenados relativamente, para que pueda decir con facilidad cuándo se produjo un evento en relación a los demás.

Temas

- Eventos de Amazon ECS
- Control de eventos de Amazon ECS

Eventos de Amazon ECS

Amazon ECS realiza el seguimiento del estado de cada una de las tareas y los servicios. Si el estado de una tarea o servicio cambia, se genera un evento que se envía a Amazon EventBridge. Estos eventos se clasifican como eventos de cambio de estado de tarea y eventos de acciones de servicio. Estos eventos y sus posibles causas se describen con mayor detalle en las secciones siguientes.

Amazon ECS genera y envía los siguientes tipos de eventos a EventBridge:

- Cambio de estado de instancia de contenedor
- Cambio de estado de tarea
- Deployment state change (Cambio de estado de implementación)
- Acción de servicio

> **Note**
>
> Amazon ECS podría agregar otros tipos de eventos, orígenes y detalles en el futuro. Si está deserializando mediante programación datos JSON de eventos, asegúrese de que la

aplicación esté preparada para tratar propiedades desconocidas para evitar problemas si se agregan estas propiedades adicionales.

En algunos casos, se generan múltiples eventos para la misma actividad. Por ejemplo, cuando se inicia una tarea en una instancia de contenedor, se genera un evento de cambio de estado de la tarea para la nueva tarea. Se genera un evento de cambio de estado de la instancia de contenedor para dar cuenta del cambio en los recursos disponibles, como la CPU, la memoria y los puertos disponibles, en la instancia de contenedor. Del mismo modo, si una instancia de contenedor se termina, se generan eventos para la instancia de contenedor, el estado de la conexión del agente del contenedor y todas las tareas que se ejecutaban en la instancia de contenedor.

Los eventos de cambio de estado del contenedor y de cambio de estado de tarea contienen dos campos `version`: uno en el cuerpo principal del evento y otro en el objeto `detail` del evento. A continuación, se describen las diferencias entre estos dos campos:

- El campo `version` en el cuerpo principal del evento se establece en `0` en todos los eventos. Para obtener más información acerca de los parámetros de EventBridge, consulte Metadatos de eventos de servicio de AWS en la Guía del usuario de Amazon EventBridge.
- El campo `version` en el objeto `detail` describe la versión del recurso asociado. Cada vez que un recurso cambia de estado, se incrementa esta versión. Dado que los eventos se pueden enviar varias veces, este campo permite identificar eventos duplicados. Los eventos duplicados tienen la misma versión en el objeto `detail`. Si está replicando el estado de las tareas y las instancias de contenedor de Amazon ECS con EventBridge, puede comparar la versión de un recurso notificada por las API de Amazon ECS con la versión notificada en EventBridge para el recurso (dentro del objeto `detail`) para verificar que la versión de la secuencia de eventos sea actual.

Los eventos de acciones de servicio solo contienen el campo `version` en el cuerpo principal.

Los eventos de acciones de servicio especifican el servicio en dos campos diferentes:

- En el caso de los eventos generados por `create-service`, el servicio está en el campo `serviceName`.
- En el caso de los eventos generados por `update-service`, el servicio está en el campo `service`.

Si utiliza herramientas automatizadas para los eventos de servicio, debe codificar ambos campos.

Para obtener más información sobre cómo crear una regla para varios servicios y categorías de eventos, consulte Eventos de acciones de servicio de Amazon ECS.

Para obtener información adicional sobre cómo integrar Amazon ECS y EventBridge, consulte Integrating Amazon EventBridge and Amazon ECS (Integración de Amazon EventBridge y Amazon ECS).

Eventos de cambio de estado de instancia de contenedor de Amazon ECS

Los siguientes escenarios provocan eventos de cambio de estado de la instancia de contenedor:

Puede llamar a las operaciones `StartTask`, `RunTask` o `StopTask` de la API, bien directamente o bien con la AWS Management Console o los SDK.

> Colocar o parar tareas en una instancia de contenedor modifica los recursos disponibles en ella (tales como CPU, memoria y puertos disponibles).

El programador de servicio de Amazon ECS comienza o detiene una tarea.

> Colocar o parar tareas en una instancia de contenedor modifica los recursos disponibles en ella (tales como CPU, memoria y puertos disponibles).

El agente de contenedor de Amazon ECS llama a la operación del API `SubmitTaskStateChange` con un estado `STOPPED` para una tarea con un estado deseado `RUNNING`.

> El agente de contenedor de Amazon ECS monitorea el estado de las tareas de las instancias de contenedor y notifica los cambios de estado. Si una tarea que se supone en estado `RUNNING` pasa a `STOPPED`, el agente libera los recursos que se asignaron a la tarea parada (tales como CPU, memoria y puertos disponibles).

El registro de la instancia de contenedor se cancela con la operación `DeregisterContainerInstance` de la API, ya sea directamente o con la AWS Management Console o los SDK.

> La anulación del registro de una instancia de contenedor cambia el estado de la instancia de contenedor y el estado de conexión del agente de contenedor de Amazon ECS.

Una tarea se paró al parar la instancia EC2.

> Cuando se para una instancia de contenedor, las tareas que se estaban ejecutando en la misma pasan al estado `STOPPED`.

El agente de contenedor de Amazon ECS registra una instancia de contenedor por primera vez.

La primera vez que el agente de contenedor de Amazon ECS registra una instancia de contenedor (durante el lanzamiento o cuando se ejecuta manualmente por primera vez), se crea un evento de cambio de estado para la instancia.

El agente de contenedor de Amazon ECS se conecta o desconecta de Amazon ECS.

Cuando el agente de contenedor de Amazon ECS se conecta o desconecta del backend de , cambia el estado `agentConnected` de la instancia de contenedor.

> **Note**
>
> El agente de contenedor de Amazon ECS se desconecta y se vuelve a conectar varias veces por hora como parte de su operación normal, por lo que cabe esperar que se produzcan eventos de conexión del agente. Estos eventos no indican que haya un problema con el agente de contenedores ni con o su instancia de contenedor.

El agente de contenedor de Amazon ECS se actualiza en una instancia.

El detalle de instancia de contenedor contiene un objeto para la versión del agente de contenedor. Si actualiza el agente, esta información de la versión cambia y genera un evento.

Example Evento de cambio de estado de instancia de contenedor

Los eventos de cambio de estado de instancia de contenedor se entregan en el siguiente formato. La sección `detail` siguiente se asemeja al objeto ContainerInstance que se devuelve de una operación de la API DescribeContainerInstances en la Referencia de la API de Amazon Elastic Container Service. Para obtener más información acerca de los parámetros de EventBridge, consulte Metadatos de eventos de servicio deAWS en la Guía del usuario de Amazon EventBridge.

```
{
  "version": "0",
  "id": "8952ba83-7be2-4ab5-9c32-6687532d15a2",
  "detail-type": "ECS Container Instance State Change",
  "source": "aws.ecs",
  "account": "111122223333",
  "time": "2016-12-06T16:41:06Z",
  "region": "us-east-1",
  "resources": [
```

```
    "arn:aws:ecs:us-east-1:111122223333:container-instance/
b54a2a04-046f-4331-9d74-3f6d7f6ca315"
  ],
  "detail": {
    "agentConnected": true,
    "attributes": [
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
      },
      {
        "name": "com.amazonaws.ecs.capability.task-iam-role-network-host"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.awslogs"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
      },
      {
        "name": "com.amazonaws.ecs.capability.privileged-container"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
      },
      {
        "name": "com.amazonaws.ecs.capability.ecr-auth"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.20"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.21"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.22"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.23"
```

```
      },
      {
        "name": "com.amazonaws.ecs.capability.task-iam-role"
      }
    ],
    "clusterArn": "arn:aws:ecs:us-east-1:111122223333:cluster/default",
    "containerInstanceArn": "arn:aws:ecs:us-east-1:111122223333:container-instance/
b54a2a04-046f-4331-9d74-3f6d7f6ca315",
    "ec2InstanceId": "i-f3a8506b",
    "registeredResources": [
      {
        "name": "CPU",
        "type": "INTEGER",
        "integerValue": 2048
      },
      {
        "name": "MEMORY",
        "type": "INTEGER",
        "integerValue": 3767
      },
      {
        "name": "PORTS",
        "type": "STRINGSET",
        "stringSetValue": [
          "22",
          "2376",
          "2375",
          "51678",
          "51679"
        ]
      },
      {
        "name": "PORTS_UDP",
        "type": "STRINGSET",
        "stringSetValue": []
      }
    ],
    "remainingResources": [
      {
        "name": "CPU",
        "type": "INTEGER",
        "integerValue": 1988
      },
      {
```

```
        "name": "MEMORY",
        "type": "INTEGER",
        "integerValue": 767
      },
      {
        "name": "PORTS",
        "type": "STRINGSET",
        "stringSetValue": [
          "22",
          "2376",
          "2375",
          "51678",
          "51679"
        ]
      },
      {
        "name": "PORTS_UDP",
        "type": "STRINGSET",
        "stringSetValue": []
      }
    ],
    "status": "ACTIVE",
    "version": 14801,
    "versionInfo": {
      "agentHash": "aebcbca",
      "agentVersion": "1.13.0",
      "dockerVersion": "DockerVersion: 1.11.2"
    },
    "updatedAt": "2016-12-06T16:41:06.991Z"
  }
}
```

Eventos de cambio de estado de tarea de Amazon ECS

Los siguientes escenarios provocan eventos de cambio de estado de la tarea:

Puede llamar a las operaciones `StartTask`, `RunTask` o `StopTask` de la API, ya sea directamente o con la AWS Management Console, la AWS CLI o los SDK.

- Comenzar o parar tareas crea nuevos recursos de tareas o modifica el estado de los recursos de tarea existentes.

El programador de servicio de Amazon ECS comienza o detiene una tarea.

Comenzar o parar tareas crea nuevos recursos de tareas o modifica el estado de los recursos de tarea existentes.

El agente de contenedor de Amazon ECS llama a la operación `SubmitTaskStateChange` de la API.

En el caso del tipo de lanzamiento de Amazon EC2, el agente de contenedor de Amazon ECS supervisa el estado de las tareas en las instancias de contenedor. El agente de contenedor de Amazon ECS informa de los cambios de estado. Los cambios de estado pueden incluir cambios de `PENDING` a `RUNNING` o de `RUNNING` a `STOPPED`.

La cancelación del registro de la instancia de contenedor subyacente se fuerza con la operación `DeregisterContainerInstance` de la API y la marca `force`, ya sea directamente o con la AWS Management Console o los SDK.

La anulación del registro de una instancia de contenedor cambia el estado de la instancia de contenedor y el estado de conexión del agente de contenedor de Amazon ECS. Si las tareas se ejecutan en la instancia de contenedor, se debe establecer la marca `force` para permitir la cancelación del registro. Esto para todas las tareas en la instancia.

La instancia de contenedor subyacente se para o se termina.

Cuando se para o se termina una instancia de contenedor, las tareas que se estaban ejecutando en la misma pasan al estado `STOPPED`.

Un contenedor en la tarea cambia de estado.

El agente de contenedor de Amazon ECS monitorea el estado de los contenedores dentro de las tareas. Por ejemplo, si un contenedor que se ejecuta dentro de una tarea se detiene, este cambio de estado del contenedor genera un evento.

Una tarea que utiliza el proveedor de capacidad de Fargate Spot recibe un aviso de terminación.

Cuando una tarea utiliza el proveedor de capacidad de `FARGATE_SPOT` y se detiene debido a una interrupción del punto de acceso, se genera un evento de cambio de estado de la tarea.

Example Evento de cambio de estado de tarea

Los eventos de cambio de estado de tarea se entregan en el siguiente formato. Tenga en cuenta lo siguiente sobre los campos:

- El estado de salud del evento no está disponible en el evento de cambio de estado de la tarea. Si necesita conocer el estado de salud de la tarea, puede ejecutar describe-tasks.
- Si sus contenedores utilizan una imagen alojada con Amazon ECR, se devuelve el campo `imageDigest`.
- Los valores de los campos `createdAt`, `connectivityAt`, `pullStartedAt`, `startedAt`, `pullStoppedAt` y `updatedAt` son marcas de tiempo de cadenas ISO.
- El valor de `detail-type` es “ECS Task State Change”.
- Cuando se genera el evento para una tarea detenida, los campos `stoppedReason` y `stoppedCode` proporcionan información adicional sobre el motivo por el que se detuvo la tarea (por ejemplo, “Iniciada por el usuario”).

Para obtener más información acerca de los parámetros de EventBridge, consulte Metadatos de eventos de servicio deAWS en la Refencia de eventos de Amazon EventBridge.

Para obtener información sobre cómo configurar una regla de eventos de Amazon EventBridge que solo captura eventos de tarea en los que se ha parado la ejecución de la tarea porque se ha terminado uno de sus contenedores esenciales, consulte Envío de alertas de Amazon Simple Notification Service para eventos de tareas detenidas de Amazon ECS

```
{
    "version": "0",
    "id": "105f6bb1-4da6-c630-4965-35383018cbca",
    "detail-type": "ECS Task State Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2025-05-06T11:02:34Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/a1173316d40a45dea9"
    ],
    "detail": {
        "attachments": [
            {
                "id": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                "type": "eni",
                "status": "ATTACHED",
                "details": [
                    {
                        "name": "subnetId",
```

```
                        "value": "subnet-0d0eab1bb38d5ca64"
                    },
                    {
                        "name": "networkInterfaceId",
                        "value": "eni-0103a2f01bad57d71"
                    },
                    {
                        "name": "macAddress",
                        "value": "0e:50:d1:c1:77:81"
                    },
                    {
                        "name": "privateDnsName",
                        "value": "ip-10-0-1-163.ec2.internal"
                    },
                    {
                        "name": "privateIPv4Address",
                        "value": "10.0.1.163"
                    }
                ]
            }
        ],
        "attributes": [
            {
                "name": "ecs.cpu-architecture",
                "value": "x86_64"
            }
        ],
        "availabilityZone": "us-east-1b",
        "capacityProviderName": "FARGATE",
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/example-cluster",
        "connectivity": "CONNECTED",
        "connectivityAt": "2025-05-06T11:02:17.19Z",
        "containers": [
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example-
cluster/a1173316d40a45dea9/a0a99b87-baa8-4bf6-b9f1-a9a95917a635",
                "lastStatus": "RUNNING",
                "name": "web",
                "image": "nginx",
                "imageDigest":
 "sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66",
                "runtimeId": "a1173316d40a45dea9-0265927825",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
```

```
                "networkInterfaces": [
                    {
                        "attachmentId": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                        "privateIpv4Address": "10.0.1.163"
                    }
                ],
                "cpu": "99",
                "memory": "100"
            },
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example-
cluster/a1173316d40a45dea9/a2010e2d-ba7c-4135-8b79-e0290ff3cd8c",
                "lastStatus": "RUNNING",
                "name": "aws-guardduty-agent-nm40lC",
                "imageDigest":
 "sha256:bf9197abdf853607e5fa392b4f97ccdd6ca56dd179be3ce8849e552d96582ac8",
                "runtimeId": "a1173316d40a45dea9-2098416933",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
                "networkInterfaces": [
                    {
                        "attachmentId": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                        "privateIpv4Address": "10.0.1.163"
                    }
                ],
                "cpu": "null"
            },
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example-
cluster/a1173316d40a45dea9/dccf0ca2-d929-471f-a5c3-98006fd4379e",
                "lastStatus": "RUNNING",
                "name": "aws-otel-collector",
                "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.32.0",
                "imageDigest":
 "sha256:7a1b3560655071bcacd66902c20ebe9a69470d5691fe3bd36baace7c2f3c4640",
                "runtimeId": "a1173316d40a45dea9-4027662657",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
                "networkInterfaces": [
                    {
                        "attachmentId": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                        "privateIpv4Address": "10.0.1.163"
                    }
                ],
```

```
                "cpu": "0"
            }
        ],
        "cpu": "256",
        "createdAt": "2025-05-06T11:02:13.877Z",
        "desiredStatus": "RUNNING",
        "enableExecuteCommand": false,
        "ephemeralStorage": {
            "sizeInGiB": 20
        },
        "group": "family:webserver",
        "launchType": "FARGATE",
        "lastStatus": "RUNNING",
        "memory": "512",
        "overrides": {
            "containerOverrides": [
                {
                    "name": "web"
                },
                {
                    "environment": [
                        {
                            "name": "CLUSTER_NAME",
                            "value": "example-cluster"
                        },
                        {
                            "name": "REGION",
                            "value": "us-east-1"
                        },
                        {
                            "name": "HOST_PROC",
                            "value": "/host_proc"
                        },
                        {
                            "name": "AGENT_RUNTIME_ENVIRONMENT",
                            "value": "ecsfargate"
                        },
                        {
                            "name": "STAGE",
                            "value": "prod"
                        }
                    ],
                    "memory": 128,
                    "name": "aws-guardduty-agent-nm40lC"
```

```
                },
                {
                    "name": "aws-otel-collector"
                }
            ]
        },
        "platformVersion": "1.4.0",
        "pullStartedAt": "2025-05-06T11:02:24.162Z",
        "pullStoppedAt": "2025-05-06T11:02:33.493Z",
        "startedAt": "2025-05-06T11:02:34.325Z",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/
webserver:5",
        "updatedAt": "2025-05-06T11:02:34.325Z",
        "version": 3
    }
}
```

Example

A continuación se muestra un ejemplo de un evento de cambio de estado de tarea para el tipo de lanzamiento EC2.

```
{
    "version": "0",
    "id": "a65cf262-f104-0dd5-ceda-4b09ba71a441",
    "detail-type": "ECS Task State Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2025-05-12T13:12:06Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ecs:us-east-1:123456789012:task/example/c1ffa94f19a540ed8d9f7e1d2a5d"
    ],
    "detail": {
        "attachments": [
            {
                "id": "52333e3b-b812-41a8-b057-9ed184bbe5e1",
                "type": "eni",
                "status": "ATTACHED",
                "details": [
                    {
```

```
                        "name": "subnetId",
                        "value": "subnet-0d0eab1bb38d5ca64"
                    },
                    {
                        "name": "networkInterfaceId",
                        "value": "eni-0ea90f746500773a4"
                    },
                    {
                        "name": "macAddress",
                        "value": "0e:d5:9b:ce:49:fb"
                    },
                    {
                        "name": "privateDnsName",
                        "value": "ip-10-0-1-37.ec2.internal"
                    },
                    {
                        "name": "privateIPv4Address",
                        "value": "10.0.1.37"
                    }
                ]
            }
        ],
        "attributes": [
            {
                "name": "ecs.cpu-architecture",
                "value": "x86_64"
            }
        ],
        "availabilityZone": "us-east-1b",
        "capacityProviderName": "Infra-ECS-Cluster-example-fa84e0cc-
AsgCapacityProvider-OseQJU9pizmp",
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/example",
        "connectivity": "CONNECTED",
        "connectivityAt": "2025-05-12T13:11:44.98Z",
        "containerInstanceArn": "arn:aws:ecs:us-east-1:123456789012:container-instance/
example/d1d84798400f49f3b21cb61610c1e",
        "containers": [
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example/
c1ffa94f19a540ed8d9f7e1d2a5d3626/197d0994-5367-4a6d-9f9a-f075e4a6",
                "lastStatus": "RUNNING",
                "name": "aws-otel-collector",
                "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.32.0",
```

```
                "imageDigest":
 "sha256:7a1b3560655071bcacd66902c20ebe9a69470d5691fe3bd36baace7c2f3c4640",
                "runtimeId":
 "8e926f0ccd8fe2b459926f49584ba6d33a3d9f61398dbabe944ee6a13a8ff3a1",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example/
c1ffa94f19a540ed8d9f7e1d2a5d",
                "networkInterfaces": [
                    {
                        "attachmentId": "52333e3b-b812-41a8-b057-9ed184bbe5e1",
                        "privateIpv4Address": "10.0.1.37"
                    }
                ],
                "cpu": "0"
            },
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example/
c1ffa94f19a540ed8d9f7e1d2a5d3626/cab39ef0-9c50-459d-844b-b9d51d73d",
                "lastStatus": "RUNNING",
                "name": "web",
                "image": "nginx",
                "imageDigest":
 "sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66",
                "runtimeId":
 "9f1c73f0094f051541d9e5c2ab1e172d83c4eb5171bcc857c4504b02770ff3b8",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example/
c1ffa94f19a540ed8d9f7e1d2a5d",
                "networkInterfaces": [
                    {
                        "attachmentId": "52333e3b-b812-41a8-b057-9ed184bbe5e1",
                        "privateIpv4Address": "10.0.1.37"
                    }
                ],
                "cpu": "99",
                "memory": "100"
            }
        ],
        "cpu": "256",
        "createdAt": "2025-05-12T13:11:44.98Z",
        "desiredStatus": "RUNNING",
        "enableExecuteCommand": false,
        "group": "family:webserver",
        "launchType": "EC2",
        "lastStatus": "RUNNING",
        "memory": "512",
```

```
        "overrides": {
            "containerOverrides": [
                {
                    "name": "aws-otel-collector"
                },
                {
                    "name": "web"
                }
            ]
        },
        "pullStartedAt": "2025-05-12T13:11:59.491Z",
        "pullStoppedAt": "2025-05-12T13:12:05.896Z",
        "startedAt": "2025-05-12T13:12:06.053Z",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example/
c1ffa94f19a540ed8d9f7e1d2a5d",
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/
webserver",
        "updatedAt": "2025-05-12T13:12:06.053Z",
        "version": 4
    }
}
```

Eventos de acciones de servicio de Amazon ECS

Amazon ECS envía eventos de acciones de servicio con el tipo de detalle ECS Service Action (Acción de servicio de ECS). A diferencia de los eventos de cambio de instancia de contenedor y de estado de tarea, los eventos de acciones de servicio no incluyen un número de versión en el campo de respuesta `details`. El siguiente es un patrón de eventos que se utiliza con el fin de crear una regla de EventBridge para los eventos de acciones de servicio de Amazon ECS. Para obtener más información, consulte Introducción a Amazon EventBridge en la Guía del usuario de Amazon EventBridge.

```
{
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Service Action"
    ]
}
```

Amazon ECS envía eventos con los tipos de eventos `INFO`, `WARN` y `ERROR`. Los siguientes son los eventos de acciones de servicio.

Eventos de acciones de servicio con el tipo de evento **`INFO`**

`SERVICE_STEADY_STATE`

El estado del servicio es correcto y el número de tareas realizadas es el deseado, por lo que se ha alcanzado un estado es estable. El programador de servicios informa periódicamente del estado, por lo que podría recibir este mensaje varias veces.

`TASKSET_STEADY_STATE`

El estado del conjunto de tareas es correcto y el número de tareas realizadas es el deseado, por lo que se ha alcanzado un estado es estable.

`CAPACITY_PROVIDER_STEADY_STATE`

Un proveedor de capacidad asociado a un servicio ha alcanzado un estado estable.

`SERVICE_DESIRED_COUNT_UPDATED`

Se envía cuando el programador de servicios actualiza el recuento calculado deseado para un servicio o conjunto de tareas. Este evento no se envía cuando un usuario actualiza manualmente el recuento deseado.

`TASKS_STOPPED`

El servicio ha detenido la tarea en ejecución.

`SERVICE_DEPLOYMENT_IN_PROGRESS`

Una implementación de servicios se encuentra en curso. La implementación del servicio puede consistir en una reversión o en una nueva revisión del servicio.

`SERVICE_DEPLOYMENT_COMPLETED`

Una implementación de servicio se encuentra en estado estable y se ha completado. La implementación del servicio puede consistir en una reversión o en la implementación de una revisión del servicio actualizada.

Eventos de acciones de servicio con el tipo de evento **`WARN`**

`SERVICE_TASK_START_IMPAIRED`

El servicio no puede iniciar sistemáticamente las tareas de forma correcta.

`SERVICE_DISCOVERY_INSTANCE_UNHEALTHY`

Un servicio que utiliza la detección de servicios contiene una tarea cuyo estado no es correcto. El programador de servicios detecta que el estado de una tarea contenida en el registro no es correcto.

`VPC_LATTICE_TARGET_UNHEALTHY`

El servicio que utiliza VPC Lattice ha detectado que uno de los destinos de VPC Lattice se encuentra en mal estado.

Eventos de acciones de servicio con el tipo de evento **`ERROR`**

`SERVICE_DAEMON_PLACEMENT_CONSTRAINT_VIOLATED`

Una tarea de un servicio que utiliza la estrategia del programador de servicios `DAEMON` ya no cumple la estrategia de delimitación de ubicación para el servicio.

`ECS_OPERATION_THROTTLED`

El programador de servicios se ha sometido a una limitación controlada debido a los límites de solicitud de la API de Amazon ECS.

`SERVICE_DISCOVERY_OPERATION_THROTTLED`

El programador de servicios se ha sometido a una limitación controlada debido a los límites de limitación controlada de la API de AWS Cloud Map. Esto puede ocurrir en los servicios configurados para usar la detección de servicios.

`SERVICE_TASK_PLACEMENT_FAILURE`

El programador de servicios no puede ubicar una tarea. La causa se describe en el campo `reason`.

Una causa común de que se genere este evento de servicio es la falta de recursos en el clúster para ubicar la tarea. Por ejemplo, no hay suficiente capacidad de CPU o memoria en las instancias de contenedor disponibles o no hay ninguna instancia de contenedor disponible. Otra causa frecuente es cuando el agente de contenedor de Amazon ECS se desconecta en la instancia de contenedor, lo que provoca que el programador no pueda ubicar la tarea.

`SERVICE_TASK_CONFIGURATION_FAILURE`

El programador de servicios no puede realizar una tarea debido a un error de configuración. La causa se describe en el campo `reason`.

Una causa común de que se genere este evento de servicio es porque se aplicaban etiquetas al servicio, pero el usuario o rol no había optado por el nuevo formato de nombre de Recurso de Amazon (ARN) en la región. Para obtener más información, consulte Nombres de recursos de Amazon (ARN) e ID. Otra causa frecuente es que Amazon ECS no podía asumir el rol de IAM de la tarea proporcionado.

`SERVICE_HEALTH_UNKNOWN`

El servicio no pudo describir los datos de estado de las tareas.

`SERVICE_DEPLOYMENT_FAILED`

Una implementación de servicio no alcanzó el nivel estable. Esto ocurre cuando se activa una instancia de CloudWatch o el interruptor detecta un error en la implementación del servicio.

Example Evento de estado estable de servicio

Los eventos de estado estable de servicio se entregan en el siguiente formato. Para obtener más información acerca de los parámetros de EventBridge, consulte Eventos de EventBridge en la Guía del usuario de Amazon EventBridge.

```
{
    "version": "0",
    "id": "af3c496d-f4a8-65d1-70f4-a69d52e9b584",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:27:22Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "INFO",
        "eventName": "SERVICE_STEADY_STATE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "createdAt": "2019-11-19T19:27:22.695Z"
    }
}
```

Example Evento de estado estable del proveedor de capacidad

Los eventos de estado estable del proveedor de capacidad se entregan en el siguiente formato.

```
{
    "version": "0",
    "id": "b9baa007-2f33-0eb1-5760-0d02a572d81f",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:37:00Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "INFO",
        "eventName": "CAPACITY_PROVIDER_STEADY_STATE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "capacityProviderArns": [
            "arn:aws:ecs:us-west-2:111122223333:capacity-provider/ASG-tutorial-
capacity-provider"
        ],
        "createdAt": "2019-11-19T19:37:00.807Z"
    }
}
```

Example Evento deteriorado de inicio de tarea de servicio

Los eventos deteriorados de inicio de tarea de servicio se entregan en el siguiente formato.

```
{
    "version": "0",
    "id": "57c9506e-9d21-294c-d2fe-e8738da7e67d",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:55:38Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "WARN",
        "eventName": "SERVICE_TASK_START_IMPAIRED",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "createdAt": "2019-11-19T19:55:38.725Z"
```

```
    }
}
```

Example Evento de error de ubicación de tarea de servicio

Los eventos de error de ubicación de tarea de servicio se entregan en el siguiente formato. Para obtener más información, consulte Eventos de EventBridge en la Guía del usuario de Amazon EventBridge.

En el ejemplo siguiente, la tarea ha intentado utilizar el proveedor de capacidad `FARGATE_SPOT`, pero el programador de servicios no ha podido adquirir ninguna capacidad de Fargate Spot.

```
{
    "version": "0",
    "id": "ddca6449-b258-46c0-8653-e0e3a6d0468b",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:55:38Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "ERROR",
        "eventName": "SERVICE_TASK_PLACEMENT_FAILURE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "capacityProviderArns": [
            "arn:aws:ecs:us-west-2:111122223333:capacity-provider/FARGATE_SPOT"
        ],
        "reason": "RESOURCE:FARGATE",
        "createdAt": "2019-11-06T19:09:33.087Z"
    }
}
```

En el siguiente ejemplo para el tipo de lanzamiento de EC2, se ha intentado lanzar la tarea en la instancia de contenedor `2dd1b186f39845a584488d2ef155c131` pero el programador de servicios no pudo realizar la tarea debido a la insuficiencia de la CPU.

```
{
  "version": "0",
```

```
  "id": "ddca6449-b258-46c0-8653-e0e3a6d0468b",
  "detail-type": "ECS Service Action",
  "source": "aws.ecs",
  "account": "111122223333",
  "time": "2019-11-19T19:55:38Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
  ],
  "detail": {
    "eventType": "ERROR",
    "eventName": "SERVICE_TASK_PLACEMENT_FAILURE",
    "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
    "containerInstanceArns": [
    "arn:aws:ecs:us-west-2:111122223333:container-instance/
default/2dd1b186f39845a584488d2ef155c131"
    ],
    "reason": "RESOURCE:CPU",
    "createdAt": "2019-11-06T19:09:33.087Z"
  }
}
```

Eventos de cambio de estado de implementación de servicios de Amazon ECS

Amazon ECS envía eventos de cambio de estado de implementación de servicios con el tipo de detalle ECS Deployment State Change (Cambio de estado de implementación de ECS). El patrón de eventos siguiente se utiliza para crear una regla de EventBridge para los eventos de cambio de estado de implementación de servicios de Amazon ECS. Para obtener más información acerca de la creación de una regla de EventBridge, consulte Introducción a Amazon EventBridge en la Guía del usuario de Amazon EventBridge.

```
{
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Deployment State Change"
    ]
}
```

Amazon ECS envía eventos con los tipos de eventos `INFO` y `ERROR`. Para obtener más información, consulte Eventos de acciones de servicio de Amazon ECS

A continuación, se incluyen los eventos de cambio de estado de implementación de servicios.

`SERVICE_DEPLOYMENT_IN_PROGRESS`

La implementación de servicios está en curso. Este evento se envía tanto para implementaciones iniciales como para implementaciones de restauración.

`SERVICE_DEPLOYMENT_COMPLETED`

Se ha terminado la implementación de servicios. Este evento se envía una vez que un servicio ha alcanzado un estado estable después de una implementación.

`SERVICE_DEPLOYMENT_FAILED`

Error en la implementación de servicios. Este evento se envía para servicios que tengan habilitada la lógica de interruptor de implementación.

Example evento de implementación de servicios en curso

Los eventos de implementación de servicios en curso se entregan cuando se comienza una implementación inicial y una implementación de restauración. La diferencia entre las dos radica en el campo `reason`. Para obtener más información acerca de los parámetros de EventBridge, consulte Metadatos de eventos de servicio de AWS en la Guía del usuario de Amazon EventBridge.

A continuación, se muestra un resultado de ejemplo del comienzo de una implementación inicial.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3a6EXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
   "time": "2020-05-23T12:31:14Z",
   "region": "us-west-2",
   "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
   ],
   "detail": {
        "eventType": "INFO",
        "eventName": "SERVICE_DEPLOYMENT_IN_PROGRESS",
        "deploymentId": "ecs-svc/123",
        "updatedAt": "2020-05-23T11:11:11Z",
        "reason": "ECS deployment deploymentId in progress."
```

```
   }
}
```

A continuación, se muestra un resultado de ejemplo del comienzo de una implementación de restauración. El campo `reason` proporciona el ID de la implementación a la que se está restaurando el servicio.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
   "time": "2020-05-23T12:31:14Z",
   "region": "us-west-2",
   "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
   ],
   "detail": {
        "eventType": "INFO",
        "eventName": "SERVICE_DEPLOYMENT_IN_PROGRESS",
        "deploymentId": "ecs-svc/123",
        "updatedAt": "2020-05-23T11:11:11Z",
        "reason": "ECS deployment circuit breaker: rolling back to
 deploymentId deploymentID."
   }
}
```

Example evento completado de implementación de servicios

Los eventos de estado de implementación de servicio completada se entregan en el siguiente formato. Para obtener más información, consulte Implementación de los servicios de Amazon ECS mediante el reemplazo de tareas.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
   "time": "2020-05-23T12:31:14Z",
   "region": "us-west-2",
```

```
    "resources": [
         "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
         "eventType": "INFO",
         "eventName": "SERVICE_DEPLOYMENT_COMPLETED",
         "deploymentId": "ecs-svc/123",
         "updatedAt": "2020-05-23T11:11:11Z",
         "reason": "ECS deployment deploymentID completed."
    }
}
```

Example Evento de error de implementación de servicios

Los eventos de estado de error de implementación de servicio se entregan en el siguiente formato. Solo se enviará un evento de estado de error de implementación de servicio para los servicios que tengan habilitada la lógica del interruptor de implementación. Para obtener más información, consulte Implementación de los servicios de Amazon ECS mediante el reemplazo de tareas.

```
{
    "version": "0",
    "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
    "detail-type": "ECS Deployment State Change",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2020-05-23T12:31:14Z",
    "region": "us-west-2",
    "resources": [
         "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
         "eventType": "ERROR",
         "eventName": "SERVICE_DEPLOYMENT_FAILED",
         "deploymentId": "ecs-svc/123",
         "updatedAt": "2020-05-23T11:11:11Z",
         "reason": "ECS deployment circuit breaker: task failed to start."
    }
}
```

Control de eventos de Amazon ECS

Amazon ECS envía los eventos siguiendo el criterio al menos una vez. Esto significa que podría recibir varias copias de un evento determinado. Además, los eventos podrían no entregarse a los escuchas de evento en el orden en que ocurrieron los eventos.

Para permitir un orden correcto de eventos, la sección `detail` de cada evento contiene una propiedad `version`. Cada vez que un recurso cambia de estado, se incrementa esta `version`. Los eventos duplicados tienen la misma `version` en el objeto `detail`. Si está replicando la instancia de contenedor de Amazon ECS y el estado de tarea con EventBridge, puede comparar la versión de un recurso notificado por las API de Amazon ECS con la `version` notificada en EventBridge para que el recurso verifique que la versión en su secuencia de eventos sea actual. Los eventos con un número de propiedad de versión más alto se deberían tratar como que han ocurrido más tarde que los eventos con números de versión más bajos.

Ejemplo: Control de eventos en una función de AWS Lambda

El ejemplo siguiente muestra una función de Lambda escrita en Python 3.9 que captura los eventos de cambio de estado de instancia de contenedor y de tarea y los guarda en una de estas dos tablas de Amazon DynamoDB:

- ECSCtrInstanceState: almacena el estado más reciente de una instancia de contenedor. El ID de tabla es el valor `containerInstanceArn` de la instancia de contenedor.
- ECSTaskState almacena el estado más reciente de una tarea. El ID de tabla es el valor `taskArn` de la tarea.

```
import json
import boto3

def lambda_handler(event, context):
    id_name = ""
    new_record = {}

    # For debugging so you can see raw event format.
    print('Here is the event:')
    print((json.dumps(event)))

    if event["source"] != "aws.ecs":
```

```
      raise ValueError("Function only supports input from events with a source type
of: aws.ecs")

   # Switch on task/container events.
   table_name = ""
   if event["detail-type"] == "ECS Task State Change":
       table_name = "ECSTaskState"
       id_name = "taskArn"
       event_id = event["detail"]["taskArn"]
   elif event["detail-type"] == "ECS Container Instance State Change":
       table_name = "ECSCtrInstanceState"
       id_name =  "containerInstanceArn"
       event_id = event["detail"]["containerInstanceArn"]
   else:
       raise ValueError("detail-type for event is not a supported type. Exiting
without saving event.")

   new_record["cw_version"] = event["version"]
   new_record.update(event["detail"])

   # "status" is a reserved word in DDB, but it appears in containerPort
   # state change messages.
   if "status" in event:
       new_record["current_status"] = event["status"]
       new_record.pop("status")


   # Look first to see if you have received a newer version of an event ID.
   # If the version is OLDER than what you have on file, do not process it.
   # Otherwise, update the associated record with this latest information.
   print("Looking for recent event with same ID...")
   dynamodb = boto3.resource("dynamodb", region_name="us-east-1")
   table = dynamodb.Table(table_name)
   saved_event = table.get_item(
       Key={
           id_name : event_id
       }
   )
   if "Item" in saved_event:
       # Compare events and reconcile.
       print(("EXISTING EVENT DETECTED: Id " + event_id + " - reconciling"))
       if saved_event["Item"]["version"] < event["detail"]["version"]:
           print("Received event is a more recent version than the stored event -
updating")
```

```
            table.put_item(
                Item=new_record
            )
        else:
            print("Received event is an older version than the stored event -
 ignoring")
    else:
        print(("Saving new event - ID " + event_id))

        table.put_item(
            Item=new_record
        )
```

El ejemplo siguiente de Fargate muestra una función de Lambda escrita en Python 3.9 que captura los eventos de cambio de estado de tarea y los guarda en la siguiente tabla de Amazon DynamoDB:

```
import json
import boto3

def lambda_handler(event, context):
    id_name = ""
    new_record = {}

    # For debugging so you can see raw event format.
    print('Here is the event:')
    print((json.dumps(event)))

    if event["source"] != "aws.ecs":
       raise ValueError("Function only supports input from events with a source type
 of: aws.ecs")

    # Switch on task/container events.
    table_name = ""
    if event["detail-type"] == "ECS Task State Change":
        table_name = "ECSTaskState"
        id_name = "taskArn"
        event_id = event["detail"]["taskArn"]
    else:
        raise ValueError("detail-type for event is not a supported type. Exiting
 without saving event.")

    new_record["cw_version"] = event["version"]
    new_record.update(event["detail"])
```

```
    # "status" is a reserved word in DDB, but it appears in containerPort
    # state change messages.
    if "status" in event:
        new_record["current_status"] = event["status"]
        new_record.pop("status")


    # Look first to see if you have received a newer version of an event ID.
    # If the version is OLDER than what you have on file, do not process it.
    # Otherwise, update the associated record with this latest information.
    print("Looking for recent event with same ID...")
    dynamodb = boto3.resource("dynamodb", region_name="us-east-1")
    table = dynamodb.Table(table_name)
    saved_event = table.get_item(
        Key={
            id_name : event_id
        }
    )
    if "Item" in saved_event:
        # Compare events and reconcile.
        print(("EXISTING EVENT DETECTED: Id " + event_id + " - reconciling"))
        if saved_event["Item"]["version"] < event["detail"]["version"]:
            print("Received event is a more recent version than the stored event -
updating")
            table.put_item(
                Item=new_record
            )
        else:
            print("Received event is an older version than the stored event -
ignoring")
    else:
        print(("Saving new event - ID " + event_id))

        table.put_item(
            Item=new_record
        )
```

Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada

Información de contenedores recopila, agrega y resume métricas y registros de las aplicaciones y microservicios en contenedores. Información de contenedores con observabilidad mejorada proporciona todas las métricas de Información de contenedores, además de métricas adicionales de tareas y contenedores.

Información de contenedores detecta todos los contenedores en ejecución en un clúster y recopila datos de rendimiento en cada capa de la pila de rendimiento. Los datos operativos se recopilan como eventos de registro de rendimiento. Son entradas que usan un esquema JSON estructurado que permite incorporar y almacenar datos de cardinalidad alta a escala. A partir de estos datos, CloudWatch crea métricas agregadas de nivel superior en el nivel de clúster, servicio y nivel de tarea como métricas de CloudWatch. Las métricas incluyen la utilización de recursos como CPU, memoria, disco y red. Las métricas están disponibles en los paneles automáticos de CloudWatch.

Las métricas solo reflejan los recursos con tareas en ejecución durante el intervalo de tiempo especificado. Por ejemplo, si tiene un clúster con un servicio, pero ese servicio no tiene tareas en un estado `RUNNING`, no se enviarán métricas a CloudWatch. Si tiene dos servicios y uno de ellos tiene tareas en ejecución y el otro no, solo se enviarán las métricas del servicio con tareas en ejecución.

El 2 de diciembre de 2024, AWS publicó Información de contenedores con observabilidad mejorada para Amazon ECS. Esta versión admite la observabilidad mejorada de los clústeres de Amazon ECS que utilizan los tipos de lanzamiento de Amazon EC2 y Fargate. Después de configurar Información de contenedores con observabilidad mejorada en Amazon ECS, Información de contenedores recopila automáticamente la telemetría detallada de la infraestructura desde el nivel del clúster hasta el nivel del contenedor en su entorno y muestra estos datos de rendimiento críticos en paneles seleccionados, lo que elimina el trabajo pesado de configurar la observabilidad. Para obtener más información sobre cómo configurar Información de contenedores con observabilidad mejorada, consulte Información de contenedores con observabilidad mejorada.

Recomendamos que utilice Información de contenedores con capacidad de observabilidad mejorada en lugar de Información de contenedores, ya que proporciona una visibilidad detallada del entorno de contenedores y reduce el tiempo medio de resolución. Para obtener más información, consulte Métricas de Información de contenedores con observabilidad mejorada de Amazon ECS en la Guía del usuario de Amazon CloudWatch.

Los eventos que puede ver son los que Amazon ECS envía a Amazon EventBridge. Para obtener más información, consulte Eventos de Amazon ECS.

> **Important**
>
> Las métricas recopiladas por CloudWatch Container Insights se cobran como métricas personalizadas. Para obtener más información acerca de los precios de CloudWatch, consulte Precios de CloudWatch.

Determine el estado de las tareas de Amazon ECS mediante comprobaciones de estado de los contenedores

Al crear una definición de tarea, puede configurar una comprobación de estado para los contenedores. Las comprobaciones de estado son comandos que se ejecutan localmente en un contenedor y validan el estado y la disponibilidad de la aplicación.

El agente de contenedor de Amazon ECS solo monitorea e informa sobre las comprobaciones de estado que están especificadas en la definición de tareas. Amazon ECS no monitorea las comprobaciones de estado de Docker que están integradas en una imagen de contenedor, pero que no están especificadas en la definición de contenedor. Los parámetros de comprobación de estado especificados en la definición de un contenedor prevalecen sobre las comprobaciones de estado de Docker que existan en la imagen del contenedor.

Cuando se define una comprobación de estado en la definición de una tarea, el contenedor ejecuta el proceso de comprobación de estado dentro del contenedor y, a continuación, evalúa el código de salida para determinar el estado de la aplicación.

La comprobación de estado consta de los siguientes parámetros:

- Comando: el comando que ejecuta el contenedor para determinar si está en buen estado. La matriz de cadenas puede comenzar por `CMD` para ejecutar los argumentos del comando directamente, o por `CMD-SHELL` para ejecutar el comando con el shell predeterminado del contenedor. Utilice `CMD-SHELL` cuando necesite funciones de shell como canalizaciones, redireccionamientos, encadenamiento de comandos o expansión de variables de entorno. Por ejemplo, `CMD-SHELL` le permite usar comandos como `curl -f http://localhost/ || exit 1` donde el shell interpreta el operador `||`. Utilice `CMD` para comandos simples que no requieren interpretación del shell.

- Intervalo: el periodo de tiempo (en segundos) entre cada comprobación de estado.
- Tiempo de espera: el periodo de tiempo (en segundos) que hay que esperar para que una comprobación de estado se lleve a cabo correctamente antes de que se considere un error.
- Reintentos: el número de veces que se reintenta una comprobación de estado fallida antes de que se considere que el contenedor está en mal estado.
- Periodo de inicio: el periodo de gracia opcional dentro del cual se puede proporcionar tiempo a los contenedores para el arranque antes de que una comprobación de estado fallida se cuente para el número máximo de reintentos.

  Si una comprobación de estado se realiza correctamente dentro del periodo indicado en `startPeriod`, se considera que el estado del contenedor es correcto y los errores posteriores se contabilizan al calcular el número máximo de intentos.

Para obtener información sobre cómo especificar una comprobación de estado en una definición de tarea, consulte Comprobación de estado.

A continuación se describen los valores de estado posibles para un contenedor:

- `HEALTHY`: la comprobación de estado del contenedor se ha superado correctamente.
- `UNHEALTHY`: error al realizar la comprobación de estado del contenedor.
- `UNKNOWN`: se está evaluando la comprobación del estado del contenedor, no se ha definido ninguna comprobación de estado del contenedor o Amazon ECS no tiene el estado de salud del contenedor.

Los comandos de comprobación de estado se ejecutan en el contenedor. Por lo tanto, debe incluir los comandos en la imagen del contenedor.

La comprobación de estado se conecta a la aplicación a través de la interfaz de bucle invertido del contenedor en `localhost` o `127.0.0.1`. Un código de salida `0` indica una correcta ejecución y un código de salida distinto de cero indica error.

Tenga en cuenta lo siguiente al utilizar las comprobaciones de estado de los contenedores:

- Las comprobaciones de estado del contenedor requieren la versión 1.17.0 o posterior del agente de contenedor de Amazon ECS.
- Se admiten comprobaciones de estado de contenedores para tareas de Fargate si utiliza la versión de la plataforma Linux `1.1.0` o posterior o la versión de la plataforma Windows `1.1.0` o posterior

Cómo determina Amazon ECS el estado de las tareas

Los contenedores que son esenciales y que incluyen el comando de comprobación de estado en la definición de la tarea son los únicos que se tienen en cuenta para determinar el estado de la tarea.

Se evalúan por orden las siguientes reglas:

1. Si el estado de un contenedor esencial es `UNHEALTHY`, entonces el estado de la tarea es `UNHEALTHY`.
2. Si el estado de un contenedor esencial es `UNKNOWN`, entonces el estado de la tarea es `UNKNOWN`.
3. Si el estado de todos los contenedores esenciales es `HEALTHY`, entonces el estado de la tarea es `HEALTHY`.

Observe el siguiente ejemplo de estado de la tarea con 2 contenedores esenciales.

Estado del contenedor 1	Estado del contenedor 2	Estado de la tarea
`UNHEALTHY`	`UNKNOWN`	`UNHEALTHY`
`UNHEALTHY`	`HEALTHY`	`UNHEALTHY`
`HEALTHY`	`UNKNOWN`	`UNKNOWN`
`HEALTHY`	`HEALTHY`	`HEALTHY`

Observe el siguiente ejemplo de estado de la tarea con 3 contenedores.

Estado del contenedor 1	Estado del contenedor 2	Estado del contenedor 3	Estado de la tarea
`UNHEALTHY`	`UNKNOWN`	`UNKNOWN`	`UNHEALTHY`
`UNHEALTHY`	`UNKNOWN`	`HEALTHY`	`UNHEALTHY`
`UNHEALTHY`	`HEALTHY`	`HEALTHY`	`UNHEALTHY`
`HEALTHY`	`UNKNOWN`	`HEALTHY`	`UNKNOWN`

Estado del contenedor 1	Estado del contenedor 2	Estado del contenedor 3	Estado de la tarea
HEALTHY	UNKNOWN	UNKNOWN	UNKNOWN
HEALTHY	HEALTHY	HEALTHY	HEALTHY

Impacto de las desconexiones de agentes en las comprobaciones de estado

Si el agente de contenedores de Amazon ECS se desconecta del servicio Amazon ECS, el contenedor no pasará al estado `UNHEALTHY`. Esto se ha diseñado para garantizar que los contenedores permanezcan en funcionamiento durante el reinicio de los agentes o cuando no estén disponibles temporalmente. El estado de la comprobación de estado es la respuesta de la “última vez que se comunicó” el agente de Amazon ECS, por lo que si se consideró que el estado del contenedor era `HEALTHY` antes de la desconexión, ese estado se mantendrá hasta que el agente se vuelva a conectar y se haga otra comprobación de estado. No se hace ninguna suposición sobre el estado de las comprobaciones de estado de los contenedores.

Visualización del estado de los contenedores de Amazon ECS

Puede ver el estado de los contenedores en la consola y mediante la API en la respuesta de `DescribeTasks`. Para obtener más información, consulte DescribeTasks en la Referencia de la API de Amazon Elastic Container Service.

Si utiliza el registro para el contenedor, por ejemplo, Registros de Amazon CloudWatch, puede configurar el comando de comprobación de estado para reenviar la salida del estado del contenedor a sus registros. Asegúrese de usar `2&1` to capturar tanto la información de `stdout` como de `stderr`. El siguiente ejemplo usa `CMD-SHELL` porque requiere funciones de shell como los operadores de tubería (>>) y OR lógico (`||`):

```
"command": [
     "CMD-SHELL",
     "curl -f http://localhost/ >> /proc/1/fd/1 2>&1  || exit 1"
   ],
```

Supervisión del estado de la instancia de contenedor de Amazon ECS

Amazon ECS proporciona supervisión del estado de las instancias de contenedores. Puede determinar rápidamente si Amazon ECS ha detectado algún problema que pudiera impedir a las instancias de contenedores ejecutar contenedores. Amazon ECS realiza comprobaciones automatizadas en cada instancia de contenedor en ejecución con versión de agente `1.57.0` o posterior para identificar problemas. Para obtener más información sobre cómo verificar la versión del agente y una instancia de contenedor, consulte Actualización del agente de contenedor de Amazon ECS.

Debe utilizar la version de AWS CLI `1.22.3` o posterior o la version de AWS CLI `2.3.6` o posterior. Para obtener más información sobre como actualizar la AWS CLI, consulte Instalación o actualización de la última versión de la AWS CLI en la Guía del usuario de AWS Command Line Interface versión 2.

Para ver el estado de la instancia de contenedor, ejecute `describe-container-instances` con la opción `CONTAINER_INSTANCE_HEALTH`.

Los siguientes son los valores válidos para `overallStatus`:

- `OK`
- `IMPAIRED`
- `INSUFFICIENT_DATA`
- `INITIALIZING`

A continuación, se muestra un ejemplo de cómo ejecutar `describe-container-instances`.

```
aws ecs describe-container-instances \
     --cluster cluster_name \
     --container-instances 47279cd2cadb41cbaef2dcEXAMPLE \
     --include CONTAINER_INSTANCE_HEALTH
```

A continuación, se muestra un ejemplo del objeto de estado de estado de la salida.

```
"healthStatus": {
 "overallStatus": "OK",
```

```
 "details": [{
  "type": "CONTAINER_RUNTIME",
  "status": "OK",
  "lastUpdated": "2021-11-10T03:30:26+00:00",
  "lastStatusChange": "2021-11-10T03:26:41+00:00"
 }]
}
```

Problemas de estado de las instancias de contenedor

Cuando el `overallStatus` sea cualquier estado que no sea `OK`, intente lo siguiente:

- Espere y, a continuación, ejecute `describe-container-instances`.
- Verifique el estado de la instancia de contenedor en la consola de EC2 o mediante el uso de la CLI.
- Revise las métricas de CloudWatch. Para obtener más información, consulte Supervisión de Amazon ECS con CloudWatch
- Consulte el AWS Health Dashboard para ver si hay algún problema con el servicio.

Identifique las oportunidades de optimización de Amazon ECS mediante los datos de seguimiento de la aplicación

Amazon ECS se integra con AWS Distro for OpenTelemetry para recopilar datos de seguimiento de su aplicación. Amazon ECS utiliza un contenedor de sidecar AWS Distro for OpenTelemetry para recopilar y enrutar datos de seguimiento a AWS X-Ray. Para obtener más información, consulte Configuración de colector AWS Distro for OpenTelemetry en Amazon ECS. Luego, puede utilizar AWS X-Ray para identificar errores y excepciones, analizar los cuellos de botella en el rendimiento y los tiempos de respuesta.

Para el colector AWS Distro for OpenTelemetry para enviar datos de seguimiento a AWS X-Ray, la aplicación debe estar configurada para crear los datos de seguimiento. Para obtener más información, consulte Instrumentación de su solicitud para AWS X-Ray en la Guía para desarrolladores de AWS X-Ray.

Permisos de IAM necesarios para la integración AWS Distro for OpenTelemetry con AWS X-Ray

La integración de Amazon ECS con AWS Distro para OpenTelemetry requiere crear un rol de tareas y especificarlo en la definición de tareas. Recomendamos configurar el elemento asociado de AWS Distro para OpenTelemetry para enrutar los registros del contenedor a Registros de CloudWatch.

> **Important**
>
> Si también recopila métricas de aplicaciones mediante la integración de AWS Distro para OpenTelemetry, asegúrese de que el rol de IAM de la tarea también contenga los permisos necesarios para esa integración. Para obtener más información, consulte Correlacionar el rendimiento de las aplicaciones de Amazon ECS mediante métricas de aplicaciones.

Cree la siguiente política y, a continuación, asóciela al rol de ejecución de tareas.

Utilización del editor de política de JSON para la creación de una política

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, elija Políticas.

   Si es la primera vez que elige Políticas, aparecerá la página Welcome to Managed Policies (Bienvenido a políticas administradas). Elija Comenzar.
3. En la parte superior de la página, seleccione Crear política.
4. En la sección Editor de políticas, seleccione la opción JSON.
5. Ingrese el siguiente documento de política JSON:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
```

```
                "logs:DescribeLogStreams",
                "logs:DescribeLogGroups",
                "logs:PutRetentionPolicy",
                "xray:PutTraceSegments",
                "xray:PutTelemetryRecords",
                "xray:GetSamplingRules",
                "xray:GetSamplingTargets",
                "xray:GetSamplingStatisticSummaries",
                "ssm:GetParameters"
            ],
            "Resource": "*"
        }
    ]
}
```

6. Elija Siguiente.

> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de política en la Guía del usuario de IAM.

7. En la página Revisar y crear, introduzca el Nombre de la política y la Descripción (opcional) para la política que está creando. Revise los Permisos definidos en esta política para ver los permisos que concede la política.
8. Elija Crear política para guardar la nueva política.

Especificación del sidecar de AWS Distro for OpenTelemetry para la integración AWS X-Ray de la definición de tarea

La consola de Amazon ECS simplifica la experiencia de crear el contenedor de sidecar de AWS Distro for OpenTelemetry usando la opción Usar la colección de seguimiento. Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

Si no está utilizando la consola de Amazon ECS, puede agregar el contenedor de sidecar de AWS Distro for OpenTelemetry según su definición de tarea. El siguiente fragmento de definición de tarea

muestra la definición de contenedor para agregar el sidecar de AWS Distro for OpenTelemetry para la integración de AWS X-Ray.

```
{
 "family": "otel-using-xray",
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryXrayRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [{
   "name": "aws-otel-emitter",
   "image": "application-image",
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "true",
     "awslogs-group": "/ecs/aws-otel-emitter",
     "awslogs-region": "us-east-1",
     "awslogs-stream-prefix": "ecs"
    }
   },
   "dependsOn": [{
    "containerName": "aws-otel-collector",
    "condition": "START"
   }]
  },
  {
   "name": "aws-otel-collector",
   "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
   "essential": true,
   "command": [
    "--config=/etc/ecs/otel-instance-metrics-config.yaml"
   ],
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "True",
     "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
     "awslogs-region": "us-east-1",
     "awslogs-stream-prefix": "ecs"
    }
   }
  }
 ],
 "networkMode": "awsvpc",
```

```
  "requiresCompatibilities": [
   "FARGATE"
  ],
  "cpu": "1024",
  "memory": "3072"
}
```

Correlacionar el rendimiento de las aplicaciones de Amazon ECS mediante métricas de aplicaciones

Amazon ECS en Fargate admite la recopilación de métricas de las aplicaciones que se ejecutan en Fargate y la exportación a Amazon CloudWatch o Amazon Managed Service for Prometheus.

Puede usar los metadatos recopilados para correlacionar los datos de rendimiento de las aplicaciones con los datos de la infraestructura subyacente, lo que reduce el tiempo medio necesario para resolver el problema.

Amazon ECS utiliza un contenedor de sidecar de AWS Distro for OpenTelemetry para recopilar y enrutar las métricas de su aplicación al destino. La experiencia de consola de Amazon ECS simplifica el proceso de agregar esta integración al crear las definiciones de tareas.

Temas

- Exportación de métricas de aplicaciones a Amazon CloudWatch
- Exportación de métricas de aplicaciones a Amazon Managed Service for Prometheus

Exportación de métricas de aplicaciones a Amazon CloudWatch

Amazon ECS en Fargate admite la exportación de métricas de aplicaciones personalizadas a Amazon CloudWatch como métricas personalizadas. Esto se hace agregando el contenedor de sidecar de AWS Distro for OpenTelemetry según su definición de tarea. La consola de Amazon ECS simplifica este proceso al agregar la opción Utilizar recopilación de métricas cuando se crea una nueva definición de tarea. Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

Las métricas de la aplicación se exportan a CloudWatch Logs con nombre de grupo de registros `/aws/ecs/application/metrics` y las métricas de se pueden ver en el espacio de nombres `ECS/AWSOTel/Application`. La aplicación debe estar instrumentada con el SDK de OpenTelemetry.

Para obtener más información, consulte Introducción a AWS Distro for OpenTelemetry en la documentación de AWS Distro for OpenTelemetry.

Consideraciones

Debe tenerse en cuenta lo siguiente al utilizar Amazon ECS en la integración de Fargate con AWS Distro for OpenTelemetry para enviar métricas de aplicaciones a Amazon CloudWatch

- Esta integración solo envía las métricas personalizadas de la aplicación a CloudWatch. Si desea obtener métricas de tarea, puede activar Información de contenedores en la configuración del clúster de Amazon ECS. Para obtener más información, consulte Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada.
- La integración de AWS Distro para OpenTelemetry se admite para cargas de trabajo de Amazon ECS alojadas en Fargate y cargas de trabajo de Amazon ECS alojadas en instancias de Amazon EC2. No se admiten instancias externas.
- CloudWatch admite un máximo de 30 dimensiones por métrica. De forma predeterminada, Amazon ECS incluye de forma predeterminada las dimensiones `TaskARN`, `ClusterARN`, `LaunchType`, `TaskDefinitionFamily` y `TaskDefinitionRevision` de las métricas. La aplicación puede definir las 25 dimensiones restantes. Si se configuran más de 30 dimensiones, CloudWatch no puede mostrarlas. Cuando esto ocurra, las métricas de la aplicación se mostrarán en el espacio de nombres métrico de CloudWatch `ECS/AWSOTel/Application` pero sin dimensiones. Puede instrumentar su aplicación para agregar dimensiones adicionales. Para obtener más información, consulte Using CloudWatch metrics with AWS Distro for OpenTelemetry (Uso de métricas CLoudWatch con Distro for OpenTelemetry) en la documentación de AWS Distro for OpenTelemetry.

Permisos de IAM necesarios para la integración de AWS Distro for OpenTelemetry Amazon CloudWatch

Integración de Amazon ECS con AWS Distro for OpenTelemetry requiere crear un rol de IAM para tareas y especificar el rol en la definición de tareas. Le recomendamos que configure también el sidecar AWS Distro for OpenTelemetry para enrutar los registros de contenedores a CloudWatch Logs, lo que requiere crear y especificar un rol de IAM de ejecución de tareas también en su definición de tarea. La consola de Amazon ECS se encarga del rol de IAM de ejecución de tareas en su nombre, pero el rol de IAM de tarea debe crearse manualmente y agregarse a la definición de tarea. Para obtener más información sobre el rol de IAM para la ejecución de tareas, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

> **Important**
>
> Si también recopila datos de seguimiento de aplicaciones mediante la integración AWS Distro para OpenTelemetry, asegúrese de que el rol de IAM de tarea también contenga los permisos necesarios para esa integración. Para obtener más información, consulte Identifique las oportunidades de optimización de Amazon ECS mediante los datos de seguimiento de la aplicación.
> Si la aplicación requiere permisos adicionales, debe agregarlos a esta política. Cada definición de tarea solo puede especificar un rol de IAM de tarea. Por ejemplo, si utiliza un archivo de configuración personalizado almacenado en Systems Manager, debe agregar el permiso `ssm:GetParameters` para esta política de IAM.

Creación de un rol de servicio de Elastic Container Service (consola de IAM)

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola de IAM, seleccione Roles y, a continuación, elija Crear rol.
3. En Tipo de entidad de confianza, elija Servicio de AWS.
4. En Servicio o caso de uso, seleccione Elastic Container Service y, a continuación, seleccione el caso de uso Tarea de Elastic Container Service.
5. Elija Siguiente.
6. En la sección Agregar permisos, busque AWSDistroOpenTelemetryPolicyForXray y, a continuación, seleccione la política.
7. (Opcional) Configure un límite de permisos. Se trata de una característica avanzada que está disponible para los roles de servicio, pero no para los roles vinculados a servicios.

   a. Abra la sección Configurar límite de permisos y, a continuación, elija Utilizar un límite de permisos para controlar los permisos que puedes tener el rol como máximo.

      IAM incluye una lista de las políticas administradas por AWS y de las políticas administradas por el cliente de cada cuenta.

   b. Seleccione la política que desea utilizar para el límite de permisos.
8. Elija Siguiente.
9. Escriba un nombre o sufijo de nombre para el rol, que pueda ayudarle a identificar su finalidad.

> **Important**
>
> Cuando asigne un nombre a un rol, tenga en cuenta lo siguiente:
>
> - Los nombres de rol deben ser únicos dentro de su Cuenta de AWS, y no se puedesn hacer únicos mediante mayúsculas y minúsculas.
>
> Por ejemplo, no puedes crear roles denominados tanto **`PRODROLE`** como **`prodrole`**. Cuando se utiliza un nombre de rol en una política o como parte de un ARN, el nombre de rol distingue entre mayúsculas y minúsculas, sin embargo, cuando un nombre de rol les aparece a los clientes en la consola, como por ejemplo durante el proceso de inicio de sesión, el nombre de rol no distingue entre mayúsculas y minúsculas.
> - Dado que otras entidades podrían hacer referencia al rol, no es posible editar el nombre del rol una vez creado.

10. (Opcional) En Descripción, ingrese una descripción para el rol.
11. (Opcional) Para editar los casos de uso y los permisos de la función, en las secciones Paso 1: Seleccionar entidades confiables o en Paso 2: Agregar permisos, elija Editar.
12. (Opcional) Para ayudar a identificar, organizar o buscar el rol, agregue etiquetas como pares clave-valor. Para obtener más información sobre el uso de etiquetas en IAM, consulte Etiquetas para recursos de AWS Identity and Access Management en la Guía del usuario de IAM.
13. Revise el rol y, a continuación, elija Crear rol.

Especificación del sidecar de AWS Distro for OpenTelemetry en la definición de tarea

La consola de Amazon ECS simplifica la experiencia de crear el contenedor del sidecar AWS Distro for OpenTelemetry mediante la opción Utilizar recopilación de métricas. Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

Si no está utilizando la consola de Amazon ECS, puede agregar el contenedor de sidecar de AWS Distro for OpenTelemetry a la definición de tarea manualmente. En el siguiente ejemplo de definición de tareas se muestra la definición de contenedor para agregar el sidecar de AWS Distro for OpenTelemetry para la integración de Amazon CloudWatch.

```
{
 "family": "otel-using-cloudwatch",
```

```
"taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryCloudWatchRole",
"executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
"containerDefinitions": [
   {
  "name": "aws-otel-emitter",
  "image": "application-image",
  "logConfiguration": {
   "logDriver": "awslogs",
   "options": {
    "awslogs-create-group": "true",
    "awslogs-group": "/ecs/aws-otel-emitter",
    "awslogs-region": "us-east-1",
    "awslogs-stream-prefix": "ecs"
   }
  },
  "dependsOn": [{
   "containerName": "aws-otel-collector",
   "condition": "START"
  }]
 },
 {
  "name": "aws-otel-collector",
  "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
  "essential": true,
  "command": [
   "--config=/etc/ecs/ecs-cloudwatch.yaml"
  ],
  "logConfiguration": {
   "logDriver": "awslogs",
   "options": {
    "awslogs-create-group": "True",
    "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
    "awslogs-region": "us-east-1",
    "awslogs-stream-prefix": "ecs"
   }
  }
 }
],
"networkMode": "awsvpc",
"requiresCompatibilities": [
 "FARGATE"
],
"cpu": "1024",
"memory": "3072"
```

```
}
```

Exportación de métricas de aplicaciones a Amazon Managed Service for Prometheus

Amazon ECS admite la exportación de las métricas de CPU, memoria, red y almacenamiento de nivel de tarea y las métricas de aplicaciones personalizadas a Amazon Managed Service for Prometheus. Esto se hace agregando el contenedor de sidecar de AWS Distro for OpenTelemetry según su definición de tarea. La consola de Amazon ECS simplifica este proceso al agregar la opción Utilizar recopilación de métricas cuando se crea una nueva definición de tarea. Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

Las métricas se exportan a Amazon Managed Service for Prometheus y se pueden ver mediante el panel de control de Amazon Managed Grafana. Su aplicación debe estar instrumentada con bibliotecas Prometheus o con el SDK de OpenTelemetry. Para obtener más información sobre cómo instrumentar la aplicación con el SDK de OpenTelemetry, consulte Introducción a AWS Distro for OpenTelemetry en la documentación de AWS Distro for OpenTelemetry.

Al utilizar las bibliotecas de Prometheus, la aplicación debe exponer un punto de conexión `/metrics` que se utiliza para extraer los datos de métricas. Para obtener más información sobre cómo instrumentar la aplicación con las bibliotecas Prometheus, consulte Bibliotecas cliente de Prometheus en la documentación de Prometheus.

Consideraciones

Debe tenerse en cuenta lo siguiente cuando utiliza Amazon ECS en la integración de Fargate con AWS Distro para OpenTelemetry a fin de enviar métricas de aplicaciones a Amazon Managed Service para Prometheus.

- La integración de AWS Distro para OpenTelemetry se admite para cargas de trabajo de Amazon ECS alojadas en Fargate y cargas de trabajo de Amazon ECS alojadas en instancias de Amazon EC2. No se admiten instancias externas en este momento.
- Por defecto, AWS Distro for OpenTelemetry incluye todas las dimensiones de nivel de tareas disponibles para las métricas de la aplicación al exportar a Amazon Managed Service for Prometheus. También puede instrumentar su aplicación para agregar dimensiones adicionales. Para obtener más información, consulte Introducción a Prometheus Remote Write Exporter para Amazon Managed Service for Prometheus en la documentación de AWS Distro for OpenTelemetry.

Permisos de IAM necesarios para la integración de AWS Distro for OpenTelemetry con Amazon Managed Service for Prometheus

La integración de Amazon ECS con Amazon Managed Service for Prometheus mediante el sidecar de AWS Distro for openTelemetry requiere crear un rol de IAM para tareas y especificar el rol en la definición de tareas. Este rol de IAM de tarea debe crearse manualmente siguiendo los pasos que se indican a continuación antes de registrar la definición de tarea.

Le recomendamos que configure también el sidecar AWS Distro for OpenTelemetry para enrutar los registros de contenedores a CloudWatch Logs, lo que requiere crear y especificar un rol de IAM de ejecución de tareas también en su definición de tarea. La consola de Amazon ECS se encarga del rol de IAM de ejecución de tareas en su nombre, pero el rol de IAM de tarea debe crearse manualmente. Para obtener más información sobre la creación de un rol de IAM para la ejecución de tareas, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

> **Important**
>
> Si también recopila datos de seguimiento de aplicaciones mediante la integración AWS Distro para OpenTelemetry, asegúrese de que el rol de IAM de tarea también contenga los permisos necesarios para esa integración. Para obtener más información, consulte Identifique las oportunidades de optimización de Amazon ECS mediante los datos de seguimiento de la aplicación.

Creación de un rol de servicio de Elastic Container Service (consola de IAM)

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola de IAM, seleccione Roles y, a continuación, elija Crear rol.
3. En Tipo de entidad de confianza, elija Servicio de AWS.
4. En Servicio o caso de uso, seleccione Elastic Container Service y, a continuación, seleccione el caso de uso Tarea de Elastic Container Service.
5. Elija Siguiente.
6. En la sección Agregar permisos, busque AmazonPrometheusRemoteWriteAccess y, a continuación, seleccione la política.

7. (Opcional) Configure un límite de permisos. Se trata de una característica avanzada que está disponible para los roles de servicio, pero no para los roles vinculados a servicios.

   a. Abra la sección Configurar límite de permisos y, a continuación, elija Utilizar un límite de permisos para controlar los permisos que puedes tener el rol como máximo.

      IAM incluye una lista de las políticas administradas por AWS y de las políticas administradas por el cliente de cada cuenta.

   b. Seleccione la política que desea utilizar para el límite de permisos.

8. Elija Siguiente.

9. Escriba un nombre o sufijo de nombre para el rol, que pueda ayudarle a identificar su finalidad.

   > **Important**
   >
   > Cuando asigne un nombre a un rol, tenga en cuenta lo siguiente:
   >
   > - Los nombres de rol deben ser únicos dentro de su Cuenta de AWS, y no se puedesn hacer únicos mediante mayúsculas y minúsculas.
   >
   > Por ejemplo, no puedes crear roles denominados tanto **`PRODROLE`** como **`prodrole`**. Cuando se utiliza un nombre de rol en una política o como parte de un ARN, el nombre de rol distingue entre mayúsculas y minúsculas, sin embargo, cuando un nombre de rol les aparece a los clientes en la consola, como por ejemplo durante el proceso de inicio de sesión, el nombre de rol no distingue entre mayúsculas y minúsculas.
   >
   > - Dado que otras entidades podrían hacer referencia al rol, no es posible editar el nombre del rol una vez creado.

10. (Opcional) En Descripción, ingrese una descripción para el rol.

11. (Opcional) Para editar los casos de uso y los permisos de la función, en las secciones Paso 1: Seleccionar entidades confiables o en Paso 2: Agregar permisos, elija Editar.

12. (Opcional) Para ayudar a identificar, organizar o buscar el rol, agregue etiquetas como pares clave-valor. Para obtener más información sobre el uso de etiquetas en IAM, consulte Etiquetas para recursos de AWS Identity and Access Management en la Guía del usuario de IAM.

13. Revise el rol y, a continuación, elija Crear rol.

Especificación del sidecar de AWS Distro for OpenTelemetry en la definición de tarea

La consola de Amazon ECS simplifica la experiencia de crear el contenedor del sidecar AWS Distro for OpenTelemetry mediante la opción Utilizar recopilación de métricas. Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.

Si no está utilizando la consola de Amazon ECS, puede agregar el contenedor de sidecar de AWS Distro for OpenTelemetry a la definición de tarea manualmente. En el siguiente ejemplo de definición de tareas se muestra la definición de contenedor para agregar la integración de sidecar de AWS Distro for OpenTelemetry for Amazon Managed Service for Prometheus.

```
{
 "family": "otel-using-cloudwatch",
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryCloudWatchRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [{
   "name": "aws-otel-emitter",
   "image": "application-image",
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "true",
     "awslogs-group": "/ecs/aws-otel-emitter",
     "awslogs-region": "aws-region",
     "awslogs-stream-prefix": "ecs"
    }
   },
   "dependsOn": [{
    "containerName": "aws-otel-collector",
    "condition": "START"
   }]
  },
  {
   "name": "aws-otel-collector",
   "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
   "essential": true,
   "command": [
    "--config=/etc/ecs/ecs-amp.yaml"
   ],
   "environment": [{
    "name": "AWS_PROMETHEUS_ENDPOINT",
    "value": "https://aps-workspaces.aws-region.amazonaws.com/workspaces/
ws-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/api/v1/remote_write"
```

```
    }],
    "logConfiguration": {
     "logDriver": "awslogs",
     "options": {
      "awslogs-create-group": "True",
      "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
      "awslogs-region": "aws-region",
      "awslogs-stream-prefix": "ecs"
     }
    }
   }
  ],
  "networkMode": "awsvpc",
  "requiresCompatibilities": [
   "FARGATE"
  ],
  "cpu": "1024",
  "memory": "3072"
}
```

Registro de llamadas a la API de Amazon ECS mediante AWS CloudTrail

Amazon Elastic Container Service se integra con AWS CloudTrail, un servicio que proporciona un registro de las medidas adoptadas por un usuario, un rol o un Servicio de AWS. CloudTrail captura todas las llamadas a la API para Amazon ECS como eventos. Las llamadas capturadas incluyen las llamadas desde la consola de Amazon ECS y las llamadas de código hacia las operaciones de la API de Amazon ECS. Mediante la información recopilada por CloudTrail, puede determinar la solicitud que se realizó a Amazon ECS, la dirección IP desde la que se realizó, 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 activado en la Cuenta de AWS cuando usted crea la cuenta y tiene acceso automático al Historial de eventos de CloudTrail. El Historial de eventos de CloudTrail proporciona un registro visible e inmutable, que se puede buscar y descargar, de los últimos 90 días de eventos de gestión registrados en una Región de AWS. Para obtener más información, consulte Trabajar con el historial de eventos de CloudTrail en la Guía del usuario de AWS CloudTrail. No se cobran cargos de CloudTrail por ver el Historial de eventos.

Para mantener un registro permanente de los eventos en su Cuenta de AWS más allá de los 90 días, cree un registro de seguimiento o un almacén de datos de eventos de CloudTrail Lake.

Registros de seguimiento de CloudTrail

Un registro de seguimiento permite a CloudTrail enviar archivos de registro a un bucket de Amazon S3. Todos los registros de seguimiento que cree con la AWS Management Console son multirregionales. Puede crear un registro de seguimiento de una sola región o multirregionales mediante la AWS CLI. Se recomienda crear un registro de seguimiento multirregional, ya que registra actividad en todas las Regiones de AWS de su 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 crear un registro de seguimiento para enviar una copia de los eventos de administración en curso en su bucket de Amazon S3 sin costo alguno desde CloudTrail; sin embargo, hay cargos por almacenamiento en Amazon S3. Para obtener más información sobre los precios de CloudTrail, consulte Precios de AWS CloudTrail. Para obtener información acerca de los precios de Amazon S3, consulte Precios de Amazon S3.

Almacenes de datos de eventos de CloudTrail Lake

CloudTrail Lake le permite ejecutar consultas basadas en SQL sobre los eventos. CloudTrail Lake convierte los eventos existentes en formato JSON basado en filas al formato ORC de Apache. ORC es un formato de almacenamiento en columnas optimizado para una recuperación rápida de datos. Los eventos se agregan en almacenes de datos de eventos, que son recopilaciones inmutables de eventos en función de criterios que se seleccionan aplicando selectores de eventos avanzados. Los selectores que se aplican a un almacén de datos de eventos controlan los eventos que perduran y están disponibles para la consulta. Para obtener más información acerca de CloudTrail Lake, consulte Trabajar con AWS CloudTrail Lake en la Guía del usuario de AWS CloudTrail.

Los almacenes de datos de eventos de CloudTrail Lake y las consultas generan costos adicionales. Cuando crea un almacén de datos de eventos, debe elegir la opción de precios que desee utilizar para él. La opción de precios determina el costo de la incorporación y el almacenamiento de los eventos, así como el período de retención predeterminado y máximo del almacén de datos de eventos. Para obtener más información sobre los precios de CloudTrail, consulte Precios de AWS CloudTrail.

Eventos de administración de Amazon ECS en CloudTrail

Los eventos de administración proporcionan información sobre las operaciones de administración que se realizan en los recursos de su Cuenta de AWS. Se denominan también operaciones del plano de control. CloudTrail registra los eventos de administración de forma predeterminada.

Amazon Elastic Container Service registra todas las operaciones del plano de control de Amazon ECS como eventos de administración. Por ejemplo, las llamadas a las secciones `CreateService`, `RunTask` y `DeleteCluster` generan entradas en los archivos de registro de CloudTrail. Para obtener una lista de las operaciones del plano de control de Amazon Elastic Container Service que Amazon ECS registra en CloudTrail, consulte la Referencia de la API de Amazon Elastic Container Service.

Ejemplos de eventos de Amazon ECS

Un evento representa una única solicitud de cualquier origen e incluye información sobre la operación de la API solicitada, la fecha y la hora de la operación o los parámetros de la solicitud, entre otras cosas. Los archivos de registro de CloudTrail no rastrean el orden en la pila de las llamadas a la API públicas, por lo que los eventos no aparecen en un orden específico.

En el siguiente ejemplo, se muestra un evento de CloudTrail que ilustra la acción `CreateService`.

> **Note**
>
> Este ejemplo se ha manipulado para mejorar la legibilidad. En un archivo de registro de CloudTrail, todas las entradas y eventos aparecen en la misma línea. Además, este ejemplo se ha limitado a una única entrada de Amazon ECS. En un archivo de registro de CloudTrail real, ve las entradas y los eventos de varios servicios de AWS.

```
{
```

```
    "eventVersion": "1.04",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:account_name",
        "arn": "arn:aws:sts::123456789012:user/Mary_Major",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-06-20T18:32:25Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Mary_Major"
            }
        }
    },
    "eventTime": "2018-06-20T19:04:36Z",
    "eventSource": "ecs.amazonaws.com",
    "eventName": "CreateCluster",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.12",
    "userAgent": "console.amazonaws.com",
    "requestParameters": {
        "clusterName": "default"
    },
    "responseElements": {
        "cluster": {
            "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/default",
            "pendingTasksCount": 0,
            "registeredContainerInstancesCount": 0,
            "status": "ACTIVE",
            "runningTasksCount": 0,
            "statistics": [],
            "clusterName": "default",
            "activeServicesCount": 0
        }
    },
    "requestID": "cb8c167e-EXAMPLE",
    "eventID": "e3c6f4ce-EXAMPLE",
```

```
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```

Para obtener información sobre el contenido de los registros de CloudTrail, consulte Contenido de los registros de CloudTrail en la Guía del usuario de AWS CloudTrail.

Visualización de Live Tail de Registros de CloudWatch para los servicios y tareas de Amazon ECS

Live Tail de Registros de CloudWatch ayuda a solucionar rápidamente los incidentes al ver una lista en streaming de los nuevos eventos de registro a medida que se incorporan. Puede ver las tareas y los eventos del servicio en la consola de Amazon ECS. Esta vista proporciona una visión coherente del estado de las tareas y servicios.

Cada tarea en un servicio de Amazon ECS tiene flujos de registro dedicados para cada contenedor. Si un servicio se escala verticalmente, la instancia de cada tarea tiene su propio conjunto de flujos de registro. La convención de nomenclatura para los flujos de registro sigue el patrón:

```
log-group-prefix/container-name/task-id
```

Una sola tarea escribe en los mismos flujos de registro durante su vida útil. El flujo de registro contiene mensajes de los contenedores de esa tarea y también cualquier salida del código de la aplicación. Todos los mensajes tienen marca de tiempo, incluidos los registros personalizados.

> **Note**
>
> Las sesiones de Live Tail generan costos según el tiempo de uso de la sesión por minuto. Para obtener más información sobre precios, consulte Precios de Amazon CloudWatch.

Permisos necesarios

Son necesarios los siguientes permisos para el usuario de IAM de la consola para iniciar y detener las sesiones de Live Tail de Registros de CloudWatch:

- `logs:DescribeLogGroups`

- `logs:StartLiveTail`
- `logs:StopLiveTail`

Procedimiento

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. Determinación del recurso que quiere ver

Recurso	Pasos
Tareas	a. En la página Clusters (Clústeres), elija el clúster. Aparecerá la página de detalles del clúster. b. Elija la pestaña Tareas. Se muestra la página de detalles de la tarea. c. Elija la pestaña Logs (Registros).
Servicios	a. En la página Clusters (Clústeres), elija el clúster. Aparecerá la página de detalles del clúster. b. Elija el servicio. Aparecerá la página de detalles del servicio. c. Elija la pestaña Logs (Registros).

3. Elija Live Tail de Registros de CloudWatch y, luego, Iniciar.
4. (Opcional) Para filtrar los flujos, en Filtrar, en Seleccionar grupos de registro, elija el grupo de registro.

Referencias de AWS CLI

También puede utilizar la AWS CLI para iniciar las sesiones de Live Tail para Registros de CloudWatch. Los siguientes recursos proporcionan información detallada sobre el uso de AWS CLI con Live Tail:

- Referencia del comando start-live-tail: sintaxis completa del comando, parámetros y ejemplos del comando `aws logs start-live-tail`.
- Guía del usuario de Live Tail de Registros de CloudWatch: guía integral que incluye el uso de AWS CLI con los modos interactivo y de solo impresión.
- Ejemplos del SDK de StartLiveTail: ejemplos programáticos del uso de la API StartLiveTail con varios SDK de AWS.

Supervisión de las cargas de trabajo mediante metadatos de Amazon ECS

Puede utilizar los metadatos de las tareas y los contenedores para solucionar problemas de las cargas de trabajo y hacer cambios de configuración en función del tiempo de ejecución.

Entre los metadatos se incluyen las categorías siguientes:

- Atributos de tarea que proporcionan información sobre dónde se está ejecutando la tarea.
- Atributos a nivel de contenedor que proporcionan el ID de Docker, el nombre y los detalles de la imagen.

  Esto proporciona visibilidad del contenedor.
- Configuración de red, como direcciones IP, subredes y modo de red.

  Esto ayuda a configurar la red y a solucionar problemas.
- Estado y condición de la tarea

  Esto le permite saber si las tareas se están ejecutando.

Puede ver metadatos con cualquiera de los siguientes métodos:

- Archivo de metadatos de contenedores

A partir de la versión 1.15.0 del agente de contenedor de Amazon ECS, hay varios metadatos disponibles en los contenedores o en la instancia de contenedor del host. Cuando se activa esta característica, puede consultar la información acerca de una tarea, un contenedor y una instancia de contenedor desde dentro del contenedor o de la instancia de contenedor del host. El archivo de metadatos se crea en la instancia de host y se monta en el contenedor como un volumen de Docker y, por lo tanto, no está disponible cuando una tarea está alojada en AWSFargate.

- Punto de enlace de metadatos de tareas

  El agente de contenedor de Amazon ECS introduce una variable de entorno en cada contenedor, denominada punto de enlace de metadatos de tareas, que proporciona varios metadatos y estadísticas de Docker de tareas al contenedor.

- Introspección de contenedor

  El agente de contenedor de Amazon ECS proporciona una operación de la API para recopilar detalles acerca de la instancia de contenedor en la que se ejecuta el agente y las tareas asociadas que se ejecutan en esa instancia.

Variables de entorno de Amazon ECS

Las variables de entorno controlan el comportamiento de una tarea, como la capacidad utilizada para ejecutarla. Amazon ECS estable las siguientes variables de entorno para sus tareas:

- `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI`: el punto de conexión de la URL HTTP completo para que el SDK lo utilice al realizar una solicitud de credenciales. Esto incluye tanto el esquema como el host. Para obtener más información, consulte Proveedor de credenciales del contenedor en la Guía de referencia de los SDK de AWS.
- `ECS_CONTAINER_METADATA_URI_V4`: la dirección de los metadatos de la tarea, versión 4. Para obtener más información, consulte Versión 4 del punto de conexión de metadatos de tareas de Amazon ECS.
- `ECS_CONTAINER_METADATA_URI`: la dirección de los metadatos de la tarea, versión 3. Para obtener más información, consulte Versión 3 del punto de conexión de los metadatos de tareas de Amazon ECS.
- `ECS_AGENT_URI`: la dirección base para los diferentes puntos de conexión compatibles con Fargate. Para obtener más información, consulte:
  - Punto de conexión de protección de reducción horizontal de tareas de Amazon ECS

  - Puntos de conexión de inyección de errores de Amazon ECS
- `AWS_EXECUTION_ENV`: la información sobre el tipo de lanzamiento en el que se ejecuta la tarea.
  - En el caso de Fargate, Amazon ECS lo establece en `AWS_ECS_FARGATE`.
  - En el caso de EC2, Amazon ECS lo establece en `AWS_ECS_EC2`.
- `AWS_DEFAULT_REGION`: la Región de AWS predeterminada para su Cuenta de AWS. Se trata de la región predeterminada donde se ejecuta su tarea.
- `AWS_REGION`: la región en la que se ejecuta la tarea. Si se define, este valor anula la `AWS_DEFAULT_REGION`.

Puede ver las variables de entorno en los metadatos de la tarea. Para obtener más información, consulte uno de los siguientes temas:

- Fargate: Metadatos de tareas de Amazon ECS disponibles para tareas en Fargate
- EC2: Metadatos de tareas disponibles para tareas de Amazon ECS en EC2

Archivo de metadatos de contenedores de Amazon ECS

A partir de la versión 1.15.0 del agente de contenedor de Amazon ECS, hay varios metadatos disponibles en los contenedores o en la instancia de contenedor del host. Cuando se activa esta característica, puede consultar la información acerca de una tarea, un contenedor y una instancia de contenedor desde dentro del contenedor o de la instancia de contenedor del host. El archivo de metadatos se crea en la instancia de host y se monta en el contenedor como un volumen de Docker y, por lo tanto, no está disponible cuando una tarea está alojada en AWSFargate.

El archivo de metadatos del contenedor se limpia en la instancia de host al mismo tiempo que esta. Puede ajustar el momento en que esto sucede con la variable del agente de contenedor `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`. Para obtener más información, consulte Limpieza automática de tareas e imágenes de Amazon ECS.

Temas

- Ubicaciones de los archivos de metadatos de los contenedores
- Activación de metadatos de contenedor de Amazon ECS
- Formato de archivo de metadatos de contenedor de Amazon ECS

Ubicaciones de los archivos de metadatos de los contenedores

De forma predeterminada, el archivo de metadatos del contenedor se escribe en las rutas de host y contenedor siguientes.

- En instancias de Linux:
  - Ruta del host: `/var/lib/ecs/data/metadata/cluster_name/task_id/container_name/ecs-container-metadata.json`

    > **Note**
    >
    > La ruta del host en Linux presupone que se utiliza la ruta de montaje del directorio de datos predeterminada (`/var/lib/ecs/data`) al iniciar el agente. Si no va a utilizar la AMI optimizada para Amazon ECS (o el paquete `ecs-init` para iniciar y mantener el agente de contenedor), asegúrese de establecer la variable de configuración del agente `ECS_HOST_DATA_DIR` en la ruta del host donde se encuentra el archivo de estado del agente de contenedor. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS.

  - Ruta del contenedor: `/opt/ecs/metadata/random_ID/ecs-container-metadata.json`
- En instancias de Windows:
  - Ruta del host: `C:\ProgramData\Amazon\ECS\data\metadata\task_id\container_name\ecs-container-metadata.json`
  - Ruta del contenedor: `C:\ProgramData\Amazon\ECS\metadata\random_ID\ecs-container-metadata.json`

Sin embargo, para facilitar el acceso, la ubicación del archivo de metadatos del contenedor se establece en la variable de entorno `ECS_CONTAINER_METADATA_FILE` en el interior del contenedor. Puede leer el contenido del archivo desde dentro del contenedor con el siguiente comando:

- En instancias de Linux:

```
cat $ECS_CONTAINER_METADATA_FILE
```

- En instancias de Windows (PowerShell):

```
Get-Content -path $env:ECS_CONTAINER_METADATA_FILE
```

Activación de metadatos de contenedor de Amazon ECS

Puede activar los metadatos de contenedor en el nivel de la instancia de contenedor ajustando la variable del agente de contenedor de `ECS_ENABLE_CONTAINER_METADATA` en `true`. Puede establecer esta variable en el archivo de configuración `/etc/ecs/ecs.config` y reiniciar el agente. También puede establecerla como variable de entorno de Docker en tiempo de ejecución, cuando se inicia el agente de contenedor. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS.

Si `ECS_ENABLE_CONTAINER_METADATA` está establecido en `true` en el momento en el que se inicia el agente, se crean archivos de metadatos para todos los contenedores creados a partir de ese momento. El agente de contenedor de Amazon ECS no puede crear archivos de metadatos para los contenedores creados antes de establecer la variable del agente de contenedor `ECS_ENABLE_CONTAINER_METADATA` en `true`. Para asegurarse de que todos los contenedores reciban archivos de metadatos, debería establecer esta variable del agente al lanzar una instancia de contenedor. El siguiente ejemplo es un script de datos de usuario que establecerá esta variable y registrará la instancia de contenedor en el clúster.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=your_cluster_name
ECS_ENABLE_CONTAINER_METADATA=true
EOF
```

Formato de archivo de metadatos de contenedor de Amazon ECS

La información siguiente se almacena en el archivo JSON de metadatos del contenedor.

`Cluster`

El nombre del clúster que está ejecutando la tarea del contenedor.

`ContainerInstanceARN`

Nombre de recurso de Amazon (ARN) de la instancia de contenedor del host.

`TaskARN`

Nombre de recurso de Amazon (ARN) de la tarea a la que pertenece el contenedor.

`TaskDefinitionFamily`

Nombre de la familia de definiciones de tareas que se está utilizando en el contenedor.

`TaskDefinitionRevision`

Revisión de definición de tarea que está utilizando el contenedor.

`ContainerID`

ID del contenedor de Docker (no el ID del contenedor de Amazon ECS).

`ContainerName`

Nombre del contenedor obtenido de la definición de tareas de Amazon ECS del contenedor.

`DockerContainerName`

Nombre de contenedor que el daemon de Docker utiliza para el contenedor (por ejemplo, el nombre que aparece en el salida del comando docker ps).

`ImageID`

Resumen del volumen de dispositivo raíz (SHA) de la imagen de Docker que se usa para iniciar el contenedor.

`ImageName`

Nombre y etiqueta de la imagen de Docker que se usa para iniciar el contenedor.

`PortMappings`

Mapeos de puertos asociadas con el contenedor.

`ContainerPort`

Puerto del contenedor que se expone.

`HostPort`

Puerto de la instancia de contenedor del host que se expone.

`BindIp`

Dirección IP de vinculación que Docker asigna al contenedor. Esta dirección IP solo se aplica con el modo de red `bridge` y únicamente está accesible desde la instancia de contenedor.

`Protocol`

Protocolo de red utilizado para el mapeo de puertos.

`Networks`

Modo de red y dirección IP del contenedor.

`NetworkMode`

Modo de red de la tarea a la que pertenece el contenedor.

`IPv4Addresses`

Direcciones IP asociadas con el contenedor.

> **Important**
>
> Si la tarea está utilizando el modo de red `awsvpc`, no se devolverá la dirección IP del contenedor. En este caso, puede recuperar la dirección IP leyendo el archivo /etc/hosts con el siguiente comando:
>
> ```
> tail -1 /etc/hosts | awk '{print $1}'
> ```

`MetadataFileStatus`

Estado del archivo de metadatos. Cuando el estado es `READY`, el archivo de metadatos está actualizado y completo. Si el archivo todavía no está listo (por ejemplo, en el momento de iniciar la tarea), está disponible una versión truncada del formato de archivo. Para evitar la probable condición de carrera en la cual el contenedor se ha iniciado pero todavía no se han escrito los metadatos, puede analizar el archivo de metadatos y esperar a que este parámetro se establezca en `READY` antes de comenzar a depender de los metadatos. Esto suele estar disponible en menos de 1 segundo a partir del momento en que se inicia el contenedor.

`AvailabilityZone`

Zona de disponibilidad en la que reside la instancia de contenedor del host.

`HostPrivateIPv4Address`

La dirección IP privada de la tarea a la que pertenece el contenedor.

`HostPublicIPv4Address`

La dirección IP pública de la tarea a la que pertenece el contenedor.

Example Archivo de metadatos de los contenedores de Amazon ECS (**READY**)

En el ejemplo siguiente se muestra un archivo de metadatos del contenedor con el estado `READY`.

```
{
    "Cluster":"arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
    "TaskARN":"arn:aws:ecs:us-east-1:123456789012:task/MyCluster/
b593651c4d6b44a6b2b583f45c957e15",
    "Family":"curltest-container",
    "Revision":"2",
    "DesiredStatus":"RUNNING",
    "KnownStatus":"RUNNING",
    "Limits":
        {
            "CPU":0.25,
            "Memory":512
        },
    "PullStartedAt":"2025-01-17T20:56:17.394610044Z",
    "PullStoppedAt":"2025-01-17T20:56:25.282708213Z",
    "AvailabilityZone":"us-east-1b",
    "LaunchType":"FARGATE",
    "Containers":[
        {
            "DockerId":"b593651c4d6b44a6b2b583f45c957e15-3356213583",
            "Name":"curltest","DockerName":"curltest",
            "Image":"public.ecr.aws/amazonlinux/amazonlinux:latest",

 "ImageID":"sha256:7f371357694782356b65c7fd60dd1ca124c47bd5ed1b1ffe7c0e17f562898367",
            "Labels":
                {
                    "com.amazonaws.ecs.cluster":"arn:aws:ecs:us-
east-1:123456789012:cluster/MyCluster",
                    "com.amazonaws.ecs.container-name":"curltest",
                    "com.amazonaws.ecs.task-arn":"arn:aws:ecs:us-
east-1:123456789012:task/MyCluster/b593651c4d6b44a6b2b583f45c957e15",
                    "com.amazonaws.ecs.task-definition-family":"curltest-
container","com.amazonaws.ecs.task-definition-version":"2"
                },
            "DesiredStatus":"RUNNING",
```

```
            "KnownStatus":"RUNNING",
            "Limits":
                {
                    "CPU":2
                },
            "CreatedAt":"2025-01-17T20:56:26.180347056Z",
            "StartedAt":"2025-01-17T20:56:26.180347056Z",
            "Type":"NORMAL",
            "LogDriver":"awslogs",
            "LogOptions":
                {
                    "awslogs-create-group":"true",
                    "awslogs-group":"/ecs/curltest-container",
                    "awslogs-region":"us-east-1",
                    "awslogs-stream":"ecs/curltest/b593651c4d6b44a6b2b583f45c957e15"
                       },
            "ContainerARN":"arn:aws:ecs:us-east-1:123456789012:container/MyCluster/
b593651c4d6b44a6b2b583f45c957e15/934575e8-5bdb-478f-b763-2341a85b690e",
            "Networks":[
                {
                    "NetworkMode":"awsvpc",
                    "IPv4Addresses":["10.0.1.58"]
                }
            ],
            "Snapshotter":"overlayfs"
        }
    ],
    "ClockDrift":
        {
            "ClockErrorBound":0.487801,"ReferenceTimestamp":"2025-01-17T20:56:02Z",
            "ClockSynchronizationStatus":"SYNCHRONIZED"
        },
    "FaultInjectionEnabled":false
}
```

Example Archivo de metadatos del contenedor de Amazon ECS incompleto (aún no está en el estado **READY**)

En el ejemplo siguiente se muestra un archivo de metadatos del contenedor que todavía no ha alcanzado el estado READY. La información del archivo se limita a algunos parámetros que se conocen de la definición de tarea. El archivo de metadatos del contenedor debería estar listo transcurrido un segundo desde que se inicia el contenedor.

```
{
    "Cluster": "default",
    "ContainerInstanceARN": "arn:aws:ecs:us-west-2:012345678910:container-instance/
default/1f73d099-b914-411c-a9ff-81633b7741dd",
    "TaskARN": "arn:aws:ecs:us-west-2:012345678910:task/default/
d90675f8-1a98-444b-805b-3d9cabb6fcd4",
    "ContainerName": "metadata"
}
```

Metadatos de tareas disponibles para tareas de Amazon ECS en EC2

El agente de contenedor de Amazon ECS proporciona un método para recuperar varios metadatos y estadísticas de Docker de tareas. Esto se conoce como el punto de enlace de metadatos de tareas. Están disponibles las siguientes versiones:

- Versión 4 del punto de enlace de metadatos de tareas: proporciona diversos metadatos y estadísticas de Docker a los contenedores. También puede proporcionar datos de velocidad de red. Disponible para las tareas de Amazon ECS lanzadas en instancias Linux de Amazon EC2 que ejecutan la versión `1.39.0` del agente de contenedor de Amazon ECS como mínimo. Para instancias de Windows de Amazon EC2 que utilizan el modo de red `awsvpc`, el agente de contenedor de Amazon ECS debe ser versión `1.54.0` como mínimo. Para obtener más información, consulte Versión 4 del punto de conexión de metadatos de tareas de Amazon ECS.
- Versión 3 del punto de enlace de metadatos de tareas: proporciona diversos metadatos y estadísticas de Docker a los contenedores. Disponible para las tareas de Amazon ECS lanzadas en instancias Linux de Amazon EC2 que ejecutan la versión `1.21.0` del agente de contenedor de Amazon ECS como mínimo. Para instancias de Windows de Amazon EC2 que utilizan el modo de red `awsvpc`, el agente de contenedor de Amazon ECS debe ser versión `1.54.0` como mínimo. Para obtener más información, consulte Versión 3 del punto de conexión de los metadatos de tareas de Amazon ECS.
- Versión 2 del punto de enlace de metadatos de tareas: disponible para tareas de Amazon ECS lanzadas en instancias de Linux de Amazon EC2 que ejecutan la versión `1.17.0` del agente de contenedor de Amazon ECS como mínimo. Para instancias de Windows de Amazon EC2 que utilizan el modo de red `awsvpc`, el agente de contenedor de Amazon ECS debe ser versión `1.54.0` como mínimo. Para obtener más información, consulte Versión 2 del punto de conexión de los metadatos de tareas de Amazon ECS.

Si su tarea de Amazon ECS está alojada en Amazon EC2, también puede acceder a los metadatos del host de tareas mediante el punto de conexión del servicio de metadatos de instancias (IMDS). El siguiente comando, cuando se ejecuta desde la instancia que aloja la tarea, muestra el ID de la instancia del host.

```
curl http://169.254.169.254/latest/meta-data/instance-id
```

La información que puede obtener desde el punto de conexión se divide en categorías como *`instance-id`*. Para obtener más información acerca de las diferentes categorías de metadatos de instancias de host que puede obtener mediante el punto de conexión, consulte Categorías de metadatos de instancias.

Versión 4 del punto de conexión de metadatos de tareas de Amazon ECS

El agente de contenedor de Amazon ECS introduce una variable de entorno en cada contenedor, denominada punto de enlace de metadatos de tareas, que proporciona varios metadatos y estadísticas de Docker de tareas al contenedor.

Los metadatos y las estadísticas de velocidad de red de las tareas se envían a CloudWatch Container Insights y se pueden consultar desde la AWS Management Console. Para obtener más información, consulte Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada.

> **Note**
>
> Amazon ECS ofrece versiones anteriores del punto de enlace de metadatos de tareas. Para no tener que crear nuevas versiones de los puntos de enlace de metadatos de tareas en el futuro, se pueden agregar otros metadatos a la salida de la versión 4. No vamos a eliminar ningún metadato existente ni a modificar los nombres de los campos de metadatos.

La variable de entorno se introduce de forma predeterminada en los contenedores de las tareas de Amazon ECS lanzadas en instancias de Linux de Amazon EC2 que ejecutan la versión `1.39.0` del agente de contenedor de Amazon ECS como mínimo. Para instancias de Windows de Amazon EC2 que utilizan el modo de red `awsvpc`, el agente de contenedor de Amazon ECS debe ser versión `1.54.0` como mínimo. Para obtener más información, consulte Administración de instancias de contenedor de Linux de Amazon ECS.

> **Note**
>
> Puede agregar compatibilidad con esta característica en instancias de Amazon EC2 que utilizan versiones anteriores del agente de contenedor de Amazon ECS si actualiza el agente a la versión más reciente. Para obtener más información, consulte Actualización del agente de contenedor de Amazon ECS.

Para ver una salida de ejemplo de metadatos de tareas, consulte Ejemplos de la versión 4 de los metadatos de tareas de Amazon ECS.

Rutas de la versión 4 del punto de enlace de metadatos de tareas

Están disponibles los siguientes puntos de enlace de metadatos de tareas para los contenedores:

`${ECS_CONTAINER_METADATA_URI_V4}`

Esta ruta devuelve metadatos del contenedor.

`${ECS_CONTAINER_METADATA_URI_V4}/task`

Esta ruta devuelve metadatos de la tarea, incluso una lista de los nombres e ID de contenedor de todos los contenedores asociados a la tarea. Para obtener más información sobre la respuesta de este punto de enlace, consulte Respuesta JSON para la versión 4 de los metadatos de tareas de Amazon ECS.

`${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`

Esta ruta muestra los metadatos de la tarea incluidos en el punto de enlace `/task` además de en las etiquetas de instancia de contenedor y tarea que se pueden recuperar mediante la API `ListTagsForResource`. Todos los errores recibidos al recuperar los metadatos de la etiqueta se incluirán en la respuesta, en el campo `Errors`.

> **Note**
>
> El campo `Errors` solo aparece en la respuesta para tareas alojadas en instancias de Linux de Amazon EC2 que ejecutan la versión `1.50.0` del agente de contenedor como mínimo. Para instancias de Windows de Amazon EC2 que utilizan el modo de red `awsvpc`, el agente contenedor de Amazon ECS debe ser versión `1.54.0` como mínimo Este punto de conexión requiere el permiso `ecs.ListTagsForResource`.

> **Important**
>
> Cuando utilice el punto de conexión `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`, tenga en cuenta que cada llamada hace hasta dos solicitudes de API a `ecs:ListTagsForResource` (una para las etiquetas de las instancias de contenedor y otra para las etiquetas de tareas) y que los contenedores asociados de la tarea pueden hacer estas llamadas en su nombre. Las llamadas frecuentes a los puntos de conexión pueden provocar una limitación de la API.
> Considere la posibilidad de implementar las estrategias de almacenamiento en caché o procesamiento por lotes para reducir la frecuencia de las llamadas, especialmente en aplicaciones de alto tráfico y depurar los problemas de limitación de la API mediante AWS CloudTrail. Para obtener información sobre la limitación de la API `ListTagsForResource`, consulte Request throttling for the Amazon ECS API en la Referencia de la API de Amazon Elastic Container Service. Para más información sobre la depuración de las llamadas a la API de Amazon ECS mediante AWS CloudTrail, consulte Registro de llamadas a la API de Amazon ECS mediante AWS CloudTrail.

`${ECS_CONTAINER_METADATA_URI_V4}/stats`

Esta ruta muestra estadísticas de Docker para el contenedor específico. Para obtener más información acerca de cada una de las estadísticas devueltas, consulte ContainerStats en la documentación del API de Docker.

Para las tareas de Amazon ECS que utilizan los modos de red `awsvpco` `bridge` alojados en instancias de Linux de Amazon EC2 que ejecutan la versión `1.43.0` del agente de contenedor como mínimo, habrá estadísticas de velocidad de red adicionales incluidas en la respuesta. Para todas las demás tareas, la respuesta solo incluirá las estadísticas de red acumuladas.

`${ECS_CONTAINER_METADATA_URI_V4}/task/stats`

Esta ruta devuelve estadísticas de Docker de todos los contenedores asociados a la tarea. Se puede utilizar en contenedores asociados para extraer métricas de red. Para obtener más información acerca de cada una de las estadísticas devueltas, consulte ContainerStats en la documentación del API de Docker.

Para las tareas de Amazon ECS que utilizan los modos de red `awsvpco` `bridge` alojados en instancias de Linux de Amazon EC2 que ejecutan la versión `1.43.0` del agente de contenedor

como mínimo, habrá estadísticas de velocidad de red adicionales incluidas en la respuesta. Para todas las demás tareas, la respuesta solo incluirá las estadísticas de red acumuladas.

Respuesta JSON para la versión 4 de los metadatos de tareas de Amazon ECS

La siguiente información se devuelve desde la respuesta de JSON (`${ECS_CONTAINER_METADATA_URI_V4}/task`) de punto de enlace de metadatos de tarea. Esto incluye los metadatos asociados a la tarea, además de los metadatos de cada contenedor dentro de la tarea.

`Cluster`

El nombre de recurso de Amazon (ARN) o el nombre corto del clúster de Amazon ECS al que pertenece la tarea.

`ServiceName`

El nombre del servicio al que pertenece la tarea. ServiceName aparecerá para las instancias de contenedor de Amazon EC2 y Amazon ECS Anywhere si la tarea está asociada a un servicio.

> **Note**
>
> Los metadatos de `ServiceName` solo se incluyen cuando se utiliza la versión `1.63.1` o una posterior del agente de contenedores de Amazon ECS.

`VPCID`

El ID de VPC de la instancia de contenedor de Amazon EC2. Este campo solo aparece para las instancias de Amazon EC2.

> **Note**
>
> Los metadatos de `VPCID` solo se incluyen cuando se utiliza la versión `1.63.1` o una posterior del agente de contenedores de Amazon ECS.

`TaskARN`

Nombre de recurso de Amazon (ARN) de la tarea al que pertenece el contenedor.

`Family`

La familia de la definición de tareas de Amazon ECS para la tarea.

`Revision`

La revisión de la definición de tareas de Amazon ECS para la tarea.

`DesiredStatus`

El estado deseado para la tarea de Amazon ECS.

`KnownStatus`

El estado conocido para la tarea de Amazon ECS.

`Limits`

Los límites de recursos especificados en el nivel de tarea, por ejemplo, CPU (expresado en vCPU) y memoria. Este parámetro se omite si no se definen límites de recurso.

`PullStartedAt`

La marca temporal del momento en que comenzó la primera extracción de la imagen del contenedor.

`PullStoppedAt`

La marca temporal del momento en que finalizó la última extracción de la imagen del contenedor.

`AvailabilityZone`

La zona de disponibilidad donde está la tarea.

> **Note**
>
> Los metadatos de la zona de disponibilidad solo están disponibles para las tareas de Fargate que utilicen la versión 1.4 o posterior (Linux) o 1.0.0 (Windows) de la plataforma.

`LaunchType`

El tipo de lanzamiento que usa la tarea. Cuando se utilizan proveedores de capacidad de clúster, indica si la tarea está utilizando la infraestructura Fargate o EC2.

> **Note**
>
> Este metadato `LaunchType` solo se incluye cuando se utiliza la versión `1.45.0` o posterior (Linux) o 1.0.0 o posterior (Windows) del agente de contenedor de Amazon ECS o una posterior.

`Containers`

Una lista de metadatos de contenedor para cada contenedor asociado con la tarea.

`DockerId`

El ID de Docker para el contenedor.

Cuando usa Fargate, el ID es un hexadecimal de 32 dígitos seguido de un número de 10 dígitos.

`Name`

El nombre del contenedor tal y como se especifica en la definición de tarea.

`DockerName`

El nombre del contenedor suministrado a Docker. El agente de contenedor de Amazon ECS genera un nombre único para el contenedor a fin de evitar conflictos de nombre cuando se ejecutan en una sola instancia varias copias de la misma definición de tareas.

`Image`

La imagen para el contenedor.

`ImageID`

El resumen SHA-256 del manifiesto de la imagen. Este es el resumen que se puede usar para extraer la imagen utilizando el formato `repository-url/image@sha256:digest`.

`Ports`

Los puertos expuestos para el contenedor. Este parámetro se omite si no hay puertos expuestos.

`Labels`

Cualquier etiqueta aplicada al contenedor. Este parámetro se omite si no hay etiquetas aplicadas.

`DesiredStatus`

El estado deseado para el contenedor procedente de Amazon ECS.

`KnownStatus`

El estado conocido para el contenedor procedente de Amazon ECS.

`ExitCode`

El código de salida para el contenedor. Este parámetro se omite si el contenedor no ha salido.

`Limits`

Los límites de recursos especificados en el nivel de contenedor, por ejemplo, CPU (expresado en unidades de CPU) y memoria. Este parámetro se omite si no se definen límites de recurso.

`CreatedAt`

La marca de hora para cuando el contenedor se creó. Este parámetro se omite si el contenedor no ha se ha creado aún.

`StartedAt`

La marca de hora para cuando el contenedor se inició. Este parámetro se omite si el contenedor no ha se ha iniciado aún.

`FinishedAt`

La marca de hora para cuando el contenedor se detuvo. Este parámetro se omite si el contenedor no ha se ha detenido aún.

`Type`

El tipo del contenedor. Los contenedores que se especifican en su definición de tarea son de tipo `NORMAL`. Puede hacer caso omiso de otros tipos de contenedores que utiliza el agente de contenedor de Amazon ECS para el aprovisionamiento de recursos para tareas internas.

`LogDriver`

El controlador de registros que utiliza el contenedor.

> **Note**
>
> Este metadato `LogDriver` solo se incluye cuando se utiliza la versión `1.45.0` del agente de contenedor de Linux de Amazon ECS o una posterior.

`LogOptions`

Opciones del controlador de registros definidas para el contenedor.

> **Note**
>
> Este metadato `LogOptions` solo se incluye cuando se utiliza la versión `1.45.0` del agente de contenedor de Linux de Amazon ECS o una posterior.

`ContainerARN`

El nombre de recurso de Amazon (ARN) del contenedor.

> **Note**
>
> Este metadato `ContainerARN` solo se incluye cuando se utiliza la versión `1.45.0` del agente de contenedor de Linux de Amazon ECS o una posterior.

`Networks`

La información de red del contenedor, como la dirección IP y el modo de red. Este parámetro se omite si no se define ninguna información de red.

`RestartCount`

La cantidad de veces que se reinició el contenedor.

> **Note**
>
> Los metadatos `RestartCount` se incluyen solo si hay una política de reinicio habilitada para el contenedor. Para obtener más información, consulte Reinicio de contenedores individuales en tareas de Amazon ECS con políticas de reinicio de contenedores.

`ExecutionStoppedAt`

La marca temporal para cuando el `DesiredStatus` de la tarea pasó a `STOPPED`. Esto ocurre cuando un contenedor esencial pasa a `STOPPED`.

Ejemplos de la versión 4 de los metadatos de tareas de Amazon ECS

En los siguientes ejemplos, se muestran resultados de ejemplo de cada uno de los puntos de enlace de metadatos de tareas.

Ejemplo de respuesta de metadatos del contenedor

Cuando se consulta el punto de enlace `${ECS_CONTAINER_METADATA_URI_V4}`, solo se devuelven los metadatos relacionados con el propio contenedor. A continuación se muestra un ejemplo del resultado de una tarea que se ejecuta como parte de un servicio (MyService).

```
{
    "DockerId": "ea32192c8553fbff06c9340478a2ff089b2bb5646fb718b4ee206641c9086d66",
    "Name": "curl",
    "DockerName": "ecs-curltest-24-curl-cca48e8dcadd97805600",
    "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
    "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
    "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/8f03e41243824aea923aca126495f665",
        "com.amazonaws.ecs.task-definition-family": "curltest",
        "com.amazonaws.ecs.task-definition-version": "24"
    },
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 10,
        "Memory": 128
    },
    "CreatedAt": "2020-10-02T00:15:07.620912337Z",
    "StartedAt": "2020-10-02T00:15:08.062559351Z",
    "Type": "NORMAL",
    "LogDriver": "awslogs",
    "LogOptions": {
        "awslogs-create-group": "true",
        "awslogs-group": "/ecs/metadata",
        "awslogs-region": "us-west-2",
        "awslogs-stream": "ecs/curl/8f03e41243824aea923aca126495f665"
    },
```

```
    "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/0206b271-
b33f-47ab-86c6-a0ba208a70a9",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "10.0.2.100"
            ],
            "AttachmentIndex": 0,
            "MACAddress": "0e:9e:32:c7:48:85",
            "IPv4SubnetCIDRBlock": "10.0.2.0/24",
            "PrivateDNSName": "ip-10-0-2-100.us-west-2.compute.internal",
            "SubnetGatewayIpv4Address": "10.0.2.1/24"
        }
    ]
}
```

Ejemplo de respuesta de metadatos de las tareas

Al consultar el punto de enlace `${ECS_CONTAINER_METADATA_URI_V4}/task`, se muestran los metadatos relacionados con la tarea de la que forma parte el contenedor, además de los metadatos de cada contenedor de la tarea. El siguiente es un ejemplo de salida.

```
{
    "Cluster": "default",
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "LaunchType": "EC2",
    "Containers": [
        {
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
```

```
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
```

```
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
}
```

Respuesta de ejemplo de metadatos de tareas con etiquetas

Cuando se consulta el punto de enlace `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`, se muestran los metadatos relacionados con la tarea, incluidas las etiquetas dela tarea y la instancia de contenedor. El siguiente es un ejemplo de salida.

```
{
    "Cluster": "default",
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "TaskTags": {
        "tag-use": "task-metadata-endpoint-test"
    },
    "ContainerInstanceTags":{
        "tag_key":"tag_value"
    },
    "LaunchType": "EC2",
    "Containers": [
        {
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
```

```
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
```

```
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
}
```

Tarea de ejemplo con etiquetas con una respuesta de metadatos de error

Cuando se consulta el punto de enlace `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`, se muestran los metadatos relacionados con la tarea, incluidas las etiquetas dela tarea y la instancia de contenedor. Si se produce un error al recuperar los datos de etiquetado, el error se muestra en la respuesta. Esta es una respuesta de ejemplo de cuando el rol de IAM asociado a la instancia de contenedor no dispone del permiso `ecs:ListTagsForResource`.

```
{
    "Cluster": "default",
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
```

```
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "Errors": [
        {
            "ErrorField": "ContainerInstanceTags",
            "ErrorCode": "AccessDeniedException",
            "ErrorMessage": "User: arn:aws:sts::111122223333:assumed-
role/ecsInstanceRole/i-0744a608689EXAMPLE is not authorized to perform:
 ecs:ListTagsForResource on resource: arn:aws:ecs:us-west-2:111122223333:container-
instance/default/2dd1b186f39845a584488d2ef155c131",
            "StatusCode": 400,
            "RequestId": "cd597ef0-272b-4643-9bd2-1de469870fa6",
            "ResourceARN": "arn:aws:ecs:us-west-2:111122223333:container-instance/
default/2dd1b186f39845a584488d2ef155c131"
        },
        {
            "ErrorField": "TaskTags",
            "ErrorCode": "AccessDeniedException",
            "ErrorMessage": "User: arn:aws:sts::111122223333:assumed-
role/ecsInstanceRole/i-0744a608689EXAMPLE is not authorized to perform:
 ecs:ListTagsForResource on resource: arn:aws:ecs:us-west-2:111122223333:task/
default/9ef30e4b7aa44d0db562749cff4983f3",
            "StatusCode": 400,
            "RequestId": "862c5986-6cd2-4aa6-87cc-70be395531e1",
            "ResourceARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/9ef30e4b7aa44d0db562749cff4983f3"
        }
    ],
    "LaunchType": "EC2",
    "Containers": [
        {
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
```

```
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
```

```
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
}
```

Respuesta de ejemplo de estadísticas del contenedor

Cuando se consulta el punto de enlace `${ECS_CONTAINER_METADATA_URI_V4}/stats`, se muestran las métricas de red del contenedor. Para las tareas de Amazon ECS que utilizan los modos de red `awsvpc` o `bridge` alojados en instancias de Amazon EC2 que ejecutan la versión `1.43.0` del agente de contenedor como mínimo, la respuesta incluirá estadísticas de velocidad

de red adicionales. Para todas las demás tareas, la respuesta solo incluirá las estadísticas de red acumuladas.

A continuación, se muestra un ejemplo del resultado de una tarea de Amazon ECS en Amazon EC2 que utiliza el modo de red `bridge`.

```
{
    "read": "2020-10-02T00:51:13.410254284Z",
    "preread": "2020-10-02T00:51:12.406202398Z",
    "pids_stats": {
        "current": 3
    },
    "blkio_stats": {
        "io_service_bytes_recursive": [

        ],
        "io_serviced_recursive": [

        ],
        "io_queue_recursive": [

        ],
        "io_service_time_recursive": [

        ],
        "io_wait_time_recursive": [

        ],
        "io_merged_recursive": [

        ],
        "io_time_recursive": [

        ],
        "sectors_recursive": [

        ]
    },
    "num_procs": 0,
    "storage_stats": {

    },
    "cpu_stats": {
```

```
        "cpu_usage": {
            "total_usage": 360968065,
            "percpu_usage": [
                182359190,
                178608875
            ],
            "usage_in_kernelmode": 40000000,
            "usage_in_usermode": 290000000
        },
        "system_cpu_usage": 13939680000000,
        "online_cpus": 2,
        "throttling_data": {
            "periods": 0,
            "throttled_periods": 0,
            "throttled_time": 0
        }
    },
    "precpu_stats": {
        "cpu_usage": {
            "total_usage": 360968065,
            "percpu_usage": [
                182359190,
                178608875
            ],
            "usage_in_kernelmode": 40000000,
            "usage_in_usermode": 290000000
        },
        "system_cpu_usage": 13937670000000,
        "online_cpus": 2,
        "throttling_data": {
            "periods": 0,
            "throttled_periods": 0,
            "throttled_time": 0
        }
    },
    "memory_stats": {
        "usage": 1806336,
        "max_usage": 6299648,
        "stats": {
            "active_anon": 606208,
            "active_file": 0,
            "cache": 0,
            "dirty": 0,
            "hierarchical_memory_limit": 134217728,
```

```
            "hierarchical_memsw_limit": 268435456,
            "inactive_anon": 0,
            "inactive_file": 0,
            "mapped_file": 0,
            "pgfault": 4185,
            "pgmajfault": 0,
            "pgpgin": 2926,
            "pgpgout": 2778,
            "rss": 606208,
            "rss_huge": 0,
            "total_active_anon": 606208,
            "total_active_file": 0,
            "total_cache": 0,
            "total_dirty": 0,
            "total_inactive_anon": 0,
            "total_inactive_file": 0,
            "total_mapped_file": 0,
            "total_pgfault": 4185,
            "total_pgmajfault": 0,
            "total_pgpgin": 2926,
            "total_pgpgout": 2778,
            "total_rss": 606208,
            "total_rss_huge": 0,
            "total_unevictable": 0,
            "total_writeback": 0,
            "unevictable": 0,
            "writeback": 0
        },
        "limit": 134217728
    },
    "name": "/ecs-curltest-26-curl-c2e5f6e0cf91b0bead01",
    "id": "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af",
    "networks": {
        "eth0": {
            "rx_bytes": 84,
            "rx_packets": 2,
            "rx_errors": 0,
            "rx_dropped": 0,
            "tx_bytes": 84,
            "tx_packets": 2,
            "tx_errors": 0,
            "tx_dropped": 0
        }
    },
```

```
    "network_rate_stats": {
        "rx_bytes_per_sec": 0,
        "tx_bytes_per_sec": 0
    }
}
```

Ejemplo de respuesta de estadísticas de las tareas

Cuando se consulta el punto de enlace `${ECS_CONTAINER_METADATA_URI_V4}/task/stats`, se devuelven las métricas de red relacionadas con la tarea de la que forma parte el contenedor. El siguiente es un ejemplo de salida.

```
{
    "01999f2e5c6cf4df3873f28950e6278813408f281c54778efec860d0caad4854": {
        "read": "2020-10-02T00:51:32.51467703Z",
        "preread": "2020-10-02T00:51:31.50860463Z",
        "pids_stats": {
            "current": 1
        },
        "blkio_stats": {
            "io_service_bytes_recursive": [

            ],
            "io_serviced_recursive": [

            ],
            "io_queue_recursive": [

            ],
            "io_service_time_recursive": [

            ],
            "io_wait_time_recursive": [

            ],
            "io_merged_recursive": [

            ],
            "io_time_recursive": [

            ],
            "sectors_recursive": [
```

```
            ]
        },
        "num_procs": 0,
        "storage_stats": {

        },
        "cpu_stats": {
            "cpu_usage": {
                "total_usage": 177232665,
                "percpu_usage": [
                    13376224,
                    163856441
                ],
                "usage_in_kernelmode": 0,
                "usage_in_usermode": 160000000
            },
            "system_cpu_usage": 13977820000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "precpu_stats": {
            "cpu_usage": {
                "total_usage": 177232665,
                "percpu_usage": [
                    13376224,
                    163856441
                ],
                "usage_in_kernelmode": 0,
                "usage_in_usermode": 160000000
            },
            "system_cpu_usage": 13975800000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "memory_stats": {
            "usage": 532480,
```

```
            "max_usage": 6279168,
            "stats": {
                "active_anon": 40960,
                "active_file": 0,
                "cache": 0,
                "dirty": 0,
                "hierarchical_memory_limit": 9223372036854771712,
                "hierarchical_memsw_limit": 9223372036854771712,
                "inactive_anon": 0,
                "inactive_file": 0,
                "mapped_file": 0,
                "pgfault": 2033,
                "pgmajfault": 0,
                "pgpgin": 1734,
                "pgpgout": 1724,
                "rss": 40960,
                "rss_huge": 0,
                "total_active_anon": 40960,
                "total_active_file": 0,
                "total_cache": 0,
                "total_dirty": 0,
                "total_inactive_anon": 0,
                "total_inactive_file": 0,
                "total_mapped_file": 0,
                "total_pgfault": 2033,
                "total_pgmajfault": 0,
                "total_pgpgin": 1734,
                "total_pgpgout": 1724,
                "total_rss": 40960,
                "total_rss_huge": 0,
                "total_unevictable": 0,
                "total_writeback": 0,
                "unevictable": 0,
                "writeback": 0
            },
            "limit": 4073377792
        },
        "name": "/ecs-curltest-26-internalecspause-a6bcc3dbadfacfe85300",
        "id": "01999f2e5c6cf4df3873f28950e6278813408f281c54778efec860d0caad4854",
        "networks": {
            "eth0": {
                "rx_bytes": 84,
                "rx_packets": 2,
                "rx_errors": 0,
```

```
                "rx_dropped": 0,
                "tx_bytes": 84,
                "tx_packets": 2,
                "tx_errors": 0,
                "tx_dropped": 0
            }
        },
        "network_rate_stats": {
            "rx_bytes_per_sec": 0,
            "tx_bytes_per_sec": 0
        }
    },
    "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af": {
        "read": "2020-10-02T00:51:32.512771349Z",
        "preread": "2020-10-02T00:51:31.510597736Z",
        "pids_stats": {
            "current": 3
        },
        "blkio_stats": {
            "io_service_bytes_recursive": [

            ],
            "io_serviced_recursive": [

            ],
            "io_queue_recursive": [

            ],
            "io_service_time_recursive": [

            ],
            "io_wait_time_recursive": [

            ],
            "io_merged_recursive": [

            ],
            "io_time_recursive": [

            ],
            "sectors_recursive": [

            ]
        },
```

```
        "num_procs": 0,
        "storage_stats": {

        },
        "cpu_stats": {
            "cpu_usage": {
                "total_usage": 379075681,
                "percpu_usage": [
                    191355275,
                    187720406
                ],
                "usage_in_kernelmode": 60000000,
                "usage_in_usermode": 310000000
            },
            "system_cpu_usage": 13977800000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "precpu_stats": {
            "cpu_usage": {
                "total_usage": 378825197,
                "percpu_usage": [
                    191104791,
                    187720406
                ],
                "usage_in_kernelmode": 60000000,
                "usage_in_usermode": 310000000
            },
            "system_cpu_usage": 13975800000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "memory_stats": {
            "usage": 1814528,
            "max_usage": 6299648,
            "stats": {
```

```
                "active_anon": 606208,
                "active_file": 0,
                "cache": 0,
                "dirty": 0,
                "hierarchical_memory_limit": 134217728,
                "hierarchical_memsw_limit": 268435456,
                "inactive_anon": 0,
                "inactive_file": 0,
                "mapped_file": 0,
                "pgfault": 5377,
                "pgmajfault": 0,
                "pgpgin": 3613,
                "pgpgout": 3465,
                "rss": 606208,
                "rss_huge": 0,
                "total_active_anon": 606208,
                "total_active_file": 0,
                "total_cache": 0,
                "total_dirty": 0,
                "total_inactive_anon": 0,
                "total_inactive_file": 0,
                "total_mapped_file": 0,
                "total_pgfault": 5377,
                "total_pgmajfault": 0,
                "total_pgpgin": 3613,
                "total_pgpgout": 3465,
                "total_rss": 606208,
                "total_rss_huge": 0,
                "total_unevictable": 0,
                "total_writeback": 0,
                "unevictable": 0,
                "writeback": 0
            },
            "limit": 134217728
        },
        "name": "/ecs-curltest-26-curl-c2e5f6e0cf91b0bead01",
        "id": "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af",
        "networks": {
            "eth0": {
                "rx_bytes": 84,
                "rx_packets": 2,
                "rx_errors": 0,
                "rx_dropped": 0,
                "tx_bytes": 84,
```

```
                "tx_packets": 2,
                "tx_errors": 0,
                "tx_dropped": 0
            }
        },
        "network_rate_stats": {
            "rx_bytes_per_sec": 0,
            "tx_bytes_per_sec": 0
        }
    }
}
```

Versión 3 del punto de conexión de los metadatos de tareas de Amazon ECS

> **Important**
>
> El punto de conexión de la versión 3 de los metadatos de la tarea ya no se mantiene activamente. Le recomendamos que actualice el punto de conexión de metadatos de la tarea versión 4 para obtener la información más reciente del punto de conexión de metadatos. Para obtener más información, consulte the section called “Versión 4 del punto de enlace de metadatos de tareas”.
> Si utiliza tareas de Amazon ECS alojadas en AWS Fargate, consulte Punto de conexión de metadatos de tareas de Amazon ECS versión 3 para tareas en Fargate.

A partir de la versión 1.21.0 del agente de contenedor de Amazon ECS, el agente introduce una variable de entorno con el nombre `ECS_CONTAINER_METADATA_URI` en cada contenedor de las tareas. Cuando consulta la versión 3 del punto de enlace de metadatos de tarea, están disponibles diversos metadatos de tarea y estadísticas de Docker para las tareas. En el caso de las tareas que utilizan el modo de red `bridge`, hay métricas de red disponibles que pueden utilizarse al consultar los puntos de enlace `/stats`.

La característica de la versión 3 del punto de enlace de metadatos de tareas está habilitada de forma predeterminada para las tareas que utilizan el tipo de lanzamiento de Fargate en la versión 1.3.0 de la plataforma o una posterior, y en las tareas que utilizan el tipo de lanzamiento de EC2 y se lanzan en una infraestructura de Linux de Amazon ECS que ejecuta la versión 1.21.0 del agente de contenedor de Amazon ECS como mínimo una infraestructura de Windows de Amazon EC2 que ejecute la versión `1.54.0` del agente de contenedor de Amazon ECS y utilice el modo de red

`awsvpc`. Para obtener más información, consulte Administración de instancias de contenedor de Linux de Amazon ECS.

Puede añadir soporte para esta característica en instancias de contenedor anteriores actualizando el agente a la versión más reciente. Para obtener más información, consulte Actualización del agente de contenedor de Amazon ECS.

> **Important**
>
> Para las tareas que utilizan el tipo de lanzamiento de Fargate y las versiones de la plataforma anteriores a la versión 1.3.0, se admite la versión 2 del punto de enlace de metadatos de tareas. Para obtener más información, consulte Versión 2 del punto de conexión de los metadatos de tareas de Amazon ECS.

Rutas de la versión 3 del punto de conexión de los metadatos de tareas

Los siguientes puntos de enlace de metadatos de tarea están disponibles para los contenedores:

`${ECS_CONTAINER_METADATA_URI}`

Esta ruta devuelve JSON de metadatos para el contenedor.

`${ECS_CONTAINER_METADATA_URI}/task`

Esta ruta devuelve JSON de metadatos para la tarea, incluida una lista de los nombres e ID del contenedor de todos los contenedores asociados con la tarea. Para obtener más información sobre la respuesta de este punto de enlace, consulte Respuesta JSON para la versión 3 de los metadatos de tareas de Amazon ECS.

`${ECS_CONTAINER_METADATA_URI}/taskWithTags`

Esta ruta muestra los metadatos de la tarea incluidos en el punto de enlace `/task`, además de en las etiquetas de las instancias de contenedor y las tareas que se pueden recuperar mediante la API `ListTagsForResource`.

`${ECS_CONTAINER_METADATA_URI}/stats`

Esta ruta devuelve JSON de estadísticas de Docker para el ID de contenedor de Docker específico. Para obtener más información acerca de cada una de las estadísticas devueltas, consulte ContainerStats en la documentación del API de Docker.

`${ECS_CONTAINER_METADATA_URI}/task/stats`

Esta ruta devuelve JSON de estadísticas de Docker de todos los contenedores asociados con la tarea. Para obtener más información acerca de cada una de las estadísticas devueltas, consulte ContainerStats en la documentación del API de Docker.

Respuesta JSON para la versión 3 de los metadatos de tareas de Amazon ECS

La siguiente información se devuelve desde la respuesta de JSON (`${ECS_CONTAINER_METADATA_URI}/task`) de punto de enlace de metadatos de tarea.

`Cluster`

El nombre de recurso de Amazon (ARN) o el nombre corto del clúster de Amazon ECS al que pertenece la tarea.

`TaskARN`

Nombre de recurso de Amazon (ARN) de la tarea al que pertenece el contenedor.

`Family`

La familia de la definición de tareas de Amazon ECS para la tarea.

`Revision`

La revisión de la definición de tareas de Amazon ECS para la tarea.

`DesiredStatus`

El estado deseado para la tarea de Amazon ECS.

`KnownStatus`

El estado conocido para la tarea de Amazon ECS.

`Limits`

Los límites de recursos especificados en el nivel de tarea, por ejemplo, CPU (expresado en vCPU) y memoria. Este parámetro se omite si no se definen límites de recurso.

`PullStartedAt`

La marca temporal del momento en que comenzó la primera extracción de la imagen del contenedor.

`PullStoppedAt`

La marca temporal del momento en que finalizó la última extracción de la imagen del contenedor.

`AvailabilityZone`

La zona de disponibilidad donde está la tarea.

> **Note**
>
> Los metadatos de la zona de disponibilidad solo están disponibles para las tareas de Fargate que utilicen la versión 1.4 o posterior (Linux) o 1.0.0 o posterior (Windows) de la plataforma.

`Containers`

Una lista de metadatos de contenedor para cada contenedor asociado con la tarea.

`DockerId`

El ID de Docker para el contenedor.

`Name`

El nombre del contenedor tal y como se especifica en la definición de tarea.

`DockerName`

El nombre del contenedor suministrado a Docker. El agente de contenedor de Amazon ECS genera un nombre único para el contenedor a fin de evitar conflictos de nombre cuando se ejecutan en una sola instancia varias copias de la misma definición de tareas.

`Image`

La imagen para el contenedor.

`ImageID`

El resumen SHA-256 del manifiesto de la imagen. Este es el resumen que se puede usar para extraer la imagen utilizando el formato `repository-url/image@sha256:digest`.

`Ports`

Los puertos expuestos para el contenedor. Este parámetro se omite si no hay puertos expuestos.

`Labels`

Cualquier etiqueta aplicada al contenedor. Este parámetro se omite si no hay etiquetas aplicadas.

`DesiredStatus`

El estado deseado para el contenedor procedente de Amazon ECS.

`KnownStatus`

El estado conocido para el contenedor procedente de Amazon ECS.

`ExitCode`

El código de salida para el contenedor. Este parámetro se omite si el contenedor no ha salido.

`Limits`

Los límites de recursos especificados en el nivel de contenedor, por ejemplo, CPU (expresado en unidades de CPU) y memoria. Este parámetro se omite si no se definen límites de recurso.

`CreatedAt`

La marca de hora para cuando el contenedor se creó. Este parámetro se omite si el contenedor no ha se ha creado aún.

`StartedAt`

La marca de hora para cuando el contenedor se inició. Este parámetro se omite si el contenedor no ha se ha iniciado aún.

`FinishedAt`

La marca de hora para cuando el contenedor se detuvo. Este parámetro se omite si el contenedor no ha se ha detenido aún.

`Type`

El tipo del contenedor. Los contenedores que se especifican en su definición de tarea son de tipo `NORMAL`. Puede hacer caso omiso de otros tipos de contenedores que utiliza el agente de contenedor de Amazon ECS para el aprovisionamiento de recursos para tareas internas.

`Networks`

La información de red del contenedor, como la dirección IP y el modo de red. Este parámetro se omite si no se define ninguna información de red.

`ClockDrift`

La información sobre la diferencia entre la hora de referencia y la hora del sistema. Esto se aplica al sistema operativo Linux. Esta capacidad utiliza el Servicio de sincronización temporal de Amazon para medir la precisión del reloj y proporcionar el límite de error de reloj de los contenedores. Para más información, consulte Set the time for your Linux instance en la Guía del usuario de Amazon EC2 para instancias de Linux.

`ReferenceTime`

La base de la precisión del reloj. Amazon ECS utiliza el estándar global de hora universal coordinada (UTC) a través de NTP, por ejemplo `2021-09-07T16:57:44Z`.

`ClockErrorBound`

La medida del error de reloj, definida como el desplazamiento a UTC. Este error es la diferencia en milisegundos entre la hora de referencia y la hora del sistema.

`ClockSynchronizationStatus`

Indica si el intento de sincronización más reciente entre la hora del sistema y la hora de referencia se ha realizado correctamente.

Los valores válidos son `SYNCHRONIZED` y `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

La marca temporal para cuando el `DesiredStatus` de la tarea pasó a `STOPPED`. Esto ocurre cuando un contenedor esencial pasa a `STOPPED`.

Ejemplos de la versión 3 de los metadatos de tareas de Amazon ECS

En los siguientes ejemplos se muestran resultados de ejemplo de los puntos de enlace de metadatos de tareas.

Ejemplo de respuesta de metadatos de contenedor

Cuando se consulta el punto de enlace `${ECS_CONTAINER_METADATA_URI}`, solo se devuelven los metadatos relacionados con el propio contenedor. El siguiente es un ejemplo de salida.

```
{
    "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
    "Name": "nginx-curl",
    "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
```

```
    "Image": "nrdlngr/nginx-curl",
    "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
    "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
    },
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 512,
        "Memory": 512
    },
    "CreatedAt": "2018-02-01T20:55:10.554941919Z",
    "StartedAt": "2018-02-01T20:55:11.064236631Z",
    "Type": "NORMAL",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "10.0.2.106"
            ]
        }
    ]
}
```

Ejemplo de respuesta de metadatos de las tareas

Cuando se consulta el punto de enlace `${ECS_CONTAINER_METADATA_URI}/task`, se devuelven los metadatos relacionados con la tarea de la que forma parte el contenedor. El siguiente es un ejemplo de salida.

La siguiente respuesta JSON es para una tarea de contenedor único.

```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
```

```
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
```

```
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
}
```

Versión 2 del punto de conexión de los metadatos de tareas de Amazon ECS

> **Important**
>
> El punto de conexión de la versión 2 de los metadatos de la tarea ya no se mantiene activamente. Le recomendamos que actualice el punto de conexión de metadatos de la tarea versión 4 para obtener la información más reciente del punto de conexión de metadatos. Para obtener más información, consulte the section called “Versión 4 del punto de enlace de metadatos de tareas”.

A partir de la versión 1.17.0 del agente de contenedor de Amazon ECS, están disponibles diversos metadatos de tarea y estadísticas de Docker para las tareas que utilizan el modo de red `awsvpc` en un punto de enlace HTTP proporcionado por el agente de contenedor de Amazon ECS.

Todos los contenedores que pertenecen a las tareas que se lanzaron con el modo de red `awsvpc` reciben una dirección IPv4 local dentro de un rango de direcciones local de enlace predefinido. Cuando un contenedor consulta el punto de enlace de metadatos, el agente de contenedor de Amazon ECS determina a qué tarea pertenece el contenedor en función de su dirección IP única y de los metadatos y estadísticas que se muestran para dicha tarea.

Habilitación de metadatos de tareas

La característica de versión 2 de metadatos de tarea está habilitada de forma predeterminada para lo siguiente:

- Tareas que utilizan el tipo de lanzamiento de Fargate que utiliza la versión 1.1.0 de la plataforma o una posterior. Para obtener más información, consulte Versiones de la plataforma Fargate para Amazon ECS.
- Tareas que utilizan el tipo de lanzamiento de EC2 y que también utilizan el modo de red `awsvpc` y se lanzan en una infraestructura de Linux de Amazon EC2 que ejecuta la versión 1.17.0 del agente de contenedor de Amazon ECS como mínimo o en una infraestructura de Windows de Amazon EC2 que ejecuta la versión `1.54.0` del agente de contenedor de Amazon ECS como mínimo. Para obtener más información, consulte Administración de instancias de contenedor de Linux de Amazon ECS.

Puede añadir soporte para esta característica en instancias de contenedor anteriores actualizando el agente a la versión más reciente. Para obtener más información, consulte Actualización del agente de contenedor de Amazon ECS.

Rutas de punto de enlace de metadatos de tareas

Los siguientes puntos de enlace de API están disponibles para los contenedores:

`169.254.170.2/v2/metadata`

Este punto de enlace devuelve metadatos JSON para la tarea, incluida una lista de los nombres e ID del contenedor de todos los contenedores asociados con la tarea. Para obtener más información sobre la respuesta de este punto de enlace, consulte Respuesta JSON de metadatos de tareas.

`169.254.170.2/v2/metadata/<container-id>`

Este punto de enlace devuelve metadatos JSON para el ID de contenedor de Docker especificado.

`169.254.170.2/v2/metadata/taskWithTags`

Esta ruta muestra los metadatos de la tarea incluidos en el punto de enlace `/task`, además de en las etiquetas de las instancias de contenedor y las tareas que se pueden recuperar mediante la API `ListTagsForResource`.

`169.254.170.2/v2/stats`

Este punto de enlace devuelve JSON de estadísticas de Docker de todos los contenedores asociados con la tarea. Para obtener más información acerca de cada una de las estadísticas devueltas, consulte ContainerStats en la documentación del API de Docker.

`169.254.170.2/v2/stats/<container-id>`

Este punto de enlace devuelve JSON de estadísticas de Docker para el ID de contenedor de Docker especificado. Para obtener más información acerca de cada una de las estadísticas devueltas, consulte ContainerStats en la documentación del API de Docker.

Respuesta JSON de metadatos de tareas

La siguiente información se devuelve desde la respuesta de JSON (`169.254.170.2/v2/metadata`) de punto de enlace de metadatos de tarea.

`Cluster`

El nombre de recurso de Amazon (ARN) o el nombre corto del clúster de Amazon ECS al que pertenece la tarea.

`TaskARN`

Nombre de recurso de Amazon (ARN) de la tarea al que pertenece el contenedor.

`Family`

La familia de la definición de tareas de Amazon ECS para la tarea.

`Revision`

La revisión de la definición de tareas de Amazon ECS para la tarea.

`DesiredStatus`

El estado deseado para la tarea de Amazon ECS.

`KnownStatus`

El estado conocido para la tarea de Amazon ECS.

`Limits`

Los límites de recursos especificados en el nivel de tarea, por ejemplo, CPU (expresado en vCPU) y memoria. Este parámetro se omite si no se definen límites de recurso.

`PullStartedAt`

La marca temporal del momento en que comenzó la primera extracción de la imagen del contenedor.

`PullStoppedAt`

La marca temporal del momento en que finalizó la última extracción de la imagen del contenedor.

`AvailabilityZone`

La zona de disponibilidad donde está la tarea.

> **Note**
>
> Los metadatos de la zona de disponibilidad solo están disponibles para las tareas de Fargate que utilicen la versión 1.4 o posterior (Linux) o 1.0.0 o posterior (Windows) de la plataforma.

`Containers`

Una lista de metadatos de contenedor para cada contenedor asociado con la tarea.

`DockerId`

El ID de Docker para el contenedor.

`Name`

El nombre del contenedor tal y como se especifica en la definición de tarea.

`DockerName`

El nombre del contenedor suministrado a Docker. El agente de contenedor de Amazon ECS genera un nombre único para el contenedor a fin de evitar conflictos de nombre cuando se ejecutan en una sola instancia varias copias de la misma definición de tareas.

`Image`

La imagen para el contenedor.

`ImageID`

El resumen SHA-256 del manifiesto de la imagen. Este es el resumen que se puede usar para extraer la imagen utilizando el formato `repository-url/image@sha256:digest`.

`Ports`

Los puertos expuestos para el contenedor. Este parámetro se omite si no hay puertos expuestos.

`Labels`

Cualquier etiqueta aplicada al contenedor. Este parámetro se omite si no hay etiquetas aplicadas.

`DesiredStatus`

El estado deseado para el contenedor procedente de Amazon ECS.

`KnownStatus`

El estado conocido para el contenedor procedente de Amazon ECS.

`ExitCode`

El código de salida para el contenedor. Este parámetro se omite si el contenedor no ha salido.

`Limits`

Los límites de recursos especificados en el nivel de contenedor, por ejemplo, CPU (expresado en unidades de CPU) y memoria. Este parámetro se omite si no se definen límites de recurso.

`CreatedAt`

La marca de hora para cuando el contenedor se creó. Este parámetro se omite si el contenedor no ha se ha creado aún.

`StartedAt`

La marca de hora para cuando el contenedor se inició. Este parámetro se omite si el contenedor no ha se ha iniciado aún.

`FinishedAt`

La marca de hora para cuando el contenedor se detuvo. Este parámetro se omite si el contenedor no ha se ha detenido aún.

`Type`

El tipo del contenedor. Los contenedores que se especifican en su definición de tarea son de tipo `NORMAL`. Puede hacer caso omiso de otros tipos de contenedores que utiliza el agente de contenedor de Amazon ECS para el aprovisionamiento de recursos para tareas internas.

`Networks`

La información de red del contenedor, como la dirección IP y el modo de red. Este parámetro se omite si no se define ninguna información de red.

`ClockDrift`

La información sobre la diferencia entre la hora de referencia y la hora del sistema. Esto se aplica al sistema operativo Linux. Esta capacidad utiliza el Servicio de sincronización temporal de Amazon para medir la precisión del reloj y proporcionar el límite de error de reloj de los contenedores. Para más información, consulte Set the time for your Linux instance en la Guía del usuario de Amazon EC2 para instancias de Linux.

`ReferenceTime`

La base de la precisión del reloj. Amazon ECS utiliza el estándar global de hora universal coordinada (UTC) a través de NTP, por ejemplo `2021-09-07T16:57:44Z`.

`ClockErrorBound`

La medida del error de reloj, definida como el desplazamiento a UTC. Este error es la diferencia en milisegundos entre la hora de referencia y la hora del sistema.

`ClockSynchronizationStatus`

Indica si el intento de sincronización más reciente entre la hora del sistema y la hora de referencia se ha realizado correctamente.

Los valores válidos son `SYNCHRONIZED` y `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

La marca temporal para cuando el `DesiredStatus` de la tarea pasó a `STOPPED`. Esto ocurre cuando un contenedor esencial pasa a `STOPPED`.

Ejemplo de respuesta de metadatos de las tareas

La siguiente respuesta JSON es para una tarea de contenedor único.

```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
```

```
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
```

```
}
```

Metadatos de tareas de Amazon ECS disponibles para tareas en Fargate

Amazon ECS alojado en Fargate proporciona un método para recuperar diversos metadatos, métricas de red y estadísticas de Docker sobre las tareas y los contenedores a los que pertenecen. Esto se conoce como el punto de enlace de metadatos de tareas. Las versiones de los puntos de enlace de metadatos de tareas disponibles para las tareas de Amazon ECS alojadas en Fargate son las siguientes:

- Versión 4 del punto de enlace de metadatos de tareas: disponible para tareas que utilizan la versión 1.4.0 de la plataforma o una posterior.
- Versión 3 del punto de enlace de metadatos de tareas: disponible para tareas que utilizan la versión 1.1.0 de la plataforma o una posterior.

Todos los contenedores que pertenecen a las tareas que se lanzaron con el modo de red `awsvpc` reciben una dirección IPv4 local dentro de un rango de direcciones local de enlace predefinido. Cuando un contenedor consulta el punto de enlace de metadatos, el agente de contenedores determina a qué tarea pertenece el contenedor en función de su dirección IP única y de los metadatos y estadísticas que se devuelven para dicha tarea.

Temas

- Versión 4 del punto de conexión de los metadatos de tareas de Amazon ECS para tareas en Fargate
- Versión 3 del punto de conexión de los metadatos de tareas de Amazon ECS para tareas en Fargate

Versión 4 del punto de conexión de los metadatos de tareas de Amazon ECS para tareas en Fargate

> **Important**
>
> Si utiliza tareas de Amazon ECS alojadas en instancias de Amazon EC2, consulte Amazon ECS task metadata endpoint.

A partir de la versión `1.4.0` de la plataforma de Fargate, se ha incluido una variable de entorno con el nombre `ECS_CONTAINER_METADATA_URI_V4` en cada contenedor de una tarea. Cuando consulta la versión 4 del punto de enlace de metadatos de tareas, hay diversos metadatos y estadísticas de Docker disponibles para las tareas.

La versión 4 del punto de enlace de metadatos de tareas funciona como la versión 3, pero proporciona metadatos de red adicionales para los contenedores y las tareas. También hay otras métricas de red que están disponibles al consultar los puntos de enlace `/stats`.

El punto de conexión de metadatos de tareas está habilitado de forma predeterminada para todas las tareas de Amazon ECS que se ejecutan en AWS Fargate y utilizan la versión de la plataforma `1.4.0` o una posterior.

> **Note**
>
> Para no tener que crear nuevas versiones de los puntos de enlace de metadatos de tareas en el futuro, se pueden agregar otros metadatos a la salida de la versión 4. No vamos a eliminar ningún metadato existente ni a modificar los nombres de los campos de metadatos.

Rutas de la versión 4 del punto de conexión de metadatos de tareas de Fargate

Los siguientes puntos de enlace de metadatos de tarea están disponibles para los contenedores:

`${ECS_CONTAINER_METADATA_URI_V4}`

Esta ruta devuelve metadatos del contenedor.

`${ECS_CONTAINER_METADATA_URI_V4}/task`

Esta ruta devuelve metadatos de la tarea, incluso una lista de los nombres e ID de contenedor de todos los contenedores asociados a la tarea. Para obtener más información sobre la respuesta de este punto de enlace, consulte Respuesta JSON para la versión 4 de los metadatos de tareas de Amazon ECS para tareas en Fargate.

`${ECS_CONTAINER_METADATA_URI_V4}/stats`

Esta ruta devuelve estadísticas de Docker del contenedor de Docker. Para obtener más información acerca de cada una de las estadísticas devueltas, consulte ContainerStats en la documentación del API de Docker.

> **Note**
>
> Las tareas de Amazon ECS alojadas en AWS Fargate requieren que el contenedor se ejecute durante ~1 segundo antes de devolver las estadísticas del contenedor.

`${ECS_CONTAINER_METADATA_URI_V4}/task/stats`

Esta ruta devuelve estadísticas de Docker de todos los contenedores asociados a la tarea. Para obtener más información acerca de cada una de las estadísticas devueltas, consulte ContainerStats en la documentación del API de Docker.

> **Note**
>
> Las tareas de Amazon ECS alojadas en AWS Fargate requieren que el contenedor se ejecute durante ~1 segundo antes de devolver las estadísticas del contenedor.

Respuesta JSON para la versión 4 de los metadatos de tareas de Amazon ECS para tareas en Fargate

La respuesta JSON de punto de enlace de metadatos de tareas (`${ECS_CONTAINER_METADATA_URI_V4}/task`) devuelve los siguientes metadatos.

`Cluster`

El nombre de recurso de Amazon (ARN) o el nombre corto del clúster de Amazon ECS al que pertenece la tarea.

`VPCID`

El ID de VPC de la instancia de contenedor de Amazon EC2. Este campo solo aparece para las instancias de Amazon EC2.

> **Note**
>
> Los metadatos de `VPCID` solo se incluyen cuando se utiliza la versión `1.63.1` o una posterior del agente de contenedores de Amazon ECS.

`TaskARN`

Nombre de recurso de Amazon (ARN) de la tarea al que pertenece el contenedor.

`Family`

La familia de la definición de tareas de Amazon ECS para la tarea.

`Revision`

La revisión de la definición de tareas de Amazon ECS para la tarea.

`DesiredStatus`

El estado deseado para la tarea de Amazon ECS.

`KnownStatus`

El estado conocido para la tarea de Amazon ECS.

`Limits`

Los límites de recursos especificados en niveles de tarea, por ejemplo, CPU (expresado en vCPU) y memoria. Este parámetro se omite si no se definen límites de recurso.

`PullStartedAt`

La marca temporal del momento en que comenzó la primera extracción de la imagen del contenedor.

`PullStoppedAt`

La marca temporal del momento en que finalizó la última extracción de la imagen del contenedor.

`AvailabilityZone`

La zona de disponibilidad donde está la tarea.

> **Note**
>
> Los metadatos de la zona de disponibilidad solo están disponibles para las tareas de Fargate que utilicen la versión 1.4 o posterior (Linux) o 1.0.0 (Windows) de la plataforma.

`LaunchType`

El tipo de lanzamiento que usa la tarea. Cuando se utilizan proveedores de capacidad de clúster, indica si la tarea está utilizando la infraestructura Fargate o EC2.

> **Note**
>
> Este metadato `LaunchType` solo se incluye cuando se utiliza la versión `1.45.0` o posterior (Linux) o 1.0.0 o posterior (Windows) del agente de contenedor de Amazon ECS o una posterior.

`EphemeralStorageMetrics`

El tamaño reservado y el uso actual del almacenamiento efímero de esta tarea.

> **Note**
>
> Fargate reserva espacio en el disco. Solo lo usa Fargate. No se cobra por esto. No se muestra en estas métricas. Sin embargo, puede ver este almacenamiento adicional en otras herramientas, como `df`.

`Utilized`

El uso actual del almacenamiento efímero (en MiB) de esta tarea.

`Reserved`

El almacenamiento efímero reservado (en MiB) de esta tarea. El tamaño de almacenamiento efímero no se puede cambiar en una tarea en ejecución. Puede especificar el objeto `ephermalStorage` en la definición de tarea para cambiar la cantidad de almacenamiento efímero. `ephermalStorage` se especifica en GiB, no en MiB. `ephermalStorage` y `EphemeralStorageMetrics` solo están disponibles en la versión 1.4.0 o posterior de la plataforma Fargate de Linux.

`Containers`

Una lista de metadatos de contenedor para cada contenedor asociado con la tarea.

`DockerId`

El ID de Docker para el contenedor.

Cuando usa Fargate, el ID es un hexadecimal de 32 dígitos seguido de un número de 10 dígitos.

`Name`

El nombre del contenedor tal y como se especifica en la definición de tarea.

`DockerName`

El nombre del contenedor suministrado a Docker. El agente de contenedor de Amazon ECS genera un nombre único para el contenedor a fin de evitar conflictos de nombre cuando se ejecutan en una sola instancia varias copias de la misma definición de tareas.

`Image`

La imagen para el contenedor.

`ImageID`

El resumen SHA-256 del manifiesto de la imagen. Este es el resumen que se puede usar para extraer la imagen utilizando el formato `repository-url/image@sha256:digest`.

`Ports`

Los puertos expuestos para el contenedor. Este parámetro se omite si no hay puertos expuestos.

`Labels`

Cualquier etiqueta aplicada al contenedor. Este parámetro se omite si no hay etiquetas aplicadas.

`DesiredStatus`

El estado deseado para el contenedor procedente de Amazon ECS.

`KnownStatus`

El estado conocido para el contenedor procedente de Amazon ECS.

`ExitCode`

El código de salida para el contenedor. Este parámetro se omite si el contenedor no ha salido.

`Limits`

Los límites de recursos especificados en el nivel de contenedor, por ejemplo, CPU (expresado en unidades de CPU) y memoria. Este parámetro se omite si no se definen límites de recurso.

`CreatedAt`

La marca de hora para cuando el contenedor se creó. Este parámetro se omite si el contenedor no ha se ha creado aún.

`StartedAt`

La marca de hora para cuando el contenedor se inició. Este parámetro se omite si el contenedor no ha se ha iniciado aún.

`FinishedAt`

La marca de hora para cuando el contenedor se detuvo. Este parámetro se omite si el contenedor no ha se ha detenido aún.

`Type`

El tipo del contenedor. Los contenedores que se especifican en su definición de tarea son de tipo `NORMAL`. Puede hacer caso omiso de otros tipos de contenedores que utiliza el agente de contenedor de Amazon ECS para el aprovisionamiento de recursos para tareas internas.

`LogDriver`

El controlador de registros que utiliza el contenedor.

> **Note**
>
> Este metadato `LogDriver` solo se incluye cuando se utiliza la versión `1.45.0` del agente de contenedor de Linux de Amazon ECS o una posterior.

`LogOptions`

Opciones del controlador de registros definidas para el contenedor.

> **Note**
>
> Este metadato `LogOptions` solo se incluye cuando se utiliza la versión `1.45.0` del agente de contenedor de Linux de Amazon ECS o una posterior.

`ContainerARN`

El nombre de recurso de Amazon (ARN) del contenedor.

> **Note**
>
> Este metadato `ContainerARN` solo se incluye cuando se utiliza la versión `1.45.0` del agente de contenedor de Linux de Amazon ECS o una posterior.

`Networks`

La información de red del contenedor, como la dirección IP y el modo de red. Este parámetro se omite si no se define ninguna información de red.

`Snapshotter`

El snapshotter que utilizó containerd para descargar la imagen de este contenedor. Los valores válidos son `overlayfs`, que es el predeterminado, y `soci` se utilizan cuando se carga en diferido con un índice SOCI. Este parámetro solo está disponible para las tareas que se ejecuten en la versión de la plataforma `1.4.0` de Linux.

`RestartCount`

La cantidad de veces que se reinició el contenedor.

> **Note**
>
> Los metadatos `RestartCount` se incluyen solo si hay una política de reinicio habilitada para el contenedor. Para obtener más información, consulte Reinicio de contenedores individuales en tareas de Amazon ECS con políticas de reinicio de contenedores.

`ClockDrift`

La información sobre la diferencia entre la hora de referencia y la hora del sistema. Esta capacidad utiliza el Servicio de sincronización temporal de Amazon para medir la precisión del reloj y proporcionar el límite de error de reloj de los contenedores. Para más información, consulte Set the time for your Linux instance en la Guía del usuario de Amazon EC2 para instancias de Linux.

`ReferenceTime`

La base de la precisión del reloj. Amazon ECS utiliza el estándar global de hora universal coordinada (UTC) a través de NTP, por ejemplo `2021-09-07T16:57:44Z`.

`ClockErrorBound`

La medida del error de reloj, definida como el desplazamiento a UTC. Este error es la diferencia en milisegundos entre la hora de referencia y la hora del sistema.

`ClockSynchronizationStatus`

Indica si el intento de sincronización más reciente entre la hora del sistema y la hora de referencia se ha realizado correctamente.

Los valores válidos son `SYNCHRONIZED` y `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

La marca temporal para cuando el `DesiredStatus` de la tarea pasó a `STOPPED`. Esto ocurre cuando un contenedor esencial pasa a `STOPPED`.

Ejemplos de la versión 4 de los metadatos de tareas de Amazon ECS para tareas en Fargate

En los siguientes ejemplos, se muestran resultados de ejemplo de los puntos de enlace de metadatos de tareas para tareas de Amazon ECS que se ejecutan en AWS Fargate.

En el contenedor, puede utilizar curl seguido por el punto de conexión de metadatos de tarea para consultar el punto de conexión, por ejemplo `curl ${ECS_CONTAINER_METADATA_URI_V4}/task`.

Ejemplo de respuesta de metadatos del contenedor

Cuando se consulta el punto de enlace `${ECS_CONTAINER_METADATA_URI_V4}`, solo se devuelven los metadatos relacionados con el propio contenedor. El siguiente es un ejemplo de salida.

```
{
    "DockerId": "cd189a933e5849daa93386466019ab50-2495160603",
    "Name": "curl",
    "DockerName": "curl",
    "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
    "ImageID":
 "sha256:25f3695bedfb454a50f12d127839a68ad3caf91e451c1da073db34c542c4d2cb",
    "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-west-2:111122223333:cluster/
default",
        "com.amazonaws.ecs.container-name": "curl",
```

```
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/default/
cd189a933e5849daa93386466019ab50",
        "com.amazonaws.ecs.task-definition-family": "curltest",
        "com.amazonaws.ecs.task-definition-version": "2"
    },
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 10,
        "Memory": 128
    },
    "CreatedAt": "2020-10-08T20:09:11.44527186Z",
    "StartedAt": "2020-10-08T20:09:11.44527186Z",
    "Type": "NORMAL",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "192.0.2.3"
            ],
            "AttachmentIndex": 0,
            "MACAddress": "0a:de:f6:10:51:e5",
            "IPv4SubnetCIDRBlock": "192.0.2.0/24",
            "DomainNameServers": [
                "192.0.2.2"
            ],
            "DomainNameSearchList": [
                "us-west-2.compute.internal"
            ],
            "PrivateDNSName": "ip-10-0-0-222.us-west-2.compute.internal",
            "SubnetGatewayIpv4Address": "192.0.2.0/24"
        }
    ],
    "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/05966557-
f16c-49cb-9352-24b3a0dcd0e1",
    "LogOptions": {
        "awslogs-create-group": "true",
        "awslogs-group": "/ecs/containerlogs",
        "awslogs-region": "us-west-2",
        "awslogs-stream": "ecs/curl/cd189a933e5849daa93386466019ab50"
    },
    "LogDriver": "awslogs",
    "Snapshotter": "overlayfs"
```

```
}
```

Ejemplos de la versión 4 de los metadatos de tareas de Amazon ECS para tareas en Fargate

Cuando se consulta el punto de enlace `${ECS_CONTAINER_METADATA_URI_V4}/task`, se devuelven los metadatos relacionados con la tarea de la que forma parte el contenedor. El siguiente es un ejemplo de salida.

```
{
  "Cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/MyEmptyCluster",
  "TaskARN": "arn:aws:ecs:us-east-1:123456789012:task/MyEmptyCluster/
bfa2636268144d039771334145e490c5",
  "Family": "sample-fargate",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Limits": {
    "CPU": 0.25,
    "Memory": 512
  },
  "PullStartedAt": "2023-07-21T15:45:33.532811081Z",
  "PullStoppedAt": "2023-07-21T15:45:38.541068435Z",
  "AvailabilityZone": "us-east-1d",
  "Containers": [
    {
      "DockerId": "bfa2636268144d039771334145e490c5-1117626119",
      "Name": "curl-image",
      "DockerName": "curl-image",
      "Image": "curlimages/curl",
      "ImageID":
 "sha256:daf3f46a2639c1613b25e85c9ee4193af8a1d538f92483d67f9a3d7f21721827",
      "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/
MyEmptyCluster",
        "com.amazonaws.ecs.container-name": "curl-image",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-east-1:123456789012:task/
MyEmptyCluster/bfa2636268144d039771334145e490c5",
        "com.amazonaws.ecs.task-definition-family": "sample-fargate",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": { "CPU": 128 },
```

```
      "CreatedAt": "2023-07-21T15:45:44.91368314Z",
      "StartedAt": "2023-07-21T15:45:44.91368314Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": ["172.31.42.189"],
          "AttachmentIndex": 0,
          "MACAddress": "0e:98:9f:33:76:d3",
          "IPv4SubnetCIDRBlock": "172.31.32.0/20",
          "DomainNameServers": ["172.31.0.2"],
          "DomainNameSearchList": ["ec2.internal"],
          "PrivateDNSName": "ip-172-31-42-189.ec2.internal",
          "SubnetGatewayIpv4Address": "172.31.32.1/20"
        }
      ],
      "ContainerARN": "arn:aws:ecs:us-east-1:123456789012:container/MyEmptyCluster/
bfa2636268144d039771334145e490c5/da6cccf7-1178-400c-afdf-7536173ee209",
      "Snapshotter": "overlayfs"
    },
    {
      "DockerId": "bfa2636268144d039771334145e490c5-3681984407",
      "Name": "fargate-app",
      "DockerName": "fargate-app",
      "Image": "public.ecr.aws/docker/library/httpd:latest",
      "ImageID":
 "sha256:8059bdd0058510c03ae4c808de8c4fd2c1f3c1b6d9ea75487f1e5caa5ececa02",
      "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/
MyEmptyCluster",
        "com.amazonaws.ecs.container-name": "fargate-app",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-east-1:123456789012:task/
MyEmptyCluster/bfa2636268144d039771334145e490c5",
        "com.amazonaws.ecs.task-definition-family": "sample-fargate",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": { "CPU": 2 },
      "CreatedAt": "2023-07-21T15:45:44.954460255Z",
      "StartedAt": "2023-07-21T15:45:44.954460255Z",
      "Type": "NORMAL",
      "Networks": [
        {
```

```
          "NetworkMode": "awsvpc",
          "IPv4Addresses": ["172.31.42.189"],
          "AttachmentIndex": 0,
          "MACAddress": "0e:98:9f:33:76:d3",
          "IPv4SubnetCIDRBlock": "172.31.32.0/20",
          "DomainNameServers": ["172.31.0.2"],
          "DomainNameSearchList": ["ec2.internal"],
          "PrivateDNSName": "ip-172-31-42-189.ec2.internal",
          "SubnetGatewayIpv4Address": "172.31.32.1/20"
        }
      ],
      "ContainerARN": "arn:aws:ecs:us-east-1:123456789012:container/MyEmptyCluster/
bfa2636268144d039771334145e490c5/f65b461d-aa09-4acb-a579-9785c0530cbc",
      "Snapshotter": "overlayfs"
    }
  ],
  "LaunchType": "FARGATE",
  "ClockDrift": {
    "ClockErrorBound": 0.446931,
    "ReferenceTimestamp": "2023-07-21T16:09:17Z",
    "ClockSynchronizationStatus": "SYNCHRONIZED"
  },
  "EphemeralStorageMetrics": {
    "Utilized": 261,
    "Reserved": 20496
  }
}
```

Ejemplo de respuesta de estadísticas de las tareas

Cuando se consulta el punto de enlace `${ECS_CONTAINER_METADATA_URI_V4}/task/stats`, se devuelven las métricas de red relacionadas con la tarea de la que forma parte el contenedor. El siguiente es un ejemplo de salida.

```
{
  "3d1f891cded94dc795608466cce8ddcf-464223573": {
    "read": "2020-10-08T21:24:44.938937019Z",
    "preread": "2020-10-08T21:24:34.938633969Z",
    "pids_stats": {},
    "blkio_stats": {
      "io_service_bytes_recursive": [
        {
          "major": 202,
```

```
          "minor": 26368,
          "op": "Read",
          "value": 638976
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Write",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Sync",
          "value": 638976
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Async",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Total",
          "value": 638976
        }
      ],
      "io_serviced_recursive": [
        {
          "major": 202,
          "minor": 26368,
          "op": "Read",
          "value": 12
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Write",
          "value": 0
        },
        {
          "major": 202,
```

```
          "minor": 26368,
          "op": "Sync",
          "value": 12
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Async",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Total",
          "value": 12
        }
      ],
      "io_queue_recursive": [],
      "io_service_time_recursive": [],
      "io_wait_time_recursive": [],
      "io_merged_recursive": [],
      "io_time_recursive": [],
      "sectors_recursive": []
    },
    "num_procs": 0,
    "storage_stats": {},
    "cpu_stats": {
      "cpu_usage": {
        "total_usage": 1137691504,
        "percpu_usage": [
          696479228,
          441212276,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
```

```
          0
        ],
        "usage_in_kernelmode": 80000000,
        "usage_in_usermode": 810000000
      },
      "system_cpu_usage": 9393210000000,
      "online_cpus": 2,
      "throttling_data": {
        "periods": 0,
        "throttled_periods": 0,
        "throttled_time": 0
      }
    },
    "precpu_stats": {
      "cpu_usage": {
        "total_usage": 1136624601,
        "percpu_usage": [
          695639662,
          440984939,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0
        ],
        "usage_in_kernelmode": 80000000,
        "usage_in_usermode": 810000000
      },
      "system_cpu_usage": 9373330000000,
      "online_cpus": 2,
      "throttling_data": {
        "periods": 0,
        "throttled_periods": 0,
        "throttled_time": 0
      }
    },
```

```
    "memory_stats": {
      "usage": 6504448,
      "max_usage": 8458240,
      "stats": {
        "active_anon": 1675264,
        "active_file": 557056,
        "cache": 651264,
        "dirty": 0,
        "hierarchical_memory_limit": 536870912,
        "hierarchical_memsw_limit": 9223372036854772000,
        "inactive_anon": 0,
        "inactive_file": 3088384,
        "mapped_file": 430080,
        "pgfault": 11034,
        "pgmajfault": 5,
        "pgpgin": 8436,
        "pgpgout": 7137,
        "rss": 4669440,
        "rss_huge": 0,
        "total_active_anon": 1675264,
        "total_active_file": 557056,
        "total_cache": 651264,
        "total_dirty": 0,
        "total_inactive_anon": 0,
        "total_inactive_file": 3088384,
        "total_mapped_file": 430080,
        "total_pgfault": 11034,
        "total_pgmajfault": 5,
        "total_pgpgin": 8436,
        "total_pgpgout": 7137,
        "total_rss": 4669440,
        "total_rss_huge": 0,
        "total_unevictable": 0,
        "total_writeback": 0,
        "unevictable": 0,
        "writeback": 0
      },
      "limit": 9223372036854772000
    },
    "name": "curltest",
    "id": "3d1f891cded94dc795608466cce8ddcf-464223573",
    "networks": {
      "eth1": {
        "rx_bytes": 2398415937,
```

```
        "rx_packets": 1898631,
        "rx_errors": 0,
        "rx_dropped": 0,
        "tx_bytes": 1259037719,
        "tx_packets": 428002,
        "tx_errors": 0,
        "tx_dropped": 0
      }
    },
    "network_rate_stats": {
      "rx_bytes_per_sec": 43.298687872232854,
      "tx_bytes_per_sec": 215.39347269466413
    }
  }
}
```

Versión 3 del punto de conexión de los metadatos de tareas de Amazon ECS para tareas en Fargate

> **Important**
>
> El punto de conexión de la versión 3 de los metadatos de la tarea ya no se mantiene activamente. Le recomendamos que actualice el punto de conexión de metadatos de la tarea versión 4 para obtener la información más reciente del punto de conexión de metadatos. Para obtener más información, consulte the section called "Versión 4 del punto de conexión de metadatos de tareas para tareas en Fargate".

A partir de la versión `1.1.0` de la plataforma de Fargate, se ha incluido una variable de entorno con el nombre `ECS_CONTAINER_METADATA_URI` en cada contenedor de una tarea. Cuando consulta la versión 3 del punto de enlace de metadatos de tarea, están disponibles diversos metadatos de tarea y estadísticas de Docker para las tareas.

La característica de punto de enlace de metadatos de tareas está habilitada de forma predeterminada para las tareas de Amazon ECS alojadas en Fargate que utilizan la versión `1.1.0` de la plataforma o una posterior. Para obtener más información, consulte Versiones de la plataforma Fargate para Amazon ECS.

Rutas del punto de conexión de metadatos de tareas para tareas en Fargate

Los siguientes puntos de enlace de API están disponibles para los contenedores:

`${ECS_CONTAINER_METADATA_URI}`

Esta ruta devuelve JSON de metadatos para el contenedor.

`${ECS_CONTAINER_METADATA_URI}/task`

Esta ruta devuelve JSON de metadatos para la tarea, incluida una lista de los nombres e ID del contenedor de todos los contenedores asociados con la tarea. Para obtener más información sobre la respuesta de este punto de enlace, consulte Respuesta JSON para la versión 3 de los metadatos de tareas de Amazon ECS para tareas en Fargate.

`${ECS_CONTAINER_METADATA_URI}/stats`

Esta ruta devuelve JSON de estadísticas de Docker para el ID de contenedor de Docker específico. Para obtener más información acerca de cada una de las estadísticas devueltas, consulte ContainerStats en la documentación del API de Docker.

`${ECS_CONTAINER_METADATA_URI}/task/stats`

Esta ruta devuelve JSON de estadísticas de Docker de todos los contenedores asociados con la tarea. Para obtener más información acerca de cada una de las estadísticas devueltas, consulte ContainerStats en la documentación del API de Docker.

Respuesta JSON para la versión 3 de los metadatos de tareas de Amazon ECS para tareas en Fargate

La siguiente información se devuelve desde la respuesta de JSON (`${ECS_CONTAINER_METADATA_URI}/task`) de punto de enlace de metadatos de tarea.

`Cluster`

El nombre de recurso de Amazon (ARN) o el nombre corto del clúster de Amazon ECS al que pertenece la tarea.

`TaskARN`

Nombre de recurso de Amazon (ARN) de la tarea al que pertenece el contenedor.

`Family`

La familia de la definición de tareas de Amazon ECS para la tarea.

`Revision`

La revisión de la definición de tareas de Amazon ECS para la tarea.

`DesiredStatus`

El estado deseado para la tarea de Amazon ECS.

`KnownStatus`

El estado conocido para la tarea de Amazon ECS.

`Limits`

Los límites de recursos especificados en el nivel de tarea, por ejemplo, CPU (expresado en vCPU) y memoria. Este parámetro se omite si no se definen límites de recurso.

`PullStartedAt`

La marca temporal del momento en que comenzó la primera extracción de la imagen del contenedor.

`PullStoppedAt`

La marca temporal del momento en que finalizó la última extracción de la imagen del contenedor.

`AvailabilityZone`

La zona de disponibilidad donde está la tarea.

> **Note**
>
> Los metadatos de la zona de disponibilidad solo están disponibles para las tareas de Fargate que utilicen la versión 1.4 o posterior (Linux) o 1.0.0 o posterior (Windows) de la plataforma.

`Containers`

Una lista de metadatos de contenedor para cada contenedor asociado con la tarea.

`DockerId`

El ID de Docker para el contenedor.

`Name`

El nombre del contenedor tal y como se especifica en la definición de tarea.

`DockerName`

El nombre del contenedor suministrado a Docker. El agente de contenedor de Amazon ECS genera un nombre único para el contenedor a fin de evitar conflictos de nombre cuando se ejecutan en una sola instancia varias copias de la misma definición de tareas.

`Image`

La imagen para el contenedor.

`ImageID`

El resumen SHA-256 del manifiesto de la imagen. Este es el resumen que se puede usar para extraer la imagen utilizando el formato `repository-url/image@sha256:digest`.

`Ports`

Los puertos expuestos para el contenedor. Este parámetro se omite si no hay puertos expuestos.

`Labels`

Cualquier etiqueta aplicada al contenedor. Este parámetro se omite si no hay etiquetas aplicadas.

`DesiredStatus`

El estado deseado para el contenedor procedente de Amazon ECS.

`KnownStatus`

El estado conocido para el contenedor procedente de Amazon ECS.

`ExitCode`

El código de salida para el contenedor. Este parámetro se omite si el contenedor no ha salido.

`Limits`

Los límites de recursos especificados en el nivel de contenedor, por ejemplo, CPU (expresado en unidades de CPU) y memoria. Este parámetro se omite si no se definen límites de recurso.

`CreatedAt`

La marca de hora para cuando el contenedor se creó. Este parámetro se omite si el contenedor no ha se ha creado aún.

`StartedAt`

La marca de hora para cuando el contenedor se inició. Este parámetro se omite si el contenedor no ha se ha iniciado aún.

`FinishedAt`

La marca de hora para cuando el contenedor se detuvo. Este parámetro se omite si el contenedor no ha se ha detenido aún.

`Type`

El tipo del contenedor. Los contenedores que se especifican en su definición de tarea son de tipo `NORMAL`. Puede hacer caso omiso de otros tipos de contenedores que utiliza el agente de contenedor de Amazon ECS para el aprovisionamiento de recursos para tareas internas.

`Networks`

La información de red del contenedor, como la dirección IP y el modo de red. Este parámetro se omite si no se define ninguna información de red.

`ClockDrift`

La información sobre la diferencia entre la hora de referencia y la hora del sistema. Esto se aplica al sistema operativo Linux. Esta capacidad utiliza el Servicio de sincronización temporal de Amazon para medir la precisión del reloj y proporcionar el límite de error de reloj de los contenedores. Para más información, consulte Set the time for your Linux instance en la Guía del usuario de Amazon EC2 para instancias de Linux.

`ReferenceTime`

La base de la precisión del reloj. Amazon ECS utiliza el estándar global de hora universal coordinada (UTC) a través de NTP, por ejemplo `2021-09-07T16:57:44Z`.

`ClockErrorBound`

La medida del error de reloj, definida como el desplazamiento a UTC. Este error es la diferencia en milisegundos entre la hora de referencia y la hora del sistema.

`ClockSynchronizationStatus`

Indica si el intento de sincronización más reciente entre la hora del sistema y la hora de referencia se ha realizado correctamente.

Los valores válidos son `SYNCHRONIZED` y `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

La marca temporal para cuando el `DesiredStatus` de la tarea pasó a `STOPPED`. Esto ocurre cuando un contenedor esencial pasa a `STOPPED`.

Ejemplos de la versión 3 de los metadatos de tareas de Amazon ECS para tareas en Fargate

La siguiente respuesta JSON es para una tarea de contenedor único.

```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
```

```
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
```

```
}
```

Introspección de contenedor de Amazon ECS

El agente de contenedor de Amazon ECS proporciona una operación de la API para recopilar detalles acerca de la instancia de contenedor en la que se ejecuta el agente y las tareas asociadas que se ejecutan en esa instancia. Puede utilizar el comando curl desde la instancia de contenedor para consultar al agente de contenedor de Amazon ECS (puerto 51678) y mostrar los metadatos de la instancia de contenedor o la información de las tareas.

> **Important**
>
> La instancia de contenedor debe disponer de un rol de IAM que permita obtener acceso a Amazon ECS para poder recuperar los metadatos. Para obtener más información, consulte Rol de IAM de instancia de contenedor de Amazon ECS.

Para ver los metadatos de la instancia de contenedor, inicie sesión en su instancia de contenedor mediante SSH y ejecute el comando siguiente. Los metadatos incluyen el ID de la instancia de contenedor, el clúster de Amazon ECS en el que está registrada la instancia de contenedor e información de la versión del agente de contenedor de Amazon ECS.

```
curl -s http://localhost:51678/v1/metadata | python3 -mjson.tool
```

Salida:

```
{
    "Cluster": "cluster_name",
    "ContainerInstanceArn": "arn:aws:ecs:region:aws_account_id:container-
instance/cluster_name/container_instance_id",
    "Version": "Amazon ECS Agent - v1.30.0 (02ff320c)"
}
```

Para ver información acerca de todas las tareas que se están ejecutando en una instancia de contenedor, inicie sesión en su instancia de contenedor mediante SSH y ejecute el comando siguiente:

```
curl http://localhost:51678/v1/tasks
```

Salida:

```
{
    "Tasks": [
        {
            "Arn": "arn:aws:ecs:us-west-2:012345678910:task/default/example5-58ff-46c9-
ae05-543f8example",
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Family": "hello_world",
            "Version": "8",
            "Containers": [
                {
          "DockerId":
 "9581a69a761a557fbfce1d0f6745e4af5b9dbfb86b6b2c5c4df156f1a5932ff1",
          "DockerName": "ecs-hello_world-8-mysql-fcae8ac8f9f1d89d8301",
          "Name": "mysql",
          "CreatedAt": "2023-10-08T20:09:11.44527186Z",
          "StartedAt": "2023-10-08T20:09:11.44527186Z",
          "ImageID":
 "sha256:2ae34abc2ed0a22e280d17e13f9c01aaf725688b09b7a1525d1a2750e2c0d1de"
        },
        {
          "DockerId":
 "bf25c5c5b2d4dba68846c7236e75b6915e1e778d31611e3c6a06831e39814a15",
          "DockerName": "ecs-hello_world-8-wordpress-e8bfddf9b488dff36c00",
          "Name": "wordpress"
        }
    ]
}
    ]
}
```

Puede ver información para una tarea particular que se está ejecutando en una instancia de contenedor. Para especificar una tarea o contenedor específico, adjunte uno de los siguientes elementos a la solicitud:

- El ARN de tarea (`?taskarn=`*`task_arn`*)
- El ID de Docker para un contenedor (`?dockerid=`*`docker_id`*)

Para obtener información de tarea con el ID de Docker de un contenedor, inicie sesión en su instancia de contenedor mediante SSH y ejecute el comando siguiente.

> **Note**
>
> Las versiones de los agentes de contenedor de Amazon ECS anteriores a la versión 1.14.2 requieren ID de contenedor de Docker completos para la API de introspección, no la versión corta que se muestra con docker ps. Puede obtener el ID de Docker completo de un contenedor ejecutando el comando docker ps --no-trunc en la instancia de contenedor.

```
curl http://localhost:51678/v1/tasks?dockerid=79c796ed2a7f
```

Salida:

```
{
    "Arn": "arn:aws:ecs:us-west-2:012345678910:task/default/e01d58a8-151b-40e8-
bc01-22647b9ecfec",
    "Containers": [
        {
            "DockerId":
 "79c796ed2a7f864f485c76f83f3165488097279d296a7c05bd5201a1c69b2920",
            "DockerName": "ecs-nginx-efs-2-nginx-9ac0808dd0afa495f001",
            "Name": "nginx",
            "CreatedAt": "2023-10-08T20:09:11.44527186Z",
            "StartedAt": "2023-10-08T20:09:11.44527186Z",
            "ImageID":
 "sha256:2ae34abc2ed0a22e280d17e13f9c01aaf725688b09b7a1525d1a2750e2c0d1de"
        }
    ],
    "DesiredStatus": "RUNNING",
    "Family": "nginx-efs",
    "KnownStatus": "RUNNING",
    "Version": "2"
}
```

Identificación de comportamientos no autorizados mediante la supervisión en tiempo de ejecución

Amazon GuardDuty es un servicio de detección de amenazas que ayuda a proteger las cuentas, los contenedores, las cargas de trabajo y los datos de su entorno de AWS. Mediante modelos de machine learning (ML) y capacidades de detección de anomalías y amenazas, GuardDuty supervisa continuamente los diferentes orígenes de registro y la actividad en tiempo de ejecución para identificar y priorizar los posibles riesgos de seguridad y actividades maliciosas en su entorno.

La supervisión en tiempo de ejecución de GuardDuty protege las cargas de trabajo que se ejecutan en instancias de contenedor de Fargate y EC2 mediante la supervisión continua de la actividad de registro y red de AWS para identificar comportamientos malintencionados o no autorizados. La supervisión en tiempo de ejecución utiliza un agente de seguridad de GuardDuty ligero y totalmente administrado que analiza el comportamiento en el host (como el acceso a los archivos, la ejecución de procesos y las conexiones de red). Esto abarca problemas como el escalado de privilegios, el uso de credenciales expuestas, la comunicación con direcciones IP malintencionadas, los dominios y la presencia de malware en las cargas de trabajo de contenedores e instancias de Amazon EC2. Para obtener más información, consulte GuardDuty Runtime Monitoring en la Guía del usuario de GuardDuty.

El administrador de seguridad habilita la supervisión en tiempo de ejecución para una o varias cuentas en AWS Organizations para GuardDuty. También selecciona si GuardDuty implementa automáticamente el agente de seguridad de GuardDuty cuando utiliza Fargate. Todos los clústeres se protegen automáticamente y GuardDuty administra el agente de seguridad en su nombre.

También puede configurar manualmente el agente de seguridad de GuardDuty en los siguientes casos:

- Al usar instancias de contenedor de EC2
- Necesita control granular para habilitar la supervisión en tiempo de ejecución a nivel de clúster

Para usar la supervisión en tiempo de ejecución, debe configurar los clústeres que están protegidos e instalar y administrar el agente de seguridad de GuardDuty en las instancias de contenedor de EC2.

Cómo funciona la supervisión en tiempo de ejecución con Amazon ECS

La supervisión en tiempo de ejecución utiliza un agente de seguridad de GuardDuty ligero que supervisa la actividad de la carga de trabajo de Amazon ECS para ver cómo las aplicaciones solicitan, obtienen acceso y consumen los recursos subyacentes del sistema.

Para las tareas de Fargate, el agente de seguridad de GuardDuty funciona como un contenedor sidecar para cada tarea.

En el caso de las instancias de contenedor de EC2, el agente de seguridad de GuardDuty se ejecuta como un proceso en la instancia.

El agente de seguridad de GuardDuty recopila datos de los siguientes recursos y, a continuación, los envía a GuardDuty para que los procese. Puede ver los resultados en la consola de GuardDuty. También puede enviarlos a otros Servicios de AWS, como AWS Security Hub o un proveedor de seguridad externo, para su agregación y corrección. Para obtener información sobre cómo ver y administrar los resultados, consulte Managing Amazon GuardDuty findings en la Guía del usuario de Amazon GuardDuty.

- Respuestas de las siguientes llamadas a la API de Amazon ECS:
  - DescribeClusters

    Los parámetros de respuesta incluyen la etiqueta de supervisión en tiempo de ejecución (si la etiqueta está configurada) al utilizar la opción `--include TAGS`.
  - DescribeTasks

    Para el tipo de lanzamiento de Fargate, los parámetros de respuesta incluyen el contenedor sidecar de GuardDuty.
  - ListAccountSettings

    Los parámetros de respuesta incluyen la configuración de la cuenta de supervisión en tiempo de ejecución, que establece el administrador de seguridad.
- Los datos de introspección del agente de contenedor. Para obtener más información, consulte Introspección de contenedor de Amazon ECS.
- El punto de conexión de metadatos de la tarea para el tipo de lanzamiento:
  - Versión 4 del punto de conexión de metadatos de tareas de Amazon ECS
  - Versión 4 del punto de conexión de los metadatos de tareas de Amazon ECS para tareas en Fargate

Consideraciones

Tenga en cuenta lo siguiente al utilizar a supervisión en tiempo de ejecución:

- La supervisión en tiempo de ejecución tiene un costo asociado. Para obtener más información, consulte Precios de Amazon GuardDuty.
- La supervisión en tiempo de ejecución no se admite en Amazon ECS Anywhere.
- La supervisión en tiempo de ejecución no es compatible con el sistema operativo Windows.
- Cuando usa Amazon ECS Exec en Fargate, debe especificar el nombre del contenedor, ya que el agente de seguridad de GuardDuty funciona como un contenedor sidecar.
- No puede usar Amazon ECS Exec en el contenedor sidecar del agente de seguridad de GuardDuty.
- El usuario de IAM que controla la supervisión en tiempo de ejecución en los clústeres debe tener los permisos de IAM adecuados para el etiquetado. Para obtener más información, consulte IAM tutorial: Define permissions to access AWS resources based on tags en la Guía del usuario de IAM.
- Las tareas de Fargate deben tener un rol de ejecución de tareas. Este rol concede a las tareas permiso para recuperar, actualizar y administrar el agente de seguridad de GuardDuty (que está almacenado en un repositorio privado de Amazon ECR) en su nombre.

Uso de los recursos

La etiqueta que agregue al clúster se tendrá en cuenta para la cuota de etiquetas del clúster.

El contenedor sidecar del agente de GuardDuty no se incluye en la cuota de contenedores por definición de tarea.

Como ocurre con la mayoría de los programas de seguridad, GuardDuty tiene una ligera sobrecarga. Para obtener información sobre los límites de memoria de Fargate, consulte CPU and memory limits en la Guía del usuario de GuardDuty. Para obtener información sobre los límites de memoria de Amazon EC2, consulte CPU and memory limit for GuardDuty agent.

Supervisión en tiempo de ejecución de las cargas de trabajo de Fargate para Amazon ECS

Si utiliza instancias de contenedor de EC2, debe configurar manualmente la supervisión en tiempo de ejecución. Para obtener más información, consulte Supervisión en tiempo de ejecución de las cargas de trabajo de EC2 en Amazon ECS.

Puede utilizar GuardDuty para administrar el agente de seguridad en las instancias de contenedor. Esta opción solo está disponible para Fargate. Esta opción (administración de agente de GuardDuty) está disponible en GuardDuty

Cuando utiliza la administración de agentes de GuardDuty, GuardDuty lleva a cabo las siguientes operaciones:

- Crea puntos de conexión de VPC para GuardDuty para cada VPC que aloja un clúster.
- Recupera e instala el último agente de seguridad de GuardDuty como contenedor sidecar en todas las nuevas tareas independientes de Fargate y en las nuevas implementaciones de servicios.

  La implementación de un nuevo servicio se produce la primera vez que se lanza un servicio o cuando actualiza un servicio existente con la opción de forzar una nueva implementación.

Activación de la supervisión en tiempo de ejecución para Amazon ECS

Puede configurar GuardDuty para administrar automáticamente el agente de seguridad de todos los clústeres de Fargate.

Requisitos previos

A continuación se indican los requisitos previos para utilizar la supervisión en tiempo de ejecución:

- Para Linux, la versión de la plataforma Fargate debe ser `1.4.0` o posterior.
- Roles de IAM y permisos para Amazon ECS:
  - Las tareas de Fargate deben tener un rol de ejecución de tareas. Este rol concede a las tareas permiso para recuperar, actualizar y administrar el agente de seguridad de GuardDuty en su nombre. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.
  - Usted controla la supervisión en tiempo de ejecución de un clúster con una etiqueta predefinida. Si sus políticas de acceso restringen el acceso en función de las etiquetas, debe conceder

permisos explícitos a los usuarios de IAM para etiquetar los clústeres. Para obtener más información, consulte IAM tutorial: Define permissions to access AWS resources based on tags en la Guía del usuario de IAM.

- Conexión al repositorio de Amazon ECR:

  El agente de seguridad de GuardDuty se almacena en un repositorio de Amazon ECR. Todas las tareas independientes y de servicio deben tener acceso al repositorio. Puede utilizar una de las siguientes opciones:

  - Para las tareas en subredes públicas, puede utilizar una dirección IP pública para la tarea o puede crear un punto de conexión de VPC para Amazon ECR en la subred en la que se ejecuta la tarea. Para obtener más información, consulte Puntos de conexión de VCP de la interfaz de Amazon ECR (AWS PrivateLink) en la Guía del usuario de Amazon Elastic Container Registry.
  - Para las tareas en subredes privadas, puede utilizar una puerta de enlace de traducción de direcciones de red (NAT) o crear un punto de conexión de VPC para Amazon ECR en la subred en la que se ejecuta la tarea.

    Para obtener más información, consulte Subred privada y puerta de enlace NAT.

- Debe tener el rol `AWSServiceRoleForAmazonGuardDuty` de GuardDuty. Para obtener más información, consulte Service-linked role permissions for GuardDuty en la Guía del usuario de Amazon GuardDuty.
- El usuario raíz debe poder acceder a todos los archivos que desee proteger con la supervisión en tiempo de ejecución. Si ha cambiado manualmente los permisos de un archivo, debe configurarlo en `755`.

A continuación, se indican los requisitos previos para utilizar la supervisión en tiempo de ejecución en instancias de contenedor de EC2:

- Debe usar la versión `20230929` o posterior de Amazon ECS-AMI.
- Debe ejecutar el agente de Amazon ECS en la versión `1.77` o posterior en las instancias de contenedor.
- Debe usar la versión del kernel `5.10` o posterior.
- Para obtener información sobre los sistemas operativos y las arquitecturas de Linux compatibles, consulte Which operating models and workloads does GuardDuty Runtime Monitoring support.

- Puede usar Systems Manager para administrar sus instancias de contenedor. Para obtener más información, consulte Configuración de Systems Manager para instancias de EC2 en la Guía del usuario de AWS Systems Manager Session Manager.

Procedimiento

La supervisión en tiempo de ejecución se activa en GuardDuty. Para obtener información sobre cómo habilitar la característica, consulte Enabling Runtime Monitoring en la Guía del usuario de Amazon GuardDuty.

Adición de la supervisión en tiempo de ejecución a las tareas existentes de Fargate para Amazon ECS

Al activar la supervisión en tiempo de ejecución, todas las nuevas tareas independientes y las nuevas implementaciones de servicios del clúster se protegen automáticamente. Para preservar la restricción de inmutabilidad, las tareas existentes no se ven afectadas.

Requisitos previos

1. Activación de la supervisión en tiempo de ejecución Para obtener más información, consulte Activación de la supervisión en tiempo de ejecución para Amazon ECS.
2. Las tareas de Fargate deben tener un rol de ejecución de tareas. Este rol concede a las tareas permiso para recuperar, actualizar y administrar el agente de seguridad de GuardDuty en su nombre. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

Procedimiento

- Para proteger una tarea de forma inmediata, debe llevar a cabo una de las siguientes acciones:
  - En el caso de las tareas independientes, detenga las tareas y, a continuación, inícielas. Para obtener más información, consulte Detención de una tarea de Amazon ECS y Ejecución de una aplicación como tarea de Amazon ECS.
  - En el caso de las tareas que forman parte de un servicio, actualice el servicio con la opción “Forzar una nueva implementación”. Para obtener más información, consulte Actualización de un servicio de Amazon ECS.

Eliminación de la supervisión en tiempo de ejecución de un clúster de Amazon ECS

Es posible que desee excluir determinados clústeres de la protección, por ejemplo, los clústeres que utilice para realizar pruebas. Esto hace que GuardDuty haga las siguientes operaciones en los recursos del clúster:

- Ya no implementa el agente de seguridad de GuardDuty en las nuevas tareas independientes de Fargate ni en nuevas implementaciones de servicios.

  Para preservar la restricción de inmutabilidad, las tareas e implementaciones existentes con supervisión en tiempo de ejecución no se ven afectadas.
- Deja de facturar y deja de aceptar eventos de tiempo de ejecución para las tareas.

Procedimiento

Siga estos pasos para eliminar la supervisión en tiempo de ejecución de un clúster.

1. Utilice la consola o la AWS CLI de Amazon ECS para configurar la clave de la etiqueta `GuardDutyManaged` del clúster en `false`. Para obtener más información, consulte Updating a cluster o Working with tags using the CLI or API. Utilice los siguientes valores para la etiqueta.

   > **Note**
   >
   > La clave y el valor distinguen entre mayúsculas y minúsculas y deben coincidir exactamente con las cadenas.

   Clave = `GuardDutyManaged`, valor = `false`
2. Elimine el punto de conexión de VPC de GuardDuty del clúster. Para obtener más información acerca de cómo eliminar los puntos de conexión de VPC, consulte Delete an interface endpoint en la Guía del usuario de AWS PrivateLink.

Eliminación de la supervisión en tiempo de ejecución para Amazon ECS de una cuenta

Si ya no desea utilizar la supervisión en tiempo de ejecución, deshabilite la característica en GuardDuty. Para obtener información sobre cómo deshabilitar la característica, consulte Enabling Runtime Monitoring en la Guía del usuario de Amazon GuardDuty.

GuardDuty lleva a cabo las siguientes operaciones:

- Elimina los puntos de conexión de VPC de GuardDuty para cada VPC que aloja un clúster.
- Ya no implementa el agente de seguridad de GuardDuty en las nuevas tareas independientes de Fargate ni en nuevas implementaciones de servicios.

  Para preservar la restricción de inmutabilidad, las tareas e implementaciones existentes no se ven afectadas hasta que se detienen, se replican o se escalan.
- Deja de facturar y deja de aceptar eventos de tiempo de ejecución para las tareas.

Supervisión en tiempo de ejecución de las cargas de trabajo de EC2 en Amazon ECS

Utilice esta opción cuando use instancias de EC2 para su capacidad o cuando necesite un control detallado de la supervisión en tiempo de ejecución del clúster en Fargate.

Para aprovisionar los clústeres para la supervisión en tiempo de ejecución, agregue una etiqueta predefinida.

Para las instancias de contenedor de EC2, debe descargar, instalar y administrar el agente de seguridad de GuardDuty.

Para Fargate, GuardDuty administra el agente de seguridad en su nombre.

Activación de la supervisión en tiempo de ejecución para Amazon ECS

Puede activar la supervisión en tiempo de ejecución de los clústeres con instancias de EC2 o cuando necesite un control detallado de la supervisión en tiempo de ejecución del clúster en Fargate.

A continuación se indican los requisitos previos para utilizar la supervisión en tiempo de ejecución:

- Para Linux, la versión de la plataforma Fargate debe ser `1.4.0` o posterior.
- Roles de IAM y permisos para Amazon ECS:
  - Las tareas de Fargate deben tener un rol de ejecución de tareas. Este rol concede a las tareas permiso para recuperar, actualizar y administrar el agente de seguridad de GuardDuty en su nombre. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

  - Usted controla la supervisión en tiempo de ejecución de un clúster con una etiqueta predefinida. Si sus políticas de acceso restringen el acceso en función de las etiquetas, debe conceder permisos explícitos a los usuarios de IAM para etiquetar los clústeres. Para obtener más información, consulte IAM tutorial: Define permissions to access AWS resources based on tags en la Guía del usuario de IAM.
- Conexión al repositorio de Amazon ECR:

  El agente de seguridad de GuardDuty se almacena en un repositorio de Amazon ECR. Todas las tareas independientes y de servicio deben tener acceso al repositorio. Puede utilizar una de las siguientes opciones:

  - Para las tareas en subredes públicas, puede utilizar una dirección IP pública para la tarea o puede crear un punto de conexión de VPC para Amazon ECR en la subred en la que se ejecuta la tarea. Para obtener más información, consulte Puntos de conexión de VCP de la interfaz de Amazon ECR (AWS PrivateLink) en la Guía del usuario de Amazon Elastic Container Registry.
  - Para las tareas en subredes privadas, puede utilizar una puerta de enlace de traducción de direcciones de red (NAT) o crear un punto de conexión de VPC para Amazon ECR en la subred en la que se ejecuta la tarea.

    Para obtener más información, consulte Subred privada y puerta de enlace NAT.
- Debe tener el rol `AWSServiceRoleForAmazonGuardDuty` de GuardDuty. Para obtener más información, consulte Service-linked role permissions for GuardDuty en la Guía del usuario de Amazon GuardDuty.
- El usuario raíz debe poder acceder a todos los archivos que desee proteger con la supervisión en tiempo de ejecución. Si ha cambiado manualmente los permisos de un archivo, debe configurarlo en `755`.

A continuación, se indican los requisitos previos para utilizar la supervisión en tiempo de ejecución en instancias de contenedor de EC2:

- Debe usar la versión `20230929` o posterior de Amazon ECS-AMI.
- Debe ejecutar el agente de Amazon ECS en la versión `1.77` o posterior en las instancias de contenedor.
- Debe usar la versión del kernel `5.10` o posterior.
- Para obtener información sobre los sistemas operativos y las arquitecturas de Linux compatibles, consulte Which operating models and workloads does GuardDuty Runtime Monitoring support.

- Puede usar Systems Manager para administrar sus instancias de contenedor. Para obtener más información, consulte Configuración de Systems Manager para instancias de EC2 en la Guía del usuario de AWS Systems Manager Session Manager.

La supervisión en tiempo de ejecución se activa en GuardDuty. Para obtener información sobre cómo habilitar la característica, consulte Enabling Runtime Monitoring en la Guía del usuario de Amazon GuardDuty.

Adición de la supervisión en tiempo de ejecución a un clúster de Amazon ECS

Configure la supervisión en tiempo de ejecución para el clúster y, a continuación, instale el agente de seguridad de GuardDuty en las instancias de contenedor de EC2.

Requisitos previos

1. Activación de la supervisión en tiempo de ejecución Para obtener más información, consulte Activación de la supervisión en tiempo de ejecución para Amazon ECS.
2. Usted controla la supervisión en tiempo de ejecución de un clúster con una etiqueta predefinida. Si sus políticas de acceso restringen el acceso en función de las etiquetas, debe conceder permisos explícitos a los usuarios de IAM para etiquetar los clústeres. Para obtener más información, consulte IAM tutorial: Define permissions to access AWS resources based on tags en la Guía del usuario de IAM.

Procedimiento

Haga las siguientes operaciones para agregar la supervisión en tiempo de ejecución a un clúster.

1. Cree un punto de conexión de VPC para GuardDuty para cada clúster de VPC. Para obtener más información, consulte Creating Amazon VPC endpoint manually en la Guía del usuario de GuardDuty.
2. Configuración de las instancias de contenedor de EC2.

    a. Actualice el agente de Amazon ECS a la versión `1.77` o posterior en las instancias de contenedor de EC2 del clúster. Para obtener más información, consulte Actualización del agente de contenedor de Amazon ECS.

    b. Instale el agente de seguridad de GuardDuty en las instancias de contenedor de EC2 del clúster. Para obtener más información, consulte Administración manual del agente de seguridad en una instancia de Amazon EC2 en la Guía del usuario de GuardDuty.

Todas las tareas nuevas y existentes y las implementaciones se protegen inmediatamente porque el agente de seguridad de GuardDuty se ejecuta como un proceso en la instancia de contenedor de EC2.

3. Utilice la consola o la AWS CLI de Amazon ECS para configurar la clave de la etiqueta `GuardDutyManaged` del clúster en `true`. Para obtener más información, consulte Updating a cluster o Working with tags using the CLI or API. Utilice los siguientes valores para la etiqueta.

> **Note**
>
> La clave y el valor distinguen entre mayúsculas y minúsculas y deben coincidir exactamente con las cadenas.

Clave = `GuardDutyManaged`, valor = `true`

Adición de la supervisión en tiempo de ejecución a las tareas existentes de Amazon ECS

Al activar la supervisión en tiempo de ejecución, todas las nuevas tareas independientes y las nuevas implementaciones de servicios del clúster se protegen automáticamente. Para preservar la restricción de inmutabilidad, las tareas existentes no se ven afectadas.

Requisitos previos

- Activación de la supervisión en tiempo de ejecución Para obtener más información, consulte Activación de la supervisión en tiempo de ejecución para Amazon ECS.

Procedimiento

- Para proteger una tarea de forma inmediata, debe llevar a cabo una de las siguientes acciones:
  - En el caso de las tareas independientes, detenga las tareas y, a continuación, inícielas. Para obtener más información, consulte Detención de una tarea de Amazon ECS y Ejecución de una aplicación como tarea de Amazon ECS.
  - En el caso de las tareas que forman parte de un servicio, actualice el servicio con la opción “Forzar una nueva implementación”. Para obtener más información, consulte Actualización de un servicio de Amazon ECS.

Eliminación de la supervisión en tiempo de ejecución de un clúster de Amazon ECS

Puede eliminar la supervisión en tiempo de ejecución de un clúster. Esto hace que GuardDuty deje de supervisar todos los recursos del clúster.

Para eliminar la supervisión en tiempo de ejecución de un clúster

1. Utilice la consola o la AWS CLI de Amazon ECS para configurar la clave de la etiqueta `GuardDutyManaged` del clúster en `false`. Para obtener más información, consulte Updating a cluster o Working with tags using the CLI or API.

   > **Note**
   >
   > La clave y el valor distinguen entre mayúsculas y minúsculas y deben coincidir exactamente con las cadenas.

   Clave = `GuardDutyManaged`, valor = `false`

2. Desinstale el agente de seguridad de GuardDuty en las instancias de contenedor de EC2 del clúster.

   Para obtener más información, consulte Desinstalación manual del agente de seguridad en la Guía del usuario de GuardDuty.

3. Elimine el punto de conexión de VPC de GuardDuty para cada VPC del clúster. Para obtener más información acerca de cómo eliminar los puntos de conexión de VPC, consulte Delete an interface endpoint en la Guía del usuario de AWS PrivateLink.

Actualización del agente de seguridad de GuardDuty en las instancias de contenedor de Amazon ECS

Para obtener información sobre cómo actualizar el agente de seguridad de GuardDuty en sus instancias de contenedor EC2, consulte Updating GuardDuty security agent en la Guía del usuario de Amazon GuardDuty.

Eliminación de la supervisión en tiempo de ejecución para Amazon ECS de una cuenta

Si ya no desea utilizar la supervisión en tiempo de ejecución, deshabilite la característica en GuardDuty. Para obtener información sobre cómo deshabilitar la característica, consulte Enabling Runtime Monitoring en la Guía del usuario de Amazon GuardDuty.

Elimine la supervisión en tiempo de ejecución de todos los clústeres. Para obtener más información, consulte Eliminación de la supervisión en tiempo de ejecución de un clúster de Amazon ECS.

Solución de problemas de la supervisión en tiempo de ejecución

Es posible que tenga que solucionar algún problema o comprobar que la supervisión en tiempo de ejecución esté activada y en ejecución en sus tareas y contenedores.

Temas

- ¿Cómo puedo saber si la supervisión en tiempo de ejecución está activa en mi cuenta?
- ¿Cómo puedo saber si la supervisión en tiempo de ejecución está activa en un clúster?
- ¿Cómo puedo saber si el agente de seguridad de GuardDuty se está ejecutando en una tarea de Fargate?
- ¿Cómo puedo saber si el agente de seguridad de GuardDuty se está ejecutando en una instancia de contenedor de EC2?
- ¿Qué ocurre cuando no existe ningún rol de ejecución de tareas para una tarea en ejecución en el clúster?
- ¿Cómo puedo saber si tengo los permisos correctos para etiquetar clústeres para la supervisión en tiempo de ejecución?
- ¿Qué ocurre cuando no hay ninguna conexión con Amazon ECR?
- ¿Cómo soluciono los errores de falta de memoria en mis tareas de Fargate después de activar la supervisión en tiempo de ejecución?

¿Cómo puedo saber si la supervisión en tiempo de ejecución está activa en mi cuenta?

En la consola de Amazon ECS, la información se encuentra en la página Configuración de la cuenta.

También puede ejecutar `list-account-settings` con la opción `effective-settings`.

```
aws ecs list-account-settings --effective-settings
```

Output

La configuración con el nombre `guardDutyActivate` y el valor establecidos en on indican que la cuenta está configurada. Debe comprobar con el administrador de GuardDuty si la administración es automática o manual.

```
{
    "setting": {
        "name": "guardDutyActivate",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type": "aws-managed"
    }
}
```

¿Cómo puedo saber si la supervisión en tiempo de ejecución está activa en un clúster?

Puede ver las estadísticas de la cobertura en la consola de GuardDuty. La información correspondiente a los recursos de Amazon ECS asociados a la cuenta propia o a las cuentas de miembro representa el porcentaje de los clústeres en buen estado sobre todos los clústeres en la Región de AWS seleccionada. Esto incluye la cobertura para los clústeres que usan los tipos de lanzamiento Fargate y EC2. Para obtener más información, consulte Revisión de las estadísticas de cobertura en la Guía del usuario de Amazon GuardDuty.

¿Cómo puedo saber si el agente de seguridad de GuardDuty se está ejecutando en una tarea de Fargate?

Para las tareas de Fargate, el agente de seguridad de GuardDuty funciona como un contenedor sidecar.

En la consola de Amazon ECS, el sidecar se muestra en Contenedores en la página de detalles de la tarea.

Puede ejecutar `describe-tasks` y buscar el contenedor con el nombre establecido en `aws-gd-agent` y LastStatus establecido en `RUNNING`.

El siguiente ejemplo muestra los resultados para el clúster predeterminado de la tarea `aws:ecs:us-east-1:123456789012:task/0b69d5c0-d655-4695-98cd-5d2d5EXAMPLE`.

```
aws ecs describe-tasks --cluster default --tasks aws:ecs:us-
east-1:123456789012:task/0b69d5c0-d655-4695-98cd-5d2d5EXAMPLE
```

Output

El contenedor denominado `gd-agent` está en estado `RUNNING`.

```
"containers": [
      {
        "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/4df26bb4-
f057-467b-a079-96167EXAMPLE",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/0b69d5c0-
d655-4695-98cd-5d2d5EXAMPLE",
        "lastStatus": "RUNNING",
        "healthStatus": "UNKNOWN",
        "memory": "string",
        "name": "aws-gd-agent"
      }
    ]
```

¿Cómo puedo saber si el agente de seguridad de GuardDuty se está ejecutando en una instancia de contenedor de EC2?

Ejecute el siguiente comando para ver el estado:

```
sudo systemctl status amazon-guardduty-agent
```

El archivo de registro se encuentra en la siguiente ubicación:

```
/var/log/amzn-guardduty-agent
```

¿Qué ocurre cuando no existe ningún rol de ejecución de tareas para una tarea en ejecución en el clúster?

Para las tareas de Fargate, la tarea se inicia sin el contenedor sidecar del agente de seguridad de GuardDuty. El panel de GuardDuty mostrará que a la tarea le falta protección en el panel de estadísticas de cobertura.

¿Cómo puedo saber si tengo los permisos correctos para etiquetar clústeres para la supervisión en tiempo de ejecución?

Para etiquetar un clúster, debe disponer de la acción `ecs:TagResource` tanto para `CreateCluster` como para`UpdateCluster`.

A continuación se muestra un fragmento de una política de ejemplo.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ecs:TagResource"
      ],
      "Resource": "*",
      "Condition": {
         "StringEquals": {
             "ecs:CreateAction" : "CreateCluster",
             "ecs:CreateAction" : "UpdateCluster",
          }
       }
    }
  ]
}
```

¿Qué ocurre cuando no hay ninguna conexión con Amazon ECR?

Para las tareas de Fargate, la tarea se inicia sin el contenedor sidecar del agente de seguridad de GuardDuty. El panel de GuardDuty mostrará que a la tarea le falta protección en el panel de estadísticas de cobertura.

¿Cómo soluciono los errores de falta de memoria en mis tareas de Fargate después de activar la supervisión en tiempo de ejecución?

El agente de seguridad de GuardDuty es un proceso ligero. Sin embargo, aún así el proceso consume recursos de acuerdo con el tamaño de la carga de trabajo. Recomendamos utilizar herramientas de seguimiento de recursos de contenedor, como Información de contenedores de Amazon CloudWatch, para organizar las implementaciones de GuardDuty en su clúster. Estas herramientas lo ayudan a descubrir el perfil de consumo del agente de seguridad de GuardDuty para

sus aplicaciones. A continuación, puede ajustar el tamaño de la tarea de Fargate, si es necesario, para evitar posibles problemas de falta de memoria.

Supervisión de los contenedores de Amazon ECS con ECS Exec

Con Amazon ECS Exec, puede interactuar directamente con los contenedores sin necesidad de interactuar primero con el sistema operativo del contenedor host, abrir puertos entrantes o administrar claves SSH. Puede utilizar ECS Exec para ejecutar comandos en un contenedor u obtener un shell en un contenedor que se ejecute en una instancia de Amazon EC2 o en AWS Fargate. Esto facilita la recopilación de información de diagnóstico y la rápida resolución de problemas. Por ejemplo, en un contexto de desarrollo, puede utilizar ECS Exec para interactuar fácilmente con varios procesos de los contenedores y solucionar problemas de las aplicaciones. Y, en escenarios de producción, puede utilizarlo para obtener acceso a los contenedores con el fin de depurar problemas.

Para ejecutar comandos en un contenedor de Linux o Windows en ejecución, puede utilizar ECS Exec desde la API de Amazon ECS, la AWS Command Line Interface (AWS CLI), el SDK de AWS o la CLI de AWS Copilot. Para obtener más información acerca del uso de ECS Exec, así como una explicación en video de cómo se utiliza la CLI de AWS Copilot, consulte la Documentación de GitHub sobre Copilot.

ECS Exec también se puede utilizar para mantener políticas de control de acceso más estrictas. Al activar selectivamente esta función, puede controlar quiénes puede ejecutar comandos y en qué tareas pueden hacerlo. Con un registro de cada comando y su salida, puede utilizar ECS Exec para ver qué tareas se ejecutaron y CloudTrail para auditar quiénes accedieron a un contenedor.

Consideraciones

Para este tema, debería estar familiarizado con los siguientes aspectos relacionados con la utilización de ECS Exec:

- ECS Exec actualmente no es compatible a través de la AWS Management Console.

  En la página de información de los clústeres se muestra la clave de registro y la clave administrada por el cliente de AWS KMS utilizada para encriptar los datos entre el cliente local y el contenedor.
- Es posible que ECS Exec no funcione como se esperaba cuando se ejecuta en sistemas operativos no compatibles con Systems Manager. Para obtener más información acerca de los sistemas operativos compatibles, consulte Tipos de sistemas operativos en la AWS Systems Manager Guía del usuario.

- ECS Exec es compatible con las tareas que se ejecutan en la siguiente infraestructura:
  - Contenedores de Linux en Amazon EC2 en cualquier AMI optimizada para Amazon ECS, incluida Bottlerocket
  - Contenedores de Linux y Windows en instancias externas (Amazon ECS Anywhere)
  - Contenedores de Linux y Windows en AWS Fargate
  - Los contenedores de Windows de Amazon EC2 de las siguientes AMI optimizadas para Amazon ECS de Windows (con la versión `1.56` del agente de contenedor o una posterior):
    - AMI de Windows Server 2022 Full optimizada para Amazon ECS
    - AMI de Windows Server 2022 Core optimizada para Amazon ECS
    - AMI de Windows Server 2019 Full optimizada para Amazon ECS
    - AMI de Windows Server 2019 Core optimizada para Amazon ECS
    - AMI de Windows Server 20H2 Core optimizada para Amazon ECS
- Si ha configurado un proxy HTTP para la tarea, defina la variable de entorno `NO_PROXY` como `"NO_PROXY=169.254.169.254,169.254.170.2"` para evitar el proxy para los metadatos de la instancia de EC2 y el tráfico del rol de IAM. Si no configura la variable de entorno `NO_PROXY`, es posible que se produzcan errores al recuperar los metadatos de la instancia o las credenciales del rol de IAM del punto de conexión de los metadatos del contenedor. Si se establece la variable de entorno en `NO_PROXY` como se recomienda, se filtran los metadatos y el tráfico de IAM para que las solicitudes `169.254.169.254 and 169.254.170.2` no pasen por el proxy `HTTP`.
- ECS Exec y Amazon VPC
  - Si utiliza puntos de conexión de VPC de interfaz de Amazon ECS, debe crear los puntos de conexión de la interfaz de Amazon VPC para Systems Manager Session Manager (`ssmmessages`). Para obtener más información sobre los puntos de conexión de VPC de Systems Manager, consulte Utilización de AWS PrivateLink para configurar un punto de conexión de VPC para Session Manager en la Guía del usuario de AWS Systems Manager.
  - Si utiliza puntos de conexión de Amazon VPC de la interfaz con Amazon ECS y está utilizando AWS KMS key para el cifrado, debe crear el punto de conexión de Amazon VPC de la interfaz para AWS KMS key. Para obtener más información, consulte Conexión a AWS KMS key a través de un punto de conexión de VPC en la Guía para desarrolladores de AWS Key Management Service.
  - Cuando tenga tareas que se ejecuten en instancias de Amazon EC2, utilice el modo de red `awsvpc`. Si no tiene acceso a Internet (por ejemplo, no están configuradas para usar una puerta de enlace de NAT), debe crear los puntos de conexión de Amazon VPC de interfaz para el Administrador de sesiones de Systems Manager (`ssmmessages`). Para obtener más

información sobre las consideraciones del modo de red `awsvpc`, consulte Consideraciones. Para obtener más información sobre los puntos de conexión de VPC de Systems Manager, consulte Utilización de AWS PrivateLink para configurar un punto de conexión de VPC para Session Manager en la Guía del usuario de AWS Systems Manager.

- ECS Exec y SSM
  - Cuando un usuario ejecuta comandos en un contenedor mediante ECS Exec, estos comandos se ejecutan como usuario `root`. SSM Agent y sus procesos secundarios se ejecutan como raíz incluso cuando se especifica un ID de usuario para el contenedor.
  - SSM Agent requiere que se pueda escribir en el sistema de archivos del contenedor para crear los directorios y archivos requeridos. Por lo tanto, no se admite que el sistema de archivos raíz se haga de solo lectura mediante el parámetro de definición de tareas `readonlyRootFilesystem` o cualquier otro método.
  - Si bien es posible iniciar sesiones de SSM fuera de la acción `execute-command`, esto hace que las sesiones no se registren ni se cuenten para el límite de sesiones. Recomendamos limitar este acceso a través de la denegación de la acción `ssm:start-session` mediante una política de IAM. Para obtener más información, consulte Limitación del acceso a la acción Iniciar sesión.
- Las siguientes características funcionan como un contenedor sidecar. Por lo tanto, debe especificar el nombre del contenedor en el que se ejecutará el comando.
  - Supervisión en tiempo de ejecución
  - Service Connect
- Los usuarios pueden ejecutar todos los comandos que están disponibles en el contexto del contenedor. Las siguientes acciones pueden dar lugar a procesos huérfanos y zombis: terminar el proceso principal del contenedor, terminar el agente de comando y eliminar dependencias. Para limpiar los procesos zombies, recomendamos agregar el indicador `initProcessEnabled` a la definición de tareas.
- ECS Exec usa parte de la CPU y de la memoria. Deberá tenerlo en cuenta al especificar las asignaciones de recursos de memoria y CPU en la definición de tareas.
- Debe utilizar la version de AWS CLI `1.22.3` o posterior o la version de AWS CLI `2.3.6` o posterior. Para obtener más información sobre como actualizar la AWS CLI, consulte Instalación o actualización de la última versión de la AWS CLI en la Guía del usuario de AWS Command Line Interface versión 2.
- Solo puede tener una sesión de ECS Exec por espacio de nombres de ID de proceso (PID). Si comparte un espacio de nombres PID en una tarea, solo puede iniciar sesiones de ECS Exec en un contenedor.

- La sesión de ECS Exec tiene un tiempo de espera de inactividad de 20 minutos. Este valor no se puede cambiar.
- No se puede activar ECS Exec para tareas existentes. Solo se puede activar para tareas nuevas.
- No puede utilizar ECS Exec cuando usa `run-task` para lanzar una tarea en un clúster que utiliza escalado administrado con ubicación asíncrona (lanzar una tarea sin instancia).
- No se puede ejecutar ECS Exec en contenedores de Microsoft Nano Server.

Requisitos previos

Antes de comenzar a utilizar ECS Exec, asegúrese de haber completado estas acciones:

- Instalar y configurar la AWS CLI. Para obtener más información, consulte Introducción a la AWS CLI.
- Instale el complemento de Session Manager para la AWS CLI. Para obtener más información, consulte Instalación del complemento de Session Manager para la AWS CLI.
- Debe utilizar un rol de tarea con los permisos adecuados para ECS Exec. Para obtener más información, consulte Rol de IAM de tarea.
- ECS Exec tiene requisitos de versión dependiendo de si las tareas están alojadas en Amazon EC2 o enAWS Fargate:
  - Si utiliza Amazon EC2, debe usar una AMI optimizada para Amazon ECS que se publicó después del 20 de enero de 2021, con un agente versión 1.50.2 o superior. Para obtener más información, consulte AMI optimizadas para Amazon ECS.
  - Si utiliza AWS Fargate, debe utilizar la versión `1.4.0` de la plataforma o una superior (Linux) o `1.0.0` (Windows). Para obtener más información, consulte Versiones de la plataforma de AWS Fargate.

Arquitectura

ECS Exec utiliza AWS Systems Manager Session Manager (SSM) para establecer una conexión con el contenedor en ejecución y las políticas de AWS Identity and Access Management (IAM) para controlar el acceso a comandos en ejecución en un contenedor en ejecución. Esto se logra mediante el montaje vinculado de los binarios de SSM Agent necesarios en el contenedor. El agente de Amazon ECS o de AWS Fargate es responsable de iniciar el agente principal de SSM dentro del contenedor junto con el código de la aplicación. Para obtener más información, consulte Systems Manager Session Manager.

Puede auditar qué usuarios accedieron al contenedor mediante el evento `ExecuteCommand` en AWS CloudTrail y registrar cada comando (y su resultado) en Amazon S3 o en los registros de Amazon CloudWatch. Para cifrar datos entre el cliente local y el contenedor con su propia clave de cifrado, debe proporcionar la clave de AWS Key Management Service (AWS KMS).

Uso de ECS Exec

Cambios opcionales en la definición de tareas

Si establece el parámetro de definición de tareas `initProcessEnabled` en `true`, se inicia el proceso de inicio dentro del contenedor. Esto elimina cualquier proceso secundario zombi del agente de SSM encontrado. A continuación, puede ver un ejemplo.

```
{
    "taskRoleArn": "ecsTaskRole",
    "networkMode": "awsvpc",
    "requiresCompatibilities": [
        "EC2",
        "FARGATE"
    ],
    "executionRoleArn": "ecsTaskExecutionRole",
    "memory": ".5 gb",
    "cpu": ".25 vcpu",
    "containerDefinitions": [
        {
            "name": "amazon-linux",
            "image": "amazonlinux:latest",
            "essential": true,
            "command": ["sleep","3600"],
            "linuxParameters": {
                "initProcessEnabled": true
            }
        }
    ],
    "family": "ecs-exec-task"
}
```

Activación de ECS Exec para las tareas y los servicios

Para activar la característica ECS Exec para los servicios y las tareas independientes, puede especificar el indicador `--enable-execute-command` cuando se utiliza uno de los siguientes comandos de la AWS CLI: `create-service`, `update-service`, `start-task` o `run-task`.

Por ejemplo, si ejecuta el siguiente comando, la característica ECS Exec se activa para un servicio recién creado que se ejecuta en Fargate. Para obtener más información acerca de la creación de servicios, consulte create-service.

```
aws ecs create-service \
    --cluster cluster-name \
    --task-definition task-definition-name \
    --enable-execute-command \
    --service-name service-name \
    --launch-type FARGATE \
     --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
 \
    --desired-count 1
```

Después de activar ECS Exec para una tarea, puede ejecutar el siguiente comando para confirmar que la tarea está lista para utilizarse. Si la propiedad `lastStatus` del `ExecuteCommandAgent` se aparece como `RUNNING` y la propiedad `enableExecuteCommand`se establece en `true`, su tarea está lista.

```
aws ecs describe-tasks \
    --cluster cluster-name \
    --tasks task-id
```

El siguiente fragmento de código resultante es un ejemplo de lo que puede ver.

```
{
    "tasks": [
        {
            ...
            "containers": [
                {
                    ...
                    "managedAgents": [
```

```
                        {
                            "lastStartedAt": "2021-03-01T14:49:44.574000-06:00",
                            "name": "ExecuteCommandAgent",
                            "lastStatus": "RUNNING"
                        }
                    ]
                }
            ],
            ...
            "enableExecuteCommand": true,
            ...
        }
    ]
}
```

Ejecución de comandos mediante ECS Exec

Después de confirmar que `ExecuteCommandAgent` se está ejecutando, puede abrir un shell interactivo en el contenedor mediante el siguiente comando. Si la tarea contiene varios contenedores, debe especificar el nombre del contenedor mediante el indicador `--container`. Amazon ECS solo admite la iniciación de sesiones interactivas, por lo que debe utilizar el indicador `--interactive`.

El siguiente comando ejecutará un comando `/bin/sh` interactivo en un contenedor denominado *`container-name`* para una tarea con un ID de *`task-id`*.

El *`task-id`* es el nombre de recurso de Amazon (ARN) de la tarea.

```
aws ecs execute-command --cluster cluster-name \
    --task task-id \
    --container container-name \
    --interactive \
    --command "/bin/sh"
```

Registro mediante ECS Exec

Activación del registro en las tareas y los servicios

> **Important**
>
> Para obtener más información acerca de los precios de CloudWatch, consulte Precios de CloudWatch. Amazon ECS también proporciona métricas de monitoreo sin costo adicional. Para obtener más información, consulte Supervisión de Amazon ECS con CloudWatch.

Amazon ECS proporciona una configuración predeterminada para registrar los comandos que se ejecutan con ECS Exec. Esta configuración predeterminada consiste en enviar registros a CloudWatch Logs mediante el uso del controlador de registros `awslogs` configurado en la definición de la tarea. Si desea utilizar una configuración personalizada, la AWS CLI admite un indicador `--configuration` para los comandos `create-cluster` y `update-cluster`. La imagen de contenedor requiere la instalación de `script` y `cat` para que los registros de comandos se carguen correctamente en Amazon S3 o CloudWatch Logs. Para obtener más información acerca de cómo crear clústeres, consulte create-cluster.

> **Note**
>
> Esta configuración solo maneja el registro de la sesión `execute-command`. No afecta el registro de la aplicación.

En el ejemplo siguiente, se crea un clúster y, a continuación, se registra el resultado en el LogGroup del registro de CloudWatch con el nombre `cloudwatch-log-group-name` y en el bucket de Amazon S3 con el nombre `s3-bucket-name`.

Debe utilizar una clave AWS KMS administrada por el cliente para cifrar el grupo de registro cuando establece la opción `CloudWatchEncryptionEnabled` en `true`. Para obtener información acerca de cómo cifrar el grupo de registros, consulte Cifrado de datos de registro en CloudWatch Logs mediante AWS Key Management Service en la Guía del usuario de Amazon CloudWatch Logs.

```
aws ecs create-cluster \
    --cluster-name cluster-name \
    --configuration executeCommandConfiguration="{ \
```

```
        kmsKeyId=string, \
        logging=OVERRIDE, \
        logConfiguration={ \
            cloudWatchLogGroupName=cloudwatch-log-group-name, \
            cloudWatchEncryptionEnabled=true, \
            s3BucketName=s3-bucket-name, \
            s3EncryptionEnabled=true, \
            s3KeyPrefix=demo \
        } \
    }"
```

La propiedad `logging` determina el comportamiento de la capacidad de registro de ECS Exec:

- `NONE`: el registro está desactivado.
- `DEFAULT`: los registros se envían al controlador `awslogs` configurado. Si el controlador no está configurado, no se guarda ningún registro.
- `OVERRIDE`: los registros se envían al LogGroup de Registros de Amazon CloudWatch proporcionado, al bucket de Amazon S3 o a ambos.

Se requieren permisos de IAM para generar registros en Amazon CloudWatch Logs o Amazon S3

Para habilitar el registro, el rol de tareas de Amazon ECS al que se hace referencia en la definición de tareas debe tener permisos adicionales. Estos permisos adicionales se pueden agregar como política al rol de tareas. Difieren en función de si dirige sus registros a Registros de Amazon CloudWatch o a Amazon S3.

Amazon CloudWatch Logs

La siguiente política de ejemplo agrega los permisos de Registros de Amazon CloudWatch requeridos.

JSON

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "logs:DescribeLogGroups"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
               "logs:CreateLogStream",
               "logs:DescribeLogStreams",
               "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:arn:aws:: 111122223333:log-group:/aws/
ecs/cloudwatch-log-group-name:*"
        }
   ]
}
```

Amazon S3

La siguiente política de ejemplo agrega los permisos de Amazon S3 requeridos.

JSON

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "s3:GetBucketLocation"
            ],
            "Resource": "*"
        },
        {
           "Effect": "Allow",
           "Action": [
               "s3:GetEncryptionConfiguration"
           ],
           "Resource": "arn:aws:s3:::s3-bucket-name"
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::s3-bucket-name/*"
        }
    ]
   }
```

Permisos de IAM requeridos para el cifrado con su propia AWS KMS key (clave de KMS)

De forma predeterminada, los datos transferidos entre el cliente local y el contenedor utilizan el cifrado TLS 1.2 que proporciona AWS. Para cifrar aún más los datos con su propia clave de KMS, debe crear una clave deKMS y agregar el permiso `kms:Decrypt` al rol de IAM para tareas. El contenedor utiliza este permiso para descifrar los datos. Para obtener más información acerca de cómo crear una clave de KMS, consulte Creación de claves.

Agregue la siguiente política insertada al rol de IAM para tareas que requiere los permisos de AWS KMS. Para obtener más información, consulte Permisos de ECS Exec.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource":
 "arn:aws:kms:arn:aws::111122223333:key/12345678-1234-1234-1234-123456789012"
        }
    ]
}
```

Para que los datos se cifren con su propia clave de KMS, se debe conceder al usuario o el grupo que utiliza la acción `execute-command` el permiso `kms:GenerateDataKey`.

La siguiente política de ejemplo para su usuario o grupo contiene el permiso requerido para utilizar su propia clave de KMS. Debe especificar el nombre de recurso de Amazon (ARN) de la clave de KMS.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:arn:aws::111122223333:key/key"
        }
    ]
}
```

Utilización de políticas de IAM para limitar el acceso a ECS Exec

Limite el acceso de los usuarios a la acción de la API execute-command mediante una o varias de las siguientes claves de condición de la política de IAM:

- `aws:ResourceTag/`*`clusterTagKey`*
- `ecs:ResourceTag/`*`clusterTagKey`*
- `aws:ResourceTag/`*`taskTagKey`*
- `ecs:ResourceTag/`*`taskTagKey`*
- `ecs:container-name`
- `ecs:cluster`
- `ecs:task`
- `ecs:enable-execute-command`

Con la siguiente política de IAM de ejemplo, los usuarios pueden ejecutar comandos en contenedores que se ejecutan dentro de tareas con una etiqueta que contiene una clave `environment` y un valor `development`, y en un clúster con el nombre `cluster-name`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:ExecuteCommand",
                "ecs:DescribeTasks"
            ],
            "Resource": [
                   "arn:aws:ecs:arn:aws::111122223333:task/cluster-name/*",
                   "arn:aws:ecs:arn:aws::111122223333:cluster/cluster-name"
            ],
            "Condition": {
                "StringEquals": {
                    "ecs:ResourceTag/environment": "development"
                }
            }
        }
    ]
}
```

Con la siguiente política de IAM de ejemplo, los usuarios no pueden utilizar la API `execute-command` cuando el nombre del contenedor es `production-app`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ecs:ExecuteCommand"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
```

```
                    "ecs:container-name": "production-app"
                }
            }
        }
    ]
}
```

Con la siguiente política de IAM, los usuarios solo pueden lanzar tareas cuando ECS Exec está desactivado.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:CreateService",
                "ecs:UpdateService"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:enable-execute-command": "false"
                }
            }
        }
    ]
}
```

> **Note**
>
> Debido a que la acción de la API `execute-command` solo contiene recursos de tareas y clústeres en una solicitud, solo se evalúan las etiquetas de clústeres y tareas.

Para obtener más información acerca de las claves de condición de la política de IAM, consulte Acciones, recursos y claves de condición de Amazon Elastic Container Service en la Referencia de autorizaciones de servicio.

Limitación del acceso a la acción Iniciar sesión

Si bien es posible iniciar sesiones de SSM en el contenedor fuera de ECS Exec, esto podría provocar que las sesiones no se registraran. Las sesiones iniciadas fuera de ECS Exec también cuentan para la cuota de sesiones. Recomendamos limitar este acceso directamente mediante la denegación de la acción `ssm:start-session` para las tareas de Amazon ECS que utilizan una política de IAM. Según las etiquetas que utilice, puede denegar el acceso a todas las tareas de Amazon ECS o a tareas específicas.

A continuación, se muestra una política de IAM de ejemplo que deniega el acceso a la acción `ssm:start-session` para las tareas de todas las regiones con un nombre de clúster especificado. Si lo desea, puede incluir un comodín con el *`cluster-name`*.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:StartSession",
            "Resource": [
            "arn:aws:ecs:arn:aws:;: 111122223333:task/cluster-name/*"
            ]
        }
    ]
}
```

A continuación, se muestra una política de IAM de ejemplo que deniega el acceso a la acción `ssm:start-session` sobre recursos en todas las regiones etiquetadas con clave de etiqueta `Task-Tag-Key` y el valor de etiqueta `Exec-Task`.

JSON

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:StartSession",
            "Resource": "arn:aws:ecs:*:*:task/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Task-Tag-Key": "Exec-Task"
                }
            }
        }
    ]
}
```

Para obtener ayuda con cualquier problema que pueda surgir al utilizar Amazon ECS Exec, consulte Troubleshooting issues with Exec.

Recomendaciones de AWS Compute Optimizer para Amazon ECS

AWS Compute Optimizer genera recomendaciones para los tamaños de tareas y contenedores de Amazon ECS. Para obtener más información, consulte ¿Qué es AWS Compute Optimizer? en la Guía del usuario de AWS Compute Optimizer.

Recomendaciones para los tamaños de tareas y contenedores de los servicios de Amazon ECS en AWS Fargate

AWS Compute Optimizer genera recomendaciones para los servicios de Amazon ECS en AWS Fargate. AWS Compute Optimizer recomienda el tamaño de la CPU de la tarea y de la memoria de la tarea y los tamaños de la CPU, la memoria del contenedor y la reserva de la memoria del contenedor. Estas recomendaciones se muestran en las siguientes páginas de la consola de Compute Optimizer.

- Página Recomendaciones para los servicios de Amazon ECS en Fargate
- Página Detalles de los servicios de Amazon ECS en Fargate

Para obtener más información, consulte Visualización de recomendaciones para servicios de Amazon ECS en Fargate en la Guía del usuario de AWS Compute Optimizer.

Solución de problemas de Amazon ECS

Es posible que tenga que solucionar problemas con sus equilibradores de carga, tareas, servicios o instancias de contenedor. Este capítulo le ayuda a buscar información de diagnóstico procedente del agente de contenedor de Amazon ECS, el daemon de Docker de la instancia de contenedor y el registro de eventos de servicio de la consola de Amazon ECS.

Para obtener información acerca de las tareas detenidas, consulte lo siguiente.

Acción	Más información
Solución de los errores de las tareas detenidas.	Visualización de los errores de las tareas detenidas de Amazon ECS
Visualización de los errores de las tareas detenidas.	Solución de los errores de las tareas detenidas de Amazon ECS
Revisión de los códigos de error de las tareas detenidas.	Mensajes de error de las tareas detenidas de Amazon ECS
Revisión de los errores de tarea CannotPullContainer.	Errores de tareas CannotPullContainer en Amazon ECS
Visualización de las solicitudes de rol de IAM de tareas.	Visualización de solicitudes de roles de IAM para tareas de Amazon ECS

Para obtener información acerca de los errores de servicio, consulte lo siguiente.

Acción	Más información
Visualización de los mensajes de eventos de servicio.	Visualización de los mensajes de eventos del servicio de Amazon ECS

Acción	Más información
Revisión de los mensajes de eventos de servicio.	Mensajes de eventos del servicio de Amazon ECS
Revisión de los problemas del equilibrador de carga.	Solución de problemas de los equilibradores de carga de servicio en Amazon ECS
Revisión de los problemas de escalado automático del servicio.	Solución de problemas de escalado automático de servicios en Amazon ECS

Para obtener información acerca de los errores de definición de tareas, consulte lo siguiente.

Acción	Más información
Solución del error de memoria en la definición de tareas.	Solución de errores de CPU o memoria no válidos en la definición de tareas de Amazon ECS

Para obtener información acerca de los errores del agente de Amazon ECS, consulte lo siguiente.

Acción	Más información
Visualización de los registros del agente de contenedor de Amazon ECS.	Visualización de los registros del agente de contenedor de Amazon ECS
Información sobre cómo recopilar los registros de Amazon ECS.	Recopilación de registros de contenedor con el recopilador de registros de Amazon ECS
Recuperación de los detalles de diagnóstico con el agente de Amazon ECS.	Recuperación de los detalles de diagnóstico de Amazon

Acción	Más información	
	ECS con la introspección del agente	

Para obtener información sobre los errores de Docker, consulte lo siguiente.

Acción	Más información	
Uso del diagnóstico de Docker.	Diagnósticos de Docker en Amazon ECS	
Activación del modo de depuración de Docker.	Configuración de la salida detallada desde el daemon de Docker en Amazon ECS	
Solución del error 500 de la API de Docker.	Solución del problema de Docker `API error (500): devmapper` en Amazon ECS	

Para obtener información sobre los errores de ECS Exec y Amazon ECS Anywhere, consulte lo siguiente.

Acción	Más información	
Solución de problemas de ECS Exec.	Solución de problemas de Amazon ECS Exec	
Solución de problemas de Amazon ECS Anywhere.	Solución de problemas de Amazon ECS Anywhere	

Para obtener información acerca de los problemas de limitación, consulte lo siguiente.

Acción	Más información
Información sobre las cuotas de limitación de Fargate.	Cuotas de limitación de AWS Fargate
Información sobre las prácticas recomendadas de limitación de Amazon ECS.	Gestión de los problemas de limitación de Amazon ECS

Para obtener información sobre los errores de la API, consulte lo siguiente.

Acción	Más información
Solución de los errores de la API.	Motivos de error de la API de Amazon ECS

Solución de los errores de las tareas detenidas de Amazon ECS

Cuando una tarea no se inicia, aparece un mensaje de error en la consola y en los parámetros de salida de `describe-tasks` (`stoppedReason` y `stoppedCode`).

Puede ver tareas detenidas en la consola durante una hora. Para ver las tareas detenidas, debe cambiar la opción de filtro. Para obtener más información, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

En las siguientes páginas, se proporciona información acerca de las tareas detenidas.

- Información sobre los cambios en los mensajes de error de las tareas detenidas.

  Actualizaciones de los mensajes de error de las tareas detenidas de Amazon ECS
- Visualización de las tareas detenidas para obtener información sobre la causa.

  Visualización de los errores de las tareas detenidas de Amazon ECS
- Información sobre los mensajes de error de las tareas detenidas y los posibles motivos de los errores.

  Mensajes de error de las tareas detenidas de Amazon ECS

- Información sobre cómo verificar la conectividad de las tareas detenidas y corregir los errores.

  Comprobación de la conectividad de las tareas detenidas de Amazon ECS

Actualizaciones de los mensajes de error de las tareas detenidas de Amazon ECS

A partir del 14 de junio de 2024, el equipo de Amazon ECS cambiará los mensajes de error de las tareas detenidas tal y como se describe en las siguientes tablas. `stopCode` no cambiará. Si sus aplicaciones dependen de cadenas de mensajes de error exactas, debe actualizarlas con las nuevas cadenas. Para obtener ayuda con preguntas o problemas, póngase en contacto con AWS Support.

> **Note**
>
> Recomendamos que no dependa de los mensajes de error para la automatización, ya que estos están sujetos a cambios.

CannotPullContainerError

Mensaje de error anterior	Nuevo mensaje de error	
CannotPullContainerError: respuesta de error del daemon: acceso de extracción denegado para el *repositorio* ; el repositorio no existe o puede requerir un “inicio de sesión en Docker”: denegado: Usuario: *roleARN*	• CannotPullContainerError: la tarea no puede extraer la imagen. Compruebe que el rol tenga los permisos para extraer imágenes del registro. Respuesta de error del daemon: acceso de extracción denegado para el *repositorio* ; el repositorio no existe o puede requerir un “inicio de sesión en Docker”: denegado: Usuario: *roleARN* no está autorizado a realizar: ecr:BatchGetImage en el	

Mensaje de error anterior	Nuevo mensaje de error	
	recurso: *image* porque ninguna política basada en la identidad permite la acción ecr:BatchGetImage. • CannotPullContainerError: la tarea no puede extraer la imagen. Compruebe si la imagen existe. Respuesta de error del daemon: acceso de extracció n denegado para el *repositorio* ; el repositor io no existe o puede requerir un “inicio de sesión en Docker”: denegado: se deniega el acceso solicitado al recurso.	
CannotPullContainerError: respuesta de error del daemon: obtener *imageURI*: net/http: la solicitud se canceló mientras se esperaba la conexión	CannotPullContainerError: la tarea no puede extraer la imagen. Compruebe la configuración de la red. Respuesta de error del daemon: obtener *imagen*: net/http: solicitud cancelada mientras se esperaba la conexión (se superó el Client.Timeout mientras se esperaban los encabezados)	

Mensaje de error anterior	Nuevo mensaje de error	
CannotPullContainerError: ref pull se ha vuelto a intentar 5 veces: no se pudo copiar: httpReadSeeker: no se pudo abrir: no se pudo procesar la solicitud: obtener *registry-uri* : marcar tcp <ip>:<port> tiempo de espera de e/s	CannotPullContainerError: la tarea no puede extraer *image-uri* del registro *registry-uri* . Hay un problema de conexión entre la tarea y el registro. Compruebe la configuración de red de la tarea. : no se pudo copiar: httpReadSeeker: no se pudo abrir: no se pudo procesar la solicitud: obtener *registry-uri* : marcar tcp <ip>:<port> tiempo de espera de e/s	

ResourceNotFoundException

Mensaje de error anterior	Nuevo mensaje de error	
Al buscar datos secretos desde AWS Secrets Manager en la región *region*: secreto *sercretARN* : ResourceNotFoundException: Secrets Manager no puede encontrar el secreto especificado.	ResourceNotFoundException: la tarea no puede recuperar el secreto con el ARN “*sercretARN* ” desde AWS Secrets Manager. Compruebe si el secreto existe en la región especificada. ResourceNotFoundException: al buscar datos secretos desde AWS Secrets Manager en la región *region*: secreto *sercretARN* : ResourceNotFoundException: Secrets Manager no puede	

Mensaje de error anterior	Nuevo mensaje de error	
	encontrar el secreto especific ado.	

ResourceInitializationError

Mensaje de error anterior	Nuevo mensaje de error	
ResourceInitializationError: no se pueden obtener secretos ni autenticación del registro: error en la recuperación del recurso de ejecución: no se puede recuperar la autenticación del registro ecr: la llamada de servicio se ha reintentado 3 veces: RequestError: error en el envío de la solicitud debido a lo siguiente: post "https://api.ecr.us-east-1.amazonaws.com/": marcar tcp <ip>:<port>: tiempo de espera de e/s. Compruebe la configuración de red de la tarea.	ResourceInitializationError: no se pueden extraer los secretos o la autenticación del registro: la tarea no puede extraer el secreto desde Amazon ECR: hay un problema de conexión entre la tarea y Amazon ECR. Compruebe la configuración de red de la tarea. RequestEr ror: error en el envío de la tarea debido a lo siguiente: post "https://api.ecr.us-east-1.amazonaws.com": marcar tcp <ip>:<port>: tiempo de espera de e/s	
ResourceInitializationError: no se pueden extraer los secretos ni la autenticación del registro: error en la recuperación del recurso de ejecución: no se pueden recuperar los secretos de ssm: se ha reintentado la llamada de servicio 5 veces: RequestCanceled: el contexto	ResourceInitializationError : no se pueden extraer los secretos o la autenticación del registro: no se pueden extraer los secretos desde ssm: la tarea no puede extraer los secretos desde AWS Systems Manager. Hay un problema de conexión entre la tarea y	

Mensaje de error anterior	Nuevo mensaje de error	
de la solicitud se canceló debido a lo siguiente: se ha superado la fecha límite de contexto	el almacén de parámetros de AWS Systems Manager. Compruebe la configuración de red de la tarea. RequestCanceled: el contexto de la solicitud se ha cancelado debido a lo siguiente: se ha superado la fecha límite de contexto	
ResourceInitializationError: no se pudieron descargar los archivos .env: comando de descarga del archivo: transmisión de error no vacía: RequestCanceled: el contexto de la solicitud se canceló debido a lo siguiente: se superó la fecha límite de contexto	ResourceInitializationError: error al descargar los archivos .env: la tarea no puede descargar los archivos de variables de entorno de Amazon S3. Hay un problema de conexión entre la tarea y Amazon S3. Compruebe la configuración de red de la tarea. La llamada de servicio se ha reintentado 5 veces: RequestCanceled: el contexto de la solicitud se canceló debido a lo siguiente: se ha superado la fecha límite de contexto	

Mensaje de error anterior	Nuevo mensaje de error	
ResourceInitializationError: no se pudo validar el registrador args::signal:killed	ResourceInitializationError : no se pudieron validar los argumentos del registrad or: la tarea no encuentra el grupo de registros de Amazon CloudWatch especificado en la definición de la tarea. Hay un problema de conexión entre la tarea y Amazon CloudWatch. Compruebe la configuración de red. : señal: interrumpida	
ResourceInitializationError : no se pueden extraer los secretos o la autenticación del registro: no se pudo extraer el comando. : señal: interrump ida	ResourceInitializationError: no se pueden extraer los secretos o la autenticación del registro: compruebe la configuración de red de la tarea. : señal: interrumpida	

Visualización de los errores de las tareas detenidas de Amazon ECS

Si tiene problemas para iniciar una tarea, es posible que la tarea se haya detenido debido a errores en la aplicación o configuración. Por ejemplo, ejecuta la tarea y esta muestra un estado `PENDING` y, a continuación, desaparece.

Si la tarea la creó un servicio de Amazon ECS, las acciones que Amazon ECS realiza para mantener el servicio se publican en los eventos del servicio. Puede ver los eventos en la AWS Management Console, la AWS CLI, los SDK de AWS, la API de Amazon ECS o las herramientas que utilizan los SDK y la API. Estos eventos incluyen que Amazon ECS detenga y reemplace una tarea porque los contenedores de la tarea han dejado de ejecutarse o no han superado demasiadas comprobaciones de estado de Elastic Load Balancing.

Si la tarea se ejecutó en una instancia de contenedor en Amazon EC2 o en equipos externos, también puede consultar los registros del tiempo de ejecución del contenedor y del agente de

Amazon ECS. Estos registros se encuentran en la instancia de Amazon EC2 de host o en un equipo externo. Para obtener más información, consulte Visualización de los registros del agente de contenedor de Amazon ECS.

Procedimiento

Console

AWS Management Console

Los siguientes pasos se pueden utilizar para comprobar si hay errores en las tareas detenidas a través de la consola. Para ver las tareas detenidas, debe cambiar la opción de filtro.

Las tareas detenidas solo aparecen en la consola durante 1 hora.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. En la página Clusters (Clústeres), elija el clúster.
4. En la página de Cluster : ***name*** (Clúster: nombre), seleccione la pestaña de Tasks (Tareas).
5. Configure el filtro para que muestre las tareas detenidas. Para Filtrar el estado deseado, seleccione Detenida.

   La opción Detenido muestra las tareas detenidas y Cualquier estado deseado muestra todas las tareas.
6. Elija la tarea detenida que desea inspeccionar.
7. En la fila correspondiente a la tarea detenida, en la columna Último estado, seleccione Detenido.

   En una ventana emergente se muestra el motivo por el que se ha detenido.

AWS CLI

1. Enumere las tareas detenidas en un clúster. El resultado contiene el nombre de recurso de Amazon (ARN) de la tarea, que necesita para describir la tarea.

```
aws ecs list-tasks \
     --cluster cluster_name \
     --desired-status STOPPED \
```

```
     --region region
```

2. Describa la tarea detenida para recuperar la información. Para obtener más información, consulte describe-task en la Referencia de la AWS Command Line Interface.

```
aws ecs describe-tasks \
     --cluster cluster_name \
     --tasks arn:aws:ecs:region:account_id:task/cluster_name/task_ID \
     --region region
```

Utilice los siguientes parámetros de salida.

- `stopCode`: el código de detención indica por qué se detuvo una tarea, por ejemplo, ResourceInitializationError.
- `StoppedReason`: el motivo por el que se detuvo la tarea.
- `reason` (en la estructura de `containers`): el motivo proporciona detalles adicionales sobre el contenedor detenido.

Pasos a seguir a continuación

Visualización de las tareas detenidas para obtener información sobre la causa. Para obtener más información, consulte Mensajes de error de las tareas detenidas de Amazon ECS.

Mensajes de error de las tareas detenidas de Amazon ECS

A continuación se muestran los posibles mensajes de error que puede recibir cuando una tarea se detiene de forma inesperada.

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

Los códigos de error de las tareas detenidas tienen una categoría asociada a ellos, por ejemplo, “ResourceInitializationError”. Para obtener más información sobre cada categoría, consulte lo siguiente:

Categoría	Más información
TaskFailedToStart	Solución de errores TaskFailedToStart de Amazon ECS
ResourceInitializationError	Solución de errores ResourceInitializationError de Amazon ECS
ResourceNotFoundException	Solución de errores ResourceNotFoundException de Amazon ECS
SpotInterruptionError	Solución de errores SpotInterruption de Amazon ECS
InternalError	Solución de errores InternalError de Amazon ECS
OutOfMemoryError	Solución de errores OutOfMemoryError de Amazon ECS
ContainerRuntimeError	Solución de errores ContainerRuntimeError de Amazon ECS Exec
ContainerRuntimeTimeoutError	Solución de errores ContainerRuntimeTimeoutError de Amazon ECS
CannotStartContainerError	Solución de errores CannotStartContainerError de Amazon ECS
CannotStopContainerError	Solución de errores CannotStopContainerError de Amazon ECS

Categoría	Más información
CannotInspectContainerError	Solución de errores CannotInspectContainerError de Amazon ECS
CannotCreateVolumeError	Solución de errores CannotCreateVolumeError de Amazon ECS
CannotPullContainer	Errores de tareas CannotPullContainer en Amazon ECS

Solución de errores TaskFailedToStart de Amazon ECS

A continuación, se muestran algunos mensajes de error de `TaskFailedToStart` y las acciones que puede llevar a cabo para corregir los errores.

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

Error inesperado de EC2 al intentar crear una interfaz de red con asignación de IP pública habilitada en la subred "***subnet-id***"

Esto ocurre cuando una tarea de Fargate que usa el modo de red `awsvpc` y se ejecuta en una subred con una dirección IP pública y la subred no tiene suficientes direcciones IP.

El número de direcciones IP disponibles está disponible en la página de detalles de la subred de la consola de Amazon EC2 o mediante `describe-subnets`. Para obtener más información, consulte View your subnet en la Guía del usuario de Amazon VPC.

Para solucionar este problema, puede crear una subred donde ejecutar la tarea.

InternalError: ***<reason>***

Este error se produce cuando se solicita un adjunto de ENI. Amazon EC2 gestiona de forma asíncrona el aprovisionamiento de la ENI. El proceso de aprovisionamiento lleva tiempo. Amazon ECS dispone de un tiempo de espera en caso de que se produzcan largos tiempos de espera o

errores no comunicados. Hay ocasiones en las que la ENI se aprovisiona, pero el informe llega a Amazon ECS después del tiempo de espera del error. En este caso, Amazon ECS ve el error de la tarea notificado con una ENI en uso.

La definición de la tarea seleccionada no es compatible con la estrategia de computación seleccionada

Este error se produce al elegir una definición de tarea con un tipo de lanzamiento que no coincide con el tipo de capacidad del clúster. Para obtener más información, consulte Tipos de lanzamiento de Amazon ECS. Tiene que seleccionar una definición de tarea que coincida con el proveedor de capacidad asignado a su clúster.

No es posible conectar la interfaz de red al índice de dispositivos no utilizados

Este error se produce cuando se utiliza un tipo de red `awsvpc` y no hay suficiente CPU o memoria para la tarea. En primer lugar, compruebe la CPU para ver las instancias. Para obtener más información, consulte las especificaciones de los tipos de instancias de Amazon EC2 en Tipos de instancias de Amazon EC2. Tome el valor de la CPU para la instancia y multiplíquelo por el número de ENI de la instancia. Ese valor “e” debe usarlo en la definición de tarea.

AGENT

La instancia de contenedor en la que se ha intentado lanzar una tarea tiene un agente que está desconectado actualmente. Para evitar tiempos de espera prolongados para la ubicación de tareas, se rechazó la solicitud.

Para obtener información acerca de cómo solucionar problemas de un agente desconectado, consulte How do I troubleshoot a disconnected Amazon ECS agent (¿Cómo soluciono problemas de un agente de Amazon ECS desconectado?).

Solución de errores ResourceInitializationError de Amazon ECS

A continuación, se muestran algunos mensajes de error de `ResourceInitialization` y las acciones que puede llevar a cabo para corregir los errores.

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

Errores

- La tarea no puede extraer la autenticación del registro de Amazon ECR. Hay un problema de conexión entre la tarea y Amazon ECR. Compruebe la configuración de red de la tarea.
- La tarea no puede descargar los archivo de variables de entorno de Amazon S3. Hay un problema de conexión entre la tarea y Amazon S3. Compruebe la configuración de red de la tarea.
- La tarea no puede extraer secretos del almacén de parámetros de AWS Systems Manager. Compruebe la conexión de red entre la tarea y AWS Systems Manager.
- La tarea no puede extraer secretos de AWS Secrets Manager. Hay un problema de conexión entre la tarea y Secrets Manager. Compruebe la configuración de red de la tarea.
- La tarea no puede extraer el secreto desde Secrets Manager. La tarea no puede extraer el secreto con el ARN 'secretARN' de Secrets Manager. Compruebe si el secreto existe en la región especificada.
- error en la extracción del comando: no se pueden extraer los secretos o la autenticación del registro. Compruebe la configuración de red de la tarea.
- La tarea no puede encontrar el grupo de registro de Amazon CloudWatch que se especifica en la definición de la tarea. Hay un problema de conexión entre la tarea y Amazon CloudWatch. Compruebe la configuración de la red.
- no se pudo inicializar el controlador de registro
- no se pudieron invocar los comandos utils de EFS para configurar los volúmenes de EFS

La tarea no puede extraer la autenticación del registro de Amazon ECR. Hay un problema de conexión entre la tarea y Amazon ECR. Compruebe la configuración de red de la tarea.

Este error indica que la tarea no se puede conectar a Amazon ECR.

Compruebe la conexión entre la tarea y Amazon ECR. Para obtener más información, consulta Comprobación de la conectividad de las tareas detenidas de Amazon ECS.

La tarea no puede descargar los archivo de variables de entorno de Amazon S3. Hay un problema de conexión entre la tarea y Amazon S3. Compruebe la configuración de red de la tarea.

Este error se produce cuando la tarea no puede descargar el archivo de entorno de Amazon S3.

Compruebe la conexión entre la tarea y el punto de conexión de VPC de Amazon S3. Para obtener más información, consulta Comprobación de la conectividad de las tareas detenidas de Amazon ECS.

La tarea no puede extraer secretos del almacén de parámetros de AWS Systems Manager. Compruebe la conexión de red entre la tarea y AWS Systems Manager.

Este error se produce cuando la tarea no puede extraer la imagen especificada en la definición de la tarea mediante las credenciales de Systems Manager.

Compruebe la conexión entre la tarea y el punto de conexión de VPC de Systems Manager. Para obtener más información, consulta Comprobación de la conectividad de las tareas detenidas de Amazon ECS.

La tarea no puede extraer secretos de AWS Secrets Manager. Hay un problema de conexión entre la tarea y Secrets Manager. Compruebe la configuración de red de la tarea.

Este error se produce cuando la tarea no puede extraer la imagen especificada en la definición de la tarea mediante las credenciales de Secrets Manager.

El error indica que hay un problema de conectividad de red entre el punto de conexión de VPC de Systems Manager y la tarea.

Para obtener información sobre cómo verificar la conectividad entre la tarea y el punto de conexión, consulte Comprobación de la conectividad de las tareas detenidas de Amazon ECS.

La tarea no puede extraer el secreto desde Secrets Manager. La tarea no puede extraer el secreto con el ARN '***secretARN***' de Secrets Manager. Compruebe si el secreto existe en la región especificada.

Este error se produce cuando la tarea no puede extraer la imagen especificada en la definición de la tarea mediante las credenciales de Secrets Manager.

Este error se debe a uno o varios de los siguientes motivos:

Causa del error…	Haga lo siguiente...	
Problema de conectividad de red entre el punto de conexión de VPC de Secrets Manager y la tarea. El problema se debe a un problema de red cuando aparece alguna de las	Compruebe la conectividad entre la tarea y el punto de conexión de Secrets Manager. Para obtener más informaci ón, consulte Comprobación de la conectividad de las tareas detenidas de Amazon ECS.	

Causa del error…	Haga lo siguiente...	
siguientes cadenas en el mensaje de error: • dial tcp • dial udp • <ip>:<port>: tiempo de espera de e/s • net/http: tiempo de espera del establecimiento de comunicación TLS • read: se ha agotado el tiempo de espera de la conexión • Se superó el Client.Timeout mientras se esperaban los encabezados • net/http: solicitud cancelada mientras se esperaba la conexión • signal: interrumpida • context: se superó la fecha límite		
El rol de ejecución de tareas especificado en la definición de la tarea no tiene los permisos para Secrets Manager.	Agregue los permisos necesarios para Secrets Manager al rol de ejecución de tareas. Para obtener más información, consulte Permisos de Secrets Manager o Systems Manager.	

Causa del error…	Haga lo siguiente...	
El ARN del secreto no existe	Compruebe si el ARN existe en Secrets Manager. Para obtener información sobre cómo ver las imágenes, consulte Find secrets in Secrets Manager en la Guía para desarrolladores de Secrets Manager.	

error en la extracción del comando: no se pueden extraer los secretos o la autenticación del registro. Compruebe la configuración de red de la tarea.

Este error se produce cuando la tarea no puede conectarse a Amazon ECR, Systems Manager o Secrets Manager. Esto se debe a una configuración incorrecta en la red.

A fin de solucionar este problema, compruebe la conectividad entre la tarea y Amazon ECR. También necesita verificar la conectividad entre la tarea y el servicio que almacena el secreto (Systems Manager o Secrets Manager). Para obtener más información, consulte Comprobación de la conectividad de las tareas detenidas de Amazon ECS.

La tarea no puede encontrar el grupo de registro de Amazon CloudWatch que se especifica en la definición de la tarea. Hay un problema de conexión entre la tarea y Amazon CloudWatch. Compruebe la configuración de la red.

Este error se produce cuando la tarea no encuentra el grupo de registro de CloudWatch que ha definido en la definición de la tarea.

El error indica que hay un problema de conectividad de red entre el punto de conexión de VPC de CloudWatch y la tarea.

Para obtener información sobre cómo verificar la conectividad entre la tarea y el punto de conexión, consulte Comprobación de la conectividad de las tareas detenidas de Amazon ECS.

no se pudo inicializar el controlador de registro

Este error se produce cuando la tarea no encuentra el grupo de registro de CloudWatch que ha definido en la definición de la tarea.

El error indica que el grupo de CloudWatch de la definición de la tarea no existe.

Siga los pasos a continuación para encontrar el CloudWatch faltante.

1. Ejecute el siguiente comando para obtener la información de definición de la tarea.

```
aws ecs describe-task-definition \
    --task-definition task-def-name
```

   Observe el resultado de cada contenedor y anote el valor `awslogs-group`.

```
"logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/example-group",
                    "awslogs-create-group": "true",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "ecs"
                },
```

2. Compruebe que el grupo existe en CloudWatch y para obtener más información, consulte Cómo trabajar con grupos de registro y flujos de registro en la Guía del usuario de Registros de Amazon CloudWatch.

   El problema es que el grupo especificado en la definición de la tarea es incorrecto o el grupo de registro no existe.

3. Solucione el problema.

El problema es...	Haga lo siguiente...	
El grupo de registro incorrecto o se especifica en la definición de la tarea.	Actualice la definición de la tarea e incluya la configuración del grupo de registro en la definición del contenedor. Para obtener información sobre cómo actualizar la definición de la tarea, consulte Actualización de una definición de tareas de	

El problema es...	Haga lo siguiente...
	Amazon ECS mediante la consola o RegisterTaskDefinition en la Referencia de la API de Amazon Elastic Container Service.
El grupo de registro no existe en CloudWatch	Cree el grupo de registro. Para obtener más información, consulte Crear un grupo de recursos en CloudWatch Logs en la Guía del usuario de Amazon CloudWatch Logs.

no se pudieron invocar los comandos utils de EFS para configurar los volúmenes de EFS

Los siguientes problemas pueden impedirle montar los volúmenes de Amazon EFS en las solicitudes:

- El sistema de archivos de Amazon EFS no está configurado correctamente.
- La tarea no cuenta con los permisos requeridos.
- Hay problemas relacionados con las configuraciones de red y VPC.

Para obtener información sobre cómo depurar y solucionar este problema, consulte ¿Por qué no puedo montar mis volúmenes de Amazon EFS en mis tareas de AWS Fargate? en AWS re:Post.

Solución de errores ResourceNotFoundException de Amazon ECS

A continuación, se muestran algunos mensajes de error de `ResourceNotFoundException` y las acciones que puede llevar a cabo para corregir los errores.

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

La tarea no puede recuperar el secreto con el ARN "*sercretARN*" desde AWS Secrets Manager. Compruebe si el secreto existe en la región especificada.

Este error se produce cuando la tarea no puede recuperar el secreto desde Secrets Manager. Esto significa que el secreto especificado en la definición de la tarea (y contenido en el mensaje de error) no existe en Secrets Manager.

La región está en el mensaje de error.

Al buscar datos secretos desde AWS Secrets Manager en la región *region*: secreto *sercretARN*: ResourceNotFoundException: Secrets Manager no puede encontrar el secreto especificado.

Para obtener información sobre cómo encontrar un secreto, consulte Find secrets in AWS Secrets Manager en la Guía del usuario de AWS Secrets Manager.

Utilice la siguiente tabla para determinar y solucionar el error.

Problema	Acciones
El secreto se encuentra en una región diferente de la definición de la tarea.	a. Cree el secreto en la misma región que la tarea. Para obtener más información, consulte Crear un secreto de AWS Secrets Manager. b. Actualice la definición de la tarea con el nuevo secreto. Para obtener más información, consulte Actualización de una definición de tareas de Amazon ECS mediante la consola o RegisterTaskDefinition en la Referencia de la API de Amazon Elastic Container Service.
La definición de la tarea tiene un ARN de secreto incorrecto.	Actualice la definición de la tarea con el secreto correcto.

Problema	Acciones
El secreto correcto existe en Secrets Manager.	Para obtener más informaci ón, consulte Actualización de una definición de tareas de Amazon ECS mediante la consola o RegisterTaskDefini tion en la Referencia de la API de Amazon Elastic Container Service.
El secreto ya no existe.	a. Cree el secreto en la misma región que la tarea. Para obtener más información, consulte Crear un secreto de AWS Secrets Manager. b. Actualice la definición de la tarea con el nuevo secreto. Para obtener más informaci ón, consulte Actualiza ción de una definición de tareas de Amazon ECS mediante la consola o RegisterTaskDefinition en la Referencia de la API de Amazon Elastic Container Service.

Solución de errores SpotInterruption de Amazon ECS

El error `SpotInterruption` tiene diferentes motivos para los tipos de lanzamiento de Fargate y EC2.

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

Tipo de lanzamiento de Fargate

El error `SpotInterruption` se produce cuando no hay capacidad de spot en Fargate o cuando Fargate recupera la capacidad de spot.

Puede hacer que sus tareas se ejecuten en varias zonas de disponibilidad para obtener más capacidad.

Tipo de lanzamiento de EC2

Este error se produce cuando no hay instancias de spot disponibles o cuando EC2 recupera la capacidad de instancias de spot.

Puede hacer que sus instancias se ejecuten en varias zonas de disponibilidad para obtener más capacidad.

Solución de errores InternalError de Amazon ECS

Se aplica a: tipo de lanzamiento de Fargate

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

El error `InternalError` se produce cuando el agente encuentra un error interno inesperado y no relacionado con el tiempo de ejecución.

Este error solo se produce si se utiliza la versión `1.4` de la plataforma o una posterior.

Para obtener información acerca de cómo depurar y resolver este problema, consulte Mensajes de error de las tareas detenidas de Amazon ECS.

Solución de errores OutOfMemoryError de Amazon ECS

A continuación, se muestran algunos mensajes de error de OutOfMemoryError y las acciones que puede llevar a cabo para corregir los errores.

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

contenedor inactivo debido al uso de memoria

Este error se produce cuando un contenedor sale porque los procesos del contenedor consumen más memoria de la que se asignó en la definición de tarea o debido a restricciones del host o sistema operativo.

Solución de errores ContainerRuntimeError de Amazon ECS Exec

A continuación, se muestran algunos mensajes de error de ContainerRuntimeError y las acciones que puede llevar a cabo para corregir los errores.

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

ContainerRuntimeError

Este error se produce cuando el agente recibe un error inesperado de `containerd` relacionado con una operación específica en tiempo de ejecución. Este error suele deberse a un problema interno del agente o el tiempo de ejecución de `containerd`.

Este error solo se produce si se utiliza la versión de la plataforma`1.4.0` o posterior (Linux) o `1.0.0` o posterior (Windows).

Para obtener información sobre cómo depurar y solucionar este problema, consulte ¿Por qué se detiene mi tarea de Amazon ECS? en AWS re:Post.

Solución de errores ContainerRuntimeTimeoutError de Amazon ECS

A continuación, se muestran algunos mensajes de error de ContainerRuntimeTimeoutError y las acciones que puede llevar a cabo para corregir los errores.

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

No se pudo realizar la transición al modo de ejecución; se agotó el tiempo de espera después de esperar 1 min o se produjo un error de tiempo de espera de Docker.

Este error se produce cuando un contenedor no puede realizar la transición a un estado `RUNNING` o `STOPPED` dentro del período de espera. El motivo y el valor del tiempo de espera se indican en el mensaje de error.

Solución de errores CannotStartContainerError de Amazon ECS

A continuación, se muestran algunos mensajes de error de CannotStartContainerError y las acciones que puede llevar a cabo para corregir los errores.

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

no se pudo obtener el estado del contenedor: ***<reason>***

Este error se produce cuando no se puede iniciar un contenedor.

Si su contenedor intenta superar la memoria especificada aquí, el contenedor se detiene. Aumento de la memoria presentada al contenedor. Este es el parámetro `memory` de la definición de la tarea. Para obtener más información, consulte the section called “Memoria”.

Solución de errores CannotStopContainerError de Amazon ECS

A continuación, se muestran algunos mensajes de error de CannotStopContainerError y las acciones que puede llevar a cabo para corregir los errores.

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

CannotStopContainerError

Este error se produce cuando no se puede detener un contenedor.

Para obtener información sobre cómo depurar y solucionar este problema, consulte ¿Por qué se detiene mi tarea de Amazon ECS? en AWS re:Post.

Solución de errores CannotInspectContainerError de Amazon ECS

A continuación, se muestran algunos mensajes de error de CannotInspectContainerError y las acciones que puede llevar a cabo para corregir los errores.

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

CannotInspectContainerError

Este error se produce cuando el agente de contenedor no puede describir el contenedor durante el tiempo de ejecución del contenedor.

Si se utiliza la versión de la plataforma `1.3` u otra anterior, el agente de Amazon ECS devuelve el motivo desde Docker.

Si se utiliza la versión de la plataforma `1.4.0` u otra posterior (Linux), o `1.0.0` u otra posterior (Windows), el agente de Fargate devuelve el motivo desde `containerd`.

Para obtener información sobre cómo depurar y solucionar este problema, consulte ¿Por qué se detiene mi tarea de Amazon ECS? en AWS re:Post.

Solución de errores CannotCreateVolumeError de Amazon ECS

A continuación, se muestran algunos mensajes de error de CannotCreateVolumeError y las acciones que puede llevar a cabo para corregir los errores.

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

CannotCreateVolumeError

Este error se produce cuando el agente no puede crear el montaje de volumen que se especifica en la definición de tareas.

Este error solo se produce si se utiliza la versión de la plataforma`1.4.0` o posterior (Linux) o `1.0.0` o posterior (Windows).

Para obtener información sobre cómo depurar y solucionar este problema, consulte ¿Por qué se detiene mi tarea de Amazon ECS? en AWS re:Post.

Errores de tareas CannotPullContainer en Amazon ECS

Los siguientes errores indican que la tarea no se pudo iniciar porque Amazon ECS no pudo recuperar la imagen del contenedor especificada.

> **Note**
>
> La versión 1.4 de la plataforma de Fargate trunca los mensajes de error largos.

Para comprobar si hay algún mensaje de error en las tareas detenidas a través de la AWS Management Console, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

Errores

- La tarea no puede extraer la imagen. Compruebe que el rol tenga los permisos para extraer imágenes del registro.
- La tarea no puede extraer el 'nombre de la imagen' del 'URI del repositorio' de Amazon ECR. Hay un problema de conexión entre la tarea y Amazon ECR. Compruebe la configuración de red de la tarea.
- La tarea no puede extraer la imagen. Compruebe la configuración de la red y vuelva a intentarlo.
- CannotPullContainerError: el manifiesto de extracción de imagen se ha reintentado 5 veces: no se pudo resolver la referencia
- Error de API (500): obtener https://111122223333.dkr.ecr.us-east-1.amazonaws.com/v2/: net/http: solicitud cancelada mientras se esperaba la conexión
- Errores de API
- escribir /var/lib/docker/tmp/GetImageBlob111111111: no queda espacio en el dispositivo
- ERROR: toomanyrequests: demasiadas solicitudes o ha alcanzado el límite de su tasa de extracción.
- Respuesta de error del daemon: obtener url: net/http: solicitud cancelada mientras se esperaba la conexión
- ref pull se ha reintentado 1 vez: no se pudo copiar: httpReaderSeeker: no se pudo abrir: código de estado inesperado
- acceso de extracción denegado
- El comando de extracción falló: pánico: error de tiempo de ejecución: dirección de memoria no válida o falta de referencia del puntero
- error al extraer la imagen conf/error al extraer la configuración de la imagen
- Contexto cancelado

La tarea no puede extraer la imagen. Compruebe que el rol tenga los permisos para extraer imágenes del registro.

Este error indica que la tarea no puede extraer la imagen especificada en la definición de la tarea debido a problemas con los permisos.

Para resolver este problema, siga estos pasos:

1. Verifique que la imagen exista en el *repositorio*. Para obtener información sobre cómo ver las imágenes, consulte Viewing image details in Amazon ECR en la Guía del usuario de Amazon Elastic Container Registry.
2. Compruebe que el rol *role-arn* tenga los permisos correctos para extraer la imagen.

   Para obtener información acerca de cómo actualizar los roles, consulte Actualización de los permisos de un rol en la Guía del usuario de AWS Identity and Access Management.

   La tarea utiliza uno de los siguientes roles:

   - Para tareas con el tipo de lanzamiento de Fargate, este es el rol de ejecución de tareas. Para obtener información acerca de los permisos adicionales para Amazon ECR, consulte Las tareas de Fargate que extraen imágenes de Amazon ECR a través de permisos de puntos de conexión de interfaz.
   - Para tareas con el tipo de lanzamiento de EC2, este es el rol de la instancia de contenedor. Para obtener información acerca de los permisos adicionales para Amazon ECR, consulte Permisos de Amazon ECR.

La tarea no puede extraer el '***nombre de la imagen***' del '***URI del repositorio***' de Amazon ECR. Hay un problema de conexión entre la tarea y Amazon ECR. Compruebe la configuración de red de la tarea.

Este error indica que la tarea no se puede conectar a Amazon ECR. Compruebe la conexión al *URI del repositorio*.

Para obtener información acerca de cómo verificar y resolver este problema, consulte Comprobación de la conectividad de las tareas detenidas de Amazon ECS.

La tarea no puede extraer la imagen. Compruebe la configuración de la red y vuelva a intentarlo.

Este error indica que la tarea no se puede conectar a Amazon ECR.

Para obtener información acerca de cómo verificar y resolver este problema, consulte Comprobación de la conectividad de las tareas detenidas de Amazon ECS.

CannotPullContainerError: el manifiesto de extracción de imagen se ha reintentado 5 veces: no se pudo resolver la referencia

Este error indica que la tarea no puede extraer la imagen.

Para resolver este problema, puede:

- Verifique que la imagen especificada en la definición de la tarea coincida con la imagen en el repositorio.
- Amazon ECS fuerza la estabilidad de la versión de la imagen. Si la imagen original ya no está disponible, aparece este error. La etiqueta de la imagen forma parte de la aplicación de este comportamiento. Cambie la imagen en la definición de la tarea: en lugar de utilizar :latest como etiqueta usa una versión específica. Para obtener más información, consulte Resolución de imagen de contenedor.

Para obtener información acerca de cómo verificar y resolver este problema, consulte Comprobación de la conectividad de las tareas detenidas de Amazon ECS.

Error de API (500): obtener https://111122223333.dkr.ecr.us-east-1.amazonaws.com/v2/: net/http: solicitud cancelada mientras se esperaba la conexión

Este error indica que se ha agotado el tiempo de espera de la conexión porque no existe una ruta a Internet.

Para solucionar este problema, puede:

- Para tareas en subredes públicas, especificar ENABLED (HABILITADO) para Auto-assign public IP (Asignar IP pública automáticamente) al ejecutar la tarea. Para obtener más información, consulte Ejecución de una aplicación como tarea de Amazon ECS.
- Para tareas en subredes privadas, especifique DISABLED (Deshabilitado) en Auto-assign public IP (Asignar IP pública automáticamente) al lanzar la tarea y configure una gateway NAT en la VPC para dirigir las solicitudes a Internet. Para obtener más información, consulte Gateways NAT en la Guía del usuario de Amazon VPC.

Errores de API

Este error indica que hay un problema de conexión con el punto de conexión de Amazon ECR.

Para obtener información sobre cómo resolver este problema, consulte ¿Cómo puedo resolver el error “CannotPullContainerError: API error” en Amazon ECS? en el sitio web de Soporte.

escribir /var/lib/docker/tmp/***GetImageBlob111111111***: no queda espacio en el dispositivo

Este error indica que no hay suficiente espacio en disco.

Para solucionar este problema, libere espacio en el disco.

Si utiliza la AMI optimizada para Amazon ECS, puede emplear el siguiente comando para recuperar los 20 archivos más grandes del sistema de archivos:

```
du -Sh / | sort -rh | head -20
```

Ejemplo de salida:

```
5.7G    /var/lib/docker/
containers/50501b5f4cbf90b406e0ca60bf4e6d4ec8f773a6c1d2b451ed8e0195418ad0d2
1.2G    /var/log/ecs
594M    /var/lib/docker/devicemapper/mnt/
c8e3010e36ce4c089bf286a623699f5233097ca126ebd5a700af023a5127633d/rootfs/data/logs
...
```

En algunos casos, un contenedor en ejecución puede llenar el volumen raíz. Si el contenedor utiliza el controlador de registros `json-file` predeterminado sin un límite `max-size`, es posible que el archivo de registro sea responsable de la mayor parte de ese espacio utilizado. Puede utilizar el comando `docker ps` para comprobar qué contenedor está utilizando el espacio mapeando el nombre del directorio de la salida anterior al ID del contenedor. Por ejemplo:

```
CONTAINER ID   IMAGE                            COMMAND             CREATED
 STATUS              PORTS                           NAMES
50501b5f4cbf   amazon/amazon-ecs-agent:latest   "/agent"            4 days ago
 Up 4 days                                             ecs-agent
```

De forma predeterminada, cuando se utiliza el controlador de registros `json-file`, Docker captura la salida estándar (y el error estándar) de todos los contenedores y los escribe en archivos con formato JSON. Puede establecer el `max-size` como una opción del controlador de registros, lo que impide que el archivo de registro ocupe demasiado espacio. Para obtener más información, consulte Controlador de registro de archivos JSON en la documentación de Docker.

A continuación se muestra un fragmento de definición de contenedor que muestra cómo utilizar esta opción:

```
{
    "log-driver": "json-file",
    "log-opts": {
```

```
        "max-size": "256m"
    }
}
```

Si los registros de contenedor ocupan demasiado espacio en el disco, una alternativa consiste en utilizar el controlador de registros `awslogs`. El controlador de registros `awslogs` envía los registros a CloudWatch, lo que libera espacio en disco que, de lo contrario, utilizarían los registros de contenedor de la instancia de contenedor. Para obtener más información, consulte Envío de registros de Amazon ECS a CloudWatch .

Es posible que deba actualizar el tamaño del disco al que Docker puede acceder.

Para más información, consulte CannotPullContainerError: no queda espacio en el dispositivo.

ERROR: toomanyrequests: demasiadas solicitudes o ha alcanzado el límite de su tasa de extracción.

Este error indica que hay un límite de velocidad de Docker Hub.

Si recibe uno de los siguientes errores, es probable que haya alcanzado los límites de tasa de Docker Hub:

Para obtener más información acerca de los límites de tasa de Docker Hub, consulte Descripción de la limitación de velocidad de Docker Hub.

Si ha aumentado el límite de tasa de Docker Hub y necesita autenticar las extracciones de Docker para las instancias de contenedor, consulte Private registry authentication for container instances.

Respuesta de error del daemon: obtener ***`url`***: net/http: solicitud cancelada mientras se esperaba la conexión

Este error indica que se ha agotado el tiempo de espera de la conexión porque no existe una ruta a Internet.

Para solucionar este problema, puede:

- Para tareas en subredes públicas, especificar ENABLED (HABILITADO) para Auto-assign public IP (Asignar IP pública automáticamente) al ejecutar la tarea. Para obtener más información, consulte Ejecución de una aplicación como tarea de Amazon ECS.
- Para tareas en subredes privadas, especifique DISABLED (Deshabilitado) en Auto-assign public IP (Asignar IP pública automáticamente) al lanzar la tarea y configure una gateway NAT en la VPC

para dirigir las solicitudes a Internet. Para obtener más información, consulte Gateways NAT en la Guía del usuario de Amazon VPC.

ref pull se ha reintentado 1 vez: no se pudo copiar: httpReaderSeeker: no se pudo abrir: código de estado inesperado

Este error indica que se ha producido un error al copiar una imagen.

Para resolver este problema, consulte uno de estos artículos:

- Para las tareas de Fargate, consulte Cómo resuelvo el error “cannotpullcontainererror” en las tareas de Amazon ECS alojadas en Fargate.
- Para otras tareas, consulte Cómo resuelvo el error “cannotpullcontainererror” en las tareas de Amazon ECS.

acceso de extracción denegado

Este error indica que no hay acceso a la imagen.

Para resolver este problema, es posible que deba autenticar su cliente de Docker con Amazon ECR. Para obtener más información, consulte Private registry authentication en la Guía del usuario de Amazon ECR.

El comando de extracción falló: pánico: error de tiempo de ejecución: dirección de memoria no válida o falta de referencia del puntero

Este error indica que no se puede acceder a la imagen debido a una dirección de memoria no válida o a una falta de referencia del puntero.

Para resolver este problema, siga estos pasos:

- Compruebe que dispone de las reglas del grupo de seguridad para acceder a Amazon S3.
- Cuando utilice puntos de conexión de puerta de enlace, debe agregar una ruta en la tabla de enrutamiento para acceder al punto de conexión.

error al extraer la imagen conf/error al extraer la configuración de la imagen

Este error indica que se ha alcanzado un límite de velocidad o que se ha producido un error de red:

Para resolver este problema, consulte Cómo puedo resolver el error “CannotPullContainerError” en mi tarea de tipo de lanzamiento de Amazon ECS EC2.

Contexto cancelado

Este error indica que se canceló el contexto.

La causa común de este error es que la VPC que está utilizando la tarea no cuenta con ninguna ruta para extraer la imagen de contenedor de Amazon ECR.

Comprobación de la conectividad de las tareas detenidas de Amazon ECS

Hay ocasiones en las que una tarea se detiene debido a un problema de conectividad de red. Puede que se trate de un problema intermitente, pero lo más probable es que se deba a que la tarea no se puede conectar a un punto de conexión.

Comprobación de la conexión de las tareas

Puede usar el manual de procedimientos `AWSSupport-TroubleshootECSTaskFailedToStart` para probar la conectividad de las tareas. Al utilizar el manual de procedimientos, deberá proporcionar la siguiente información sobre los recursos:

- El ID de la tarea

  Use el ID de la tarea fallida más reciente.
- El clúster en el que estaba la tarea

Para obtener información sobre cómo usar el manual de procedimientos, consulte `AWSSupport-TroubleshootECSTaskFailedToStart` en la Referencia del manual de procedimientos de Automatización de AWS Systems Manager.

El manual de procedimientos analiza la tarea. Puede ver los resultados en la sección Salida para los siguientes problemas que pueden impedir el inicio de una tarea:

- Conectividad de red con el registro de contenedores configurado
- Conectividad del punto de conexión de VPC
- Configuración de regla de grupo de seguridad

Solución de problemas de punto de conexión de VPC

Cuando el resultado del manual de procedimientos `AWSSupport-TroubleshootECSTaskFailedToStart` indique un problema de punto de conexión de VPC, compruebe la siguiente configuración:

- La VPC en la que se crea el punto de conexión debe usar un DNS privado.
- Asegúrese de tener un punto de conexión de AWS PrivateLink para el servicio al que la tarea no se pueda conectar en la misma VPC que la tarea. Para obtener más información, consulte lo siguiente:

Servicio	Información de punto de conexión de VPC para el servicio
Amazon ECR	Puntos de conexión de VCP de la interfaz de Amazon ECR (AWS PrivateLink)
Systems Manager	Mejora de la seguridad de las instancias de EC2 mediante puntos de conexión de VPC para Systems Manager
Secrets Manager	Uso de un punto de conexión de VPC de AWS Secrets Manager
CloudWatch	Punto de conexión de VPC de CloudWatch
Amazon S3	AWS PrivateLink para Amazon S3

- Configure una regla de salida para la subred de tareas que permita el tráfico HTTPS en el puerto 443 DNS (TCP). Para obtener más información, consulte Configuración de reglas de grupos de seguridad en la Guía del usuario de Amazon Elastic Compute Cloud.

- Si utiliza un servidor de dominio de nombre personalizado, confirme la configuración de la consulta de DNS. La consulta debe tener acceso de salida en el puerto 53 y usar los protocolos UDP y TCP. Además, debe tener acceso HTTPS en el puerto 443. Para obtener más información, consulte Configuración de reglas de grupos de seguridad en la Guía del usuario de Amazon Elastic Compute Cloud.
- Si la subred tiene una ACL de red, se requieren las siguientes reglas de ACL:
  - Una regla de salida que permita el tráfico en los puertos 1024-65535.
  - Una regla de entrada que permita el tráfico TCP en el puerto 443.

  Para obtener información sobre cómo configurar las reglas, consulte Control del tráfico hacia las subredes utilizando las ACL de red en la Guía del usuario de Amazon Virtual Private Cloud.

Solución de problemas de red

Cuando el resultado del manual de procedimientos `AWSSupport-TroubleshootECSTaskFailedToStart` indique un problema de red, compruebe la siguiente configuración:

Tareas que utilizan el modo de red awsvpc en una subred pública

Aplique la siguiente configuración según el manual de procedimientos:

- Para tareas en subredes públicas, especificar ENABLED (HABILITADO) para Auto-assign public IP (Asignar IP pública automáticamente) al ejecutar la tarea. Para obtener más información, consulte Ejecución de una aplicación como tarea de Amazon ECS.
- Necesita una puerta de enlace para gestionar el tráfico de Internet. La tabla de enrutamiento de la subred de tareas debe tener una ruta para el tráfico a la puerta de enlace.

  Para obtener más información, consulte Agregar y eliminar rutas de una tabla de enrutamiento en la Guía del usuario de Amazon Virtual Private Cloud.

Tipo de puerta de enlace	Destino de la tabla de enrutamiento	Objetivo de la tabla de enrutamiento
NAT	0.0.0.0/0	ID de gateway NAT

Tipo de puerta de enlace	Destino de la tabla de enrutamiento	Objetivo de la tabla de enrutamiento
Puerta de enlace de Internet	0.0.0.0/0	ID de la puerta de enlace de Internet

- Si la subred de tareas tiene una ACL de red, se requieren las siguientes reglas de ACL:
  - Una regla de salida que permita el tráfico en los puertos 1024-65535.
  - Una regla de entrada que permita el tráfico TCP en el puerto 443.

  Para obtener información sobre cómo configurar las reglas, consulte Control del tráfico hacia las subredes utilizando las ACL de red en la Guía del usuario de Amazon Virtual Private Cloud.

Tareas que utilizan el modo de red awsvpc en una subred privada

Aplique la siguiente configuración según el manual de procedimientos:

- Seleccione DESHABILITADO para Asignar automáticamente la IP pública al iniciar la tarea.
- Configure una puerta de enlace de NAT en su VPC para enrutar las solicitudes a Internet. Para obtener más información, consulte Puertas de enlace NAT en la Guía del usuario de Amazon Virtual Private Cloud.
- La tabla de enrutamiento de la subred de tareas debe tener una ruta para el tráfico a la puerta de enlace de NAT.

  Para obtener más información, consulte Agregar y eliminar rutas de una tabla de enrutamiento en la Guía del usuario de Amazon Virtual Private Cloud.

Tipo de puerta de enlace	Destino de la tabla de enrutamiento	Objetivo de la tabla de enrutamiento
NAT	0.0.0.0/0	ID de gateway NAT

- Si la subred de tareas tiene una ACL de red, se requieren las siguientes reglas de ACL:
  - Una regla de salida que permita el tráfico en los puertos 1024-65535.
  - Una regla de entrada que permita el tráfico TCP en el puerto 443.

Para obtener información sobre cómo configurar las reglas, consulte Control del tráfico hacia las subredes utilizando las ACL de red en la Guía del usuario de Amazon Virtual Private Cloud.

Tareas que no utilizan el modo de red awsvpc en una subred pública

Aplique la siguiente configuración según el manual de procedimientos:

- Seleccione Activar para Asignar la IP automáticamente en Redes para instancias de Amazon EC2 al crear el clúster.

  Esta opción asigna una dirección IP pública a la interfaz de red principal de la instancia.
- Necesita una puerta de enlace para gestionar el tráfico de Internet. La tabla de enrutamiento de la subred de la instancia debe tener una ruta para el tráfico a la puerta de enlace.

  Para obtener más información, consulte Agregar y eliminar rutas de una tabla de enrutamiento en la Guía del usuario de Amazon Virtual Private Cloud.

Tipo de puerta de enlace	Destino de la tabla de enrutamiento	Objetivo de la tabla de enrutamiento
NAT	0.0.0.0/0	ID de gateway NAT
Puerta de enlace de Internet	0.0.0.0/0	ID de la puerta de enlace de Internet

- Si la subred de la instancia tiene una ACL de red, se requieren las siguientes reglas de ACL:
  - Una regla de salida que permita el tráfico en los puertos 1024-65535.
  - Una regla de entrada que permita el tráfico TCP en el puerto 443.

  Para obtener información sobre cómo configurar las reglas, consulte Control del tráfico hacia las subredes utilizando las ACL de red en la Guía del usuario de Amazon Virtual Private Cloud.

Tareas que no utilizan el modo de red awsvpc en una subred privada

Aplique la siguiente configuración según el manual de procedimientos:

- Seleccione Desactivar para Asignar la IP automáticamente en Redes para instancias de Amazon EC2 al crear el clúster.

- Configure una puerta de enlace de NAT en su VPC para enrutar las solicitudes a Internet. Para obtener más información, consulte Gateways NAT en la Guía del usuario de Amazon VPC.
- La tabla de enrutamiento de la subred de la instancia debe tener una ruta para el tráfico a la puerta de enlace de NAT.

  Para obtener más información, consulte Agregar y eliminar rutas de una tabla de enrutamiento en la Guía del usuario de Amazon Virtual Private Cloud.

Tipo de puerta de enlace	Destino de la tabla de enrutamiento	Objetivo de la tabla de enrutamiento
NAT	0.0.0.0/0	ID de gateway NAT

- Si la subred de tareas tiene una ACL de red, se requieren las siguientes reglas de ACL:
  - Una regla de salida que permita el tráfico en los puertos 1024-65535.
  - Una regla de entrada que permita el tráfico TCP en el puerto 443.

  Para obtener información sobre cómo configurar las reglas, consulte Control del tráfico hacia las subredes utilizando las ACL de red en la Guía del usuario de Amazon Virtual Private Cloud.

Visualización de solicitudes de roles de IAM para tareas de Amazon ECS

Cuando utiliza un proveedor para las credenciales de las tareas en un rol de IAM, las solicitudes del proveedor se guardan en un registro de auditoría. El registro de auditoría hereda la misma configuración de rotación de registro que el registro del agente de contenedor. Las variables de configuración de agente de contenedor `ECS_LOG_ROLLOVER_TYPE`, `ECS_LOG_MAX_FILE_SIZE_MB` y `ECS_LOG_MAX_ROLL_COUNT` se pueden establecer para que afecten al comportamiento del registro de auditoría. Para obtener más información, consulte Parámetros de configuración del registro del agente de contenedor de Amazon ECS.

Para el agente de contenedor versión 1.36.0 y posterior, el registro de auditoría se encuentra en `/var/log/ecs/audit.log`. Cuando rota el registro, se agrega una marca temporal en formato *YYYY*-*MM*-*DD*-*HH* al final del nombre del archivo de registro.

Para el agente de contenedor versión 1.35.0 y anterior, el registro de auditoría se encuentra en `/var/log/ecs/audit.log.`*YYYY*-*MM*-*DD*-*HH*.

El formato de entrada de registro es el siguiente:

- Timestamp
- Código de respuesta HTTP
- Dirección IP y número de puerto del origen de solicitud
- URI relativa del proveedor de credenciales
- El agente de usuario que realizó la solicitud
- El ARN de la tarea a la que pertenece el contenedor solicitante
- El nombre de API `GetCredentials` y su número de versión
- El nombre del clúster de Amazon ECS en el que está registrada la instancia de contenedor
- El ARN de la instancia de contenedor

Puede usar el siguiente comando para ver los archivos de registro.

```
cat /var/log/ecs/audit.log.2016-07-13-16
```

Salida:

```
2016-07-13T16:11:53Z 200 172.17.0.5:52444 "/v1/credentials" "python-requests/2.7.0
 CPython/2.7.6 Linux/4.4.14-24.50.amzn1.x86_64" TASK_ARN GetCredentials
 1 CLUSTER_NAME CONTAINER_INSTANCE_ARN
```

Visualización de los mensajes de eventos del servicio de Amazon ECS

Si está solucionando un problema con un servicio, primero debe comprobar la información de diagnóstico del registro de eventos de servicio. Para consultar los eventos de servicio, utilice la API `DescribeServices`, la AWS CLI o la AWS Management Console.

Cuando se visualizan mensajes de eventos de servicio mediante la API de Amazon ECS, solo se devuelven los eventos del programador de servicios. Estos incluyen los eventos de estado de instancia y ubicación de tareas más recientes. Sin embargo, la consola de Amazon ECS muestra eventos de servicio provenientes de las siguientes fuentes.

- Eventos de estado de instancia y ubicación de las tareas provenientes del programador de servicios de Amazon ECS. Estos eventos tienen el prefijo service ***(service-name)***. Para

garantizar que la vista de eventos sirve de utilidad, solo mostramos los `100` eventos más recientes. Los mensajes de eventos del servicio duplicados se omiten hasta que resuelve la causa de esta duplicación o a las seis horas. Si la causa no se resuelve dentro de las seis horas, recibirá otro mensaje de evento del servicio por ese motivo.

- Eventos de escalado automático de servicios Estos eventos tienen un prefijo de Mensaje y solo se producen cuando se configura un servicio con una política de escalado de Application Auto Scaling.

Siga estos pasos para ver los mensajes de eventos de servicio actuales.

Console

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. En la página Clusters (Clústeres), elija el clúster.
4. Elija el servicio que desea inspeccionar.
5. Elija Deployments and events (Implementaciones y eventos). En Events (Eventos), vea los mensajes.

AWS CLI

Utilice el comando describe-services para ver los mensajes de eventos de servicio para un servicio especificado.

El siguientes ejemplo de la AWS CLI describe el servicio *`service-name`* del clúster *`predeterminado`*, que proporcionará los mensajes de eventos de servicio más recientes.

```
aws ecs describe-services \
    --cluster default \
    --services service-name \
    --region us-west-2
```

Mensajes de eventos del servicio de Amazon ECS

A continuación, se proporcionan ejemplos de mensajes de eventos de servicio que es posible que vea en la consola de Amazon ECS.

El servicio (*service-name*) ha alcanzado un estado estable.

El programador del servicio envía un evento de servicio `service (service-name) has reached a steady state.` cuando el servicio esté en buen estado y con el número deseado de tareas, alcanzando así un estado estable.

El programador de servicios informa periódicamente del estado, por lo que podría recibir este mensaje varias veces.

El servicio (*service-name*) no ha podido asignar una tarea porque ninguna instancia de contenedor cumplía todos los requisitos.

El programador del servicio envía este mensaje de evento cuando no puede encontrar los recursos disponibles para agregar otra tarea. Las causas posibles de este error son:

Utilice proveedores de capacidad para escalar automáticamente las instancias de EC2. Para obtener más información, consulte Proveedores de capacidad de Amazon ECS para el tipo de lanzamiento de EC2 .

> Si pretendía utilizar un proveedor de capacidad, asegúrese de que esté pasando una estrategia de proveedor de capacidad o tenga una estrategia de proveedor de capacidad predeterminada asociada a su clúster y no esté pasando el tipo de inicialización y la estrategia de proveedor de capacidad como entrada.

No se ha encontrado en su clúster ninguna instancia de contenedor

> Si no se ha registrado ninguna instancia de contenedor en el clúster en el que intenta ejecutar una tarea, recibe este error. Debería añadir instancias de contenedor a su clúster. Para obtener más información, consulte Lanzamiento de una instancia de contenedor de Linux de Amazon ECS.

No hay puertos suficientes

> Si la tarea utiliza mapeo de puertos de host fijo (por ejemplo, la tarea usa el puerto 80 en el host para un servidor web), debe tener al menos una instancia de contenedor por tarea, dado que solo un contenedor puede utilizar un único puerto de host a la vez. Debería agregar instancias de contenedor a su clúster o reducir el número de tareas deseadas.

Demasiados puertos registrados

La instancia de contenedor más coincidente para la ubicación de tareas no puede exceder el límite máximo permitido de puerto reservado de 100 puertos de host por instancia de contenedor. El uso del mapeo dinámico de los puertos de host puede solucionar el problema.

Puerto ya en uso

La definición de tarea de esta tarea utiliza el mismo puerto en su asignación de puertos que una tarea que ya se ejecuta en la instancia de contenedor que se ha elegido. El mensaje de evento de servicio tendría el ID de instancia de contenedor elegido como parte del siguiente mensaje.

```
The closest matching container-instance is already using a port required by your
 task.
```

No hay memoria suficiente

Si la definición de tareas especifica 1 000 MiB de memoria y las instancias de contenedor en el clúster tienen cada una 1 024 MiB de memoria, solo puede ejecutar una copia de esta tarea por instancia de contenedor. Puede experimentar con menos memoria en la definición de tareas para que pueda lanzar más de una tarea por instancia de contenedor o lanzar más instancias de contenedor en su clúster.

> **Note**
>
> Si intenta maximizar la utilización de los recursos proporcionando a las tareas la mayor cantidad de memoria posible para un tipo de instancia determinado, consulte Reserva de la memoria de instancias de contenedor de Linux de Amazon ECS.

No hay suficiente CPU

Una instancia de contenedor tiene 1 024 unidades de CPU por cada núcleo de CPU. Si la definición de tareas especifica 1 000 unidades de CPU y las instancias de contenedor en el clúster tienen cada una 1 024 unidades de CPU, solo puede ejecutar una copia de esta tarea por instancia de contenedor. Puede experimentar con menos unidades de CPU en la definición de tarea para poder lanzar más de una tarea por instancia de contenedor o lanzar más instancias de contenedor en su clúster.

No hay suficientes puntos de asociación de ENI

Cada una de las tareas que utilizan el modo de red `awsvpc` recibe su propia interfaz de red elástica (ENI), que se asocia a la instancia de contenedor que la aloja. Las instancias de Amazon EC2 tienen un límite en cuanto al número de ENI que se les pueden asociar y no hay ninguna instancia de contenedor en el clúster que disponga capacidad de ENI.

El límite de ENI para instancias de contenedor individuales depende de las siguientes condiciones:

- Si no se ha inscrito en el ajuste de cuenta `awsvpcTrunking`, el límite de ENI para cada instancia de contenedor depende del tipo de instancia. Para obtener más información, consulte Direcciones IP por interfaz de red por tipo de instancia en la Guía del usuario de Amazon EC2.
- Si se ha inscrito en el ajuste de cuenta `awsvpcTrunking`, pero no ha lanzado nuevas instancias de contenedor con un tipo de instancia compatible después de la inscripción, el límite de ENI de cada instancia de contenedor sigue siendo el valor predeterminado. Para obtener más información, consulte Direcciones IP por interfaz de red por tipo de instancia en la Guía del usuario de Amazon EC2.
- Si se ha inscrito en el ajuste de cuenta `awsvpcTrunking` y ha lanzado nuevas instancias de contenedor con un tipo de instancia compatible después de la inscripción, están disponibles ENI adicionales. Para obtener más información, consulte Instancias admitidas para un aumento de las interfaces de red de contenedores de Amazon ECS.

Para obtener más información acerca de la inscripción en el ajuste de cuenta `awsvpcTrunking`, consulte Aumento de las interfaces de red de instancias de contenedor de Linux de Amazon ECS.

Puede añadir instancias de contenedor a su clúster para proporcionar más adaptadores de red disponibles.

Falta atributo requerido en instancia de contenedor

Algunos parámetros de definición de tareas requieren que se instale en la instancia de contenedor una versión de API remota de Docker específica. Otras, como las opciones de controlador de registros, requieren que las instancias de contenedor registren dichos controladores de registro con la variable de configuración del agente `ECS_AVAILABLE_LOGGING_DRIVERS`. Si la definición de tareas contiene un parámetro que requiere un atributo de instancia de contenedor específico y no tiene instancias de contenedor disponibles que puedan satisfacer este requisito, la tarea no se puede colocar.

Una causa común de este error es si el servicio utiliza tareas que utilizan el modo de red `awsvpc` y el tipo de lanzamiento de EC2. El clúster especificado no tiene una instancia de contenedor registrada en la misma subred que se especificó en la `awsvpcConfiguration` cuando se creó el servicio.

Puede usar el manual de procedimientos AWSSupport-TroubleshootECSContainerInstance para solucionar problemas. Este manual de procedimientos comprueba si los datos de usuario de la instancia contienen la información de clúster correcta, si el perfil de instancia contiene los permisos necesarios y si hay problemas de configuración de la red. Para obtener más información, consulte AWSSupport-TroubleshootECSContainerInstance en la Guía del usuario de referencia del manual de procedimientos de Automatización de AWS Systems Manager.

Para obtener más información sobre qué atributos son necesarios para parámetros de definición de tareas específicas y variables de configuración de agentes, consulte Parámetros en la definición de tareas de Amazon ECS para el tipo de lanzamiento de Fargate y Configuración del agente de contenedor de Amazon ECS.

El servicio (***service-name***) no ha podido asignar una tarea porque ninguna instancia de contenedor cumplía todos los requisitos. El ***container-instance-id*** de contenedor-instancia más parecida tiene disponibles unidades de CPU insuficientes.

La instancia de contenedor más parecida para la colocación de tareas no contiene suficientes unidades de CPU para satisfacer los requisitos en la definición de tareas. Revise los requisitos de CPU en los parámetros de tamaño de tarea y de definición de contenedor de la definición de tarea.

El servicio (***service-name***) no ha podido asignar una tarea porque ninguna instancia de contenedor cumplía todos los requisitos. Las instancia de contenedor más parecida ***id-instancia-contenedor*** encontró el error "AGENT".

El agente de contenedor de Amazon ECS en la instancia de contenedor que más coincide para la ubicación de la tarea está desconectado. Si puede conectar a la instancia de contenedor con SSH, puede examinar los registros de agente; para obtener más información, consulte Parámetros de configuración del registro del agente de contenedor de Amazon ECS. También debe verificar que el agente se está ejecutando en la instancia. Si utiliza la AMI optimizada para Amazon ECS puede intentar detener y reiniciar el agente mediante el siguiente comando.

- Para la AMI de Amazon Linux 2 optimizada para Amazon ECS y la AMI de Amazon Linux 2023 optimizada para Amazon ECS

```
sudo systemctl restart ecs
```

- Para la AMI de Amazon Linux optimizada para Amazon ECS

```
sudo stop ecs && sudo start ecs
```

(la instancia ***instance-id***) de (la tarea ***task-id***) del servicio (***service-name***) no está en buen estado en (elb ***elb-name***) debido a (razón por la que la instancia ha fallado o al menos el número de comprobaciones de estado UnhealthyThreshold consecutivas).

Este servicio está registrado con un balanceador de carga y las comprobaciones de estado del balanceador de carga están fallando. El mensaje incluye el ID de la tarea para ayudar a identificar qué tarea específica no supera las comprobaciones de estado. Para obtener más información, consulte Solución de problemas de los equilibradores de carga de servicio en Amazon ECS.

El servicio (***service-name***) no puede iniciar las tareas de forma consistente y con éxito.

Este servicio contiene tareas que no se han podido iniciar después de varios intentos consecutivos. En este momento, el programador de servicio comienza a aumentar incrementalmente el tiempo entre intentos. Debe encontrar el motivo por el que las tareas no se pueden lanzar. Para obtener más información, consulte Lógica de limitación controlada de servicios de Amazon ECS.

Una vez que el servicio se actualice, por ejemplo con una definición de tarea actualizada, el programador de servicio continuará funcionando con normalidad.

Se están limitando de las operaciones del servicio (***service-name***) de manera controlada. Vuelva a intentarlo más adelante.

Este servicio no puede lanzar más tareas debido a los límites controlados de API. Una vez que el programador de servicios pueda lanzar más tareas, se reanudará.

Para solicitar un aumento de la cuota del límite de tasa de la API, abra la página AWS Support Center, inicie sesión si es necesario y elija Create case (Crear caso). Seleccione Service limit increase (Aumento del límite de servicio). Rellene y envíe el formulario.

servicio (*service-name*) no pudo detener o iniciar tareas durante una implementación debido a la configuración de implementación del servicio. Actualice el valor minimumHealthyPercent o MaximumPercent y vuelva a intentarlo.

Este servicio no puede detener o iniciar tareas durante una implementación de servicio debido a la configuración de implementación. La configuración de la implementación consta de los valores `minimumHealthyPercent` y `maximumPercent` que se definen al crear el servicio. Estos valores también se pueden actualizar en un servicio existente.

El valor `minimumHealthyPercent` representa el límite mínimo del número de tareas que se deben ejecutar para un servicio durante una implementación o cuando una instancia de contenedor se está agotando. Es un porcentaje del número deseado de tareas para el servicio. Este valor se redondea hacia arriba. Por ejemplo, si el porcentaje mínimo en buen estado es `50` y el número de tareas deseado es cuatro, el programador puede detener dos tareas existentes antes de iniciar dos nuevas. Del mismo modo, si el porcentaje mínimo en buen estado es del 75 % y el recuento de tareas deseado es dos, entonces el programador no puede detener ninguna tarea debido a que el valor resultante también es dos.

El valor `maximumPercent` representa el límite máximo del número de tareas que se deben ejecutar para un servicio durante una implementación o cuando una instancia de contenedor se está agotando. Es un porcentaje del número deseado de tareas para un servicio. Este valor se redondea hacia abajo. Por ejemplo, si el porcentaje máximo es `200` y el recuento de tareas deseado es cuatro, el programador puede iniciar cuatro tareas nuevas antes de detener cuatro tareas existentes. Del mismo modo, si el porcentaje máximo es `125` y el recuento de tareas deseado es tres, el programador no puede iniciar ninguna tarea debido a que el valor resultante también es tres.

Al establecer un porcentaje mínimo o uno máximo en buen estado, debe asegurarse de que el programador pueda detener o iniciar al menos una tarea cuando se desencadena una implementación.

servicio (*service-name*) no pudo ubicar una tarea. Motivo: Ha alcanzado el límite del número de tareas que puede ejecutar en forma simultánea

Puede solicitar un aumento de cuota para el recurso que provocó el error. Para obtener más información, consulte Service Quotas. Para solicitar un aumento de cuota, consulte Solicitud de un aumento de cuota en la Guía de usuario de Service Quotas.

servicio (*service-name*) no pudo ubicar una tarea. Motivo: Error interno.

A continuación, se presenta el posible motivo para este error:

El servicio no puede iniciar una tarea debido a que una subred se encuentra en una zona de disponibilidad no compatible.

Para obtener más información sobre las regiones de Fargate y las zonas de disponibilidad admitidas, consulte the section called “Regiones de AWS Fargate”.

Para obtener más información acerca de cómo ver la zona de disponibilidad de las subredes, consulte Ver la subred en la Guía del usuario de Amazon VPC.

servicio (*service-name*) no pudo ubicar una tarea. Motivo: La configuración de CPU solicitada supera el límite.

Puede solicitar un aumento de cuota para el recurso que provocó el error. Para obtener más información, consulte Service Quotas. Para solicitar un aumento de cuota, consulte Solicitud de un aumento de cuota en la Guía de usuario de Service Quotas.

servicio (*service-name*) no pudo ubicar una tarea. Motivo: La configuración de MEMORIA solicitada supera el límite.

Puede solicitar un aumento de cuota para el recurso que provocó el error. Para obtener más información, consulte Service Quotas. Para solicitar un aumento de cuota, consulte Solicitud de un aumento de cuota en la Guía de usuario de Service Quotas.

servicio (*service-name*) no pudo ubicar una tarea. Motivo: Ha alcanzado el límite del número de vCPU que puede ejecutar de forma simultánea

AWS Fargate está pasando de cuotas basadas en el recuento de tareas a cuotas basadas en vCPU.

Puede solicitar un aumento de todas las cuotas basadas en vCPU de Fargate. Para obtener más información, consulte Service Quotas. Para solicitar un aumento de cuota de Fargate, consulte Solicitud de aumento de cuota en la Guía del usuario de Service Quotas.

El servicio (*service-name*) no pudo alcanzar un estado estable porque el conjunto de tareas (*taskSet-ID*) no se pudo reducir horizontalmente. Motivo: el número de tareas protegidas es superior al recuento de tareas deseado

El servicio tiene más tareas protegidas que el número de tareas deseado. Puede elegir una de las siguientes opciones:

- Espere a que caduque la protección de las tareas actuales para poder finalizarlas.
- Determine qué tareas pueden detenerse y utilice la API `UpdateTaskProtection` con la opción `protectionEnabled` configurada en `false` para desactivar la protección de estas tareas.
- Aumente el recuento de tareas deseado del servicio a un número mayor al de tareas protegidas.

El servicio (*service-name*) no pudo alcanzar un estado estable. Motivo: no se encontró ninguna instancia de contenedor en su proveedor de capacidad.

El programador del servicio envía este mensaje de evento cuando no puede encontrar los recursos disponibles para agregar otra tarea. Las causas posibles de este error son:

No hay ningún proveedor de capacidad asociado al clúster

Use `describe-services` para comprobar que tiene un proveedor de capacidad asociado al clúster. Puede actualizar la estrategia del proveedor de capacidad para el servicio.

Compruebe que haya capacidad disponible en el proveedor de capacidad. En el caso del tipo de lanzamiento de EC2, asegúrese de que las instancias de contenedor cumplan con los requisitos de definición de tareas.

No se ha encontrado en su clúster ninguna instancia de contenedor

Si no se ha registrado ninguna instancia de contenedor en el clúster en el que intenta ejecutar una tarea, recibe este error. Debería añadir instancias de contenedor a su clúster. Para obtener más información, consulte Lanzamiento de una instancia de contenedor de Linux de Amazon ECS.

No hay puertos suficientes

Si la tarea utiliza la asignación de puertos de host fija (por ejemplo, la tarea usa el puerto 80 en el host para un servidor web), debe tener al menos una instancia de contenedor por tarea. Solo un contenedor puede usar un solo puerto de host a la vez. Debería agregar instancias de contenedor a su clúster o reducir el número de tareas deseadas.

Demasiados puertos registrados

La instancia de contenedor más coincidente para la ubicación de tareas no puede exceder el límite máximo permitido de puerto reservado de 100 puertos de host por instancia de contenedor. El uso del mapeo dinámico de los puertos de host puede solucionar el problema.

Puerto ya en uso

La definición de tarea de esta tarea utiliza el mismo puerto en su asignación de puertos que una tarea que ya se ejecuta en la instancia de contenedor que se ha elegido. El mensaje de evento de servicio tendría el ID de instancia de contenedor elegido como parte del siguiente mensaje.

```
The closest matching container-instance is already using a port required by your
 task.
```

No hay memoria suficiente

Si la definición de tareas especifica 1 000 MiB de memoria y las instancias de contenedor en el clúster tienen cada una 1 024 MiB de memoria, solo puede ejecutar una copia de esta tarea por instancia de contenedor. Puede experimentar con menos memoria en la definición de tareas para que pueda lanzar más de una tarea por instancia de contenedor o lanzar más instancias de contenedor en su clúster.

> **Note**
>
> Si está intentando maximizar la utilización de los recursos proporcionando a las tareas la mayor cantidad de memoria posible para un tipo de instancia determinado, consulte Reserva de la memoria de instancias de contenedor de Linux de Amazon ECS.

No hay suficientes puntos de asociación de ENI

Cada una de las tareas que utilizan el modo de red `awsvpc` recibe su propia interfaz de red elástica (ENI), que se asocia a la instancia de contenedor que la aloja. Las instancias de Amazon EC2 tienen un límite en cuanto al número de ENI que se les pueden adjuntar y no hay ninguna instancia de contenedor en el clúster que disponga de capacidad de ENI.

El límite de ENI para instancias de contenedor individuales depende de las siguientes condiciones:

- Si no se ha inscrito en el ajuste de cuenta `awsvpcTrunking`, el límite de ENI para cada instancia de contenedor depende del tipo de instancia. Para obtener más información, consulte Direcciones IP por interfaz de red por tipo de instancia en la Guía del usuario de Amazon EC2.
- Si se ha inscrito en el ajuste de cuenta `awsvpcTrunking`, pero no ha lanzado nuevas instancias de contenedor con un tipo de instancia compatible después de la inscripción, el límite de ENI de cada instancia de contenedor sigue siendo el valor predeterminado. Para obtener más información, consulte Direcciones IP por interfaz de red por tipo de instancia en la Guía del usuario de Amazon EC2.
- Si se ha inscrito en el ajuste de cuenta `awsvpcTrunking` y ha lanzado nuevas instancias de contenedor con un tipo de instancia compatible después de la inscripción, están disponibles ENI adicionales. Para obtener más información, consulte Instancias admitidas para un aumento de las interfaces de red de contenedores de Amazon ECS.

Para obtener más información acerca de la inscripción en el ajuste de cuenta `awsvpcTrunking`, consulte Aumento de las interfaces de red de instancias de contenedor de Linux de Amazon ECS.

Puede añadir instancias de contenedor a su clúster para proporcionar más adaptadores de red disponibles.

Falta atributo requerido en instancia de contenedor

Algunos parámetros de definición de tareas requieren que se instale en la instancia de contenedor una versión de API remota de Docker específica. Otras, como las opciones de controlador de registros, requieren que las instancias de contenedor registren dichos controladores de registro con la variable de configuración del agente `ECS_AVAILABLE_LOGGING_DRIVERS`. Si la definición de tareas contiene un parámetro que requiere un atributo de instancia de contenedor específico y no tiene instancias de contenedor disponibles que puedan satisfacer este requisito, la tarea no se puede colocar.

Una causa común de este error es si el servicio utiliza tareas que emplean el modo de red `awsvpc` y el tipo de lanzamiento de EC2, y el clúster especificado no tiene una instancia de contenedor registrada en la misma subred que se especificó en `awsvpcConfiguration` cuando se creó el servicio.

Puede usar el manual de procedimientos AWSSupport-TroubleshootECSContainerInstance para solucionar problemas. Este manual de procedimientos comprueba si los datos de usuario de la instancia contienen la información de clúster correcta, si el perfil de instancia contiene los permisos necesarios y si hay problemas de configuración de la red. Para obtener más

información, consulte AWSSupport-TroubleshootECSContainerInstance en la Guía del usuario de referencia del manual de procedimientos de Automatización de AWS Systems Manager.

Para obtener más información sobre qué atributos son necesarios para parámetros de definición de tareas específicas y variables de configuración de agentes, consulte Parámetros en la definición de tareas de Amazon ECS para el tipo de lanzamiento de Fargate y Configuración del agente de contenedor de Amazon ECS.

servicio (*service-name*) no pudo ubicar una tarea. Motivo: la capacidad no está disponible en este momento. Vuelva a intentarlo más tarde o en una zona de disponibilidad diferente.

Actualmente no hay capacidad disponible para ejecutar el servicio.

Puede elegir una de las siguientes opciones:

- Espere hasta que las instancias de contenedor de EC2 o la capacidad de Fargate estén disponibles.
- Vuelva a iniciar el servicio y especifique subredes adicionales.

no se pudo implementar el servicio (*service-name*): no se pudieron iniciar las tareas.

No se pudieron iniciar las tareas de su servicio.

Para obtener información sobre cómo depurar las tareas detenidas, consulte Mensajes de error de las tareas detenidas de Amazon ECS.

se agotó el tiempo de espera del servicio (*service-name*) al esperar que se iniciara el agente de Amazon ECS. Compruebe los registros en /var/log/ecs/ecs-agent.log.

El agente de contenedor de Amazon ECS en la instancia de contenedor que más coincide para la ubicación de la tarea está desconectado. Si puede conectarse a la instancia de contenedor con SSH, puede examinar los registros de agente. Para obtener más información, consulte Parámetros de configuración del registro del agente de contenedor de Amazon ECS. También debe verificar que el agente se está ejecutando en la instancia. Si utiliza la AMI optimizada para Amazon ECS puede intentar detener y reiniciar el agente mediante el siguiente comando.

- Para la AMI de Amazon Linux 2 optimizada para Amazon ECS

```
sudo systemctl restart ecs
```

- Para la AMI de Amazon Linux optimizada para Amazon ECS

```
sudo stop ecs && sudo start ecs
```

el conjunto de tareas (*taskSet-ID*) el servicio (*service-name*) (la tarea *task-id*) no está en buen estado en el grupo de destino (*targetGroup-ARN*) debido a **TARGET GROUP IS NOT FOUND**.

El conjunto de tareas del servicio no pasa las comprobaciones de estado porque no se encuentra el grupo de destino. El mensaje incluye el ID de la tarea para ayudar a identificar qué tarea específica no supera las comprobaciones de estado. Debe eliminar y volver a crear el servicio. No elimine ningún grupo de destino de Elastic Load Balancing a menos que ya se haya eliminado el servicio de Amazon ECS correspondiente.

el conjunto de tareas (*taskSet-ID*) el servicio (*service-name*) (la tarea *task-id*) no está en buen estado en el grupo de destino (*targetGroup-ARN*) debido a **TARGET IS NOT FOUND**.

El conjunto de tareas del servicio no pasa las comprobaciones de estado porque no se encuentra el destino. El mensaje incluye el ID de la tarea para ayudar a identificar qué tarea específica no supera las comprobaciones de estado.

Mensajes de eventos que no están en buen estado del servicio de Amazon ECS

A continuación, se proporcionan ejemplos de mensajes de eventos que no están en buen estado del servicio.

EC2: (servicio *service-name*) (tarea *task-id*) (instancia *instance-id*) (puerto *port-number*) no está en buen estado en (target-group *target-group-name*) debido a (motivo *failure-reason*)

Este mensaje indica que una tarea que se ejecuta en una instancia EC2 no supera las comprobaciones de estado Para obtener más información, consulte los siguientes temas:

- ¿Cómo logro que mis tareas de Amazon ECS que utilizan el tipo de lanzamiento de Amazon EC2 pasen la comprobación de estado del equilibrador de carga de aplicación?

EC2 con taskSet: (servicio ***service-name***, taskSet ***taskSet-id***) (tarea ***task-id***) (instancia ***instance-id***) (puerto ***port-number***) no está en buen estado en (target-group ***target-group-name***) debido a (motivo ***failure-reason***)

Este mensaje indica que una tarea de un conjunto de tareas que se ejecuta en una instancia EC2 no supera las comprobaciones de estado. Para obtener más información, consulte los siguientes temas:

- ¿Cómo logro que mis tareas de Amazon ECS que utilizan el tipo de lanzamiento de Amazon EC2 pasen la comprobación de estado del equilibrador de carga de aplicación?

Fargate: (servicio ***service-name***) (tarea ***task-id***) (puerto ***port-number***) no está en buen estado en (target-group ***target-group-name***) debido a (motivo ***failure-reason***)

Este mensaje indica que una tarea de Fargate no supera las comprobaciones de estado.

Para obtener más información sobre la solución de errores en las comprobaciones de estado en las tareas de Fargate, consulte ¿Cómo se solucionan los errores de las comprobaciones de estado de las tareas de Amazon ECS en Fargate?

Fargate con taskSet: (servicio ***service-name***, taskSet ***taskSet-id***) (tarea ***task-id***) (puerto ***port-number***) no está en buen estado en (target-group ***target-group-name***) debido a (motivo ***failure-reason***)

Este mensaje indica que una tarea de un conjunto de tareas que se ejecuta en Fargate no supera las comprobaciones de estado.

Para obtener más información sobre la solución de errores de las comprobaciones de estado en las tareas de Fargate, consulte ¿Cómo se solucionan los errores de las comprobaciones de estado de las tareas de Amazon ECS en Fargate?.

Mensajes de eventos del servicio de reequilibrio del servicio de zonas de disponibilidad de Amazon ECS

A continuación, se proporcionan ejemplos de mensajes de eventos de servicio que podría ver.

service (*nombre-del-servicio*) is not AZ balanced with *número-de-tareas* tasks in *zona de disponibilidad 1*, *número-de-tareas* in *zona de disponibilidad 2*, and *número-de-tareas* in *zona de disponibilidad 3*. El reequilibrio de la zona de disponibilidad está en curso.

El programador de servicios envía un evento de servicio `service (service-name) is not AZ balanced` cuando el número de tareas no se distribuye uniformemente en las zonas de disponibilidad. No hay ninguna acción que realizar. Se trata de un evento informativo.

service (*nombre-del-servicio*) is AZ balanced with *número-de-tareas* tasks in *zona de disponibilidad 1*, *número-de-tareas* tasks in *zona de disponibilidad 2*, and *número-de-tareas* tasks in *zona de disponibilidad 3*.

El programador de servicios envía un evento de servicio `service (service-name) is AZ balanced` cuando se completa el reequilibrio del servicio de la zona de disponibilidad. No hay ninguna acción que realizar. Se trata de un evento informativo.

nombre-del-servicio has started *número-de-tareas* tasks in *zona de disponibilidad* to AZ Rebalance: *id-de-tareas*.

El programador de servicios envía un evento de servicio "*nombre-del-servicio*/*nombre-del-conjunto-de-tareas* has started *número-de-tareas* tasks in *zona de disponibilidad*" cuando inicia tareas en una zona de disponibilidad debido al reequilibrio del servicio. No hay ninguna acción que realizar. Se trata de un evento informativo.

nombre-del-servicio has stopped *número-de-tareas* running tasks in *zona de disponibilidad* due to AZ rebalancing: *id-de-tarea*.

El programador de servicios envía un evento de servicio "*nombre-de-servicio*/*nombre-conjunto-tareas* ha detenido *número* de tareas en *zona de disponibilidad*" cuando detiene tareas en una zona de disponibilidad debido al reequilibrio del servicio. No hay ninguna acción que realizar. Se trata de un evento informativo.

service (*nombre-del-servicio*) is unable to place a task in *zona de disponibilidad* because no container instance met all of its requirements.

El programador de servicio envía un evento de servicio "*nombre-del-servicio* is unable to place a task in *zona de disponibilidad*" porque ninguna instancia de contenedor cumplió todos los requisitos. Para solucionar el problema, lance las instancias en la zona de disponibilidad.

service (*nombre-del-servicio*) is unable to place a task in *zona de disponibilidad*.

El programador de servicios envía un evento de servicio "*nombre-del-servicio* is unable to place a task in *zona de disponibilidad*" cuando se utiliza el tipo de lanzamiento de Fargate y no hay capacidad disponible.

Puede agregar otras subredes en la zona de disponibilidad que se muestra en el mensaje de error o ponerse en contacto con Soporte para obtener capacidad adicional.

service (*nombre-del-servicio*) was unable to AZ Rebalance because *nombre-del-conjunto-de-tareas* was unable to scale in due to *motivo*.

El programador del servicio envía un evento de servicio "*nombre-del-servicio* was unable to AZ Rebalance because *nombre-del-conjunto-de-tareas* was unable to scale in due to *motivo*" al utilizar la protección de reducción horizontal de tareas.

Puede elegir una de las siguientes opciones:

- Espere a que caduque la protección de las tareas actuales para poder finalizarlas.
- Determine qué tareas pueden detenerse y utilice la API `UpdateTaskProtection` con la opción `protectionEnabled` configurada en `false` para desactivar la protección de estas tareas.
- Aumente el recuento de tareas deseado del servicio a un número mayor al de tareas protegidas.

service (*nombre-del-servicio*) stopped AZ Rebalancing.

El programador de servicios envía un evento de servicio "*nombre-del-servicio* stopped AZ Rebalancing" cuando se detuvo la operación de reequilibrio de la zona de disponibilidad. Se trata de un evento informativo. Amazon ECS envía eventos adicionales que proporcionan más información.

Solución de problemas de los equilibradores de carga de servicio en Amazon ECS

Los servicios de Amazon ECS pueden registrar tareas con un balanceador de carga de Elastic Load Balancing. Los errores de configuración de los balanceadores de carga son habitualmente la causa de que se paren las tareas. Si las tareas paradas se iniciaron mediante servicios que utilizan un balanceador de carga, tenga en cuenta las siguientes causas posibles.

No existe el rol vinculado al servicio de Amazon ECS

El rol vinculado al servicio de Amazon ECS permite a los servicios de Amazon ECS registrar instancias de contenedor en los balanceadores de carga de Elastic Load Balancing. El rol vinculado al servicio se debe crear en la cuenta. Para obtener más información, consulte Uso de roles vinculados al servicio para Amazon ECS.

Grupo de seguridad de instancia de contenedor

Si el contenedor está asignado al puerto 80 en su instancia de contenedor, el grupo de seguridad de la instancia de contenedor debe permitir el tráfico de entrada en el puerto 80 para que se superen las comprobaciones de estado del balanceador de carga.

El equilibrador de carga de Elastic Load Balancing no se ha configurado para todas las zonas de disponibilidad

El balanceador de carga se debe configurar para que utilice todas las zonas de disponibilidad de una región o, al menos, todas en las que residen sus instancias de contenedor. Si un servicio utiliza un equilibrador de carga e inicia una tarea en una instancia de contenedor que reside en la zona de disponibilidad que el equilibrador de carga no tiene configurada para utilizar, la tarea no supera nunca la comprobación de estado. Esto supone la cancelación de la tarea.

Comprobación de estado del equilibrador de carga de Elastic Load Balancing mal configurada

Los parámetros de comprobación de estado del balanceador de carga pueden ser demasiado restrictivos ni señalar a recursos que no existen. Si se determina que una instancia de contenedor no está en buen estado, se elimina del equilibrador de carga. Asegúrese de comprobar que los parámetros siguientes estén configurados correctamente para el balanceador de carga de su servicio.

Ping Port

El valor Ping Port de una comprobación de estado del balanceador de carga es el puerto en las instancias de contenedor que comprueba el balanceador de carga para determinar si está

en buen estado. Si este puerto está mal configurado, el balanceador de carga probablemente cancele el registro de su instancia de contenedor desde sí mismo. Este puerto debe estar configurado para que utilice el valor `hostPort` para el contenedor en la definición de tareas del servicio que está utilizando con la comprobación de estado.

Ping Path

Esto forma parte de la comprobación del estado del equilibrador de cargas. Es un punto de conexión de la aplicación que puede devolver un código de estado correcto (por ejemplo, 200) cuando la aplicación está en buen estado. Este valor se suele establecer en `index.html`, pero si el servicio no responde a esa solicitud, la comprobación de estado falla. Si el contenedor no tiene un archivo `index.html`, puede definirlo como `/` para alcanzar la URL base de la instancia de contenedor.

Response Timeout

Es la cantidad de tiempo de la que dispone el contenedor para devolver una respuesta al ping de comprobación de estado. Si este valor es inferior a la cantidad de tiempo requerida para una respuesta, la comprobación de estado falla.

Intervalo de comprobación de estado

Es la cantidad de tiempo entre pings de comprobación de estado. Mientras más cortos sean los intervalos de comprobación de estado, antes podrá alcanzar la instancia de contenedor el umbral en mal estado.

Unhealthy Threshold

Es el número de veces que puede fallar la comprobación de estado antes de que se considere que la instancia de contenedor está en mal estado. Si tiene un umbral en mal estado de 2 y un intervalo de comprobación de estado de 30 segundos, entonces la tarea tiene 60 segundos para responder al ping de comprobación de estado antes de que se suponga que tiene mal estado. Puede aumentar el umbral en mal estado o el intervalo de comprobación de estado para dar a sus tareas más tiempo para responder.

No se puede actualizar el servicio *`servicename`*: se ha cambiado el nombre o el puerto del contenedor del equilibrador de carga en la definición de tarea

Si su servicio utiliza un equilibrador de carga, puede utilizar la AWS CLI o el SDK para modificar su configuración. Para obtener más información acerca de cómo modificar la configuración, consulte UpdateService en la Referencia de la API de Amazon Elastic Container Service. Si actualiza la definición de la tarea del servicio, el nombre y el puerto del contenedor que se

especificaron en la configuración del equilibrador de carga deben permanecer en la definición de la tarea.

Ha alcanzado el límite del número de tareas que puede ejecutar de forma simultánea.

Para una nueva cuenta, sus cuotas podrían ser inferiores a las cuotas de servicio. La cuota de servicio de su cuenta se puede consultar en la consola de Service Quotas. Para solicitar un aumento de cuota, consulte Solicitud de un aumento de cuota en la Guía de usuario de Service Quotas.

Solución de problemas de escalado automático de servicios en Amazon ECS

Application Auto Scaling desactiva los procesos de reducción horizontal mientras las implementaciones de Amazon ECS están en curso, y se reanudan una vez finalizada la implementación. Sin embargo, los procesos de escalado horizontal continúan produciéndose durante una implementación, a menos que se suspendan. Para obtener más información, consulte Suspensión y reanudación del escalado para Application Auto Scaling.

Solución de errores de CPU o memoria no válidos en la definición de tareas de Amazon ECS

Al registrar una definición de tareas mediante la API de Amazon ECS o la AWS CLI, si especifica un valor de `cpu` o `memory` no válido, se devuelve el siguiente error.

```
An error occurred (ClientException) when calling the RegisterTaskDefinition operation:
 Invalid 'cpu' setting for task.
```

> **Note**
>
> Cuando se utiliza Terraform, se puede devolver el siguiente error.
>
> ```
> Error: ClientException: No Fargate configuration exists for given values.
> ```

Para solucionar este problema, debe especificar un valor admitido para la CPU y la memoria de la tarea en la definición de esta. El valor `cpu` se puede expresar en unidades de CPU o vCPU en una

definición de tarea. Cuando se registra la definición de tarea, se convierte en un entero que indica las unidades de CPU. El valor `memory` se puede expresar en unidades de MiB o GB en una definición de tarea. Cuando se registra la definición de tarea, se convierte en un entero que indica los MiB.

Para las definiciones de tareas en las que se especifica `FARGATE` para el parámetro `requiresCompatibilities` (incluso si también se especifica `EC2`), debe utilizar uno de los valores que se indican en la siguiente tabla. Estos valores determinan el rango de los valores admitidos para el parámetro de CPU y memoria.

Para las tareas alojadas en Fargate, en la siguiente tabla, se muestran las combinaciones de CPU y memoria válidas. Los valores de memoria del archivo JSON se especifican en MiB. Puede convertir el valor de GB a MiB multiplicando el valor por 1024. Por ejemplo, 1 GB = 1024 MiB.

Valor de CPU	Valor de memoria	Sistemas operativos admitidos por AWS Fargate
256 (0,25 vCPU)	512 MiB, 1 GB, 2 GB	Linux
512 (0,5 vCPU)	1 GB, 2 GB, 3 GB, 4 GB	Linux
1024 (1 vCPU)	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB	Linux, Windows
2048 (2 vCPU)	Entre 4 GB y 16 GB en incrementos de 1 GB	Linux, Windows
4096 (4 vCPU)	Entre 8 GB y 30 GB en incrementos de 1 GB	Linux, Windows
8192 (8 vCPU) **Note** Esta opción requiere una plataforma Linux `1.4.0` o posterior.	Entre 16 GB y 60 GB en incrementos de 4 GB	Linux
16 384 (16 vCPU)	Entre 32 GB y 120 GB en incrementos de 8 GB	Linux

Valor de CPU	Valor de memoria	Sistemas operativos admitidos por AWS Fargate
Note Esta opción requiere una plataforma Linux `1.4.0` o posterior.		

Para las tareas alojadas en Amazon EC2, los valores admitidos de CPU para tareas están entre 0,25 vCPU y 192 vCPU.

El mecanismo de control de la CPU difiere entre los tipos de lanzamiento de Amazon EC2 y Fargate:

- Para las tareas alojadas en Amazon EC2: Amazon ECS utiliza el periodo de CPU y la cuota de CPU para controlar el tamaño de la tarea y los límites máximos de la CPU. Al especificar la vCPU en la definición de la tarea, Amazon ECS traduce el valor a la configuración del periodo y cuota de CPU que se aplica a `cgroup`.
- Para las tareas alojadas en Fargate: Amazon ECS utiliza recursos compartidos de CPU para controlar la asignación de CPU. Los valores de cuota y período de CPU no se utilizan para limitar la CPU en las tareas de Fargate.

Para las tareas de Amazon EC2, la cuota de CPU controla la cantidad de tiempo de CPU que se concede a `cgroup` durante un periodo de CPU determinado. Ambas configuraciones se expresan en microsegundos. Cuando la cuota de CPU es igual al periodo de CPU, significa que `cgroup` se puede ejecutar hasta el 100 % en una vCPU (o cualquier otra fracción que sume el 100 % en varias vCPU). La cuota de CPU tiene un máximo de 1 000 000 µs y el periodo de CPU es un mínimo de 1 ms. Puede usar estos valores para establecer los límites del recuento de CPU. Cuando se cambia el periodo de CPU sin cambiar la cuota de CPU, los límites efectivos son diferentes a los que se especifican en la definición de la tarea.

El periodo de 100 ms permite que las vCPU oscilen entre 0,125 y 10.

> **Note**
>
> Los parámetros de CPU y memoria de nivel de tarea se omiten para los contenedores de Windows.

Visualización de los registros del agente de contenedor de Amazon ECS

Amazon ECS almacena los registros en la carpeta `/var/log/ecs` de las instancias de contenedor. Hay registros disponibles del agente de contenedor de Amazon ECS y del servicio `ecs-init` que controla el estado del agente (comenzar/detener) en la instancia de contenedor. Puede ver estos archivos de registro conectando a una instancia de contenedor mediante SSH.

> **Note**
>
> Si no está seguro de cómo recopilar todos los registros de las instancias de contenedor, puede utilizar el recopilador de registros de Amazon ECS. Para obtener más información, consulte Recopilación de registros de contenedor con el recopilador de registros de Amazon ECS .

Sistema operativo Linux

El proceso `ecs-init` almacena registros en `/var/log/ecs/ecs-init.log`.

El archivo `ecs-init.log` contiene información sobre la administración del ciclo de vida, la configuración y el arranque del agente de contenedor.

Puede usar el siguiente comando para ver los archivos de registro.

```
cat /var/log/ecs/ecs-init.log
```

Salida:

```
2018-02-16T18:13:54Z [INFO] pre-start
2018-02-16T18:13:56Z [INFO] start
```

```
2018-02-16T18:13:56Z [INFO] No existing agent container to remove.
2018-02-16T18:13:56Z [INFO] Starting Amazon Elastic Container Service Agent
```

Sistema operativo Windows

Puede utilizar el recopilador de registros de Amazon ECS para Windows. Para obtener más información, consulte Amazon ECS Logs Collector For Windows en Github.

1. Conecte con la instancia .
2. Abra PowerShell y ejecute los siguientes comandos con privilegios administrativos. Los comandos descargan el script y recopilan los registros.

   ```
   Invoke-WebRequest -OutFile ecs-logs-collector.ps1 https://
   raw.githubusercontent.com/awslabs/aws-ecs-logs-collector-for-windows/master/ecs-
   logs-collector.ps1
   .\ecs-logs-collector.ps1
   ```

Puede activar el registro de depuración para el agente de Amazon ECS y el daemon de Docker. Esta opción permite que el script recopile los registros antes de activar el modo de depuración. El script reinicia el daemon de Docker y el agente de Amazon ECS y, a continuación, cierra todos los contenedores que están en ejecución en la instancia. Antes de ejecutar el siguiente comando, vacíe la instancia de contenedor y traslade las tareas importantes a otras instancias de contenedor.

Ejecute el siguiente comando para activar el registro.

```
.\ecs-logs-collector.ps1 -RunMode debug
```

Recopilación de registros de contenedor con el recopilador de registros de Amazon ECS

Si no está seguro de cómo recopilar todos los diversos registros en las instancias de contenedor, puede utilizar el recopilador de registros de Amazon ECS. Está disponible en GitHub tanto para Linux como para Windows. El script recopila registros generales del sistema operativo, así como registros del agente de contenedor de Amazon ECS y de Docker, que pueden resultar útiles para solucionar problemas en los casos de AWS Support. A continuación, comprime y archiva la información recopilada en un solo archivo que se puede compartir fácilmente con fines de diagnóstico. También

permite habilitar el modo de depuración para el daemon de Docker y el agente de contenedor de Amazon ECS en variantes de Amazon Linux como, por ejemplo, la AMI optimizada para Amazon ECS. Actualmente, el recopilador de registros de Amazon ECS admite los siguientes sistemas operativos:

- Amazon Linux
- Red Hat Enterprise Linux 7
- Debian 8
- Ubuntu 14.04
- Ubuntu 16.04
- Ubuntu 18.04
- Windows Server 2016

> **Note**
>
> El código fuente del recopilador de registros de Amazon ECS está disponible en GitHub tanto para Linux como para Windows. Le recomendamos enviar solicitudes de inserción para los cambios que le gustaría que incluyamos. No obstante, Amazon Web Services actualmente no permite ejecutar copias modificadas de este software.

Para descargar y ejecutar el recopilador de registros de Amazon ECS para Linux

1. Conéctese a la instancia de contenedor.
2. Descargue el script del recopilador de registros de Amazon ECS.

```
curl -O https://raw.githubusercontent.com/awslabs/ecs-logs-collector/master/ecs-logs-collector.sh
```

3. Ejecute el script para recopilar los registros y crear el archivo.

> **Note**
>
> Para habilitar el modo de depuración para el daemon de Docker y el agente de contenedor de Amazon ECS, agregue la opción `--mode=enable-debug` al siguiente comando. Esto puede reiniciar el daemon de Docker, que cierra todos los contenedores que están en ejecución en la instancia. Sopese la posibilidad de drenar la instancia de

contenedor y transferir las tareas importantes a otras instancias de contenedor antes de activar el modo de depuración. Para obtener más información, consulte Drenaje de instancias de contenedor de Amazon ECS.

```
[ec2-user ~]$ sudo bash ./ecs-logs-collector.sh
```

> **Important**
>
> Recomendamos que edite los registros y elimine toda la información confidencial de los archivos. Puede buscar datos conocidos y también variables de entorno, como `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` y `AWS_SESSION_TOKEN`, en el archivo.

Después de haber ejecutado el script, puede examinar los registros recopilados en la carpeta `collect` creada por el script. El archivo `collect.tgz` es un archivo comprimido que contiene todos los registros, y lo puede compartir con AWS Support para facilitar el diagnóstico.

Para descargar y ejecutar el recopilador de registros de Amazon ECS para Windows

1. Conéctese a la instancia de contenedor. Para obtener más información, consulte Conectarse a una instancia de Windows mediante RDP en la Guía del usuario de Amazon EC2.
2. Descargue el script del recopilador de registros de Amazon ECS a través de PowerShell.

```
Invoke-WebRequest -OutFile ecs-logs-collector.ps1 https://
raw.githubusercontent.com/awslabs/aws-ecs-logs-collector-for-windows/master/ecs-
logs-collector.ps1
```

3. Ejecute el script para recopilar los registros y crear el archivo.

> **Note**
>
> Para habilitar el modo de depuración para el daemon de Docker y el agente de contenedor de Amazon ECS, agregue la opción `-RunMode debug` al siguiente comando. Esto reinicia el daemon de Docker, que cierra todos los contenedores que están en ejecución en la instancia. Sopese la posibilidad de drenar la instancia de contenedor y transferir las tareas importantes a otras instancias de contenedor antes

de activar el modo de depuración. Para obtener más información, consulte Drenaje de instancias de contenedor de Amazon ECS.

```
.\ecs-logs-collector.ps1
```

> **Important**
>
> Recomendamos que edite los registros y elimine toda la información confidencial de los archivos. Puede buscar datos conocidos y también variables de entorno, como `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` y `AWS_SESSION_TOKEN`, en el archivo.

Después de haber ejecutado el script, puede examinar los registros recopilados en la carpeta `collect` creada por el script. El archivo `collect.tgz` es un archivo comprimido con todos los registros, que puede compartir con AWS Support para facilitar el diagnóstico.

Recuperación de los detalles de diagnóstico de Amazon ECS con la introspección del agente

La API de introspección del agente de Amazon ECS proporciona información sobre el estado general del agente de Amazon ECS y las instancias de contenedor.

Puede utilizar la API de introspección del agente para obtener el ID de Docker de un contenedor en su tarea. Puede utilizar la API de introspección del agente conectando a una instancia de contenedor mediante SSH.

> **Important**
>
> La instancia de contenedor debe tener un rol de IAM que permita el acceso a Amazon ECS para poder conectar con la API de introspección. Para obtener más información, consulte Rol de IAM de instancia de contenedor de Amazon ECS.

En el ejemplo siguiente se muestran dos tareas: una que está actualmente en ejecución y otra que se detuvo.

> **Note**
>
> El comando siguiente se canaliza a través de python -mjson.tool para mayor legibilidad.

```
curl http://localhost:51678/v1/tasks | python -mjson.tool
```

Salida:

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1095  100  1095    0     0   117k      0 --:--:-- --:--:-- --:--:--  133k
{
    "Tasks": [
        {
            "Arn": "arn:aws:ecs:us-west-2:aws_account_id:task/090eff9b-1ce3-4db6-848a-
a8d14064fd24",
            "Containers": [
                {
                    "DockerId":
 "189a8ff4b5f04affe40e5160a5ffadca395136eb5faf4950c57963c06f82c76d",
                    "DockerName": "ecs-console-sample-app-static-6-simple-
app-86caf9bcabe3e9c61600",
                    "Name": "simple-app"
                },
                {
                    "DockerId":
 "f7f1f8a7a245c5da83aa92729bd28c6bcb004d1f6a35409e4207e1d34030e966",
                    "DockerName": "ecs-console-sample-app-static-6-busybox-
ce83ce978a87a890ab01",
                    "Name": "busybox"
                }
            ],
            "Family": "console-sample-app-static",
            "KnownStatus": "STOPPED",
            "Version": "6"
        },
        {
            "Arn": "arn:aws:ecs:us-west-2:aws_account_id:task/1810e302-eaea-4da9-
a638-097bea534740",
            "Containers": [
                {
```

```
                    "DockerId":
 "dc7240fe892ab233dbbcee5044d95e1456c120dba9a6b56ec513da45c38e3aeb",
                    "DockerName": "ecs-console-sample-app-static-6-simple-app-
f0e5859699a7aecfb101",
                    "Name": "simple-app"
                },
                {
                    "DockerId":
 "096d685fb85a1ff3e021c8254672ab8497e3c13986b9cf005cbae9460b7b901e",
                    "DockerName": "ecs-console-sample-app-static-6-
busybox-92e4b8d0ecd0cce69a01",
                    "Name": "busybox"
                }
            ],
            "DesiredStatus": "RUNNING",
            "Family": "console-sample-app-static",
            "KnownStatus": "RUNNING",
            "Version": "6"
        }
    ]
}
```

En el ejemplo anterior, la tarea detenida (*090eff9b-1ce3-4db6-848a-a8d14064fd24*) tiene dos contenedores. Puede utilizar docker inspect ***container-ID*** para consultar información detallada en cada contenedor. Para obtener más información, consulte Introspección de contenedor de Amazon ECS.

Diagnósticos de Docker en Amazon ECS

Docker ofrece varias herramientas de diagnóstico que le ayudan a solucionar problemas en sus contenedores y tareas. Para obtener más información sobre todas las utilidades de línea de comandos de Docker disponibles, consulte la referencia de la CLI de Docker en la documentación de Docker. Puede obtener acceso a las utilidades de línea de comando de Docker conectando a una instancia de contenedor mediante SSH.

Los códigos de salida que notifican los contenedores de Docker también pueden facilitar información de diagnóstico (por ejemplo, el código de salida 137 significa que el contenedor recibió una señal `SIGKILL`). Para obtener más información, consulte Exit Status en la documentación de Docker.

Enumeración de los contenedores de Docker en Amazon ECS

Puede utilizar el comando docker ps en la instancia de contenedor para enumerar los contenedores en ejecución. En el ejemplo siguiente, solo se está ejecutando el agente de contenedor de Amazon ECS. Para obtener más información, consulte docker ps en la documentación de Docker.

```
docker ps
```

Salida:

```
CONTAINER ID        IMAGE                               COMMAND             CREATED
      STATUS              PORTS                         NAMES
cee0d6986de0        amazon/amazon-ecs-agent:latest   "/agent"            22 hours ago
      Up 22 hours         127.0.0.1:51678->51678/tcp   ecs-agent
```

Puede utilizar el comando docker ps -a para ver todos los contenedores (incluso los contenedores parados o cancelados). Esto resulta útil para mostrar contenedores que se han parado de forma imprevista. En el ejemplo siguiente, el contenedor `f7f1f8a7a245` finalizó hace 9 segundos, por lo que no aparece en una salida docker ps sin el indicador `-a`.

```
docker ps -a
```

Salida:

```
CONTAINER ID        IMAGE                                          COMMAND
 CREATED             STATUS                         PORTS                        NAMES
db4d48e411b1        amazon/ecs-emptyvolume-base:autogenerated   "not-applicable"
  19 seconds ago                                                                  ecs-
console-sample-app-static-6-internalecs-emptyvolume-source-c09288a6b0cba8a53700
f7f1f8a7a245        busybox:buildroot-2014.02                   "\"sh -c '/bin/sh -c
  22 hours ago        Exited (137) 9 seconds ago                                  ecs-
console-sample-app-static-6-busybox-ce83ce978a87a890ab01
189a8ff4b5f0        httpd:2                                     "httpd-foreground"
  22 hours ago        Exited (137) 40 seconds ago                                 ecs-
console-sample-app-static-6-simple-app-86caf9bcabe3e9c61600
0c7dca9321e3        amazon/ecs-emptyvolume-base:autogenerated   "not-applicable"
  22 hours ago                                                                    ecs-
console-sample-app-static-6-internalecs-emptyvolume-source-90fefaa68498a8a80700
```

```
cee0d6986de0        amazon/amazon-ecs-agent:latest              "/agent"
  22 hours ago        Up 22 hours                   127.0.0.1:51678->51678/tcp   ecs-
agent
```

Visualización de los registros de Docker en Amazon ECS

Puede ver los flujos `STDOUT` y `STDERR` para un contenedor con el comando docker logs. En este ejemplo, los registros se muestran para el contenedor *`dc7240fe892a`* y se canalizan a través del comando head por razones de brevedad. Para obtener más información, acceda a docker logs en la documentación de Docker.

> **Note**
>
> Los registros de Docker solo están disponibles en la instancia del contenedor si utiliza el controlador de registro `json` predeterminado. Si ha configurado las tareas para que utilicen el controlador de registros `awslogs`, sus registros de contenedor estarán disponibles en CloudWatch Logs. Para obtener más información, consulte Envío de registros de Amazon ECS a CloudWatch .

```
docker logs dc7240fe892a | head
```

Salida:

```
AH00558: httpd: Could not reliably determine the server's fully qualified domain name,
 using 172.17.0.11. Set the 'ServerName' directive globally to suppress this message
AH00558: httpd: Could not reliably determine the server's fully qualified domain name,
 using 172.17.0.11. Set the 'ServerName' directive globally to suppress this message
[Thu Apr 23 19:48:36.956682 2015] [mpm_event:notice] [pid 1:tid 140327115417472]
 AH00489: Apache/2.4.12 (Unix) configured -- resuming normal operations
[Thu Apr 23 19:48:36.956827 2015] [core:notice] [pid 1:tid 140327115417472] AH00094:
 Command line: 'httpd -D FOREGROUND'
10.0.1.86 - - [23/Apr/2015:19:48:59 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:48:59 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:49:28 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:49:29 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:49:50 +0000] "-" 408 -
10.0.0.154 - - [23/Apr/2015:19:49:50 +0000] "-" 408 -
10.0.1.86 - - [23/Apr/2015:19:49:58 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:49:59 +0000] "GET / HTTP/1.1" 200 348
```

```
10.0.1.86 - - [23/Apr/2015:19:50:28 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:50:29 +0000] "GET / HTTP/1.1" 200 348
time="2015-04-23T20:11:20Z" level="fatal" msg="write /dev/stdout: broken pipe"
```

Inspección de los contenedores de Docker en Amazon ECS

Si dispone del ID de Docker de un contenedor, puede inspeccionarlo con el comando docker inspect. La inspección de contenedores ofrece la vista más detallada del entorno en el que se puede lanzar un contenedor. Para obtener más información, consulte docker inspect en la documentación de Docker.

```
docker inspect dc7240fe892a
```

Salida:

```
[{
    "AppArmorProfile": "",
    "Args": [],
    "Config": {
        "AttachStderr": false,
        "AttachStdin": false,
        "AttachStdout": false,
        "Cmd": [
            "httpd-foreground"
        ],
        "CpuShares": 10,
        "Cpuset": "",
        "Domainname": "",
        "Entrypoint": null,
        "Env": [
            "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/
local/apache2/bin",
            "HTTPD_PREFIX=/usr/local/apache2",
            "HTTPD_VERSION=2.4.12",
            "HTTPD_BZ2_URL=https://www.apache.org/dist/httpd/httpd-2.4.12.tar.bz2"
        ],
        "ExposedPorts": {
            "80/tcp": {}
        },
        "Hostname": "dc7240fe892a",
...
```

Configuración de la salida detallada desde el daemon de Docker en Amazon ECS

Si tiene problemas con los contenedores o imágenes de Docker, puede activar el modo de depuración en su daemon de Docker. El uso de la depuración proporciona una salida más detallada del daemon. Puede utilizarla para recuperar los mensajes de error que se envían desde los registros de contenedores, como Amazon ECR.

> **Important**
>
> Este procedimiento está escrito para la AMI de Amazon Linux optimizada para Amazon ECS. Para los demás sistemas operativos, consulte Enable debugging y Control and configure Docker with systemd en la documentación de Docker.

Para usar el modo de depuración del daemon de Docker en la AMI de Amazon Linux optimizada para Amazon ECS

1. Conéctese a la instancia de contenedor.
2. Abra el archivo de opciones de Docker con un editor de texto, como vi. Para la AMI de Amazon Linux optimizada para Amazon ECS, el archivo de opciones de Docker se encuentra en `/etc/sysconfig/docker`.
3. Busque la declaración de opciones de Docker y añada la opción `-D` a la cadena, dentro de las comillas.

   > **Note**
   >
   > Si la declaración de opciones de Docker comienza por #, elimine ese carácter para quitar el comentario de la declaración y habilitar las opciones.

   Para la AMI de Amazon Linux optimizada para Amazon ECS, la declaración de opciones de Docker se llama `OPTIONS`. Por ejemplo:

```
# Additional startup options for the Docker daemon, for example:
# OPTIONS="--ip-forward=true --iptables=true"
# By default we limit the number of open files per container
```

```
OPTIONS="-D --default-ulimit nofile=1024:4096"
```

4. Guarde el archivo y salga del editor de texto.
5. Reinicie el daemon de Docker.

```
sudo service docker restart
```

La salida es la siguiente:

```
Stopping docker:                                           [  OK  ]
Starting docker: .                                   [  OK  ]
```

6. Reinicie el agente de Amazon ECS.

```
sudo service ecs restart
```

Sus registros de Docker ahora deberían mostrar una salida más detallada.

```
time="2015-12-30T21:48:21.907640838Z" level=debug msg="Unexpected response from
 server: \"{\\\"errors\\\":[{\\\"code\\\":\\\"DENIED\\\",\\\"message\\\":\\\"User:
 arn:aws:sts::1111:assumed-role/ecrReadOnly/i-abcdefg is not authorized to perform:
 ecr:InitiateLayerUpload on resource: arn:aws:ecr:us-east-1:1111:repository/nginx_test
\\\"}]}\\n\" http.Header{\"Connection\":[]string{\"keep-alive\"}, \"Content-Type\":
[]string{\"application/json; charset=utf-8\"}, \"Date\":[]string{\"Wed, 30 Dec 2015
 21:48:21 GMT\"}, \"Docker-Distribution-Api-Version\":[]string{\"registry/2.0\"},
 \"Content-Length\":[]string{\"235\"}}"
```

Solución del problema de Docker `API error (500): devmapper` en Amazon ECS

El siguiente error de Docker indica que el almacenamiento en grupo fino en la instancia de contenedor está lleno y que el daemon de Docker no puede crear nuevos contenedores:

```
CannotCreateContainerError: API error (500): devmapper: Thin Pool has 4350 free data
 blocks which is less than minimum required 4454 free data blocks. Create more free
 space in thin pool or use dm.min_free_space option to change behavior
```

La versión `2015.09.d` y posteriores de las AMI de Amazon Linux optimizadas para Amazon ECS se lanzan, de forma predeterminada, con un volumen de 8 GiB para el sistema operativo que se asocia a `/dev/xvda` y se monta como la raíz del sistema de archivos. Existe un volumen adicional de 22 GiB asociado a `/dev/xvdcz` que utiliza Docker para el almacenamiento de metadatos e imágenes. Si este espacio de almacenamiento está lleno, el daemon de Docker no puede crear nuevos contenedores.

La forma más sencilla de agregar almacenamiento a sus instancias de contenedor consiste en terminar las instancias existentes y lanzar otras nuevas con volúmenes de almacenamiento de datos mayores. No obstante, si no puede hacerlo, puede agregar almacenamiento al grupo de volumen que utiliza Docker y ampliar el volumen lógico mediante los procedimientos que se indican en AMI de Linux optimizadas para Amazon ECS.

Si el almacenamiento de la instancia de contenedor se llena demasiado rápido, hay algunas acciones que puede realizar para reducir este efecto:

- Para consultar la información de sondeo ligero, ejecute el siguiente comando en la instancia de contenedor:

  ```
  docker info
  ```

- (Agente de contenedor de Amazon ECS, versión 1.8.0 y posteriores) Puede reducir la cantidad de tiempo que los contenedores detenidos o cerrados permanecen en las instancias de contenedor. La variable de configuración del agente `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` establece la duración de tiempo para esperar desde que se para una tarea hasta que se elimina el contenedor de Docker (de forma predeterminada, este valor es 3 horas). Esto elimina los datos del contenedor de Docker. Si este valor se establece demasiado bajo, es posible que no pueda inspeccionar los contenedores detenidos o ver los registros antes de que se supriman. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS.
- Puede eliminar contenedores sin ejecución e imágenes no utilizadas de las instancias de contenedor. Puede utilizar los siguientes comandos de ejemplo para eliminar manualmente contenedores parados e imágenes sin utilizar. Los contenedores eliminados no se pueden inspeccionar más adelante, y las imágenes eliminadas se deben volver a extraer antes de comenzar nuevos contenedores desde ellas.

  Para eliminar contenedores que no están en ejecución, ejecute el siguiente comando en la instancia de contenedor:

```
docker rm $(docker ps -aq)
```

Para eliminar imágenes sin utilizar, ejecute el siguiente comando en la instancia de contenedor:

```
docker rmi $(docker images -q)
```

- Puede eliminar bloques de datos no utilizados dentro de contenedores. Puede utilizar el comando siguiente para ejecutar fstrim en cualquier contenedor en ejecución y descartar cualquier bloque de datos que no utilice el sistema de archivos del contenedor.

```
sudo sh -c "docker ps -q | xargs docker inspect --format='{{ .State.Pid }}' | xargs -
IZ fstrim /proc/Z/root/"
```

Solución de problemas de Amazon ECS Exec

A continuación, se incluyen notas de solución de problemas que ayudan a diagnosticar por qué puede aparecer un error al utilizar ECS Exec.

Verificación mediante Exec Checker

El script de ECS Exec Checker proporciona una forma de verificar y validar que el clúster y la tarea de Amazon ECS cumplan los requisitos previos para utilizar la característica ECS Exec. El script de ECS Exec Checker verifica que tanto su entorno de AWS CLI como su clúster y tareas estén listos para ECS Exec, llamando a varias API en su nombre. La herramienta requiere la versión más reciente de la AWS CLI y que esté disponible `jq`. Para obtener más información, consulte ECS Exec Checker en GitHub.

Error al ejecutar **`execute-command`**

Si se produce un error `The execute command failed`, las causas pueden ser estas.

- La tarea no cuenta con los permisos requeridos. Compruebe que la definición de tareas utilizada para lanzar la tarea tenga un rol de IAM para tareas definido y que el rol cuente con los permisos requeridos. Para obtener más información, consulte Permisos de ECS Exec.
- SSM Agent no está instalado o no está en ejecución.
- Existe un punto de conexión de Amazon VPC de interfaz para Amazon ECS, pero no hay uno para el Administrador de sesiones de Systems Manager.

Solución de problemas de Amazon ECS Anywhere

Amazon ECS Anywhere admite el registro de una instancia externa, por ejemplo, un servidor ubicado en las instalaciones o una máquina virtual (VM), en el clúster de Amazon ECS. A continuación, se indican problemas comunes que puede encontrar y recomendaciones generales para solucionarlos.

Temas

- Problemas de registro de instancias externas
- Problemas de red de instancias externas
- Problemas al ejecutar tareas en la instancia externa

Problemas de registro de instancias externas

Cuando se registra una instancia externa en el clúster de Amazon ECS, se deben cumplir los siguientes requisitos:

- Se debe recuperar una activación de AWS Systems Manager, que consiste en un ID de activación y un código de activación. Se utiliza para registrar la instancia externa como una instancia administrada por Systems Manager. Cuando se solicita una activación de Systems Manager, especifique un límite de registro y una fecha de vencimiento. El límite de registro especifica el número máximo de instancias que se pueden registrar mediante la activación. El valor predeterminado para el límite de registro es 1 instancia. La fecha de vencimiento especifica cuándo vence la activación. El valor de predeterminado es 24 horas. Si la activación de Systems Manager que está utilizando para registrar su instancia externa no es válida, solicite una nueva. Para obtener más información, consulte Registro de una instancia externa en un clúster de Amazon ECS.
- Se utiliza una política de IAM para proporcionar a la instancia externa los permisos que necesita para comunicarse con las operaciones de las API de AWS. Si esta política administrada no se crea correctamente y no contiene los permisos requeridos, se produce un error de registro de la instancia externa. Para obtener más información, consulte Rol de IAM de Amazon ECS Anywhere.
- Amazon ECS proporciona un script de instalación que instala Docker, el agente de contenedor de Amazon ECS y Systems Manager Agent en la instancia externa. Si el script de instalación falla, es probable que no se pueda volver a ejecutar en la misma instancia sin que se produzca un error. Si esto sucede, siga el proceso de limpieza para borrar los recursos de AWS de la instancia y poder ejecutar de nuevo el script de instalación. Para obtener más información, consulte Anulación del registro de una instancia externa de Amazon ECS.

> **Note**
>
> Tenga en cuenta que, si el script de instalación solicitó y utilizó correctamente la activación de Systems Manager, la ejecución del script de instalación por segunda vez vuelve a utilizar la activación de Systems Manager. Esto podría, a su vez, hacer que alcance el límite de registros para esa activación. Si se alcanza este límite, debe crear una nueva activación.

- Al ejecutar el script de instalación en una instancia externa para las cargas de trabajo de GPU, si el controlador NVIDIA no se detecta o configura correctamente, se producirá un error. El script de instalación utiliza el comando `nvidia-smi` para confirmar la existencia del controlador NVIDIA.

Problemas de red de instancias externas

Para comunicar cualquier cambio, la instancia externa requiere una conexión de red a AWS. Si su instancia externa pierde la conexión de red a AWS, las tareas que se están ejecutando en las instancias continúan haciéndolo de todos modos, a menos que se detengan manualmente. Una vez que se restablece la conexión con AWS, las credenciales de AWS que utilizan el agente de contenedor de Amazon ECS y Systems Manager Agent en la instancia externa se renuevan automáticamente. Para obtener más información acerca de los dominio de AWS que se utilizan para la comunicación entre la instancia externa y AWS, consulte Red .

Problemas al ejecutar tareas en la instancia externa

Las causas más comunes de que las tareas o los contenedores no se ejecutan en la instancia externa son la red o los permisos relacionados. Si los contenedores extraen las imágenes de Amazon ECR o están configurados para enviar registros de contenedores a CloudWatch Logs, la definición de tareas debe especificar un rol de IAM de ejecución de tareas válido. Sin un rol de IAM de ejecución de tareas válido, los contenedores no se iniciarán. Para obtener más información acerca de los problemas relacionados con la red, consulte Problemas de red de instancias externas.

> **Important**
>
> Amazon ECS proporciona la herramienta de recopilación de registros de Amazon ECS. Puede utilizarla para recopilar registros de las instancias externas para fines de resolución de

problemas. Para obtener más información, consulte Recopilación de registros de contenedor con el recopilador de registros de Amazon ECS .

Cuotas de limitación de AWS Fargate

AWS Fargate limita las tareas de Amazon ECS y las tarifas de lanzamiento de los pods de Amazon EKS a las cuotas (antes denominadas límites) mediante un algoritmo de bucket de token para cada cuenta de AWS por región. Con este algoritmo, su cuenta tiene un bucket que contiene un número específico de tokens. El número de tokens del bucket representa su cuota de tasa en un segundo determinado. Cada cuenta de cliente tiene un bucket de tokens de tareas y pods que se agota en función del número de tareas y pods lanzados por la cuenta de cliente. Este bucket de tokens tiene un máximo de bucket que le permite realizar periódicamente un mayor número de solicitudes y una tasa de recarga que le permite mantener una tasa constante de solicitudes durante el tiempo que sea necesario.

Por ejemplo, el tamaño del bucket de tokens para tareas y pods de una cuenta de cliente de Fargate es de 100 tokens y la tasa de recarga es de 20 tokens por segundo. Por lo tanto, puede lanzar inmediatamente hasta 100 tareas de Amazon ECS y pods de Amazon EKS por cuenta de cliente, con una tasa de lanzamiento sostenida de 20 tareas de Amazon ECS y pods de Amazon EKS por segundo.

Acciones	Capacidad máxima del bucket (o velocidad de ráfaga)	Tasa de recarga del bucket (o tasa sostenida)
Cuota de tasa de recursos de Fargate para tareas bajo demanda de Amazon ECS y pods de Amazon EKS[1]	100	20
Cuota de tasa de recursos de Fargate para tareas de Spot Amazon ECS	100	20

[1]Las cuentas que lanzan solo los pods de Amazon EKS tienen una tasa de ráfaga de 20, con una tasa de lanzamiento sostenido de pods de 20 lanzamientos de pods por segundo cuando se utilizan las versiones de plataforma indicadas en las Versiones de la plataforma de Amazon EKS.

Limitación de la API **RunTask** en Fargate

Además, Fargate limita la tasa de solicitudes al lanzar tareas con la API `RunTask` de Amazon ECS utilizando una cuota independiente. Fargate limita las solicitudes de la API `RunTask` de Amazon ECS de manera controlada para cada cuenta de AWS por región. Cada solicitud que realice elimina un token del bucket. Hacemos esto para mejorar el rendimiento del servicio y garantizar un utilización justa para todos los clientes de Fargate. Las llamadas de las API están sujetas a los cuotas de solicitud tanto si se originan en la consola de Amazon Elastic Container Service, en herramientas de la línea de comandos o en una aplicación de terceros. La cuota de tarifas para las llamadas a la API `RunTask` de Amazon ECS es de 20 llamadas por segundo (ráfaga y sostenida). Sin embargo, cada llamada a esta API puede iniciar hasta 10 tareas. Esto significa que puede lanzar 100 tareas en un segundo haciendo 10 llamadas a esta API, solicitando que se inicien 10 tareas en cada llamada. Del mismo modo, también podría realizar 20 llamadas a esta API, solicitando que se inicien 5 tareas en cada llamada. Para obtener más información acerca de la limitación de API para la API `RunTask` de Amazon ECS, consulte API request throttling en la Referencia de la API de Amazon ECS.

En la práctica, las tasas de lanzamiento de tareas y pods también dependen de otras consideraciones, como las imágenes de contenedores que se van a descargar y desempaquetar, las comprobaciones de estado y otras integraciones habilitadas, como registrar tareas o pods en un equilibrador de carga. Los clientes ven variaciones en las tasas de lanzamiento de tareas y pods en comparación con las cuotas representadas anteriormente según las características que los clientes habilitan.

Ajuste de cuotas de tarifas en Fargate

Puede solicitar un aumento de las cuotas de limitación de Fargate para su cuenta de AWS. Para obtener más información, consulte Requesting a quota increase (Solicitud de un aumento de cuota) en la Guía del usuario de Cuotas de servicio.

Motivos de error de la API de Amazon ECS

Cuando una acción de API que haya desencadenado a través de la API de Amazon ECS, la consola o la AWS CLI genera un mensaje de error `failures`, puede seguir estos pasos para ayudar a solucionar la causa. El error devuelve un motivo y el nombre de recurso de Amazon (ARN) del recurso asociado al error.

Muchos recursos son específicos de la región, por lo que debe asegurarse de configurar la región correcta para los recursos cuando utilice la consola. Cuando utilice la AWS CLI, asegúrese de que

los comandos de la AWS CLI se envíen a la región correcta mediante el parámetro `--region` *`region`*.

Para obtener más información acerca de la estructura del tipo de datos `Failure`, consulte Error en la Referencia de la API de Amazon Elastic Container Service.

A continuación, se muestran ejemplos de mensajes de error que puede recibir al ejecutar comandos de la API.

Acción de la API	Motivo del error o motivo de la detención	Causa
`DescribeClusters`	`MISSING`	No se encontró el clúster especificado. Compruebe la ortografía del nombre del clúster.
`DescribeInstances`	`MISSING`	No se encontró la instancia de contenedor especificada. Compruebe que se haya especificado el clúster en el que está registrada la instancia de contenedor y que tanto el ARN como el ID de la instancia del contenedor sean correctos.
`DescribeServices`	`MISSING`	No se encontró el servicio especificado. Compruebe que se haya especificado el clúster o la región correctos y que el ARN o el nombre del servicio sean válidos.
`DescribeTasks`	`MISSING`	No se encontró la tarea especificada. Compruebe que se haya especificado el clúster o la región correctos y que

Acción de la API	Motivo del error o motivo de la detención	Causa
		tanto el ID como el ARN de la tarea sean válidos.

Acción de la API	Motivo del error o motivo de la detención	Causa
`DescribeTasks`	`TaskFailedToStart: RESOURCE:*`	En el caso de los errores de `RESOURCE:CPU` , la cantidad de CPU solicitadas por la tarea no está disponible en la instancia de contenedor. Esto suele ocurrir cuando el requisito de unidades de CPU en la definición de la tarea es mayor que el tamaño de la CPU de las instancias de Amazon EC2 definidas en el grupo de escalado automátic o asignado al proveedor de capacidad. Debe comprobar la configuración del proveedor de capacidad. En el caso de los errores de `RESOURCE:MEMORY` , la cantidad de memoria solicitada por la tarea no está disponible en la instancia de contenedor. Esto suele ocurrir cuando el requisito de cantidad de memoria en la definición de la tarea es mayor que la memoria admitida en las instancias de Amazon EC2 definidas en el grupo de escalado automátic o asignado al proveedor de capacidad. Debe comprobar

Acción de la API	Motivo del error o motivo de la detención	Causa
		la configuración del proveedor de capacidad.
	`TaskFailedToStart: AGENT`	La instancia de contenedor en la que se ha intentado lanzar una tarea tiene un agente que está desconectado actualmente. Para evitar tiempos de espera prolongados para la ubicación de tareas, se rechazó la solicitud. Para obtener información acerca de cómo solucionar problemas de un agente desconectado, consulte How do I troubleshoot a disconnected Amazon ECS agent (¿Cómo soluciono problemas de un agente de Amazon ECS desconectado?).
	`TaskFailedToStart: MemberOf placement constraint unsatisfied`	No hay ninguna instancia de contenedor que cumpla con las restricciones de ubicación definidas en la definición de la tarea.

Acción de la API	Motivo del error o motivo de la detención	Causa
	`TaskFailedToStart: ATTRIBUTE`	La definición de tareas contiene un parámetro que requiere un atributo de instancia de contenedo r específico que no está disponible en las instancias de contenedor. Por ejemplo, si su tarea usa el modo de red `awsvpc`, pero no hay ninguna instancia en sus subredes especificadas con el atributo `ecs.capability.tas k-eni` . Para obtener más información sobre qué atributos son necesarios para determinados parámetro s de definición de tareas y variables de configuración del agente, consulte Parámetro s en la definición de tareas de Amazon ECS para el tipo de lanzamiento de Fargate y Configuración del agente de contenedor de Amazon ECS.

Acción de la API	Motivo del error o motivo de la detención	Causa
	`TaskFailedToStart: NO ACTIVE INSTANCES`	No hay instancias activas en su proveedor de capacidad . Para obtener informaci ón sobre cómo administr ar grupos de escalado automático, consulte Grupos de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling.
	`TaskFailedToStart: EMPTY CAPACITY PROVIDER`	No hay instancias en su clúster. Lo más probable es que se deba a un proveedor de capacidad vacío o a que las instancias del proveedor de capacidad no estén registradas en el clúster. Para obtener información sobre cómo administrar grupos de escalado automático, consulte Grupos de escalado automátic o en la Guía del usuario de Amazon EC2 Auto Scaling.
`GetTaskProtection`	`MISSING`	No se encontró la tarea especificada. Compruebe que el nombre o ARN del clúster y el ID o ARN de la tarea sean válidos.

Acción de la API	Motivo del error o motivo de la detención	Causa
	`TASK_NOT_VALID`	La tarea especificada no forma parte de un servicio de Amazon ECS. Solo se pueden proteger las tareas administr adas por servicios de Amazon ECS. Compruebe el ARN o ID de la tarea e inténtelo de nuevo.

Acción de la API	Motivo del error o motivo de la detención	Causa
`RunTask` o `StartTask`	`RESOURCE:*`	Los recursos solicitados por la tarea no están disponibles en la instancia de contenedor del clúster. Si el recurso es CPU, memoria, puertos o interfaces de red elásticas, es posible que tenga que agregar instancias de contenedor al clúster. En el caso de errores `RESOURCE:ENI` , el clúster no tiene ningún punto de asociación disponible a la interfaz de red elástica. Las tareas que utilizan el modo de red `awsvpc` necesitan estos puntos de asociación. Las instancias de Amazon EC2 presentan un límite en cuanto al número de interfaces de red que pueden tener asociadas, y la interfaz de red principal cuenta como una. Para obtener más información acerca de cuántas interfaces de red se admiten para cada tipo de instancias, consulte Direcciones IP por interfaz de red por tipo de instancia en la Guía del usuario de Amazon EC2.

Acción de la API	Motivo del error o motivo de la detención	Causa
		En el caso de errores `RESOURCE:GPU` , el número de GPU solicitadas por la tarea no está disponible y es posible que tenga que agregar instancias de contenedor habilitadas para GPU al clúster. Para obtener más información, consulte Definiciones de tareas de Amazon ECS para cargas de trabajo de GPU.
	`AGENT`	La instancia de contenedor en la que se ha intentado lanzar una tarea tiene un agente que está desconectado actualmente. Para evitar tiempos de espera prolongados para la ubicación de tareas, se rechazó la solicitud. Para obtener información acerca de cómo solucionar problemas de un agente desconectado, consulte How do I troubleshoot a disconnected Amazon ECS agent (¿Cómo soluciono problemas de un agente de Amazon ECS desconectado?).

Acción de la API	Motivo del error o motivo de la detención	Causa
	`LOCATION`	La instancia de contenedor en la que intentó lanzar una tarea se encuentra en una zona de disponibilidad distinta a la de las subredes especificadas en la `awsVpcConfiguration`.
	`ATTRIBUTE`	La definición de tareas contiene un parámetro que requiere un atributo de instancia de contenedor específico que no está disponible en las instancias de contenedor. Por ejemplo, si su tarea usa el modo de red `awsvpc`, pero no hay ninguna instancia en sus subredes especificadas con el atributo `ecs.capability.task-eni`. Para obtener más información sobre qué atributos son necesarios para determinados parámetros de definición de tareas y variables de configuración del agente, consulte Parámetros en la definición de tareas de Amazon ECS para el tipo de lanzamiento de Fargate y Configuración del agente de contenedor de Amazon ECS.

Acción de la API	Motivo del error o motivo de la detención	Causa
`StartTask`	`MISSING`	No se encuentra la instancia de contenedor en la que ha intentado lanzar la tarea. Compruebe si se ha especificado un clúster o una región incorrectos o si el ID o el ARN de la instancia de contenedor estén mal escritos.
	`INACTIVE`	Anteriormente se anuló el registro en Amazon ECS de la instancia de contenedor en la que ha intentado lanzar una tarea, y no se puede utilizar.
`StopServiceDeployment`	`ECS deployment failed`	Una cuenta fraudulenta ejecutó la API `StopServiceDeployment` .
`TagResource`	`InvalidParameterException`	El ARN del servicio que está etiquetando tiene el formato corto. Debe migrar al formato largo. Para obtener información acerca de migrar el ARN, consulte Migración de un ARN de servicio corto de Amazon ECS a un ARN largo.

Acción de la API	Motivo del error o motivo de la detención	Causa
`UpdateTaskProtection`	`DEPLOYMENT_BLOCKED`	No se puede configurar la protección de tareas, ya que una o más tareas protegida s impiden que la implement ación del servicio alcance un estado estable. Desactive la protección en las tareas existentes o espere hasta que esta caduque.
	`MISSING`	No se encontró la tarea especificada. Compruebe que el nombre o ARN del clúster y el ID o ARN de la tarea sean válidos.
	`TASK_NOT_VALID`	La tarea especificada no forma parte de un servicio de Amazon ECS. Solo se pueden proteger las tareas administr adas por servicios de Amazon ECS. Compruebe el ARN o ID de la tarea e inténtelo de nuevo.

> **Note**
>
> Además de las situaciones de error descritas aquí, las operaciones de las API también pueden fallar debido a excepciones, lo que genera respuestas de error. Para obtener una lista de estas excepciones, consulte Common Errors (Errores comunes).

Seguridad en Amazon Elastic Container Service

La seguridad en la nube de AWS es la máxima prioridad. Como cliente de AWS, se beneficia de una arquitectura de red y un centro de datos que se han diseñado para satisfacer los requisitos de seguridad de las organizaciones más exigentes.

La seguridad es una responsabilidad compartida entre AWS y el usuario. 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 los servicios de AWS en la nube de AWS. AWS también proporciona servicios que puedes utilizar de forma segura. Auditores independientes prueban y verifican periódicamente la eficacia de nuestra seguridad en el marco de los programas de conformidad de AWS. Para obtener más información acerca de los programas de conformidad que se aplican a Amazon Elastic Container Service, consulte Servicios de AWS en el ámbito del programa de conformidad.
- Seguridad en la nube: su responsabilidad se determina según el servicio de AWSque utiliza. También eres responsable de otros factores, incluida la confidencialidad de los datos, los requisitos de la empresa y la legislación y la normativa aplicables.

Esta documentación lo ayuda a comprender cómo aplicar el modelo de responsabilidad compartida cuando se utiliza Amazon ECS. En los siguientes temas, se mostrará cómo configurar Amazon ECS a fin de cumplir sus objetivos de seguridad y conformidad. Además, descubra cómo utilizar otros servicios de AWS que ayudan a monitorear y proteger los recursos de Amazon ECS.

Temas

- Identity and Access Management para Amazon Elastic Container Service
- Registro y monitoreo en Amazon Elastic Container Service
- Validación de conformidad para Amazon Elastic Container Service
- AWS Fargate Estándar de procesamiento de la información federal (FIPS, Federal Information Processing Standard 140)
- Seguridad de la infraestructura de Amazon Elastic Container Service
- AWS shared responsibility model for Amazon ECS
- Prácticas recomendadas de seguridad de red para Amazon ECS
- Prácticas recomendadas de seguridad para las tareas y contenedores de Amazon ECS

Identity and Access Management para Amazon Elastic Container Service

AWS Identity and Access Management (IAM) es un Servicio de AWS que ayuda a los gestionadores a controlar de forma segura el acceso a los recursos de AWS. Los administradores de IAM controlan quién está autenticado (ha iniciado sesión) y autorizado (tiene permisos) para utilizar recursos de Amazon ECS. IAM es un servicio de Servicio de AWS que se puedes utilizar sin cargo adicional.

Temas

- Público
- Autenticación con identidades
- Administración de acceso mediante políticas
- Cómo funciona Amazon Elastic Container Service con IAM
- Ejemplos de políticas basadas en identidades de Amazon Elastic Container Service
- Políticas administradas por AWS para Amazon Elastic Container Service
- Uso de roles vinculados al servicio para Amazon ECS
- Roles de IAM para Amazon ECS
- Permisos necesarios para la consola de Amazon ECS
- Permisos necesarios para las funciones de Lambda en las implementaciones azul/verde de Amazon ECS
- Permisos de IAM necesarios para el escalado automático del servicio de Amazon ECS
- Conceder permisos para etiquetar recursos durante la creación
- Solución de problemas de identidad y acceso de Amazon Elastic Container Service
- Prácticas recomendadas de IAM para Amazon ECS

Público

La forma en que utiliza AWS Identity and Access Management (IAM) difiere en función del trabajo que realiza en Amazon ECS.

Usuario de servicio: si utiliza el servicio de Amazon ECS para realizar su trabajo, su administrador le proporciona las credenciales y los permisos que necesita. A medida que vaya utilizando más características de Amazon ECS para realizar su trabajo, es posible que necesite permisos

adicionales. Entender cómo se administra el acceso puede ayudarle a solicitar los permisos correctos al administrador. Si no puede acceder a una característica de Amazon ECS, consulte Solución de problemas de identidad y acceso de Amazon Elastic Container Service.

Administrador de servicio: si está a cargo de los recursos de Amazon ECS de su empresa, es probable que tenga acceso completo a Amazon ECS. Su trabajo consiste en determinar a qué características y recursos de Amazon ECS deben acceder sus usuarios del servicio. Luego, debe enviar solicitudes a su gestionador de IAM para cambiar los permisos de los usuarios de su servicio. Revise la información de esta página para conocer los conceptos básicos de IAM. Para obtener más información acerca de cómo la empresa puede utilizar IAM con Amazon ECS, consulte Cómo funciona Amazon Elastic Container Service con IAM.

Administrador de IAM: si es administrador de IAM, es posible que desee obtener detalles sobre cómo escribir políticas para administrar el acceso a Amazon ECS. Para consultar ejemplos de políticas de Amazon ECS basadas en identidades que puede utilizar en IAM, consulte Ejemplos de políticas basadas en identidades de Amazon Elastic Container Service.

Autenticación con identidades

La autenticación es la manera de iniciar sesión en AWS mediante credenciales de identidad. Debe estar autenticado (haber iniciado sesión en AWS) como Usuario raíz de la cuenta de AWS, como un usuario de IAM o asumiendo un rol de IAM.

Puede iniciar sesión en AWS como una identidad federada mediante las credenciales proporcionadas a través de una fuente de identidad. AWS IAM Identity Center Los usuarios (del IAM Identity Center), la autenticación de inicio de sesión único de su empresa y sus credenciales de Google o Facebook son ejemplos de identidades federadas. Al iniciar sesión como una identidad federada, su gestionador habrá configurado previamente la federación de identidades mediante roles de IAM. Cuando accede a AWS mediante la federación, está asumiendo un rol de forma indirecta.

Según el tipo de usuario que sea, puede iniciar sesión en AWS Management Console o en el portal de acceso AWS. Para obtener más información sobre el inicio de sesión en AWS, consulta Cómo iniciar sesión en su Cuenta de AWS en la Guía del usuario de AWS Sign-In.

Si accede a AWS mediante programación, AWS proporciona un kit de desarrollo de software (SDK) y una interfaz de la línea de comandos (CLI) para firmar criptográficamente las solicitudes mediante el uso de las credenciales. Si no usa las herramientas de AWS, debe firmar las solicitudes. Para obtener más información sobre la firma de solicitudes, consulte AWSSignature Versión 4 para solicitudes API en la Guía del usuario de IAM.

Independientemente del método de autenticación que use, es posible que deba proporcionar información de seguridad adicional. Por ejemplo, AWS le recomienda el uso de la autenticación multifactor (MFA) para aumentar la seguridad de su cuenta. Para obtener más información, consulte Autenticación multifactor en la Guía del usuario de AWS IAM Identity Center y Autenticación multifactor AWSen IAM en la Guía del usuario de IAM.

Usuario raíz de Cuenta de AWS

Cuando se crea una Cuenta de AWS, se comienza con una identidad de inicio de sesión que tiene acceso completo a todos los recursos y Servicios de AWS de la cuenta. Esta identidad recibe el nombre de usuario raíz de la Cuenta de AWS y se accede a ella iniciando sesión con el email y la contraseña que utilizó para crear la cuenta. Recomendamos encarecidamente que no utiliza el usuario raíz para sus tareas diarias. Proteja las credenciales del usuario raíz y utilícelas solo para las tareas que solo el usuario raíz pueda realizar. Para ver la lista completa de las tareas que requieren que inicie sesión como usuario raíz, consulta Tareas que requieren credenciales de usuario raíz en la Guía del usuario de IAM.

Identidad federada

Como práctica recomendada, solicite que los usuarios humanos, incluidos los que requieren acceso de administrador, utilizan la federación con un proveedor de identidades para acceder a los Servicios de AWS utilizando credenciales temporales.

Una identidad federada es un usuario del directorio de usuarios de su empresa, un proveedor de identidad web, el AWS Directory Service, el directorio del Identity Center, o cualquier usuario que acceda a Servicios de AWS utilizando credenciales proporcionadas a través de un origen de identidad. Cuando identidades federadas acceden a Cuentas de AWS, asumen roles y los roles proporcionan credenciales temporales.

Para una administración de acceso centralizada, le recomendamos que utiliza AWS IAM Identity Center. Puede crear usuarios y grupos en el IAM Identity Center o puede conectarse y sincronizar con un conjunto de usuarios y grupos de su propia fuente de identidad para usarlos en todas sus aplicaciones y Cuentas de AWS. Para obtener más información, consulta ¿Qué es el Centro de identidades de IAM? en la Guía del usuario de AWS IAM Identity Center.

Usuarios y grupos de IAM

Un usuario de IAM es una identidad de la Cuenta de AWSque dispone de permisos específicos para una sola persona o aplicación. Siempre que sea posible, recomendamos emplear credenciales temporales, en lugar de crear usuarios de IAM que tengan credenciales de larga duración como

contraseñas y claves de acceso. No obstante, si tiene casos de uso específicos que requieran credenciales de larga duración con usuarios de IAM, recomendamos rotar las claves de acceso. Para más información, consulte Rotar las claves de acceso periódicamente para casos de uso que requieran credenciales de larga duración en la Guía del usuario de IAM.

Un grupo de IAM es una identidad que especifica un conjunto de usuarios de IAM. No puedes iniciar sesión como grupo. Puedes usar los grupos para especificar permisos para varios usuarios a la vez. Los grupos facilitan la administración de los permisos de grandes conjuntos de usuarios. Por ejemplo, podría tener un grupo cuyo nombre fuese IAMAdmins y conceder permisos a dicho grupo para administrar los recursos de IAM.

Los usuarios son diferentes de los roles. Un usuario se asocia exclusivamente a una persona o aplicación, pero la intención es que cualquier usuario pueda asumir un rol que necesite. Los usuarios tienen credenciales de larga duración permanentes; no obstante, los roles proporcionan credenciales temporales. Para obtener más información, consulte Casos de uso para usuarios de IAM en la Guía del usuario de IAM.

Roles de IAM

Un rol de IAM es una identidad de la Cuenta de AWS que dispone de permisos específicos. Es similar a un usuario de IAM, pero no está asociado a una persona determinada. Para asumir temporalmente un rol de IAM en la AWS Management Console, puede cambiar de una función de usuario a un rol de IAM (consola). Puedes asumir un rol llamando a una operación de la AWS CLI o de la API de AWS, o utilizando una URL personalizada. Para más información sobre los métodos para el uso de roles, consulta Métodos para asumir un rol en la Guía del usuario de IAM.

Los roles de IAM con credenciales temporales son útiles en las siguientes situaciones:

- Acceso de usuario federado: para asignar permisos a una identidad federada, puede crear un rol y definir sus permisos. Cuando se autentica una identidad federada, se asocia la identidad al rol y se le conceden los permisos define el rol. Para obtener información acerca de roles de federación, consulte Crear un rol para un proveedor de identidad de terceros (federación) en la Guía de usuario de IAM. Si utiliza el IAM Identity Center, debe configurar un conjunto de permisos. IAM Identity Center correlaciona el conjunto de permisos con un rol en IAM para controlar a qué pueden acceder las identidades después de autenticarse. Para obtener información acerca de los conjuntos de permisos, consulta Conjuntos de permisos en la Guía del usuario de AWS IAM Identity Center.
- Permisos de usuario de IAM temporales: un usuario de IAM puede asumir un rol de IAM para recibir temporalmente permisos distintos que le permitan realizar una tarea concreta.

- Acceso entre cuentas: puede utilizar un rol de IAM para permitir que alguien (una entidad principal de confianza) de otra cuenta acceda a los recursos de la cuenta. Los roles son la forma principal de conceder acceso entre cuentas. No obstante, con algunos Servicios de AWS se puede adjuntar una política directamente a un recurso (en lugar de utilizar un rol como representante). Para obtener información acerca de la diferencia entre los roles y las políticas basadas en recursos para el acceso entre cuentas, consulta Acceso a recursos entre cuentas en IAM en la Guía del usuario de IAM.
- Acceso entre servicios: algunos Servicios de AWS utilizan características de otros Servicios de AWS. Por ejemplo, cuando realiza una llamada en un servicio, es común que ese servicio ejecute aplicaciones en Amazon EC2 o almacene objetos en Amazon S3. Es posible que un servicio haga esto usando los permisos de la entidad principal, usando un rol de servicio o usando un rol vinculado a servicios.
  - Reenviar sesiones de acceso (FAS): cuando utiliza un rol o un usuario de IAM para llevar a cabo las acciones en AWS, se le considera una entidad principal. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. FAS utiliza los permisos de la entidad principal para llamar a un Servicio de AWS, combinados con el Servicio de AWS solicitante para realizar solicitudes a servicios posteriores. Las solicitudes de FAS solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros Servicios de AWS o recursos para completarse. En este caso, debe tener permisos para realizar ambas acciones. Para obtener información sobre las políticas a la hora de realizar solicitudes de FAS, consulte Reenviar sesiones de acceso.
  - Rol de servicio: un rol de servicio es un rol de IAM que adopta 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 Creación de un rol para delegar permisos a un Servicio de AWS en la Guía del usuario de IAM.
  - Rol vinculado a los servicios: un rol vinculado a servicios 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 a servicios aparecen en la 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.
- Aplicaciones que se ejecutan en Amazon EC2: puede utilizar un rol de IAM que le permita administrar credenciales temporales para las aplicaciones que se ejecutan en una instancia de EC2 y realizan solicitudes a la AWS CLI o a la API de AWS. Es preferible hacerlo de este modo a almacenar claves de acceso en la instancia de EC2. Para asignar un rol de AWS a una instancia de EC2 y ponerla a disposición de todas las aplicaciones, cree un perfil de instancia adjuntado a

la instancia. Un perfil de instancia contiene el rol y permite a los programas que se ejecutan en la instancia de EC2 obtener credenciales temporales. Para obtener más información, consulte Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias Amazon EC2 en la Guía del usuario de IAM.

Administración de acceso mediante políticas

Para controlar el acceso en AWS, se crean políticas y se adjuntan a identidades o recursos de AWS. Una política es un objeto de AWS que, cuando se asocia a una identidad o un recurso, define sus permisos. AWS evalúa estas políticas cuando una entidad principal (sesión de rol, usuario o usuario raíz) realiza una solicitud. Los permisos en las políticas determinan si la solicitud se permite o se deniega. La mayoría de las políticas se almacenan en AWS como documentos JSON. Para obtener más información sobre la estructura y el contenido de los documentos de política JSON, consulte Información general de políticas JSON en la Guía del usuario de IAM.

Los administradores pueden utilizar las políticas JSON de AWS para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

De forma predeterminada, los usuarios y los roles no tienen permisos. Un administrador de IAM puede crear políticas de IAM para conceder permisos a los usuarios para realizar acciones en los recursos que necesitan. A continuación, el administrador puede añadir las políticas de IAM a roles y los usuarios pueden asumirlos.

Las políticas de IAM definen permisos para una acción independientemente del método que se utiliza para realizar la operación. Por ejemplo, suponga que dispone de una política que permite la acción `iam:GetRole`. Un usuario con dicha política puede obtener información del usuario de la AWS Management Console, la AWS CLI o la API de AWS.

Políticas basadas en identidades

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 identidad, consulte Creación de políticas de IAM en la Guía del usuario de IAM.

Las políticas basadas en identidades pueden clasificarse además como políticas insertadas o políticas administradas. Las políticas insertadas se integran directamente en un único usuario, grupo

o rol. Las políticas administradas son políticas independientes que puedes adjuntar a varios usuarios, grupos y roles de su Cuenta de AWS. Las políticas gestionadas incluyen las políticas gestionadas de AWS y las políticas gestionadas por el cliente. Para obtener más información sobre cómo elegir una política administrada o una política insertada, consulte Elegir entre políticas administradas y políticas insertadas en la Guía del usuario de IAM.

Políticas basadas en recursos

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 en función de recursos. Las entidades principales pueden incluir cuentas, usuarios, roles, usuarios federados o Servicios de AWS.

Las políticas basadas en recursos son políticas insertadas que se encuentran en ese servicio. No se puede utilizar políticas de IAM administradas de AWS en una política basada en recursos.

Listas de control de acceso (ACL)

Las listas de control de acceso (ACL) controlan qué entidades principales (miembros de cuentas, usuarios o roles) tienen permisos para acceder a un recurso. Las ACL son similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de políticas JSON.

Amazon S3, AWS WAF y Amazon VPC son ejemplos de servicios que admiten las ACL. Para obtener más información sobre las ACL, consulta Información general de Lista de control de acceso (ACL) en la Guía para desarrolladores de Amazon Simple Storage Service.

Otros tipos de políticas

AWS admite otros tipos de políticas adicionales menos frecuentes. Estos tipos de políticas pueden establecer el máximo de permisos que los tipos de políticas más frecuentes le conceden.

- Límites de permisos: un límite de permisos es una característica avanzada que le permite establecer los permisos máximos que una política basada en identidad puede conceder a una entidad de IAM (usuario o rol de IAM). Puedes establecer un límite de permisos para una entidad. Los permisos resultantes son la intersección de las políticas basadas en la identidad de la entidad

y los límites de permisos. Las políticas basadas en recursos que especifiquen el usuario o rol en el campo `Principal` no estarán restringidas por el límite de permisos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para obtener más información sobre los límites de los permisos, consulta Límites de permisos para las entidades de IAM en la Guía del usuario de IAM.

- Políticas de control de servicio (SCP): las SCP son políticas de JSON que especifican los permisos máximos de una organización o una unidad organizativa en AWS Organizations. AWS Organizations es un servicio que le permite agrupar y administrar de manera centralizada varias Cuentas de AWS que posea su empresa. Si habilita todas las características en una empresa, entonces podrá aplicar políticas de control de servicio (SCP) a una o todas sus cuentas. Una SCP limita los permisos para las entidades de las cuentas de miembro, incluido cada Usuario raíz de la cuenta de AWS. Para obtener más información acerca de SCP y Organizations, consulta Políticas de control de servicios en la Guía del usuario de AWS Organizations.
- Políticas de control de recursos (RCP): las RCP son políticas JSON que permiten establecer los permisos máximos disponibles para los recursos de las cuentas sin actualizar las políticas de IAM asociadas a cada recurso que posea. La RCP limita los permisos de los recursos en las cuentas de miembros y puede afectar a los permisos efectivos de las identidades, incluidos los Usuario raíz de la cuenta de AWS, independientemente de si pertenecen a su organización. Para obtener más información sobre Organizations y RCP, incluida una lista de los Servicios de AWS que admiten RCP, consulte Políticas de control de recursos (RCP) en la Guía del usuario de AWS Organizations.
- Políticas de sesión: las políticas de sesión son políticas avanzadas que se pasan como parámetro cuando se crea una sesión temporal mediante programación para un rol o un usuario federado. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades del rol y las políticas de la sesión. Los permisos también pueden proceder de una política en función de recursos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para más información, consulte Políticas de sesión en la Guía del usuario de IAM.

Varios tipos de políticas

Cuando se aplican varios tipos de políticas a una solicitud, los permisos resultantes son más complicados de entender. Para obtener información acerca de cómo AWS decide si permitir o no una solicitud cuando hay varios tipos de políticas implicados, consulte Lógica de evaluación de políticas en la Guía del usuario de IAM.

Cómo funciona Amazon Elastic Container Service con IAM

Antes de utilizar IAM para administrar el acceso a Amazon ECS, conozca qué características de IAM se encuentran disponibles con Amazon ECS.

Característica de IAM	Compatibilidad con Amazon ECS
Políticas con base en identidad	Sí
Políticas basadas en recursos	No
Acciones de políticas	Sí
Recursos de políticas	Parcial
Claves de condición de política	Sí
ACL	No
ABAC (etiquetas en políticas)	Sí
Credenciales temporales	Sí
Sesiones de acceso directo (FAS)	Sí
Roles de servicio	Sí
Roles vinculados al servicio	Sí

Para obtener una perspectiva general sobre cómo funcionan Amazon ECS y otros servicios de AWS con la mayoría de las características de IAM, consulte Servicios de AWS que funcionan con IAM en la Guía del usuario de IAM.

Políticas de Amazon ECS basadas en identidades

Admite 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 identidad, consulte Creación de políticas de IAM 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. No es posible especificar la entidad principal en una política basada en identidad porque se aplica al usuario o rol al que está asociada. Para obtener más información sobre los elementos que puede utilizar en una política de JSON, consulte Referencia de los elementos de las políticas de JSON de IAM en la Guía del usuario de IAM.

Ejemplos de políticas basadas en identidades de Amazon ECS

Para ver ejemplos de políticas de Amazon ECS basadas en identidades, consulte Ejemplos de políticas basadas en identidades de Amazon Elastic Container Service.

Políticas basadas en recursos de Amazon ECS

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 en función de recursos. Las entidades principales pueden incluir cuentas, usuarios, roles, usuarios federados o Servicios de AWS.

Para habilitar el acceso entre cuentas, puedes especificar toda una cuenta o entidades de IAM de otra cuenta como la entidad principal de una política en función de recursos. Añadir a una política en función de recursos una entidad principal entre cuentas es solo una parte del establecimiento de una relación de confianza. Cuando la entidad principal y el recurso se encuentran en Cuentas de AWS diferentes, un gestionador de IAM de la cuenta de confianza también debe conceder a la entidad principal (usuario o rol) permiso para acceder al recurso. Para conceder el permiso, adjunte la entidad a una política basada en identidad. Sin embargo, si la política basada en recursos concede acceso a una entidad principal de la misma cuenta, no es necesaria una política basada en identidad adicional. Para obtener más información, consulte Cross account resource access in IAM en la Guía del usuario de IAM.

Acciones políticas para Amazon ECS

Admite acciones de política: sí

Los administradores pueden utilizar las políticas JSON de AWS 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. Las acciones de la política generalmente tienen el mismo nombre que la operación de API de AWS asociada. Hay algunas excepciones, como acciones de solo permiso que no tienen una operación de API coincidente. También hay algunas operaciones que requieren varias acciones en una política. Estas acciones adicionales se denominan acciones dependientes.

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 ECS, consulte Acciones definidas por Amazon Elastic Container Service en la Referencia de autorizaciones de servicio.

Las acciones de políticas de Amazon ECS utilizan el siguiente prefijo antes de la acción:

```
ecs
```

Para especificar varias acciones en una única instrucción, sepárelas con comas.

```
"Action": [
      "ecs:action1",
      "ecs: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": "ecs:Describe*"
```

Para ver ejemplos de políticas de Amazon ECS basadas en identidades, consulte Ejemplos de políticas basadas en identidades de Amazon Elastic Container Service.

Recursos de políticas para Amazon ECS

Admite recursos de políticas: en forma parcial

Los administradores pueden utilizar las políticas JSON de AWS 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. Las instrucciones deben contener un elemento `Resource` o `NotResource`. Como práctica recomendada, especifique un recurso utilizando el Nombre de recurso de Amazon (ARN). Puedes hacerlo para acciones que admitan un tipo de recurso específico, conocido como permisos de nivel de recurso.

Para las acciones que no admiten permisos de nivel de recurso, como las operaciones de descripción, utiliza un carácter comodín (*) para indicar que la instrucción se aplica a todos los recursos.

```
"Resource": "*"
```

Para ver una lista de los tipos de recursos Amazon ECS y sus ARN, consulte Recursos definidos por Amazon Elastic Container Service en la Referencia de autorizaciones de servicio. Para obtener información acerca de las acciones con las que puede especificar el ARN de cada recurso, consulte Acciones definidas por Amazon Elastic Container Service.

Algunas acciones de la API de Amazon ECS admiten varios recursos. Por ejemplo, se puede hacer referencia a varios clústeres al llamar a la acción de la API `DescribeClusters`. Para especificar varios recursos en una única instrucción, separe los ARN con comas.

```
"Resource": [
      "EXAMPLE-RESOURCE-1",
      "EXAMPLE-RESOURCE-2"
```

Por ejemplo, el recurso del clúster de Amazon ECS tiene el siguiente ARN:

```
arn:${Partition}:ecs:${Region}:${Account}:cluster/${clusterName}
```

Para especificar el clúster de `my-cluster-1` y `my-cluster-2` en su instrucción, utilice el siguiente ARN:

```
"Resource": [
        "arn:aws:ecs:us-east-1:123456789012:cluster/my-cluster-1",
        "arn:aws:ecs:us-east-1:123456789012:cluster/my-cluster-2"
```

Para especificar todos los clústeres que pertenecen a una cuenta específica, utilice el carácter comodín (*):

```
"Resource": "arn:aws:ecs:us-east-1:123456789012:cluster/*"
```

Para las definiciones de tareas, puede especificar la última revisión o una revisión específica.

Para especificar todas las revisiones de la definición de tareas, utilice el carácter comodín (*):

```
"Resource:arn:${Partition}:ecs:${Region}:${Account}:task-definition/
${TaskDefinitionFamilyName}:*"
```

Para especificar una revisión de definición de tarea específica, utilice ${TaskDefinitionRevisionNumber}:

```
"Resource:arn:${Partition}:ecs:${Region}:${Account}:task-definition/
${TaskDefinitionFamilyName}:${TaskDefinitionRevisionNumber}"
```

Para ver ejemplos de políticas de Amazon ECS basadas en identidades, consulte Ejemplos de políticas basadas en identidades de Amazon Elastic Container Service.

Claves de condición de políticas para Amazon ECS

Admite claves de condición de políticas específicas del servicio: sí

Los administradores pueden utilizar las políticas JSON de AWS 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` (o bloque de `Condition`) permite especificar condiciones en las que entra en vigor una instrucción. El elemento `Condition` es opcional. Puedes 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.

Si especifica varios elementos de `Condition` en una instrucción o varias claves en un único elemento de `Condition`, AWS las evalúa mediante una operación `AND` lógica. Si especifica varios

valores para una única clave de condición, AWS evalúa la condición con una operación lógica `OR`. Se deben cumplir todas las condiciones antes de que se concedan los permisos de la instrucción.

También puedes utilizar variables de marcador de posición al especificar condiciones. Por ejemplo, puedes conceder un permiso de usuario de IAM para acceder a un recurso solo si está etiquetado con su nombre de usuario de IAM. Para más información, consulta Elementos de la política de IAM: variables y etiquetas en la Guía del usuario de IAM.

AWS admite claves de condición globales y claves de condición específicas del servicio. Para ver todas las claves de condición globales de AWS, consulte Claves de contexto de condición globales de AWS en la Guía del usuario de IAM.

Amazon ECS admite las siguientes claves de condición específicas de servicios que puede utilizar para proporcionar filtrado detallado para sus políticas de IAM:

Clave de condición	Descripción	Tipos de evaluación
aws:RequestTag/${TagKey}	La clave de contexto tiene el formato `"aws:RequestTag/ tag-key":"tag-value "`, donde *`tag-key`* y *`tag-value`* son un par formado por una clave y un valor de una etiqueta. Compruebe que el par de clave-valor de la etiqueta está presente en una solicitud de AWS. Por ejemplo, podría comprobar que la solicitud incluya la clave de etiqueta `"Dept"` y que tenga el valor `"Accounting"` .	Cadena
aws:ResourceTag/${TagKey}	La clave de contexto tiene el formato `"aws:ResourceTag/ tag-key":"tag-value "`, donde *`tag-key`* y *`tag-value`* son un par formado por una clave y un valor de una etiqueta. Comprueba que la etiqueta asociada al recurso de identidad (usuario o rol) coincida con el valor y el nombre de la clave especificada.	Cadena
aws:TagKeys	Esta clave de contexto tiene el formato `"aws:TagKeys":" tag-key"`, donde *`tag-key`* es una lista de	Cadena

Clave de condición	Descripción	Tipos de evaluación
	claves de etiqueta sin valores (por ejemplo, `["Dept","Cost-Center"]`). Comprueba las claves de etiqueta presentes en una solicitud de AWS.	
ecs:ResourceTag/${TagKey}	La clave de contexto tiene el formato `"ecs:ResourceTag/`*`tag-key`*`":"`*`tag-value`*`"`, donde *`tag-key`* y *`tag-value`* son un par formado por una clave y un valor de una etiqueta. Comprueba que la etiqueta asociada al recurso de identidad (usuario o rol) coincida con el valor y el nombre de la clave especificada.	Cadena
ecs:account-setting	La clave de contexto utiliza el formato `"ecs:account-setting":"`*`account-setting`*`"`, donde *`account-setting`* es el nombre de la configuración de cuenta.	Cadena
ecs:auto-assign-public-ip	La clave de contexto adquiere un formato de `"ecs:auto-assign-public-ip":"`*`value`*`"` donde *`value-`* es `"true"` o `"false"`.	Cadena
ecs:capacity-provider	La clave de contexto tiene el formato `"ecs:capacity-provider":"`*`capacity-provider-arn`*`"`, donde *`capacity-provider-arn`* es el ARN del proveedor de capacidad.	ARN, Null
ecs:cluster	La clave de contexto utiliza el formato `"ecs:cluster":"`*`cluster-arn`*`"`, donde *`cluster-arn`* es el ARN del clúster de Amazon ECS.	ARN, Null
ecs:compute-compatibility	La clave del contexto tiene el formato `"ecs:compute-compatability":"`*`compatability-type`*`"`, donde *`tipo-compatibilidad`* es “`FARGATE`”, “`EC2`” o “`EXTERNAL`”.	Cadena

Clave de condición	Descripción	Tipos de evaluación
ecs:container-instances	La clave de contexto utiliza el formato "ecs:container-instances":" *container-instance-arns*", donde *container-instance-arns* es uno o varios ARN de instancia de contenedor.	ARN, Null
ecs:container-name	La clave de contexto tiene formato "ecs:container-name":" *container-name* " donde *container-instance-* es el nombre de un contenedor de Amazon ECS que se define en la definición de tarea.	Cadena
ecs:enable-execute-command	La clave de contexto adquiere un formato de "ecs:enable-execute-command":" *value*" donde *value-* es "true" o "false".	Cadena
ecs:enable-service-connect	La clave de contexto adquiere un formato "ecs:enable-service-connect":" *value*" donde *value* es "true" o "false".	Cadena
ecs:enable-ebs-volumes	La clave de contexto adquiere un formato "ecs:enable-ebs-volumes":" *value*" donde *value* es "true" o "false".	Cadena
ecs:enable-managed-tags	La clave de contexto adquiere un formato "ecs:enable-managed-tags":" *value*" donde *value* es "true" o "false".	Cadena
ecs:enable-vpc-lattice	La clave de contexto adquiere un formato "ecs:vpc-lattice":" *value*" donde *value* es "true" o "false".	Cadena
ecs:fargate-ephemeral-storage-kms-key	La clave de contexto tiene el formato "ecs:fargate-ephemeral-storage-kms-key":" *key*", donde *key* es el ID de la clave AWS KMS.	Cadena

Clave de condición	Descripción	Tipos de evaluación
ecs:namespace	La clave de contexto utiliza el formato `"ecs:namespace":"` *`namespace-arn`* `"`, donde *`namespace-arn`* es el ARN del espacio de nombres de AWS Cloud Map.	ARN, Null
ecs:propagate-tags	La clave de contexto tiene el formato `"ecs:propagate-tags":"` *`value`*`"`, donde *`value`* es `"TASK_DEFINITION"` , `"SERVICE"` o `"NONE"`.	Cadena
ecs:service	La clave de contexto utiliza el formato `"ecs:service":"` *`service-arn`* `"`, donde *`service-arn`* es el ARN del servicio de Amazon ECS.	ARN, Null
ecs:task-definition	La clave de contexto utiliza el formato `"ecs:task-definition":"` *`task-definition-arn`* `"`, donde *`task-definition-arn`* es el ARN de la definición de tarea de Amazon ECS.	ARN, Null
ecs:subnet	La clave de contexto tiene el formato `"ecs:subnet":"` *`subnets`*`"` donde *`subnets`* son las subredes de una tarea o servicio que usa el modo `awsvpc`.	Cadena
ecs:task	La clave de contexto tiene el formato `"ecs:task":"` *`task-arn`*`"`, donde *`task-arn`* es el ARN del servicio de Amazon ECS.	ARN, Null
ecs:task-cpu	La clave de contexto tiene el formato `"ecs:task-cpu":"`*`task-cpu`*`"`, donde *`task-cpu`* es la CPU de la tarea, como un entero con 1024 = 1 vCPU.	Entero
ecs:task-memory	La clave de contexto tiene el formato `"ecs:task-memory":"` *`task-memory`* `"`, donde *`task-memory`* es la memoria de la tarea en MiB.	Entero

Para obtener una lista de las claves de condición de Amazon ECS, consulte Claves de condición de Amazon Elastic Container Service en la Referencia de autorizaciones de servicio. Para obtener más información acerca de las acciones y los recursos con los que puede utilizar una clave de condición, consulte Acciones definidas por Amazon Elastic Container Service.

Para ver ejemplos de políticas de Amazon ECS basadas en identidades, consulte Ejemplos de políticas basadas en identidades de Amazon Elastic Container Service.

Listas de control de acceso (ACL) de Amazon ECS

Admite ACL: no

Las listas de control de acceso (ACL) controlan qué entidades principales (miembros de cuentas, usuarios o roles) tienen permisos para acceder a un recurso. Las ACL son similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de políticas JSON.

Control de acceso basado en atributos (ABAC) con Amazon ECS

> **Important**
>
> Amazon ECS admite el control de acceso basado en atributos para todos los recursos de Amazon ECS. Para determinar si puede utilizar los atributos para definir el alcance de una acción, utilice la tabla Acciones definidas por Amazon ECS en la Referencia de autorizaciones de servicios. En primer lugar, compruebe que haya un recurso en la columna Recursos. A continuación, utilice la columna Claves de condición para ver las claves de la combinación de acción y recurso.

Admite ABAC (etiquetas en las políticas): sí

El control de acceso basado en atributos (ABAC) es una estrategia de autorización que define permisos en función de atributos. En AWS, estos atributos se denominan etiquetas. Puedes adjuntar etiquetas a entidades de IAM (usuarios o roles) y a muchos recursos de AWS. El etiquetado de entidades y recursos es el primer paso de ABAC. A continuación, designa las políticas de ABAC para permitir operaciones cuando la etiqueta de la entidad principal coincida con la etiqueta del recurso al que se intenta acceder.

ABAC es útil en entornos que crecen con rapidez y ayuda en situaciones en las que la administración de las políticas resulta engorrosa.

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.

Para obtener más información sobre el etiquetado de recursos de Amazon ECS, consulte Etiquetado de los recursos de Amazon ECS.

Para consultar un ejemplo de política basada en la identidad para limitar el acceso a un recurso en función de las etiquetas de ese recurso, consulte Descripción de los servicios de Amazon ECS basados en etiquetas.

Uso de credenciales temporales con Amazon ECS

Admite credenciales temporales: sí

Algunos Servicios de AWS no funcionan cuando inicia sesión con credenciales temporales. Para obtener información adicional, incluida la información sobre qué Servicios de AWS funcionan con credenciales temporales, consulta Servicios de AWS que funcionan con IAM en la Guía del usuario de IAM.

Utiliza credenciales temporales si inicia sesión en la AWS Management Console con cualquier método, excepto un nombre de usuario y una contraseña. Por ejemplo, cuando accede a AWS utilizando el enlace de inicio de sesión único (SSO) de la empresa, ese proceso crea automáticamente credenciales temporales. También crea automáticamente credenciales temporales cuando inicia sesión en la consola como usuario y luego cambia de rol. Para obtener más información sobre el cambio de roles, consulte Cambio de un usuario a un rol de IAM (consola) en la Guía del usuario de IAM.

Puedes crear credenciales temporales de forma manual mediante la AWS CLI o la API de AWS. A continuación, puedes usar esas credenciales temporales para acceder a AWS. AWS recomienda generar credenciales temporales de forma dinámica en lugar de usar claves de acceso a largo plazo. Para obtener más información, consulte Credenciales de seguridad temporales en IAM.

Sesiones de acceso directo para Amazon ECS

Admite sesiones de acceso directo (FAS): sí

Cuando utiliza un usuario o un rol de IAM para llevar a cabo acciones en AWS, se le considera una entidad principal. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. FAS utiliza los permisos de la entidad principal para llamar a un Servicio de AWS, combinados con el Servicio de AWS solicitante para realizar solicitudes a servicios posteriores. Las solicitudes de FAS solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros Servicios de AWS o recursos para completarse. En este caso, debe tener permisos para realizar ambas acciones. Para obtener información sobre las políticas a la hora de realizar solicitudes de FAS, consulte Reenviar sesiones de acceso.

Roles de servicio para Amazon ECS

Admite 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 Creación de un rol para delegar permisos a un Servicio de AWS en la Guía del usuario de IAM.

> **Warning**
>
> Cambiar los permisos de un rol de servicio podría interrumpir la funcionalidad de Amazon ECS. Edite los roles de servicio solo cuando Amazon ECS proporcione orientación para hacerlo.

Roles vinculado a servicios para Amazon ECS

Admite roles vinculados al servicio: sí

Un rol vinculado a servicios es un tipo de rol de servicio que está vinculado a un Servicio de AWS. El servicio puedes asumir el rol para realizar una acción en su nombre. Los roles vinculados a servicios aparecen en la 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 detalles acerca de cómo crear o administrar roles vinculados a servicios de Amazon ECS, consulte Uso de roles vinculados al servicio para Amazon ECS.

Ejemplos de políticas basadas en identidades de Amazon Elastic Container Service

De forma predeterminada, los usuarios y roles no tienen permiso para crear ni modificar recursos de Amazon ECS. Tampoco puede realizar tareas mediante la AWS Management Console, la AWS Command Line Interface (AWS CLI) o la API de AWS. Un administrador de IAM puede crear políticas de IAM para conceder permisos a los usuarios para realizar acciones en los recursos que necesitan. A continuación, el administrador puede añadir las políticas de IAM a roles y los usuarios pueden asumirlos.

Para obtener información acerca de cómo crear una política basada en identidad de IAM mediante el uso de estos documentos de políticas JSON de ejemplo, consulte Creación de políticas de IAM (consola) en la Guía del usuario de IAM.

Para obtener más información sobre las acciones y los tipos de recursos definidos por Amazon ECS, incluido el formato de los ARN para cada uno de los tipos de recursos, consulte Acciones, recursos y claves de condición de Amazon Cognito en la Referencia de autorizaciones de servicio.

Temas

- Prácticas recomendadas sobre políticas de Amazon ECS
- Consentimiento para que los usuarios de Amazon ECS vean sus propios permisos
- Ejemplos de clústeres de Amazon ECS
- Ejemplos de instancias de contenedor de Amazon ECS
- Ejemplos de definición de tarea de Amazon ECS
- Ejemplo de tarea de ejecución de Amazon ECS
- Ejemplo de tarea de inicio de Amazon ECS
- Ejemplos de enumeración y descripción de tareas de Amazon ECS
- Ejemplo de creación de servicios de Amazon ECS
- Ejemplo de actualización de servicios de Amazon ECS
- Descripción de los servicios de Amazon ECS basados en etiquetas
- Ejemplo de denegación de anulación del espacio de nombres de Amazon ECS Service Connect

Prácticas recomendadas sobre políticas de Amazon ECS

Las políticas basadas en identidades determinan si alguien puede crear, eliminar o acceder a los recursos de Amazon ECS de su cuenta. Estas acciones pueden generar costos adicionales para su Cuenta de AWS. Siga estas directrices y recomendaciones al crear o editar políticas basadas en identidades:

- Comienza con las políticas administradas por AWSy continúa con los permisos de privilegio mínimo: a fin de comenzar a conceder permisos a los usuarios y las cargas de tarea, utiliza las políticas administradas por AWS, que conceden permisos para muchos casos de uso comunes. Están disponibles en su Cuenta de AWS. Se recomienda definir políticas gestionadas por el cliente de AWS específicas para sus casos de uso a fin de reducir aún más los permisos. Con el fin de obtener más información, consulta las políticas administradas por AWS o las políticas administradas por AWS para funciones de tarea en la Guía de usuario de IAM.
- Aplique permisos de privilegio mínimo: cuando establezca permisos con políticas de IAM, conceda solo los permisos necesarios para realizar una tarea. Para ello, debe definir las acciones que se pueden llevar a cabo en determinados recursos en condiciones específicas, también conocidos como permisos de privilegios mínimos. Con el fin de obtener más información sobre el uso de IAM para aplicar permisos, consulta Políticas y permisos en IAM en la Guía del usuario de IAM.
- Utilice condiciones en las políticas de IAM para restringir aún más el acceso: puede agregar una condición a sus políticas para limitar el acceso a las acciones y los recursos. Por ejemplo, puedes escribir una condición de políticas para especificar que todas las solicitudes deben enviarse utilizando SSL. También puedes usar condiciones para conceder acceso a acciones de servicios si se emplean a través de un Servicio de AWS determinado como, por ejemplo, AWS CloudFormation. Para obtener más información, consulta Elementos de la política de JSON de IAM: Condición en la Guía del usuario de IAM.
- Utiliza el analizador de acceso de IAM para validar las políticas de IAM con el fin de garantizar la seguridad y funcionalidad de los permisos: el analizador de acceso de IAM valida políticas nuevas y existentes para que respeten el lenguaje (JSON) de las políticas de IAM y las prácticas recomendadas de IAM. El analizador de acceso de IAM proporciona más de 100 verificaciones de políticas y recomendaciones procesables para ayudar a crear políticas seguras y funcionales. Para más información, consulte Validación de políticas con el Analizador de acceso de IAM en la Guía del usuario de IAM.
- Solicite la autenticación multifactor (MFA): si se encuentra en una situación en la que necesite usuarios raíz o de IAM en su Cuenta de AWS, active la MFA para obtener una mayor seguridad. Para exigir la MFA cuando se invoquen las operaciones de la API, añada condiciones de MFA

a sus políticas. Para más información, consulte Acceso seguro a la API con MFA en la Guía del usuario de IAM.

Para obtener más información sobre las prácticas recomendadas de IAM, consulte Prácticas recomendadas de seguridad en IAM en la Guía del usuario de IAM.

Consentimiento para que los usuarios de Amazon ECS vean sus propios permisos

En este ejemplo, se muestra cómo podría crear una política que permita a los usuarios de IAM ver las políticas gestionadas e insertadas que se asocian a la identidad de sus usuarios. Esta política incluye permisos para llevar a cabo esta acción en la consola o mediante programación con la AWS CLI o la API de AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
```

```
        }
    ]
}
```

Ejemplos de clústeres de Amazon ECS

La siguiente política de IAM concede permiso para crear y mostrar la lista de clústeres. Las acciones `CreateCluster` y `ListClusters` no aceptan ningún recurso, por tanto la definición de recursos se define en `*` para todos los recursos.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:CreateCluster",
                "ecs:ListClusters"
            ],
            "Resource": ["*"]
        }
    ]
}
```

La siguiente política de IAM concede permiso para describir y eliminar un clúster determinado. Las acciones `DescribeClusters` y `DeleteCluster` aceptan ARN de clústeres como recursos.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:DescribeClusters",
                "ecs:DeleteCluster"
            ],
            "Resource": ["arn:aws:ecs:us-east-1:<aws_account_id>:cluster/
<cluster_name>"]
        }
    ]
}
```

La siguiente política de IAM se puede asociar a un usuario o grupo y solo permitiría a dicho usuario o grupo realizar operaciones en un clúster determinado.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ecs:Describe*",
                "ecs:List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "ecs:DeleteCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:ListContainerInstances",
                "ecs:RegisterContainerInstance",
                "ecs:SubmitContainerStateChange",
                "ecs:SubmitTaskStateChange"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:ecs:us-east-1:<aws_account_id>:cluster/default"
        },
        {
            "Action": [
                "ecs:DescribeContainerInstances",
                "ecs:DescribeTasks",
                "ecs:ListTasks",
                "ecs:UpdateContainerAgent",
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:RunTask"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "ArnEquals": {"ecs:cluster": "arn:aws:ecs:us-
east-1:<aws_account_id>:cluster/default"}
            }
        }
```

```
    ]
}
```

Ejemplos de instancias de contenedor de Amazon ECS

El agente de Amazon ECS se encarga del registro de instancias de contenedor, pero puede haber ocasiones en las que desee permitir a un usuario anular el registro de una instancia manualmente desde un clúster. Quizás la instancia de contenedor se registró accidentalmente en el clúster equivocado o la instancia se terminó mientras había tareas en ejecución.

La siguiente política de IAM permite a un usuario crear una lista y anular el registro de las instancias de contenedor de un clúster determinado:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:DeregisterContainerInstance",
                "ecs:ListContainerInstances"
            ],
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:cluster/
<cluster_name>"]
        }
    ]
}
```

La siguiente política de IAM permite a un usuario describir una instancia de contenedor específica en un clúster determinado. Para abrir este permiso a todas las instancias de contenedor en un clúster, puede sustituir el UUID de la instancia de contenedor por `*`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:DescribeContainerInstances"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
```

```
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:container-instance/
<cluster_name>/<container_instance_UUID>"]
        }
    ]
}
```

Ejemplos de definición de tarea de Amazon ECS

Las políticas de IAM de definición de tareas no admiten los permisos de nivel de recursos, pero la siguiente política de IAM permite a un usuario registrar, enumerar y describir definiciones de tareas:

Si utiliza la consola, debe agregar `CloudFormation: CreateStack` como un `Action`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:RegisterTaskDefinition",
                "ecs:ListTaskDefinitions",
                "ecs:DescribeTaskDefinition"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Ejemplo de tarea de ejecución de Amazon ECS

Los recursos para `RunTask` son definiciones de tareas. Para limitar los clústeres en los que un usuario puede ejecutar definiciones de tareas, puede especificarlos en el bloque `Condition`. La ventaja es que no tiene que enumerar ambas definiciones de tarea y clústeres en los recursos para permitir el acceso adecuado. Puede aplicar uno, el otro o ambos.

La siguiente política de IAM concede permiso para ejecutar cualquier revisión de una definición de tarea específica en un clúster determinado:

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

```
            "Effect": "Allow",
            "Action": ["ecs:RunTask"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:task-definition/
<task_family>:*"]
        }
    ]
}
```

Ejemplo de tarea de inicio de Amazon ECS

Los recursos para `StartTask` son definiciones de tareas. Para limitar los clústeres e instancias de contenedor en los que un usuario puede comenzar definiciones de tareas, puede especificarlos en el bloque `Condition`. La ventaja es que no tiene que enumerar ambas definiciones de tarea y clústeres en los recursos para permitir el acceso adecuado. Puede aplicar uno, el otro o ambos.

La siguiente política de IAM concede permiso para comenzar cualquier revisión de una definición de tarea específica en un clúster determinado y una instancia de contenedor en particular.

> **Note**
>
> En este ejemplo, si llama a la API `StartTask` con la AWS CLI u otro SDK de AWS, debe especificar la revisión de la definición de tarea para que mapeo de `Resource` coincida.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:StartTask"],
            "Condition": {
                "ArnEquals": {
                    "ecs:cluster": "arn:aws:ecs:<region>:<aws_account_id>:cluster/
<cluster_name>",
                    "ecs:container-instances":
 ["arn:aws:ecs:<region>:<aws_account_id>:container-instance/<cluster_name>/
<container_instance_UUID>"]
                }
```

```
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:task-definition/
<task_family>:*"]
        }
    ]
}
```

Ejemplos de enumeración y descripción de tareas de Amazon ECS

La siguiente política de IAM concede permiso a un usuario para mostrar tareas para un clúster determinado:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:ListTasks"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["*"]
        }
    ]
}
```

La siguiente política de IAM concede permiso a un usuario para describir una tarea especificada en un clúster determinado:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:DescribeTasks"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:task/<cluster_name>/
<task_UUID>"]
```

```
        }
    ]
}
```

Ejemplo de creación de servicios de Amazon ECS

La siguiente política de IAM permite a un usuario crear servicios de Amazon ECS en la AWS Management Console:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:Describe*",
                "application-autoscaling:PutScalingPolicy",
                "application-autoscaling:RegisterScalableTarget",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "ecs:List*",
                "ecs:Describe*",
                "ecs:CreateService",
                "elasticloadbalancing:Describe*",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:ListAttachedRolePolicies",
                "iam:ListRoles",
                "iam:ListGroups",
                "iam:ListUsers"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Ejemplo de actualización de servicios de Amazon ECS

La siguiente política de IAM permite a un usuario actualizar servicios de Amazon ECS en la AWS Management Console:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:Describe*",
                "application-autoscaling:PutScalingPolicy",
                "application-autoscaling:DeleteScalingPolicy",
                "application-autoscaling:RegisterScalableTarget",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "ecs:List*",
                "ecs:Describe*",
                "ecs:UpdateService",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:ListAttachedRolePolicies",
                "iam:ListRoles",
                "iam:ListGroups",
                "iam:ListUsers"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Descripción de los servicios de Amazon ECS basados en etiquetas

Puede utilizar condiciones en la política basada en la identidad para controlar el acceso a los recursos de Amazon ECS basados en etiquetas. En este ejemplo se muestra cómo crear una política que permita describir sus servicios. Sin embargo, los permisos solo se conceden si la etiqueta de servicio `Owner` tiene el valor del nombre de usuario de dicho usuario. Esta política también proporciona los permisos necesarios para llevar a cabo esta acción en la consola.

JSON

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

```
        {
            "Sid": "DescribeServices",
            "Effect": "Allow",
            "Action": "ecs;:DescribeServices",
            "Resource": "*"
        },
        {
            "Sid": "ViewServiceIfOwner",
            "Effect": "Allow",
            "Action": "ecs:DescribeServices",
            "Resource": "arn:aws:ecs:*:*:service/*",
            "Condition": {
                "StringEquals": {"ecs:ResourceTag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

También puede asociar esta política al usuario de IAM en su cuenta. Si un usuario con el nombre `richard-roe` intenta describir un servicio de Amazon ECS, el servicio debe tener una etiqueta `Owner=richard-roe` u `owner=richard-roe`. De lo contrario, se le deniega el acceso. La clave de la etiqueta de condición `Owner` coincide con los nombres de las claves de condición `Owner` y `owner` porque no distinguen entre mayúsculas y minúsculas. Para obtener más información, consulte Elementos de la política de JSON de IAM: Condición en la Guía del usuario de IAM.

Ejemplo de denegación de anulación del espacio de nombres de Amazon ECS Service Connect

La siguiente política de IAM impide a un usuario anular el espacio de nombres predeterminado de Service Connect en una configuración de servicio. El espacio de nombres predeterminado se establece en el clúster. Sin embargo, puede anularlo en una configuración de servicio. Para mantener la coherencia, considere configurar todos los servicios nuevos para que usen el mismo espacio de nombres. Utilice las siguientes claves de contexto para exigir a los servicios que usen un espacio de nombres específico. Sustituya los `<region>`, `<aws_account_id>`, `<cluster_name>` y `<namespace_id>` del ejemplo siguiente por el suyo propio.

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

```
            "Effect": "Allow",
            "Action": [
                "ecs:CreateService",
                "ecs:UpdateService"
            ],
            "Condition": {
                "ArnEquals": {
                    "ecs:cluster": "arn:aws:ecs:<region>:<aws_account_id>:cluster/
<cluster_name>",
                    "ecs:namespace":
 "arn:aws:servicediscovery:<region>:<aws_account_id>:namespace/<namespace_id>"
                }
            },
            "Resource": "*"
        }
    ]
}
```

Políticas administradas por AWS para Amazon Elastic Container Service

Para agregar permisos a usuarios, grupos y roles, es más fácil utilizar las políticas administradas de AWS que escribirlas uno mismo. Se necesita tiempo y experiencia para crear políticas administradas por el cliente de IAM que le brinden a su equipo solo los permisos necesarios. Para comenzar a hacerlo con rapidez, puede utilizar nuestras políticas administradas de AWS. Estas políticas cubren casos de uso comunes y están disponibles en su cuenta de AWS. Para obtener más información sobre las políticas administradas de AWS, consulte Políticas administradas de AWS en la Guía del usuario de IAM.

Los servicios de AWS mantienen y actualizan las políticas administradas de AWS. No puede cambiar los permisos en las políticas gestionadas de AWS. En ocasiones, los servicios agregan permisos adicionales a una política administrada de AWS para admitir características nuevas. Este tipo de actualización afecta a todas las identidades (usuarios, grupos y roles) donde se asocia la política. Es más probable que los servicios actualicen una política gestionada por AWScuando se lanza una nueva característica o cuando se ponen a disposición nuevas operaciones. Los servicios no quitan los permisos de una política administrada de AWS, por lo tanto, las actualizaciones de las políticas no deteriorarán los permisos existentes.

Además, AWS admite políticas administradas para funciones de trabajo que abarcan varios servicios. Por ejemplo, la política administrada de AWS ReadOnlyAccess proporciona acceso de solo lectura a

todos los servicios y los recursos de AWS. Cuando un servicio lanza una nueva característica, AWS agrega permisos de solo lectura para las operaciones y los recursos nuevos. Para obtener una lista y descripciones de las políticas de funciones de trabajo, consulte Políticas administradas de AWS para funciones de trabajo en la Guía del usuario de IAM.

Amazon ECS y Amazon ECR proporcionan varias políticas administradas y relaciones de confianza que se pueden asociar a usuarios, grupos, roles, instancias de Amazon EC2 y tareas de Amazon ECS para permitir diferentes niveles de control sobre los recursos y las operaciones de la API. Puede aplicar estas políticas directamente o puede usarlas como punto de partida para crear las suyas propias. Para obtener más información acerca de las políticas administradas por Amazon ECR, consulte Políticas administradas por Amazon ECR.

AmazonECS_FullAccess

Puede adjuntar la política `AmazonECS_FullAccess` a las identidades de IAM. Esta política otorga acceso administrativo a los recursos de Amazon ECS y le otorga a una identidad de IAM (como un usuario, grupo o rol) acceso a los servicios de AWS con los que Amazon ECS está integrado para utilizar todas las características de Amazon ECS. El uso de esta política permite acceder a todas las características de Amazon ECS que están disponibles en la AWS Management Console.

Para ver los permisos de esta política, consulte AmazonECS_FullAccess en la Referencia de políticas administradas de AWS.

AmazonECSInfrastructureRolePolicyForVolumes

Puede adjuntar la política administrada de `AmazonECSInfrastructureRolePolicyForVolumes` a las entidades de IAM.

La política concede los permisos que necesita Amazon ECS para hacer llamadas a la API de AWS en su nombre. Puede adjuntar esta política al rol de IAM que proporciona con la configuración de su volumen al lanzar las tareas y los servicios de Amazon ECS. El rol permite que Amazon ECS administre los volúmenes adjuntos a sus tareas. Para obtener más información, consulte Rol de IAM para la infraestructura de Amazon ECS.

Para ver los permisos de esta política, consulte AmazonECSInfrastructureRolePolicyForVolumes en la Referencia de políticas administradas de AWS.

AmazonEC2ContainerServiceforEC2Role

Puede adjuntar la política `AmazonEC2ContainerServiceforEC2Role` a las identidades de IAM. Esta política concede permisos administrativos que permiten a las instancias de contenedor de

Amazon ECS realizar llamadas a AWS en su nombre. Para obtener más información, consulte Rol de IAM de instancia de contenedor de Amazon ECS.

Amazon ECS asocia esta política a una función de servicio que permite a Amazon ECS realizar acciones en su nombre contra las instancias de Amazon EC2 o las instancias externas.

Para ver los permisos de esta política, consulte AmazonEC2ContainerServiceforEC2Role en la Referencia de políticas administradas de AWS.

Consideraciones

Debe tener en cuenta las siguientes recomendaciones y consideraciones al utilizar la política de IAM administrada `AmazonEC2ContainerServiceforEC2Role`.

- Siguiendo el consejo de seguridad estándar de concesión de privilegios mínimos, puede modificar la política administrada `AmazonEC2ContainerServiceforEC2Role` para que se adapte a sus necesidades específicas. Si alguno de los permisos otorgados en la política administrada no resulta necesario para su caso de uso, cree una política personalizada y agregue solo los permisos que necesite. Por ejemplo, el permiso `UpdateContainerInstancesState` se proporciona para el vaciado de instancias de spot. Si ese permiso no es necesario para su caso de uso, exclúyalo mediante una política personalizada.
- Los contenedores que se ejecutan en sus instancias de contenedor tienen acceso a todos los permisos que se suministran al rol de instancia de contenedor a través de metadatos de instancia. Le recomendamos que limite los permisos en el rol de instancia de contenedor a la lista mínima de permisos que se proporciona en la política administrada `AmazonEC2ContainerServiceforEC2Role`. Si los contenedores de sus tareas necesitan permisos adicionales que no se muestran aquí, le recomendamos que proporcione a esas tareas sus propios roles de IAM. Para obtener más información, consulte Rol de IAM de tarea de Amazon ECS.

  Puede evitar que los contenedores se encuentren en el puente `docker0` accedan a los permisos proporcionados al rol de instancia de contenedor. Para hacer esto sin dejar de permitir los permisos proporcionados por Rol de IAM de tarea de Amazon ECS, ejecute el comando iptables en sus instancias de contenedor. Los contenedores no pueden consultar metadatos de instancia con esta regla en vigor. Este comando supone que se aplica la configuración puente de Docker predeterminada y no funcionará para contenedores que utilicen el modo de red `host`. Para obtener más información, consulte Modo de red.

```
sudo yum install -y iptables-services; sudo iptables --insert DOCKER USER 1 --in-
interface docker+ --destination 169.254.169.254/32 --jump DROP
```

Debe guardar esta regla de iptables en la instancia de contenedor para que se conserve tras un reinicio. Utilice siguientes comandos para la AMI optimizada para Amazon ECS. Para otros sistemas operativos, consulte la documentación correspondiente a dicho sistema operativo.

- Para la AMI de Amazon Linux 2 optimizada para Amazon ECS:

  ```
  sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --
  now iptables
  ```

- Para la AMI de Amazon Linux optimizada para Amazon ECS:

  ```
  sudo service iptables save
  ```

AmazonEC2ContainerServiceEventsRole

Puede adjuntar la política `AmazonEC2ContainerServiceEventsRole` a las identidades de IAM. Esta política concede permisos que permiten a Amazon EventBridge (anteriormente CloudWatch Events) ejecutar tareas en su nombre. Esta política se puede asociar al rol de IAM que se especifica al crear tareas programadas. Para obtener más información, consulte Rol de IAM de EventBridge de Amazon ECS.

Para ver los permisos de esta política, consulte AmazonEC2ContainerServiceEventsRole en la Referencia de políticas administradas de AWS.

AmazonECSTaskExecutionRolePolicy

La política de IAM administrada `AmazonECSTaskExecutionRolePolicy` otorga los permisos que necesita el agente de contenedor de Amazon ECS y los agentes de contenedor de AWS Fargate para hacer llamadas a la API de AWS en su nombre. Esta política se puede agregar a su rol de IAM de ejecución de tareas. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

Para ver los permisos de esta política, consulte AmazonECSTaskExecutionRolePolicy en la Referencia de políticas administradas de AWS.

AmazonECSServiceRolePolicy

La política de IAM `AmazonECSServiceRolePolicy` administrada permite que Amazon Elastic Container Service administre su clúster. Esta política puede ser añadida a su función vinculada al servicio AWSServiceRoleForECS .

Para ver los permisos de esta política, consulte AmazonECSServiceRolePolicy en la Referencia de políticas administradas de AWS.

`AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecur`

También puede asociar la política `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity` a sus entidades de IAM. Esta política concede acceso administrativo a AWS Private Certificate Authority, Secrets Manager y otros servicios de AWS necesarios para administrar las características de TLS de Amazon ECS Service Connect en su nombre.

Para ver los permisos de esta política, consulte AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity en la Referencia de políticas administradas de AWS.

`AWSApplicationAutoscalingECSServicePolicy`

No puede asociar `AWSApplicationAutoscalingECSServicePolicy` a sus entidades IAM. Esta política se asocia a un rol vinculado al servicio que permite a Application Auto Scaling realizar acciones en su nombre. Para obtener más información, consulte Roles vinculados a servicios para Aplication Auto Scaling.

Para ver los permisos de esta política, consulte AWSApplicationAutoscalingECSServicePolicy en la Referencia de políticas administradas de AWS.

`AWSCodeDeployRoleForECS`

No puede asociar `AWSCodeDeployRoleForECS` a sus entidades IAM. Esta política se asocia a un rol vinculado al servicio que permite a CodeDeploy realizar acciones en su nombre. Para obtener más información, consulte Creación de una función de servicio para CodeDeploy en la Guía del usuario de AWS CodeDeploy.

Para ver los permisos de esta política, consulte AWSCodeDeployRoleForECS en la Referencia de políticas administradas de AWS.

AWSCodeDeployRoleForECSLimited

No puede asociar `AWSCodeDeployRoleForECSLimited` a sus entidades IAM. Esta política se asocia a un rol vinculado al servicio que permite a CodeDeploy realizar acciones en su nombre. Para obtener más información, consulte Creación de una función de servicio para CodeDeploy en la Guía del usuario de AWS CodeDeploy.

Para ver los permisos de esta política, consulte AWSCodeDeployRoleForECSLimited en la Referencia de políticas administradas de AWS.

AmazonECSInfrastructureRolePolicyForLoadBalancers

También puede asociar la política `AmazonECSInfrastructureRolePolicyForLoadBalancers` a sus entidades de IAM. Esta política concede permisos que permiten a Amazon ECS administrar los recursos de Elastic Load Balancing en su nombre. La política incluye:

- Permisos de solo lectura para describir los oyentes, las reglas, los grupos de destino y el estado de los destinos
- Permisos para registrar los destinos y anular el registro de estos con los grupos de destino
- Permisos para modificar los oyentes de los equilibradores de carga de aplicación y los equilibradores de carga de red
- Permisos para modificar las reglas de los equilibradores de carga de aplicación

Estos permisos permiten a Amazon ECS administrar automáticamente las configuraciones del equilibrador de carga cuando se crean o actualizan los servicios, lo que garantiza el enrutamiento correcto del tráfico a los contenedores.

Para ver los permisos de esta política, consulte AmazonECSInfrastructureRolePolicyForLoadBalancers en la referencia de políticas administradas de AWS.

AmazonECSInfrastructureRolePolicyForVpcLattice

También puede asociar la política `AmazonECSInfrastructureRolePolicyForVpcLattice` a sus entidades de IAM. Esta política proporciona acceso a otros recursos de servicio de AWS necesarios para administrar en su nombre la característica de VPC Lattice en las cargas de trabajo de Amazon ECS.

Para ver los permisos de esta política, consulte AmazonECSInfrastructureRolePolicyForVpcLattice en la Referencia de políticas administradas de AWS.

Proporciona acceso a otros recursos de servicio de AWS necesarios para administrar en su nombre la característica de VPC Lattice en las cargas de trabajo de Amazon ECS.

Actualizaciones de Amazon ECS para políticas administradas por AWS

Consultar los detalles sobre las actualizaciones de las políticas administradas por AWS para Amazon ECS ya que este servicio comenzó a realizar el seguimiento de estos cambios. Para obtener alertas automáticas sobre cambios en esta página, suscríbase a la fuente RSS en la página de historial de documentos de Amazon ECS.

Cambio	Descripción	Fecha
Agregar permisos a the section called “AmazonEC2ContainerServiceforEC2Role”	La política de IAM administrada `AmazonEC2ContainerServiceforEC2Role` se actualizó para incluir el permiso `ecs:ListTagsForResource`. Este permiso permite al agente de Amazon ECS recuperar las etiquetas de tareas y de instancias de contenedores a través del punto de conexión de metadatos de la tarea (`${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`).	4 de agosto de 2025
Agregue permisos a AmazonECSInfrastructureRolePolicyForLoadBalancers.	La política de IAM administrada `AmazonECSInfrastructureRolePolicyForLoadBalancers` se actualizó con nuevos permisos para describir los grupos de	25 de julio de 2025

Cambio	Descripción	Fecha
	destino, registrarlos y anular el registro de estos.	
Adición de la nueva política the section called “`AmazonECSInfrastructureRolePolicyForLoadBalancers`”	Se agregó la nueva política AmazonECSInfrastructureRolePolicyForLoadBalancers, que proporciona acceso a otros recursos de servicio de AWS necesarios para administrar los equilibradores de carga asociados a las cargas de trabajo de Amazon ECS.	15 de julio de 2025
Agregue permisos a AmazonECSServiceRolePolicy.	La política de IAM administrada `AmazonECSServiceRolePolicy` se actualizó con nuevos permisos AWS Cloud Map que permiten a Amazon ECS actualizar los atributos de los servicios de AWS Cloud Map que administra Amazon ECS.	15 de julio de 2025
Agregar permisos a AmazonECSServiceRolePolicy	La política de IAM administrada `AmazonECSServiceRolePolicy` se actualizó con nuevos permisos AWS Cloud Map que permiten a Amazon ECS actualizar los atributos de los servicios de AWS Cloud Map que administra Amazon ECS.	24 de junio de 2025

Cambio	Descripción	Fecha
Adición de permisos a AmazonECSInfrastructureRolePolicyForVolumes	Se actualizó la política `AmazonECSInfrastructureRolePolicyForVolumes` para agregar el permiso `ec2:DescribeInstances`. El permiso ayuda a prevenir la colisión de nombres de dispositivos para volúmenes de Amazon EBS que están asociados a tareas de Amazon ECS que se ejecutan en la misma instancia de contenedor.	2 de junio de 2025
Adición de una nueva política AmazonECSInfrastructureRolePolicyForVpcLattice	Proporciona acceso a otros recursos de servicio de AWS necesarios para administrar en su nombre la característica de VPC Lattice en las cargas de trabajo de Amazon ECS.	18 de noviembre de 2024
Adición de permisos a AmazonECSInfrastructureRolePolicyForVolumes	La política `AmazonECSInfrastructureRolePolicyForVolumes` se ha actualizado para permitir a los clientes crear un volumen de Amazon EBS a partir de una instantánea.	10 de octubre de 2024

Cambio	Descripción	Fecha
Permisos agregados de the section called “AmazonECS_FullAccess”.	La política `AmazonECS_FullAccess` se actualizó para añadir permisos `iam:PassRole` para los roles de IAM para una función denominada `ecsInfrastructureRole`. Este es el rol de IAM predeterminado que crea AWS Management Console destinado a usarse como un rol de infraestructura de ECS que permite a Amazon ECS administrar los volúmenes de Amazon EBS adjuntos a las tareas de ECS.	13 de agosto de 2024
Adición de la nueva política AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity	Se agregó la nueva política AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity, que proporciona acceso administrativo a AWS KMS, AWS Private Certificate Authority y Secrets Manager y permite que las características de la TLS de Amazon ECS Service Connect funcionen correctamente.	22 de enero de 2024

Cambio	Descripción	Fecha
Adición de la nueva política AmazonECSInfrastructureRolePolicyForVolumes	Se agregó la política `AmazonECSInfrastructureRolePolicyForVolumes` . La política concede los permisos que Amazon ECS necesita para hacer llamadas a la API de AWS con el fin de administrar los volúmenes de Amazon EBS asociados a las cargas de trabajo de Amazon ECS.	11 de enero de 2024
Agregar permisos a AmazonECSServiceRolePolicy	La política de IAM administrada `AmazonECSServiceRolePolicy` se actualizó con los nuevos permisos `events` y los permisos adicionales `autoscaling` y `autoscaling-plans` .	4 de diciembre de 2023
Adición de permisos a AmazonEC2ContainerServiceEventsRole	La política de IAM administrada `AmazonECSServiceRolePolicy` se actualizó para permitir el acceso a la operación de la API de AWS Cloud Map `DiscoverInstancesRevision` .	4 de octubre de 2023

Cambio	Descripción	Fecha
Agregar permisos a AmazonEC2ContainerServiceforEC2Role	La política `AmazonEC2ContainerServiceforEC2Role` se modificó para agregar el permiso `ecs:TagResource`, que incluye una condición que limita el permiso solo a clústeres recién creados e instancias de contenedor registradas.	6 de marzo de 2023
Agregar permisos a the section called “AmazonECS_FullAccess”	La política `AmazonECS_FullAccess` se modificó para agregar el permiso `elasticloadbalancing:AddTags`, que incluye una condición que limita el permiso solo a equilibradores de carga, grupos de destino, reglas y oyentes recién creados. Este permiso no permite agregar etiquetas a ningún recurso de Elastic Load Balancing ya creado.	4 de enero de 2023
Amazon ECS comenzó a realizar el seguimiento de los cambios	Amazon ECS comenzó a realizar el seguimiento de los cambios de las políticas administradas por AWS.	8 de junio de 2021

Eliminación gradual de las políticas de IAM administradas de AWS para Amazon Elastic Container Service

Las siguientes políticas de IAM administradas por AWS se eliminaron gradualmente. Estas políticas ahora se sustituyen por las políticas actualizadas. Se recomienda actualizar los usuarios o roles para utilizar las políticas actualizadas.

AmazonEC2ContainerServiceFullAccess

> **Important**
>
> La política de IAM administrada por `AmazonEC2ContainerServiceFullAccess` se eliminó gradualmente a partir del 29 de enero de 2021, en respuesta a un problema de seguridad detectado en el permiso `iam:passRole`. Este permiso concede acceso a todos los recursos, incluso a las credenciales de los roles de la cuenta. Ahora que la política se eliminó gradualmente, no la puede asociar a ningún nuevo usuario o rol. Cualquier usuario o rol que ya tenga asociada la política puede continuar utilizándola. No obstante, es aconsejable que actualice sus usuarios o roles para que utilicen la política administrada `AmazonECS_FullAccess`. Para obtener más información, consulte Migración a la política administrada `AmazonECS_FullAccess`.

AmazonEC2ContainerServiceRole

> **Important**
>
> La política de IAM administrada `AmazonEC2ContainerServiceRole` se eliminó gradualmente. Ahora se sustituye por el rol vinculado al servicio de Amazon ECS. Para obtener más información, consulte Uso de roles vinculados al servicio para Amazon ECS.

AmazonEC2ContainerServiceAutoscaleRole

> **Important**
>
> La política de IAM administrada `AmazonEC2ContainerServiceAutoscaleRole` se eliminó gradualmente. Ahora se sustituye por el rol vinculado al servicio Application Auto

Scaling de Amazon ECS. Para obtener más información, consulte Roles vinculados a servicios de Application Auto Scaling en la Guía del usuario de Application Auto Scaling.

Migración a la política administrada **`AmazonECS_FullAccess`**

La política de IAM administrada `AmazonEC2ContainerServiceFullAccess` se eliminó gradualmente el 29 de enero de 2021, en respuesta a un problema de seguridad detectado en el permiso `iam:passRole`. Este permiso concede acceso a todos los recursos, incluso a las credenciales de los roles de la cuenta. Ahora que la política se eliminó gradualmente, no la puede asociar a grupos, usuarios o roles nuevos. Todos los grupos, usuarios o roles que ya tengan asociada la política, pueden continuar utilizándola. No obstante, es aconsejable que actualice sus grupos, usuarios o roles para que utilicen la política administrada `AmazonECS_FullAccess`.

Los permisos que concede la política `AmazonECS_FullAccess` incluye la lista completa de permisos necesarios para utilizar ECS como administrador. Si actualmente utiliza permisos concedidos por la política `AmazonEC2ContainerServiceFullAccess` que no están en la política `AmazonECS_FullAccess`, puede agregarlos a una instrucción de política en línea. Para obtener más información, consulte Políticas administradas por AWS para Amazon Elastic Container Service.

Siga estos pasos para determinar si tiene grupos, usuarios o roles que actualmente estén utilizando la política de IAM administrada `AmazonEC2ContainerServiceFullAccess`. A continuación, actualícelos para separar desasociar la política anterior y asociar la `AmazonECS_FullAccess`.

Para actualizar un grupo, usuario o rol a fin de que utilice la política AmazonECS_FullAccess (AWS Management Console)

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Policies (Políticas) y busque y seleccione la política `AmazonEC2ContainerServiceFullAccess`.
3. Elija la pestaña Policy usage (Uso de políticas) que muestre cualquier rol de IAM que actualmente esté utilizando esta política.
4. Para cada rol de IAM que actualmente esté utilizando la política `AmazonEC2ContainerServiceFullAccess`, seleccione el rol y siga estos pasos para desasociar la política obsoleta y adjuntar la política `AmazonECS_FullAccess`.

    a. En la página Permissions (Permisos), elija la X junto a la política AmazonEC2ContainerServiceFullAccess.

b. Elija Añadir permisos.

c. Elija Attach existing policies directly (Asociar políticas existentes directamente), busque y seleccione la política AmazonECS_FullAccess y, a continuación, elija Next: Review (Siguiente: Revisar).

d. Revise los cambios y, luego, seleccione Add permissions (Agregar permisos).

e. Repita estos pasos para cada grupo, usuario o rol que utilice la política `AmazonEC2ContainerServiceFullAccess`.

Para actualizar un grupo, usuario o rol a fin de que utilice la política **`AmazonECS_FullAccess`** (AWS CLI)

1. Utilice el comando generate-service-last-accessed-details para generar un informe que incluya detalles sobre cuándo se usó la política obsoleta por última vez.

```
aws iam generate-service-last-accessed-details \
     --arn arn:aws:iam::aws:policy/AmazonEC2ContainerServiceFullAccess
```

Ejemplo de salida:

```
{
    "JobId": "32bb1fb0-1ee0-b08e-3626-ae83EXAMPLE"
}
```

2. Utilice el ID de trabajo del resultado anterior con el comando get-service-last-accessed-details para recuperar el último informe del servicio al que se accedió. Este informe muestra el nombre de recurso de Amazon (ARN) de las entidades de IAM que usaron la política obsoleta por última vez.

```
aws iam get-service-last-accessed-details \
      --job-id 32bb1fb0-1ee0-b08e-3626-ae83EXAMPLE
```

3. Utilice uno de los siguientes comandos para desasociar la política `AmazonEC2ContainerServiceFullAccess` de un grupo, usuario o rol.

   - detach-group-policy
   - detach-role-policy
   - detach-user-policy

4. Utilice uno de los siguientes comandos para asociar la política `AmazonECS_FullAccess` a un grupo, usuario o rol.

   - attach-group-policy
   - attach-role-policy
   - attach-user-policy

Uso de roles vinculados al servicio para Amazon ECS

Amazon Elastic Container Service utiliza roles vinculados al servicio de AWS Identity and Access Management (IAM). Un rol vinculado al servicio es un tipo único de rol de IAM que se encuentra vinculado directamente a Amazon ECS. El rol vinculado al servicio está predefinido por Amazon ECS e incluye todos los permisos que el servicio requiere para llamar a otros servicios de AWS en su nombre.

Un rol vinculado al servicio simplifica la configuración de Amazon ECS porque ya no tendrá que agregar manualmente los permisos requeridos. Amazon ECS define los permisos de sus roles vinculados al servicio y, a menos que esté definido de otra manera, solo Amazon ECS puede asumir sus roles. Los permisos definidos incluyen las políticas de confianza y de permisos, y que la política de permisos no se pueda asociar a ninguna otra entidad de IAM.

Para obtener información sobre otros servicios que admiten roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que muestran Sí en la columna Roles vinculados a servicios. Elija una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Permisos de roles vinculados al servicio para Amazon ECS

Amazon ECS usa el rol vinculado al servicio denominado AWSServiceRoleForECS.

El rol vinculado al servicio AWSServiceRoleForECS depende de los siguientes servicios para asumir el rol:

- `ecs.amazonaws.com`

La política de permisos de rol denominada AmazonECSServiceRolePolicy permite a Amazon ECS llevar a cabo las siguientes acciones en los recursos especificados:

- Acción: al utilizar el modo de red `awsvpc` para las tareas de Amazon ECS, este administra el ciclo de vida de las interfaces de red elásticas asociadas a la tarea. Esto también incluye las etiquetas que Amazon ECS agrega a sus interfaces de red elástica.
- Acción: al utilizar un equilibrador de carga con su servicio de Amazon ECS, este administra el registro y la anulación del registro de los recursos con el equilibrador de carga.
- Acción: al utilizar la detección de servicios de Amazon ECS, este administra los recursos de AWS Cloud Map y Route 53 necesarios para que la detección de servicios funcione.
- Acción: al utilizar el escalado automático de servicios de Amazon ECS, este administra los recursos de escalado automático necesarios.
- Acción: Amazon ECS crea y administra alarmas y flujos de registro de CloudWatch que ayudan a la supervisión de sus recursos de Amazon ECS.
- Acción: al utilizar Amazon ECS Exec, Amazon ECS administra los permisos necesarios para iniciar sesiones de Amazon ECS Exec en sus tareas.
- Acción: al utilizar Amazon ECS Service Connect, Amazon ECS administra los recursos de AWS Cloud Map necesarios para utilizar la función.
- Acción: al utilizar proveedores de capacidad de Amazon ECS, este servicio administra los permisos necesarios para modificar el grupo de escalado automático y sus instancias de Amazon EC2.
- Acción: Amazon ECS puede actualizar los atributos de los servicios de AWS Cloud Map que administra Amazon ECS.

Debe configurar permisos para permitir a una entidad de IAM (como un usuario, grupo o rol) crear, editar o eliminar un rol vinculado a servicios. Para obtener más información, consulte Permisos de roles vinculados a servicios en la Guía del usuario de IAM.

Creación de un rol vinculado al servicio para Amazon ECS

En la mayoría de los casos, no es necesario crear manualmente roles vinculados a servicios. Cuando crea un clúster o crea o actualiza un servicio en la AWS Management Console, la AWS CLI, o la API de AWS, Amazon ECS crea el rol vinculado al servicio. Si no ve el rol AWSServiceRoleForECS después de crear un clúster, lleve a cabo lo siguiente para solucionar el problema:

- Verifique y configure los permisos para permitir a Amazon ECS crear, editar o eliminar un rol vinculado a un servicio en su nombre. Para obtener más información, consulte Permisos de roles vinculados a servicios en la Guía del usuario de IAM.

- Reintente la operación de creación del clúster o cree manualmente el rol vinculado a servicios.

  Puede utilizar la consola de IAM para crear el rol vinculado al servicio AWSServiceRoleForECS. En la AWS CLI o la API de AWS, cree un rol vinculado al servicio con el nombre de servicio `ecs.amazonaws.com`. Para obtener más información, consulte Creación de un rol vinculado a un servicio en la Guía del usuario de IAM.

> **Important**
>
> Este rol vinculado a servicios puede aparecer en su cuenta si se ha completado una acción en otro servicio que utilice las características compatibles con este rol.

Si elimina este rol vinculado a servicios y necesita crearlo de nuevo, puede utilizar el mismo proceso para volver a crear el rol en su cuenta. Cuando crea un clúster o crea o actualiza un servicio, Amazon ECS vuelve a crear el rol vinculado al servicio.

Si elimina este rol vinculado al servicio, puede utilizar el mismo proceso de IAM para volver a crear el rol.

Edición de un rol vinculado al servicio para Amazon ECS

Amazon ECS no permite editar el rol vinculado al servicio AWSServiceRoleForECS. Después de crear un rol vinculado al servicio, no podrá cambiar el nombre del rol, ya que varias entidades podrían hacer referencia al rol. Sin embargo, sí puede editar la descripción del rol con IAM. Para obtener más información, consulte Actualizar un rol vinculado a servicios en la Guía del usuario de IAM.

Eliminación de un rol vinculado al servicio para Amazon ECS

Si ya no necesita usar una característica o servicio que requieran un rol vinculado a un servicio, le recomendamos que elimine dicho rol. Así no tendrá una entidad no utilizada que no se supervise ni mantenga de forma activa. Sin embargo, debe limpiar los recursos de su rol vinculado al servicio antes de eliminarlo manualmente.

> **Note**
>
> Si el servicio de Amazon ECS utiliza el rol al intentar eliminar los recursos, se podría producir un error en la eliminación. En tal caso, espere unos minutos e intente de nuevo la operación.

Para comprobar si el rol vinculado a servicio tiene una sesión activa

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Roles y seleccione el nombre AWSServiceRoleForECS (no la casilla de verificación).
3. En la página Summary, elija Access Advisor y revise la actividad reciente del rol vinculado a servicio.

   > **Note**
   >
   > Si no está seguro de si Amazon ECS utiliza el rol AWSServiceRoleForECS, puede intentar eliminar el rol para comprobarlo. Si el servicio utiliza el rol, este no podrá eliminarse y podrá ver las regiones en las que se utiliza. Si el rol se está utilizando, debe esperar que la sesión finalice para poder eliminarlo. No se puede revocar la sesión de un rol vinculado a servicios.

Para eliminar los recursos de Amazon ECS que utiliza el rol vinculado al servicio AWSServiceRoleForECS

Debe eliminar todos los clústeres de Amazon ECS de todas las regiones de AWS para poder eliminar el rol AWSServiceRoleForECS.

1. Reduzca los servicios de Amazon ECS hasta el recuento deseado de 0 en todas las regiones y, a continuación, elimine los servicios. Para obtener más información, consulte Actualización de un servicio de Amazon ECS y Eliminación de un servicio de Amazon ECS mediante la consola.
2. Fuerce la cancelación del registro de todas las instancias de contenedor de todos los clústeres de todas las regiones. Para obtener más información, consulte Anulación del registro de una instancia de contenedor de Amazon ECS.
3. Elimine todos los clústeres de Amazon ECS de todas las regiones. Para obtener más información, consulte Eliminación de un clúster de Amazon ECS.

Para eliminar manualmente el rol vinculado a servicios mediante IAM

Utilice la consola de IAM, la AWS CLI o la API de AWS para eliminar el rol vinculado al servicio AWSServiceRoleForECS. Para obtener más información, consulte Eliminación de un rol vinculado a servicios en la Guía del usuario de IAM.

Regiones admitidas para los roles vinculados al servicio de Amazon ECS

Amazon ECS admite el uso de roles vinculados al servicio en todas las regiones en las que se encuentra disponible el servicio. Para obtener más información, consulte Puntos de enlace y regiones de AWS.

Roles de IAM para Amazon ECS

Un rol de IAM es una identidad de IAM que puede crear en su cuenta y que tiene permisos específicos. En Amazon ECS, puede crear roles para conceder permisos a los recursos de Amazon ECS, como contenedores o servicios.

Los roles que Amazon ECS necesita dependen del tipo de lanzamiento de la definición de la tarea y de las características que utilice. Utilice la siguiente tabla para determinar qué roles de IAM necesita para Amazon ECS.

Rol	Definición	Cuando sea necesario	Más información
Rol de ejecución de tareas	Este rol permite que Amazon ECS utilice otros servicios de AWS en su nombre.	La tarea está alojada en AWS Fargate o en instancias externas y: • extrae una imagen de contenedor de un repositorio privado de Amazon ECR. • extrae una imagen de contenedor de un repositorio privado de Amazon ECR en una cuenta	Rol de IAM de ejecución de tareas de Amazon ECS

Rol	Definición	Cuando sea necesario	Más información
		diferente de la cuenta que ejecuta la tarea. • envía registros de contenedor a CloudWatch Logs con el controlador de registros `awslogs`. La tarea está alojada en AWS Fargate o en instancias de Amazon EC2 y: • utiliza la autenticación de registros privados. • utiliza la supervisión en tiempo de ejecución. • la definición de la tarea hace referencia a información confidencial mediante secretos de Secrets Manager o parámetros del Almacén de parámetros de AWS Systems Manager.	

Rol	Definición	Cuando sea necesario	Más información
Rol de la tarea	Este rol permite que el código de la aplicación (en el contenedor) utilice otros servicios de AWS.	La aplicación obtiene acceso a otros servicios de AWS, como Amazon S3.	Rol de IAM de tarea de Amazon ECS
Rol de la instancia de contenedor	Este rol permite que sus instancias de EC2 o instancias externas se registren en el clúster.	La tarea está alojada en instancias de Amazon EC2 o en una instancia externa.	Rol de IAM de instancia de contenedor de Amazon ECS
Rol de Amazon ECS Anywhere	Este rol permite que sus instancias externas accedan a las API de AWS.	La tarea está alojada en instancias externas.	Rol de IAM de Amazon ECS Anywhere
Infraestructura de Amazon ECS para el rol de equilibradores de carga	Este rol permite a Amazon ECS administrar los recursos del equilibrador de carga en los clústeres en su nombre para las implementaciones azul/verde.	Quiere utilizar las implementaciones azul/verde de Amazon ECS.	Rol de IAM de infraestructura de Amazon ECS para los equilibradores de carga
Rol de CodeDeploy de Amazon ECS	Este rol permite a CodeDeploy llevar a cabo actualizaciones en sus servicios.	Utilice el tipo de implementación azul/verde de CodeDeploy para implementar servicios.	Rol de IAM de CodeDeploy de Amazon ECS

Rol	Definición	Cuando sea necesario	Más información
Rol de EventBridge de Amazon ECS	Este rol permite a EventBridge llevar a cabo actualizaciones en sus servicios.	Utilice las reglas y los objetivos de EventBridge para programar las tareas.	Rol de IAM de EventBridge de Amazon ECS
Rol de infraestructura de Amazon ECS	Este rol permite a Amazon ECS administrar los recursos de infraestructura de sus clústeres.	• Adjunte volúmenes de Amazon EBS a sus tareas de Amazon ECS del tipo lanzamiento de Fargate o EC2. El rol de infraestructura permite a Amazon ECS administrar los volúmenes de Amazon EBS para sus tareas. • Utilice la seguridad de la capa de transporte (TLS) para cifrar el tráfico entre sus servicios de Amazon ECS Service Connect. • Quiere crear grupos de destino de VPC Lattice.	Rol de IAM de infraestructura de Amazon ECS

Prácticas recomendadas de roles de IAM en Amazon ECS

Los roles que Amazon ECS necesita dependen del tipo de lanzamiento de la definición de la tarea y de las características que utilice. Recomendamos que cree roles separados en la tabla en lugar de compartir roles.

Rol	Definición	Cuando sea necesario	Más información
Rol de ejecución de tareas	Este rol permite que Amazon ECS utilice otros servicios de AWS en su nombre.	La tarea está alojada en AWS Fargate o en instancias externas y: • extrae una imagen de contenedor de un repositorio privado de Amazon ECR. • extrae una imagen de contenedor de un repositorio privado de Amazon ECR en una cuenta diferente de la cuenta que ejecuta la tarea. • envía registros de contenedor a CloudWatch Logs con el controlador de registros `awslogs`. La tarea está alojada en AWS Fargate o en instancias de Amazon EC2 y:	Rol de IAM de ejecución de tareas de Amazon ECS

Rol	Definición	Cuando sea necesario	Más información
		• utiliza la autenticación de registros privados. • utiliza la supervisión en tiempo de ejecución. • la definición de la tarea hace referencia a información confidencial mediante secretos de Secrets Manager o parámetros del Almacén de parámetros de AWS Systems Manager.	
Rol de la tarea	Este rol permite que el código de la aplicación (en el contenedor) utilice otros servicios de AWS.	La aplicación obtiene acceso a otros servicios de AWS, como Amazon S3.	Rol de IAM de tarea de Amazon ECS
Rol de la instancia de contenedor	Este rol permite que sus instancias de EC2 o instancias externas se registren en el clúster.	La tarea está alojada en instancias de Amazon EC2 o en una instancia externa.	Rol de IAM de instancia de contenedor de Amazon ECS

Rol	Definición	Cuando sea necesario	Más información
Rol de Amazon ECS Anywhere	Este rol permite que sus instancias externas accedan a las API de AWS.	La tarea está alojada en instancias externas.	Rol de IAM de Amazon ECS Anywhere
Rol de CodeDeploy de Amazon ECS	Este rol permite a CodeDeploy llevar a cabo actualizaciones en sus servicios.	Utilice el tipo de implementación azul/verde de CodeDeplo y para implementar servicios.	Rol de IAM de CodeDeploy de Amazon ECS
Rol de EventBridge de Amazon ECS	Este rol permite a EventBridge llevar a cabo actualizaciones en sus servicios.	Utilice las reglas y los objetivos de EventBrid ge para programar las tareas.	Rol de IAM de EventBridge de Amazon ECS

Rol	Definición	Cuando sea necesario	Más información
Rol de infraestructura de Amazon ECS	Este rol permite a Amazon ECS administrar los recursos de infraestructura de sus clústeres.	• Adjunte volúmenes de Amazon EBS a sus tareas de Amazon ECS del tipo lanzamiento de Fargate o EC2. El rol de infraestructura permite a Amazon ECS administrar los volúmenes de Amazon EBS para sus tareas. • Utilice la seguridad de la capa de transporte (TLS) para cifrar el tráfico entre sus servicios de Amazon ECS Service Connect. • Quiere crear grupos de destino de VPC Lattice.	Rol de IAM de infraestructura de Amazon ECS

Rol de la tarea

Le recomendamos que asigne un rol de tarea. Su rol puede distinguirse del rol de la instancia de Amazon EC2 en la que se ejecutan. La asignación de un rol a cada tarea se ajusta al principio de acceso con privilegio mínimo y permite un mayor control pormenorizado de acciones y recursos.

Al agregar un rol de tarea a una definición de tarea, el agente contenedor de Amazon ECS crea automáticamente un token con un ID de credencial único (por ejemplo, `12345678-90ab-cdef-1234-567890abcdef`) para la tarea. A continuación, este token y las credenciales del

rol se agregan a la memoria caché interna del agente. El agente rellena la variable de entorno `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` del contenedor con el URI del ID de la credencial (por ejemplo, `/v2/credentials/12345678-90ab-cdef-1234-567890abcdef`).

Puede recuperar manualmente las credenciales del rol temporal desde el interior de un contenedor adjuntando la variable de entorno a la dirección IP del agente de contenedor de Amazon ECS y ejecutando el comando `curl` en la cadena resultante.

```
curl 169.254.170.2$AWS_CONTAINER_CREDENTIALS_RELATIVE_URI
```

El resultado esperado es el siguiente:

```
{
 "RoleArn": "arn:aws:iam::123456789012:role/SSMTaskRole-SSMFargateTaskIAMRole-
DASWWSF2WGD6",
 "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
 "Token": "IQoJb3JpZ2luX2VjEEM/Example==",
 "Expiration": "2021-01-16T00:51:53Z"
}
```

Las versiones más recientes de los SDK de AWS obtienen automáticamente estas credenciales de la variable de entorno `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` al realizar llamadas a la API de AWS. Para obtener información sobre cómo renovar las credenciales, consulte Renovación de credenciales de AWS en rePost.

El resultado incluye un par de claves de acceso compuesto por un ID de clave de acceso secreta y una clave secreta que la aplicación utiliza para acceder a los recursos de AWS. También incluye un token que AWS utiliza para comprobar que las credenciales son válidas. De forma predeterminada, las credenciales asignadas a las tareas que utilizan roles de tarea son válidas durante seis horas. Después de eso, el agente de contenedor de Amazon ECS los rota automáticamente.

Rol de ejecución de tareas

El rol de IAM de ejecución de tareas se utiliza para conceder permiso al agente de contenedor de Amazon ECS para llamar a acciones específicas de la API de AWS en su nombre. Por ejemplo, cuando usa AWS Fargate, Fargate necesita un rol de IAM que le permita extraer imágenes de Amazon ECR y escribir registros en CloudWatch Logs. También se requiere un rol de IAM cuando una tarea hace referencia a un secreto almacenado en AWS Secrets Manager, como un secreto de extracción de imágenes.

> **Note**
>
> Si extrae imágenes como usuario autenticado, es menos probable que le afecten los cambios que se hayan producido en los límites de la tasa de extracción de Docker Hub. Para obtener más información, consulte Autenticación de registros privados para instancias de contenedor. Al utilizar Amazon ECR y Amazon ECR Public, puede evitar los límites impuestos por Docker. Si extrae imágenes de Amazon ECR, esto también ayuda a acortar los tiempos de extracción de la red y reduce los cambios en la transferencia de datos cuando el tráfico abandona la VPC.

> **Important**
>
> Cuando utilice Fargate, debe autenticarse en un registro de imágenes privado utilizando `repositoryCredentials`. No es posible configurar las variables de entorno del agente de contenedor de Amazon ECS `ECS_ENGINE_AUTH_TYPE` o `ECS_ENGINE_AUTH_DATA` ni modificar el archivo `ecs.config` para las tareas alojadas en Fargate. Para obtener más información, consulte Autenticación de registros privados para tareas.

Rol de la instancia de contenedor

La política de IAM administrada `AmazonEC2ContainerServiceforEC2Role` debe incluir los siguientes permisos. Siguiendo el consejo de seguridad estándar de concesión de privilegios mínimos, la política administrada `AmazonEC2ContainerServiceforEC2Role` se puede utilizar como guía. Si no necesita ninguno de los permisos que se conceden en la política administrada para su caso de uso, cree una política personalizada y agregue solo los permisos que necesite.

- `ec2:DescribeTags`: (opcional) permite que una entidad principal describa las etiquetas asociadas a una instancia de Amazon EC2. El agente contenedor de Amazon ECS utiliza este permiso para admitir la propagación de etiquetas de recursos. Para obtener más información, consulte Cómo se etiquetan los recursos.
- `ecs:CreateCluster`: permite a una entidad principal crear un clúster de Amazon ECS (opcional). Este permiso lo utiliza el agente de contenedor de Amazon ECS para crear un clúster `default` si aún no existe ninguno.
- `ecs:DeregisterContainerInstance`: permite a una entidad principal anular el registro de una instancia de contenedor de Amazon ECS de un clúster (opcional). El agente de contenedor de

Amazon ECS no llama a esta operación de la API, pero este permiso se conserva para ayudar a garantizar la compatibilidad con versiones anteriores.

- `ecs:DiscoverPollEndpoint`: esta acción devuelve puntos de conexión que el agente de contenedor de Amazon ECS utiliza para buscar actualizaciones (obligatorio).
- `ecs:Poll`: permite que el agente de contenedor de Amazon ECS se comunique con el plano de control de Amazon ECS y le informe los cambios en el estado de la tarea (obligatorio).
- `ecs:RegisterContainerInstance`: permite que una entidad principal registre una instancia de contenedor con un clúster (obligatorio). El agente contenedor de Amazon ECS utiliza este permiso para registrar la instancia de Amazon EC2 con un clúster, así como para admitir la propagación de etiquetas de recursos.
- `ecs:StartTelemetrySession`: permite que el agente de contenedor de Amazon ECS se comunique con el plano de control de Amazon ECS para enviar la información de estado y las métricas de cada contenedor y tarea (opcional).

  Aunque este permiso no es obligatorio, recomendamos que lo agregue para que las métricas de la instancia de contenedor puedan iniciar acciones de escalado y también recibir informes relacionados con los comandos de comprobación de estado.
- `ecs:TagResource`: permite que el agente de contenedor de Amazon ECS etiquete el clúster al crearlo y etiquete las instancias de contenedor cuando se registran en un clúster (opcional).
- `ecs:UpdateContainerInstancesState`: permite que un usuario principal modifique el estado de una instancia de contenedor de Amazon ECS. El agente contenedor de Amazon ECS utiliza este permiso para el vaciado de instancias de spot.
- `ecs:Submit*`: esto incluye las acciones de la API `SubmitAttachmentStateChanges`, `SubmitContainerStateChange` y `SubmitTaskStateChange` (obligatorio). Los utiliza el agente de contenedores de Amazon ECS para informar los cambios de estado de cada recurso al plano de control de Amazon ECS. El agente de contenedor de Amazon ECS ya no utiliza el permiso `SubmitContainerStateChange`, pero se conserva para ayudar a garantizar la compatibilidad con versiones anteriores.
- `ecr:GetAuthorizationToken`: permite que una entidad principal recupere un token de autorización (opcional). Un token de autorización representa las credenciales de autenticación de IAM y se puede utilizar para acceder a cualquier registro de Amazon ECR al que tenga acceso el elemento principal de IAM. El token de autorización recibido es válido durante 12 horas.
- `ecr:BatchCheckLayerAvailability`: cuando se inserta una imagen de contenedor en un repositorio privado de Amazon ECR, se comprueba cada capa de imagen para verificar si ya se insertó (opcional). Si se insertó, se omite la capa de imagen.

- `ecr:GetDownloadUrlForLayer`: cuando se extrae una imagen de contenedor de un repositorio privado de Amazon ECR, se llama a esta API una vez para cada capa de imagen que aún no está almacenada en caché (opcional).
- `ecr:BatchGetImage`: cuando se extrae una imagen de contenedor de un repositorio privado de Amazon ECR, se llama una vez a esta API para recuperar el manifiesto de la imagen (opcional).
- `logs:CreateLogStream`: permite que una entidad principal cree un flujo de registro de los registros de CloudWatch para un grupo de registro especificado (opcional).
- `logs:PutLogEvents`: permite que una entidad principal cargue un lote de eventos de registro en un flujo de registro especificado (opcional).

Roles vinculados a servicios

Puede utilizar el rol vinculado al servicio para que Amazon ECS conceda el permiso de servicio de Amazon ECS para llamar a otras API del servicio en su nombre. Amazon ECS necesita los permisos para crear y eliminar interfaces de red, registrar y anular el registro de objetivos en un grupo objetivo. También necesita los permisos necesarios para crear y eliminar políticas de escalado. Estos permisos se conceden mediante el rol vinculado al servicio. Este rol se crea en su nombre la primera vez que utiliza el servicio.

> **Note**
>
> Si elimina de forma inadvertida el rol vinculado al servicio, puede volver a crearlo. Para obtener instrucciones, consulte Crear el rol vinculado al servicio.

Recomendaciones de roles

Al configurar los roles y las políticas de tarea de IAM, recomendamos que realice las siguientes acciones.

Bloqueo del acceso a metadatos de Amazon EC2

Cuando ejecute sus tareas en instancias de Amazon EC2, le recomendamos encarecidamente que bloquee el acceso a los metadatos de Amazon EC2 para evitar que sus contenedores hereden el rol asignado a esas instancias. Si sus aplicaciones tienen que solicitar una acción de API de AWS, utilice en su lugar los roles de IAM para las tareas.

Para evitar que las tareas que se ejecutan en modo puente accedan a los metadatos de Amazon EC2, ejecute el siguiente comando o actualice los datos de usuario de la instancia. Para obtener más instrucciones sobre cómo actualizar los datos de usuario de una instancia, consulte este artículo de soporte técnico de AWS. Para obtener más información sobre el modo puente de definición de tareas, consulte el modo red de definición de tareas.

```
sudo yum install -y iptables-services; sudo iptables --insert FORWARD 1 --in-interface
 docker+ --destination 169.254.169.254/32 --jump DROP
```

Para que este cambio persista tras un reinicio, ejecute el siguiente comando específico para su imagen de máquina de Amazon (AMI):

- Amazon Linux 2

  ```
  sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now
   iptables
  ```

- Amazon Linux

  ```
  sudo service iptables save
  ```

Para las tareas que utilizan el modo de red de `awsvpc`, defina la variable de entorno `ECS_AWSVPC_BLOCK_IMDS` y `true` en el archivo `/etc/ecs/ecs.config`.

Debe establecer la variable `ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST` en `false` en el archivo `ecs-agent config` para evitar que los contenedores que se ejecutan en la red `host` accedan a los metadatos de Amazon EC2.

Uso del modo de red **`awsvpc`**

Utilice el modo de red `awsvpc` para restringir el flujo de tráfico entre diferentes tareas o entre sus tareas y otros servicios que se ejecutan en su VPC de Amazon. Esto agrega una capa de seguridad adicional. El modo de red `awsvpc` proporciona aislamiento de red a nivel de tarea para las tareas que se ejecutan en Amazon EC2. Se trata del modo predeterminado en AWS Fargate. Es el único modo de red que puede utilizar para asignar un grupo de seguridad a las tareas.

Uso de la información del último acceso para limitar los roles

Le recomendamos que elimine cualquier acción que no se haya utilizado nunca o que no se haya utilizado durante algún tiempo. Esto evita que se produzcan accesos no deseados. Para ello, revise

la información del último acceso que proporciona IAM y, a continuación, elimine las acciones que no se hayan utilizado nunca o que no se hayan utilizado recientemente. Para ello, siga los siguientes pasos.

Utilice el siguiente comando para generar un informe que muestre la información del último acceso de la política de referencia:

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

utilice el `JobId` que estaba en el resultado para ejecutar el siguiente comando. A continuación, podrá ver los resultados del informe.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

Para obtener más información, consulte Ajuste de permisos en AWS con información sobre los últimos accesos.

Supervisar AWS CloudTrail para detectar actividades sospechosas

Puede supervisar AWS CloudTrail para detectar cualquier actividad sospechosa. La mayoría de las llamadas a la API de AWS se registran en AWS CloudTrail como eventos. AWS CloudTrail Insights las analiza e informa de cualquier comportamiento sospechoso asociado a las llamadas a la API de `write`. Esto podría incluir un aumento en el volumen de llamadas. Estas alertas incluyen información como la hora en que se produjo la actividad inusual y el ARN de identidad principal que contribuyó a las API.

Puede identificar las acciones que realizan las tareas con un rol de IAM en AWS CloudTrail consultando la propiedad de `userIdentity` del evento. En el siguiente ejemplo, el `arn` incluye el nombre del rol asumido, `s3-write-go-bucket-role`, seguido del nombre de la tarea, `7e9894e088ad416eb5cab92afExample`.

```
"userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROA36C6WWEJ2YEXAMPLE:7e9894e088ad416eb5cab92afExample",
    "arn": "arn:aws:sts::123456789012:assumed-role/s3-write-go-bucket-
role/7e9894e088ad416eb5cab92afExample",
    ...
}
```

> **Note**
>
> Cuando las tareas que asumen un rol se ejecutan en instancias de contenedor de Amazon EC2, el agente de contenedor de Amazon ECS registra una solicitud en el registro de auditoría del agente que se encuentra en una dirección con el formato `/var/log/ecs/audit.log.YYYY-MM-DD-HH`. Para obtener más información, consulte Registro de roles de IAM y Registrar eventos de Insights en Trails.

Rol de IAM de ejecución de tareas de Amazon ECS

El rol de ejecución de tareas concede al agente de contenedor de Amazon ECS y al agente de Fargate permiso para realizar llamadas a la API de AWS en su nombre. El rol de IAM de ejecución de tareas es necesario en función de los requisitos de la tarea. Puede tener varios roles de ejecución de tareas asociados a su cuenta para distintos fines y servicios.

> **Note**
>
> Amazon ECS pone estos permisos a disposición del agente que se ejecuta en la instancia y le envía periódicamente las credenciales temporales del rol, pero los contenedores de la tarea no pueden acceder a estos directamente. Para ver los permisos de IAM que el código de la aplicación adentro del contenedor necesita para ejecutarse, consulte Rol de IAM de tarea de Amazon ECS.

A continuación, se indican los casos de uso comunes para un rol de IAM de ejecución de tareas:

- La tarea está alojada en AWS Fargate o en una instancia externa y:
  - extrae una imagen de contenedor de un repositorio privado de Amazon ECR.
  - extrae una imagen de contenedor de un repositorio privado de Amazon ECR en una cuenta diferente de la cuenta que ejecuta la tarea.
  - envía registros de contenedor a CloudWatch Logs con el controlador de registros `awslogs`. Para obtener más información, consulte Envío de registros de Amazon ECS a CloudWatch .
- Las tareas están alojadas en AWS Fargate o en instancias de Amazon EC2 y:
  - utiliza la autenticación de registros privados. Para obtener más información, consulte permisos para la autenticación de registros privados.
  - utiliza la supervisión en tiempo de ejecución.

- la definición de la tarea hace referencia a información confidencial mediante secretos de Secrets Manager o parámetros del Almacén de parámetros de AWS Systems Manager. Para obtener más información, consulte Permisos de Secrets Manager o Systems Manager.

> **Note**
>
> El rol de ejecución de tareas es compatible con la versión 1.16.0 y posteriores del agente de contenedor de Amazon ECS.

Amazon ECS proporciona la política administrada con el nombre `AmazonECSTaskExecutionRolePolicy`, que contiene los permisos que requieren los casos de uso comunes descritos anteriormente. Para obtener más información, consulte AmazonECSTaskExecutionRolePolicy en la Guía de referencia de políticas administradas de AWS. Puede ser necesario agregar políticas en línea al rol de ejecución de tareas para los casos de uso especiales.

La consola de Amazon ECS crea un rol de ejecución de tareas. Puede asociar manualmente la política de IAM administrada para tareas a fin de permitir que Amazon ECS agregue permisos para futuras características y mejoras a medida que se vayan introduciendo. Puede utilizar la búsqueda en la consola de IAM para buscar `ecsTaskExecutionRole` y ver si la cuenta ya dispone del rol de ejecución de tareas. Para obtener más información, consulte IAM console search en la Guía del usuario de IAM.

Si extrae imágenes como usuario autenticado, es menos probable que le afecten los cambios que se hayan producido en el uso y los límites de Docker Hub. Para obtener más información, consulte Autenticación de registros privados para instancias de contenedor.

Al utilizar Amazon ECR y Amazon ECR Public, puede evitar los límites impuestos por Docker. Si extrae imágenes de Amazon ECR, esto también ayuda a acortar los tiempos de extracción de la red y reduce los cambios en la transferencia de datos cuando el tráfico abandona la VPC.

Cuando utilice Fargate, debe autenticarse en un registro de imágenes privado utilizando `repositoryCredentials`. No es posible configurar las variables de entorno del agente de contenedor de Amazon ECS `ECS_ENGINE_AUTH_TYPE` o `ECS_ENGINE_AUTH_DATA` ni modificar el archivo `ecs.config` para las tareas alojadas en Fargate. Para obtener más información, consulte Autenticación de registros privados para tareas.

Creación del rol de de ejecución de tareas

Si su cuenta aún no tiene un rol de ejecución de tareas, siga los pasos siguientes para crearlo.

AWS Management Console

Creación de un rol de servicio de Elastic Container Service (consola de IAM)

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola de IAM, seleccione Roles y, a continuación, elija Crear rol.
3. En Tipo de entidad de confianza, elija Servicio de AWS.
4. En Servicio o caso de uso, seleccione Elastic Container Service y, a continuación, seleccione el caso de uso Tarea de Elastic Container Service.
5. Elija Siguiente.
6. En la sección Agregar permisos, busque AmazonECSTaskExecutionRolePolicy y, a continuación, seleccione la política.
7. Elija Siguiente.
8. En Nombre del rol, ingrese ecsTaskExecutionRole.
9. Revise el rol y, a continuación, elija Crear rol.

AWS CLI

Sustituya cada *`entrada del usuario`* por información propia.

1. Cree un archivo con el nombre `ecs-tasks-trust-policy.json`, que contenga la política de confianza que se va a utilizar para el rol de IAM. El archivo debe contener lo siguiente:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
      },
```

```
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Cree un rol de IAM con el nombre `ecsTaskExecutionRole`, que utilice la política de confianza creada en el paso anterior.

```
aws iam create-role \
      --role-name ecsTaskExecutionRole \
      --assume-role-policy-document file://ecs-tasks-trust-policy.json
```

3. Asocie la política `AmazonECSTaskExecutionRolePolicy` administrada por AWS al rol `ecsTaskExecutionRole`.

```
aws iam attach-role-policy \
      --role-name ecsTaskExecutionRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy
```

Después de crear el rol, agregue permisos adicionales al rol para las siguientes características.

Característica	Permisos adicionales
Obtención de imágenes de contenedores de registros privados externos a AWS (como Docker Hub, Quay.io o su propio registro privado) con las credenciales de Secrets Manager	permisos para la autenticación de registros privados
Transferencia de datos confidenciales con Systems Manager o Secrets Manager	Permisos de Secrets Manager o Systems Manager
Configuración de las tareas de Fargate de modo que extraigan imágenes de Amazon ECR a través de puntos de conexión de interfaz	Las tareas de Fargate que extraen imágenes de Amazon ECR a través de permisos de puntos de conexión de interfaz
Archivos de configuración del host en un bucket de Amazon S3	Permisos de almacenamiento de archivos de Amazon S3

Característica	Permisos adicionales
Configuración de la Información de contenedor para ver los eventos del ciclo de vida de Amazon ECS	Permisos necesarios para habilitar los eventos del ciclo de vida de Amazon ECS en Información de contenedores
Visualización de los eventos del ciclo de vida de Amazon ECS en la información de contenedor	Permisos necesarios para ver los eventos del ciclo de vida de Amazon ECS en Información de contenedores

permisos para la autenticación de registros privados

La autenticación de registro privado permite que sus tareas de Amazon ECS obtengan imágenes de contenedores de registros privados externos a AWS (como Docker Hub, Quay.io, o su propio registro privado) que requieren credenciales de autenticación. Esta característica utiliza Secrets Manager para almacenar de forma segura las credenciales de registro, que luego se referencian en la definición de la tarea mediante el parámetro `repositoryCredentials`.

Para obtener más información sobre la configuración de la autenticación de registro privado, consulte Uso de imágenes de contenedor no AWS en Amazon ECS.

Para proporcionar acceso a los secretos que contienen las credenciales de su registro privado, agregue los siguientes permisos como una política insertada al rol de ejecución de tareas. Para obtener más información, consulte Adición y eliminación de políticas de IAM.

- `secretsmanager:GetSecretValue`: obligatorio para recuperar las credenciales de registro privado de Secrets Manager.
- `kms:Decrypt`: obligatorio solo si el secreto utiliza una clave de KMS personalizada y no la clave predeterminada. Se debe agregar el nombre de recurso de Amazon (ARN) de la clave de personalizada como un recurso.

Es siguiente es un ejemplo de política insertada que agrega los permisos.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "kms:Decrypt",
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:secret_name",
                "arn:aws:kms:<region>:<aws_account_id>:key/key_id"
            ]
        }
    ]
}
```

Permisos de Secrets Manager o Systems Manager

El permiso para dejar que el agente de contenedor extraiga los recursos necesarios de AWS Systems Manager o Secrets Manager. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.

Utilización de Secrets Manager

Para proporcionar acceso a los secretos de Secrets Manager que cree, agregue manualmente el siguiente permiso al rol de ejecución de tareas. Para obtener información sobre cómo administrar los permisos, consulte Adición y eliminación de permisos de identidad de IAM en la Guía del usuario de IAM.

- `secretsmanager:GetSecretValue`: obligatorio si se hace referencia a un secreto de Secrets Manager. Agrega el permiso para recuperar el secreto de Secrets Manager.

La siguiente política de ejemplo agrega los permisos necesarios.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
      ]
    }
```

```
  ]
}
```

Utilización de Systems Manager

> **Important**
>
> En el caso de las tareas que utilizan el tipo de lanzamiento de EC2, debe utilizar la variable de configuración del agente de ECS `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` para utilizar esta característica. Puede añadirla al archivo `./etc/ecs/ecs.config` durante la creación de la instancia de contenedor o puede añadirla a una instancia existente y, a continuación, reiniciar el agente de ECS. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS.

Para proporcionar acceso a los parámetros del almacén de parámetros de Systems Manager que cree, agregue manualmente los siguientes permisos como política al rol de ejecución de tareas. Para obtener información sobre cómo administrar los permisos, consulte Adición y eliminación de permisos de identidad de IAM en la Guía del usuario de IAM.

- `ssm:GetParameters`: obligatorio si se hace referencia a un parámetro del almacén de parámetros de Systems Manager en una definición de tareas. Agrega el permiso para recuperar los parámetros de Systems Manager.
- `secretsmanager:GetSecretValue`: obligatorio si se hace referencia a un secreto de Secrets Manager directamente o si el parámetro del almacén de parámetros de Systems Manager hace referencia a un secreto de Secrets Manager en una definición de tareas. Agrega el permiso para recuperar el secreto de Secrets Manager.
- `kms:Decrypt`: obligatorio solo si el secreto utiliza una clave administrada por el cliente y no la clave predeterminada. El ARN de su clave personalizada debe añadirse como un recurso. Agrega el permiso para descifrar la clave administrada por el cliente.

La siguiente política de ejemplo agrega los permisos necesarios:

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

```
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetParameters",
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:ssm:region:aws_account_id:parameter/parameter_name",
        "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name",
        "arn:aws:kms:region:aws_account_id:key/key_id"
      ]
    }
  ]
}
```

Las tareas de Fargate que extraen imágenes de Amazon ECR a través de permisos de puntos de conexión de interfaz

Al lanzar las tareas que usan el tipo de lanzamiento de Fargate que extrae imágenes de Amazon ECR cuando Amazon ECR está configurado para utilizar un punto de enlace de la VPC de interfaz, puede restringir el acceso de las tareas a una VPC o a un punto de enlace de la VPC específicos. Para hacerlo, cree un rol de ejecución de tareas para que las tareas utilicen claves de condición de IAM.

Utilice las siguientes claves de condición globales de IAM para restringir el acceso a una VPC o a un punto de enlace de la VPC específicos. Para obtener más información, consulte Claves de contexto de condición globales de AWS.

- `aws:SourceVpc`: restringe el acceso a una VPC específica. Puede restringir la VPC a la VPC que aloja la tarea y el punto de conexión.
- `aws:SourceVpce`: restringe el acceso a un punto de enlace de la VPC específico.

En la siguiente política del rol de ejecución de tareas se ofrece un ejemplo para añadir claves de condición:

JSON

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

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:GetAuthorizationToken",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "arn:aws:ec2:arn:aws::111122223333:vpc-
endpoint/vpce-0123456789abcdef0",
                    "arn:aws:ec2:arn:aws::111122223333:vpc/vpc-0123456789abcdef0"
                }
            }
        }
    ]
}
```

Permisos de Amazon ECR

Los siguientes permisos son necesarios cuando se necesita extraer imágenes de contenedores de repositorios privados de Amazon ECR. El rol de ejecución de tareas debería tener estos permisos para permitir que los agentes de contenedores de Amazon ECS y Fargate extraigan imágenes de contenedores en su nombre. Para las implementaciones básicas de ECS, estos permisos deben agregarse al rol de ejecución de tareas en lugar de al rol de IAM de la tarea.

La política administrada del rol de ejecución de tareas de Amazon ECS (`AmazonECSTaskExecutionRolePolicy`) incluye los permisos necesarios para extraer imágenes de Amazon ECR. Si está utilizando la política administrada, no necesita agregar estos permisos por separado.

Si va a crear una política personalizada, incluya los siguientes permisos para poder extraer imágenes de Amazon ECR:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Tenga en cuenta que estos permisos son diferentes de los permisos que podrían ser necesarios en el rol de IAM de la tarea si el código de su aplicación necesita interactuar directamente con las API de Amazon ECR. Para obtener más información sobre los permisos del rol de IAM de la tarea para Amazon ECR, consulte Permisos de Amazon ECR.

Permisos de almacenamiento de archivos de Amazon S3

Cuando especifica un archivo de configuración alojado en Amazon S3, el rol de ejecución de tareas debe incluir el permiso `s3:GetObject` para el archivo de configuración y el permiso `s3:GetBucketLocation` en el bucket de Amazon S3 en el que se encuentra el archivo. Para obtener más información, consulte Acciones de políticas para Amazon S3 en la Guía del usuario de Amazon Simple Storage Service.

La siguiente política de ejemplo agrega los permisos necesarios para recuperar un archivo de Amazon S3. Especifique el nombre del bucket de Amazon S3 y el nombre del archivo de configuración.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/folder_name/config_file_name"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket"
      ]
    }
  ]
}
```

Rol de IAM de tarea de Amazon ECS

Sus tareas de Amazon ECS pueden tener un rol de IAM asociado a ellas. Los permisos concedidos en el rol de IAM se ofrecen a los contenedores que se ejecutan en la tarea. Este rol permite que el código de la aplicación (que se ejecuta en el contenedor) utilice otros servicios de AWS. El rol de tarea es obligatorio cuando la aplicación accede a otros servicios de AWS, como Amazon S3.

> **Note**
>
> Ni el contenedor de Amazon ECS ni los agentes de Fargate acceden a estos permisos. Para conocer los permisos de IAM que Amazon ECS necesita para extraer imágenes de contenedores y ejecutar la tarea, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

A continuación, se describen las ventajas de usar roles de tarea:

- Separación de preocupaciones: si utiliza el tipo de lanzamiento EC2, los roles de IAM de tareas le permiten especificar los permisos de IAM para sus contenedores sin necesidad de especificar estos permisos mediante perfiles de instancia de EC2 (para obtener más información, consulte Uso de perfiles de instancia en la Guía del usuario de AWS Identity and Access Management). Por lo tanto, puede implementar sus aplicaciones de forma independiente y uniforme en las instancias de contenedor de ECS sin necesidad de modificar los permisos de IAM asociados a las instancias de EC2.
- Auditabilidad: el acceso y el registro de eventos se encuentran disponibles a través de CloudTrail con el fin de garantizar una auditoría retrospectiva. Las credenciales de tareas tienen un contexto de '`taskArn`' que se asocia a la sesión, de modo que los registros de CloudTrail muestran la tarea para la que se ofrecieron las credenciales de rol.
- Entrega uniforme de credenciales: ECS proporciona las credenciales de los roles de IAM a sus contenedores y las hace accesibles mediante una interfaz bien definida, independientemente del tipo de lanzamiento asociado a sus tareas. En ECS Fargate, los perfiles de instancias de EC2 no están disponibles para los contenedores de sus tareas. Los roles de IAM de tareas le permiten asociar permisos de IAM a sus contenedores, independientemente del tipo de lanzamiento, cuando utiliza el SDK de AWS o la CLI de AWS en sus contenedores. Para obtener más información sobre cómo el SDK de AWS accede a estas credenciales, consulte Proveedor de credenciales de contenedor.

> **Important**
>
> Los contenedores no son un límite de seguridad y el uso de roles de IAM en las tareas no cambia esta situación. Cada tarea que se ejecuta en Fargate tiene su propio límite de aislamiento y no comparte el kernel subyacente, los recursos de CPU, los recursos de memoria ni la interfaz de red elástica con ninguna otra tarea. En el caso de las instancias de contenedores externos y de EC2 en ECS, no hay aislamiento de tareas (a diferencia de Fargate) y los contenedores podrían acceder a las credenciales de otras tareas de la misma instancia de contenedor. También pueden acceder a los permisos asignados a la función de instancia de contenedor de ECS. Siga las recomendaciones en Recomendaciones de roles para bloquear el acceso al servicio de metadatos de instancias de Amazon EC2 para contenedores (para obtener más información, consulte Uso del servicio de metadatos de instancias para acceder a los metadatos de la instancia en la Guía del usuario de Amazon EC2).

Cuando se especifica un rol de IAM para una tarea, la CLI de AWS u otros SDK de los contenedores de esa tarea utilizan las credenciales de AWS que proporciona el rol de tarea exclusivamente, y ya no heredan ningún permiso de IAM de Amazon EC2 o de la instancia externa en la que se ejecutan.

Creación del rol de IAM de tareas

Al crear una política de IAM para que utilicen sus tareas, la política debe incluir los permisos que desea que asuman los contenedores de sus tareas. Puede utilizar una política administrada de AWS existente o puede crear una política personalizada desde cero que satisfaga sus necesidades específicas. Para obtener más información, consulte Creación de políticas de IAM en la Guía del usuario de IAM.

> **Important**
>
> Para las tareas de Amazon ECS (para todos los tipos de lanzamientos), le recomendamos que utilice el rol y la política de IAM para sus tareas. Estas credenciales permiten a su tarea llevar a cabo peticiones a la API de AWS sin necesidad de llamar a `sts:AssumeRole` para asumir el mismo rol que ya está asociado a la tarea. Si su tarea requiere que un rol se asuma a sí mismo, debe crear una política de confianza que permita explícitamente que ese rol se asuma a sí mismo. Para obtener más información, consulte Actualización de una política de confianza de rol en la Guía del usuario de IAM.

Una vez creada la política de IAM, puede crear un rol de IAM que incluya esa política a la que hace referencia en la definición de la tarea de Amazon ECS. Puede crear el rol usando el caso de uso Elastic Container Service Task (Tarea de Elastic Container Service) en la consola de IAM. A continuación, puede adjuntar la política de IAM específica al rol que proporciona a los contenedores de la tarea los permisos que desea. Los procedimientos siguientes describen cómo hacerlo.

Si tiene varias definiciones de tareas o servicios que requieran permisos de IAM, debería plantearse la creación de un rol para cada definición de tarea específica o servicio con los permisos mínimos requeridos para que las tareas operen, a fin de poder minimizar el acceso que proporciona para cada tarea.

Para obtener información acerca del punto de conexión de servicio para su región, consulte Service endpoints en la Guía de Referencia general de Amazon Web Services.

El rol de tarea de IAM debe tener una política de confianza que especifique el servicio `ecs-tasks.amazonaws.com`. El permite `sts:AssumeRole` permite que las tareas asuman el rol de IAM diferente de la que utiliza la instancia de Amazon EC2. De esta forma, la tarea no hereda el rol asociado con la instancia de Amazon EC2. A continuación, se muestra un ejemplo de una política de confianza. Sustituya el identificador de región y especifique el número de cuenta de AWS que utiliza al lanzar tareas.

> **Important**
>
> Al crear el rol de IAM de su tarea, se recomienda utilizar las claves de condición `aws:SourceAccount` o `aws:SourceArn` en la relación de confianza asociada al rol para definir aún más el ámbito del permiso y evitar el problema de seguridad del suplente confuso. El uso de la clave de condición `aws:SourceArn` para especificar un clúster concreto no se admite actualmente, debe utilizar el comodín para especificar todos los clústeres. Para obtener más información sobre el problema del suplente confuso y cómo proteger su cuenta de AWS, consulte El problema del suplente confuso en la Guía del usuario de IAM.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":[
               "ecs-tasks.amazonaws.com"
            ]
         },
         "Action":"sts:AssumeRole",
         "Condition":{
            "ArnLike":{
            "aws:SourceArn":"arn:aws:ecs:us-west-2:111122223333:*"
            },
            "StringEquals":{
               "aws:SourceAccount":"111122223333"
            }
         }
      }
```

```
   ]
}
```

Utilice el siguiente procedimiento que creará una política para recuperar objetos de Amazon S3 con un ejemplo de política. Sustituya cada *entrada del usuario* por valores propios.

AWS Management Console

Utilización del editor de política de JSON para la creación de una política

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, elija Políticas.

   Si es la primera vez que elige Políticas, aparecerá la página Welcome to Managed Policies (Bienvenido a políticas administradas). Elija Comenzar.
3. En la parte superior de la página, seleccione Crear política.
4. En la sección Editor de políticas, seleccione la opción JSON.
5. Ingrese el siguiente documento de política JSON:

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject"
         ],
         "Resource":[
            "arn:aws:s3:::my-task-secrets-bucket/*"
         ]
      }
   ]
}
```

6. Elija Siguiente.

> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de política en la Guía del usuario de IAM.

7. En la página Revisar y crear, introduzca el Nombre de la política y la Descripción (opcional) para la política que está creando. Revise los Permisos definidos en esta política para ver los permisos que concede la política.
8. Elija Crear política para guardar la nueva política.

AWS CLI

Sustituya cada *entrada del usuario* por valores propios.

1. Cree un archivo denominado `s3-policy.json` con el siguiente contenido.

   JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject"
         ],
         "Resource":[
            "arn:aws:s3:::my-task-secrets-bucket/*"
         ]
      }
   ]
}
```

2. Utilice el siguiente comando de la para crear la política de IAM con el archivo de documento de política de JSON. Sustituya cada *entrada del usuario* por valores propios.

```
aws iam create-policy \
      --policy-name taskRolePolicy \
      --policy-document file://s3-policy.json
```

Utilice el siguiente procedimiento para crear el rol de servicio.

AWS Management Console

Creación de un rol de servicio de Elastic Container Service (consola de IAM)

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola de IAM, seleccione Roles y, a continuación, elija Crear rol.
3. En Tipo de entidad de confianza, elija Servicio de AWS.
4. En Servicio o caso de uso, seleccione Elastic Container Service y, a continuación, seleccione el caso de uso Tarea de Elastic Container Service.
5. Elija Siguiente.
6. En Agregar permisos, busque y seleccione la política que haya creado.
7. Elija Siguiente.
8. Escriba un nombre para el rol en Nombre de rol. En este ejemplo, escriba `AmazonECSTaskS3BucketRole` para nombrar el rol.
9. Revise el rol y, a continuación, elija Crear rol.

AWS CLI

1. Cree un archivo con el nombre `ecs-tasks-trust-policy.json` que contenga la política de confianza que se va a utilizar para el rol de IAM de la tarea. El archivo debe contener lo siguiente. Sustituya el identificador de región y especifique el número de cuenta de AWS que utiliza al lanzar tareas.

   JSON

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

```
    "Statement":[
       {
          "Effect":"Allow",
          "Principal":{
             "Service":[
                "ecs-tasks.amazonaws.com"
             ]
          },
          "Action":"sts:AssumeRole",
          "Condition":{
             "ArnLike":{
             "aws:SourceArn":"arn:aws:ecs:us-west-2:111122223333:*"
             },
             "StringEquals":{
                "aws:SourceAccount":"111122223333"
             }
          }
       }
    ]
 }
```

2. Cree un rol de IAM con el nombre `ecsTaskRole`, que utilice la política de confianza creada en el paso anterior.

```
aws iam create-role \
      --role-name ecsTaskRole \
      --assume-role-policy-document file://ecs-tasks-trust-policy.json
```

3. Recupere el ARN de la política de IAM que creó mediante el siguiente comando. Sustituya *`taskRolePolicy`* por el nombre de la política que haya creado.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`taskRolePolicy`].Arn'
```

4. Adjunte la política de IAM que haya creado al rol `ecsTaskRole`. Sustituya el valor de `policy-arn` por el ARN de la política que haya creado.

```
aws iam attach-role-policy \
      --role-name ecsTaskRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/taskRolePolicy
```

Después de crear el rol, agregue permisos adicionales al rol para las siguientes características.

Característica	Permisos adicionales
Uso de ECS Exec	Permisos de ECS Exec
Usar una imagen de un repositorio privado de Amazon ECR	Permisos de Amazon ECR
Uso de instancias de EC2 (Windows y Linux)	Configuración adicional de instancias de Amazon EC2
Uso de instancias externas	Configuración adicional de las instancias externas
Uso de instancias de EC2 de Windows	Configuración adicional de las instancias de Amazon EC2 de Windows

Permisos de Amazon ECR

Los siguientes permisos son necesarios cuando el código de su aplicación necesita interactuar directamente con los repositorios de Amazon ECR. Tenga en cuenta que, para la implementación básica, en la que solo necesita extraer imágenes de Amazon ECR, estos permisos no son necesarios en el nivel del rol de IAM de la tarea. En su lugar, el rol de ejecución de tareas de Amazon ECS debería tener estos permisos. Para obtener más información sobre el rol de ejecución de tareas, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

Si el código de su aplicación que se ejecuta en el contenedor necesita interactuar directamente con las API de Amazon ECR, debe agregar los siguientes permisos a un rol de IAM de la tarea e incluir el rol de IAM de la tarea en su definición de tarea. Para obtener más información, consulte Adición y eliminación de políticas de IAM en la Guía del usuario de IAM.

Utilice la siguiente política para su rol de IAM de la tarea para agregar los permisos de Amazon ECR necesarios para las aplicaciones de contenedor que necesitan interactuar directamente con Amazon ECR:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Permisos de ECS Exec

La característica ECS Exec requiere un rol de IAM para tareas para conceder a los contenedores los permisos necesarios para la comunicación entre el SSM Agent administrado (agente `execute-command`) y el servicio de SSM. Debe agregar los siguientes permisos a un rol de IAM para tareas e incluir el rol de IAM para tareas en la definición de tareas. Para obtener más información, consulte Adición y eliminación de políticas de IAM en la Guía del usuario de IAM.

Utilice la siguiente política para el rol de IAM para tareas a fin de agregar los permisos SSM requeridos.

JSON

```
{
   "Version": "2012-10-17",
   "Statement": [
       {
       "Effect": "Allow",
       "Action": [
            "ssmmessages:CreateControlChannel",
            "ssmmessages:CreateDataChannel",
            "ssmmessages:OpenControlChannel",
            "ssmmessages:OpenDataChannel"
```

```
        ],
       "Resource": "*"
       }
    ]
}
```

Configuración adicional de instancias de Amazon EC2

Recomendamos que limite los permisos en el rol de instancia de contenedor a la lista mínima de permisos que se proporciona en la política de IAM administrada `AmazonEC2ContainerServiceforEC2Role`.

Las instancias de Amazon EC2 requieren al menos la versión `1.11.0` del agente de contenedor para utilizar los roles para tareas; sin embargo, recomendamos utilizar la versión más reciente. Para obtener información sobre la comprobación de la versión del agente y la actualización a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS. Si utiliza una AMI optimizada para Amazon ECS, la instancia también necesita al menos `1.11.0-1` del paquete `ecs-init`. Si las instancias utilizan la versión de AMI optimizada para Amazon ECS más reciente, entonces contienen las versiones requeridas del agente de contenedor y `ecs-init`. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.

Si no está utilizando la AMI optimizada para Amazon ECS para las instancias de contenedor, agregue la opción `--net=host` al comando docker run que inicia el agente, junto con las siguientes variables de configuración de agente para la configuración deseada (para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS):

`ECS_ENABLE_TASK_IAM_ROLE=true`

Usa roles de IAM para las tareas de contenedores con los modos de red `bridge` y `default`.

`ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true`

Usa roles de IAM para las tareas de contenedores con el modo de red `host`. Esta variable solo se admite en versiones de agente 1.12.0 y posterior.

Para ver un ejemplo de un comando de ejecución, consulte Actualización manual del agente de contenedor de Amazon ECS (para AMI no optimizadas para Amazon ECS). También deberá establecer los siguientes comandos de redes en su instancia de contenedor para que los contenedores de sus tareas puedan recuperar las credenciales de AWS:

```
sudo sysctl -w net.ipv4.conf.all.route_localnet=1
sudo iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
sudo iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Debe guardar estas reglas de iptables en su instancia de contenedor para que no se vean afectadas por un reinicio. Puede utilizar los comandos iptables-save e iptables-restore para guardar las reglas de iptables y restaurarlas en el arranque. Para obtener más información, consulte la documentación del sistema operativo específico.

Para impedir que los contenedores de las tareas que usan el modo de red `awsvpc` obtengan acceso a la información sobre credenciales proporcionada al perfil de instancia de Amazon EC2, pero permitiendo los permisos que concede el rol de tarea, establezca la variable de configuración del agente `ECS_AWSVPC_BLOCK_IMDS` en `true` en el archivo de configuración del agente y reinicie este agente. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS.

Para impedir que los contenedores de las tareas que utilizan el modo de red `bridge` obtengan acceso a la información sobre credenciales proporcionada al perfil de instancia de Amazon EC2, pero permitiendo los permisos que concede el rol de tarea, ejecute el siguiente comando iptables en las instancias de Amazon EC2. Este comando no afecta a los contenedores de las tareas que usan los modos de red `host` o `awsvpc`. Para obtener más información, consulte Modo de red.

- ```
  sudo yum install -y iptables-services; sudo iptables --insert DOCKER-USER 1 --in-
  interface docker+ --destination 169.254.169.254/32 --jump DROP
  ```

  Debe guardar esta regla de iptables en la instancia de Amazon EC2 para que se conserve tras un reinicio. Puede usar los siguientes comandos para la AMI optimizada para Amazon ECS. Para otros sistemas operativos, consulte la documentación correspondiente a dicho sistema operativo.

  ```
  sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now
   iptables
  ```

Configuración adicional de las instancias externas

Las instancias externas requieren al menos la versión `1.11.0` del agente de contenedor para utilizar los roles de IAM para tareas; sin embargo, recomendamos utilizar la versión más reciente. Para
  ```

obtener información sobre la comprobación de la versión del agente y la actualización a la versión más reciente, consulte Actualización del agente de contenedor de Amazon ECS. Si utiliza una AMI optimizada para Amazon ECS, la instancia también necesita al menos `1.11.0-1` del paquete `ecs-init`. Si las instancias utilizan la versión de AMI optimizada para Amazon ECS más reciente, entonces contienen las versiones requeridas del agente de contenedor y `ecs-init`. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.

Si no está utilizando la AMI optimizada para Amazon ECS para las instancias de contenedor, agregue la opción `--net=host` al comando docker run que inicia el agente, junto con las siguientes variables de configuración de agente para la configuración deseada (para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS):

`ECS_ENABLE_TASK_IAM_ROLE=true`

Usa roles de IAM para las tareas de contenedores con los modos de red `bridge` y `default`.

`ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true`

Usa roles de IAM para las tareas de contenedores con el modo de red `host`. Esta variable solo se admite en versiones de agente 1.12.0 y posterior.

Para ver un ejemplo de un comando de ejecución, consulte Actualización manual del agente de contenedor de Amazon ECS (para AMI no optimizadas para Amazon ECS). También deberá establecer los siguientes comandos de redes en su instancia de contenedor para que los contenedores de sus tareas puedan recuperar las credenciales de AWS:

```
sudo sysctl -w net.ipv4.conf.all.route_localnet=1
sudo iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
sudo iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Debe guardar estas reglas de iptables en su instancia de contenedor para que no se vean afectadas por un reinicio. Puede utilizar los comandos iptables-save e iptables-restore para guardar las reglas de iptables y restaurarlas en el arranque. Para obtener más información, consulte la documentación del sistema operativo específico.

Configuración adicional de las instancias de Amazon EC2 de Windows

> **Important**
>
> Esto se aplica únicamente a los contenedores de Windows en EC2 que utilizan roles de tareas.

El rol de tareas con características de Windows requiere una configuración adicional en EC2.

- Cuando se lanzan las instancias de contenedor, se debe establecer la opción `-EnableTaskIAMRole` en el script de datos de usuario de las instancias de contenedor. El rol `EnableTaskIAMRole` activa la característica de roles de IAM de tareas para las tareas. Por ejemplo:

```
<powershell>
Import-Module ECSTools
Initialize-ECSAgent -Cluster 'windows' -EnableTaskIAMRole
</powershell>
```

- Debe arrancar el contenedor con los comandos de redes que se proporcionan en Script de arranque del contenedor de Amazon ECS.
- Debe crear un rol y una política de IAM para las tareas. Para obtener más información, consulte Creación del rol de IAM de tareas.
- Los roles de IAM para el proveedor de credenciales de tareas utilizan el puerto 80 en la instancia de contenedor. Por lo tanto, si configura roles de IAM para las tareas en la instancia de contenedor, los contenedores no pueden utilizar el puerto 80 como puerto del host en ningún mapeo de puertos. Para exponer los contenedores en el puerto 80, recomendamos configurar un servicio para ellos que utilice el balanceo de carga. Puede utilizar el puerto 80 en el balanceador de carga. De este modo, el tráfico se puede dirigir a otro puerto del host en las instancias de contenedor. Para obtener más información, consulte Uso del equilibrador de carga para distribuir el tráfico de servicio de Amazon ECS.
- Si se reinicia la instancia de Windows, debe eliminar la interfaz de proxy y volver a inicializar el agente de contenedor de Amazon ECS para que se vuelva a activar el proxy de credenciales.

Script de arranque del contenedor de Amazon ECS

Antes de que los contenedores puedan obtener acceso al proxy de credenciales en la instancia de contenedor para obtener credenciales, el contenedor se debe arrancar con los comandos de redes requeridos. El siguiente script de ejemplo de código se debe ejecutar en los contenedores cuando se inician.

> **Note**
>
> No es necesario ejecutar este script cuando se utiliza el modo de red `awsvpc` en Windows.

Si ejecuta contenedores de Windows que incluyen Powershell, utilice el siguiente script:

```
# Copyright Amazon.com Inc. or its affiliates. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License"). You may
# not use this file except in compliance with the License. A copy of the
# License is located at
#
# http://aws.amazon.com/apache2.0/
#
# or in the "license" file accompanying this file. This file is distributed
# on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
# express or implied. See the License for the specific language governing
# permissions and limitations under the License.

$gateway = (Get-NetRoute | Where { $_.DestinationPrefix -eq '0.0.0.0/0' } | Sort-Object
 RouteMetric | Select NextHop).NextHop
$ifIndex = (Get-NetAdapter -InterfaceDescription "Hyper-V Virtual Ethernet*" | Sort-
Object | Select ifIndex).ifIndex
New-NetRoute -DestinationPrefix 169.254.170.2/32 -InterfaceIndex $ifIndex -NextHop
 $gateway -PolicyStore ActiveStore # credentials API
New-NetRoute -DestinationPrefix 169.254.169.254/32 -InterfaceIndex $ifIndex -NextHop
 $gateway -PolicyStore ActiveStore # metadata API
```

Si ejecuta contenedores de Windows que solo tienen el shell de comandos, utilice el siguiente script:

```
# Copyright Amazon.com Inc. or its affiliates. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License"). You may
```

```
# not use this file except in compliance with the License. A copy of the
# License is located at
#
# http://aws.amazon.com/apache2.0/
#
# or in the "license" file accompanying this file. This file is distributed
# on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
# express or implied. See the License for the specific language governing
# permissions and limitations under the License.

for /f "tokens=1" %i  in ('netsh interface ipv4 show interfaces ^| findstr /x /r
 ".*vEthernet.*"') do set interface=%i
for /f "tokens=3" %i  in ('netsh interface ipv4 show addresses %interface% ^| findstr /
x /r ".*Default.Gateway.*"') do set gateway=%i
netsh interface ipv4 add route prefix=169.254.170.2/32 interface="%interface%"
 nexthop="%gateway%" store=active # credentials API
netsh interface ipv4 add route prefix=169.254.169.254/32 interface="%interface%"
 nexthop="%gateway%" store=active # metadata API
```

Rol de IAM de instancia de contenedor de Amazon ECS

Las instancias de contenedor de Amazon ECS, incluidas las instancias de Amazon EC2 y las externas, ejecutan el agente de contenedor de Amazon ECS y requieren un rol de IAM para que el servicio sepa que el agente le pertenece. Antes de lanzar instancias de contenedor y registrarlas en un clúster, debe crear un rol de IAM para que lo utilicen las instancias de contenedor. El rol se crea en la cuenta que utiliza para iniciar sesión en la consola o ejecutar los comandos AWS CLI.

> **Important**
>
> Si está registrando instancias externas en el clúster, el rol de IAM que utiliza también requiere permisos de Systems Manager. Para obtener más información, consulte Rol de IAM de Amazon ECS Anywhere.

Amazon ECS proporciona la política de IAM administrada `AmazonEC2ContainerServiceforEC2Role`, que contiene los permisos necesarios para utilizar el conjunto completo de características de Amazon ECS. Esta política administrada se puede asociar a un rol de IAM y a las instancias de contenedor. También puede utilizar la política administrada como guía cuando crea la política personalizada que va a utilizar. El rol de instancia de contenedor proporciona los permisos necesarios para que el agente de contenedor de Amazon ECS y el daemon

de Docker puedan llamar a las API de AWS en su nombre. Para obtener más información acerca de la política administrada, consulte AmazonEC2ContainerServiceforEC2Role.

Creación del rol de instancia de contenedor

> **Important**
>
> Si está registrando instancias externas en el clúster, consulte Rol de IAM de Amazon ECS Anywhere.

Puede crear el rol manualmente y asociarlo a la política de IAM administrada para instancias de contenedor a fin de permitir que Amazon ECS agregue permisos para futuras características y mejoras a medida que se vayan introduciendo. Utilice el siguiente procedimiento para adjuntar la política de IAM administrada si es necesario.

AWS Management Console

Creación de un rol de servicio de Elastic Container Service (consola de IAM)

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola de IAM, seleccione Roles y, a continuación, elija Crear rol.
3. En Tipo de entidad de confianza, elija Servicio de AWS.
4. En Servicio o caso de uso, seleccione Elastic Container Service y, a continuación, seleccione el caso de uso Rol de EC2 para Elastic Container Service.
5. Elija Siguiente.
6. En la sección Políticas de permisos, compruebe que la política AmazonEC2ContainerServiceforEC2Role esté seleccionada.

   > **Important**
   >
   > La política administradaAmazonEC2ContainerServiceforEC2Role debe estar asociada al rol de IAM de instancia de contenedor; de lo contrario, recibirá un error al utilizar la AWS Management Console para crear clústeres.

7. Elija Siguiente.

8. En Nombre del rol, ingrese ecsInstanceRole
9. Revise el rol y, a continuación, elija Crear rol.

AWS CLI

Sustituya cada *entrada del usuario* por valores propios.

1. Cree un archivo denominado `instance-role-trust-policy.json` con el siguiente contenido.

   JSON

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": { "Service": "ec2.amazonaws.com"},
         "Action": "sts:AssumeRole"
       }
     ]
   }
   ```

2. Utilice el siguiente comando para crear el rol de IAM de la instancia con el documento de política de confianza.

   ```
   aws iam create-role \
       --role-name ecsInstanceRole \
       --assume-role-policy-document file://instance-role-trust-policy.json
   ```

3. Cree un perfil de instancia denominado `ecsInstanceRole-profile` mediante el comando create-instance-profile.

   ```
   aws iam create-instance-profile --instance-profile-name ecsInstanceRole-profile
   ```

   Ejemplo de respuesta

   ```
   {
       "InstanceProfile": {
           "InstanceProfileId": "AIPAJTLBPJLEGREXAMPLE",
   ```

```
        "Roles": [],
        "CreateDate": "2022-04-12T23:53:34.093Z",
        "InstanceProfileName": "ecsInstanceRole-profile",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/ecsInstanceRole-
profile"
    }
}
```

4. Añada el rol *ecsInstanceRole* al perfil de instancia *ecsInstanceRole-profile*.

```
aws iam add-role-to-instance-profile \
    --instance-profile-name ecsInstanceRole-profile \
    --role-name ecsInstanceRole
```

5. Adjunte la política administrada `AmazonEC2ContainerServiceRoleForEC2Role` al rol con el siguiente comando.

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role \
    --role-name ecsInstanceRole
```

Después de crear el rol, agregue permisos adicionales al rol para las siguientes características.

Característica	Permisos adicionales
Amazon ECR tiene la imagen del contenedor	Permisos de Amazon ECR
Configuración de Registros de CloudWatch de modo que supervise las instancias de contenedores	Supervisión de los permisos de instancias de contenedores
Archivos de configuración del host en un bucket de Amazon S3	Acceso de solo lectura a Amazon S3

Permisos de Amazon ECR

El rol de la instancia de contenedor de Amazon ECS que utiliza con sus instancias de contenedor debe tener los siguientes permisos de política de IAM para Amazon ECR.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Si usa la política administrada `AmazonEC2ContainerServiceforEC2Role` para sus instancias de contenedor, el rol debe tener los permisos adecuados. Para comprobar si el rol es compatible con Amazon ECR, consulte Rol de IAM de instancia de contenedor de Amazon ECS en la Guía para desarrolladores de Amazon Elastic Container Service.

Permisos necesarios para establecer la configuración de cuenta awsvpcTrunking

Amazon ECS es compatible con el lanzamiento de instancias de contenedor con mayor densidad de ENI al usar tipos de instancias de Amazon EC2 compatibles. Al utilizar esta característica, le recomendamos crear dos roles de instancia de contenedor. Habilite la configuración de la cuenta de `awsvpcTrunking` para un rol y use ese rol para las tareas que requieran el enlace troncal de ENI. Para obtener información acerca de la configuración de la cuenta de `awsvpcTrunking`, consulte Acceso a las características de Amazon ECS con la configuración de la cuenta.

El rol de la instancia de contenedor que utiliza con sus instancias de contenedor debe tener los siguientes permisos de política de IAM para establecer la configuración de la cuenta.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:ListAccountSettings",
                "ecs:ListAttributes",
                "ecs:PutAccountSetting"
            ],
            "Resource": "*"
        }
    ]
}
```

Para utilizar el rol de instancia de contenedor, agregue lo siguiente a los datos de usuario de la instancia:

```
#!/bin/bash
aws ecs put-account-setting --name awsvpcTrunking --value enabled --region region
ECS_CLUSTER=MyCluster>> /etc/ecs/ecs.config
EOF
```

Para obtener más información acerca de cómo agregar datos de usuario a sus instancias de EC2, consulte Ejecutar comandos en la instancia de Linux durante la inicialización en la Guía del usuario de Amazon EC2.

Acceso de solo lectura a Amazon S3

El almacenamiento de la información de configuración en un bucket privado en Amazon S3 y la concesión de acceso de solo lectura al rol de IAM de instancia de contenedor es una forma práctica y segura de permitir la configuración de instancia de contenedor en el momento del lanzamiento. Puede almacenar una copia del archivo `ecs.config` en un bucket privado, utilizar los datos de usuario de Amazon EC2 para instalar la AWS CLI y, a continuación, copiar la información de configuración en `/etc/ecs/ecs.config` cuando se lance la instancia.

Para obtener más información acerca de cómo crear un archivo `ecs.config`, almacenarlo en Amazon S3 y lanzar instancias con esta configuración, consulte Almacenamiento de la configuración de instancia de contenedor de Amazon ECS en Amazon S3.

Puede utilizar el siguiente comando de la AWS CLI para conceder a Amazon S3 acceso de solo lectura para su rol de la instancia de contenedor. Sustituya *`ecsInstanceRole`* por el nombre del rol que haya creado.

```
aws iam attach-role-policy \
      --role-name ecsInstanceRole \
      --policy-arn arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess
```

También puede utilizar la consola de IAM para agregar el acceso de solo lectura de Amazon S3 (`AmazonS3ReadOnlyAccess`) a su rol. Para obtener más información, consulte Updating permissions for a role en la Guía del usuario de AWS Identity and Access Management.

Permisos necesarios para establecer la configuración de cuenta awsvpcTrunking

El rol de la instancia de contenedor de Amazon ECS que utiliza con las instancias de contenedor debe tener los siguientes permisos de la política de IAM para establecer la configuración de la cuenta

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:ListAccountSettings",
                "ecs:ListAttributes",
                "ecs:PutAccountSetting"
            ],
            "Resource": "*"
        }
    ]
}
```

Supervisión de los permisos de instancias de contenedores

Para que las instancias de contenedor puedan enviar datos de registros a Registros de CloudWatch, debe crear una política de IAM que permita al agente de Amazon ECS escribir los registros de aplicación del cliente en CloudWatch (normalmente gestionado a través del controlador `awslogs`). Después de crear la política, adjúntela a `ecsInstanceRole`.

AWS Management Console

Utilización del editor de política de JSON para la creación de una política

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, elija Políticas.

   Si es la primera vez que elige Políticas, aparecerá la página Welcome to Managed Policies (Bienvenido a políticas administradas). Elija Comenzar.
3. En la parte superior de la página, seleccione Crear política.
4. En la sección Editor de políticas, seleccione la opción JSON.
5. Ingrese el siguiente documento de política JSON:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": ["arn:aws:logs:*:*:*"]
        }
    ]
}
```

6. Elija Siguiente.

> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de política en la Guía del usuario de IAM.

7. En la página Revisar y crear, introduzca el Nombre de la política y la Descripción (opcional) para la política que está creando. Revise los Permisos definidos en esta política para ver los permisos que concede la política.
8. Elija Crear política para guardar la nueva política.

Después de crear la política, adjúntela al rol de la instancia de contenedor. Para obtener información acerca de cómo asociar la política al rol, consulte Updating permissions for a role en la Guía del usuario de AWS Identity and Access Management.

AWS CLI

1. Cree un archivo denominado `instance-cw-logs.json` con el siguiente contenido.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": ["arn:aws:logs:*:*:*"]
        }
    ]
}
```

2. Utilice el siguiente comando de la para crear la política de IAM con el archivo de documento de política de JSON.

```
aws iam create-policy \
      --policy-name cwlogspolicy \
      --policy-document file://instance-cw-logs.json
```

3. Recupere el ARN de la política de IAM que creó mediante el siguiente comando. Sustituya *cwlogspolicy* por el nombre de la política que haya creado.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`cwlogspolicy`].Arn'
```

4. Utilice el siguiente comando para adjuntar la política al rol de IAM de la instancia de contenedor mediante el ARN de la política.

```
aws iam attach-role-policy \
      --role-name ecsInstanceRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/cwlogspolicy
```

Rol de IAM de Amazon ECS Anywhere

Cuando registra un servidor ubicado en las instalaciones o una máquina virtual (VM) en el clúster, el servidor o la VM requieren un rol de IAM para comunicarse con las API de AWS. Solo tiene que crear este rol de IAM una vez por cada cuenta de AWS. Sin embargo, este rol de IAM debe asociarse a cada servidor o máquina virtual que se registre en un clúster. Este rol es el `ECSAnywhereRole`. Este rol se puede crear manualmente. Como alternativa, Amazon ECS puede crear el rol en su nombre cuando registre una instancia externa en la AWS Management Console. Puede utilizar la búsqueda en la consola de IAM para buscar `ecsAnywhereRole` y ver si la cuenta ya dispone del rol. Para obtener más información, consulte IAM console search en la Guía del usuario de IAM.

AWS proporciona dos políticas de IAM administradas que se pueden utilizar al crear las políticas de rol de IAM, `AmazonSSMManagedInstanceCore` y `AmazonEC2ContainerServiceforEC2Role` de ECS Anywhere. La política `AmazonEC2ContainerServiceforEC2Role` incluye permisos que probablemente proporcionen más acceso del que necesita. Por lo tanto, según su caso de uso específico, le recomendamos que cree una política personalizada que agregue solo los permisos de esa política que necesite incluir en ella. Para obtener más información, consulte Rol de IAM de instancia de contenedor de Amazon ECS.

El rol de IAM de ejecución de tareas concede permiso al agente de contenedor de Amazon ECS para realizar llamadas a la API de AWS en su nombre. Cuando se utiliza un rol de IAM de ejecución de tareas, debe especificarse en la definición de tareas. Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

Se requiere el rol de ejecución de tareas si se aplica alguna de las siguientes condiciones:

- Está enviando registros de contenedor a CloudWatch Logs mediante el controlador de registros `awslogs`.
- La definición de tareas especifica una imagen de contenedor alojada en un repositorio privado de Amazon ECR. Sin embargo, si el rol `ECSAnywhereRole` asociado a su instancia externa también incluye los permisos necesarios para extraer imágenes de Amazon ECR, no es necesario que se incluyan en el rol de ejecución de tareas.

Creación del rol de Amazon ECS Anywhere

Sustituya cada *`entrada del usuario`* por información propia.

1. Cree un archivo local denominado `ssm-trust-policy.json` con la siguiente política de confianza.

   JSON

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

2. Utilice el siguiente comando de la AWS CLI para crear el rol y adjuntar la política de confianza.

   ```
   aws iam create-role --role-name ecsAnywhereRole --assume-role-policy-document
    file://ssm-trust-policy.json
   ```

3. Utilice el siguiente comando para adjuntar las políticas administradas de AWS.

```
aws iam attach-role-policy --role-name ecsAnywhereRole --policy-arn
 arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
aws iam attach-role-policy --role-name ecsAnywhereRole --policy-arn
 arn:aws:iam::aws:policy/service-role/AmazonEC2ContainerServiceforEC2Role
```

También puede utilizar el flujo de trabajo de la política de confianza personalizada de IAM para crear el rol. Para obtener instrucciones, consulte Creating a role using custom trust policies (console) en la Guía del usuario de IAM.

Rol de IAM de infraestructura de Amazon ECS

Un rol de IAM de infraestructura de Amazon ECS permite a Amazon ECS administrar los recursos de infraestructura de sus clústeres en su nombre y se utiliza cuando:

- Adjunte volúmenes de Amazon EBS a sus tareas de Amazon ECS del tipo lanzamiento de Fargate o EC2. El rol de infraestructura permite a Amazon ECS administrar los volúmenes de Amazon EBS para sus tareas.
- Utilice la seguridad de la capa de transporte (TLS) para cifrar el tráfico entre sus servicios de Amazon ECS Service Connect.
- Quiere crear grupos de destino de Amazon VPC Lattice.

Cuando Amazon ECS asume este rol para tomar medidas en su nombre, los eventos estarán visibles en AWS CloudTrail. Si Amazon ECS utiliza el rol para administrar los volúmenes de Amazon EBS adjuntos a sus tareas, el valor de `roleSessionName` del registro de CloudTrail será `ECSTaskVolumesForEBS`. Si el rol se utiliza para cifrar el tráfico entre los servicios de Service Connect, el `roleSessionName` del registro de CloudTrail será `ECSServiceConnectForTLS`. Si se usa el rol para crear grupos de destino para VPC Lattice, el elemento `roleSessionName` del registro de CloudTrail será `ECSNetworkingWithVPCLattice`. Para usar este nombre para buscar eventos en la consola de CloudTrail, filtre por Nombre de usuario.

Amazon ECS proporciona políticas administradas que contienen los permisos necesarios para adjuntar volúmenes y TLS. Para obtener más información, consulte AmazonECSInfrastructureRolePolicyForVolumes, AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity y AmazonECSInfrastructureRolePolicyForVpcLattice en la Guía de referencia de políticas administradas de AWS.

Creación del rol de infraestructura de Amazon ECS

Sustituya cada *entrada del usuario* por información propia.

1. Cree un archivo con el nombre `ecs-infrastructure-trust-policy.json`, que contenga la política de confianza que se va a utilizar para el rol de IAM. El archivo debe contener lo siguiente:

   JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessToECSForInfrastructureManagement",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Utilice el siguiente comando de la AWS CLI para crear un rol con el nombre `ecsInfrastructureRole` mediante la política de confianza que creó en el paso anterior.

```
aws iam create-role \
      --role-name ecsInfrastructureRole \
      --assume-role-policy-document file://ecs-infrastructure-trust-policy.json
```

3. Según el caso de uso, asocie la política `AmazonECSInfrastructureRolePolicyForVolumes`, `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity` o `AmazonECSInfrastructureRolePolicyForVpcLattice` administrada por AWS al rol de `ecsInfrastructureRole`.

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRole \
```

```
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSInfrastructureRolePolicyForVolumes
```

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity
```

También puede usar el flujo de trabajo de Política de confianza personalizada de la consola de IAM para crear el rol. Para obtener instrucciones, consulte Creating a role using custom trust policies (console) en la Guía del usuario de IAM.

> **Important**
>
> Si Amazon ECS utiliza el rol de infraestructura de ECS para administrar los volúmenes de Amazon EBS adjuntos a sus tareas, asegúrese de lo siguiente antes de detener las tareas que utilizan volúmenes de Amazon EBS.
>
> - El rol no se ha eliminado.
> - La política de confianza del rol no se modifica para eliminar el acceso a Amazon ECS (`ecs.amazonaws.com`).
> - La política administrada `AmazonECSInfrastructureRolePolicyForVolumes` no se ha eliminado. Si debe modificar los permisos del rol, retenga al menos `ec2:DetachVolume`, `ec2:DeleteVolume` y `ec2:DescribeVolumes` para eliminar el volumen.
>
> Si se elimina o modifica el rol antes de detener las tareas con volúmenes de Amazon EBS adjuntos, las tareas se atascarán en `DEPROVISIONING` y los volúmenes de Amazon EBS asociados no se podrán eliminar. Amazon ECS volverá a intentarlo automáticamente a intervalos regulares para detener la tarea y eliminar el volumen hasta que se restablezcan los permisos necesarios. Puede ver el estado del adjunto del volumen de una tarea y el motivo del estado asociado mediante la API DescribeTasks.

Después de crear el archivo, debe conceder a su usuario permiso para transferir el rol a Amazon ECS.

Permiso para transferir el rol de infraestructura a Amazon ECS

Para utilizar un rol de IAM de infraestructura de ECS, debe conceder a su usuario permiso para transferir el rol a Amazon ECS. Adjunte el siguiente permiso `iam:PassRole` a su usuario. Sustituya *`ecsInstanceRole`* por el nombre del rol de infraestructura que haya creado.

JSON

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

        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource": ["arn:aws:iam::*:role/ecsInfrastructureRole"],
            "Condition": {
                "StringEquals": {"iam:PassedToService": "ecs.amazonaws.com"}
            }
        }
    ]
}
```

Para obtener más información sobre `iam:Passrole` y la actualización de permisos de su usuario, consulte Granting a user permissions to pass a role to an AWS service y Changing permissions for an IAM user en la Guía del usuario de AWS Identity and Access Management.

Rol de IAM de infraestructura de Amazon ECS para los equilibradores de carga

Un rol de IAM de infraestructura de Amazon ECS para los equilibradores de carga permite a Amazon ECS administrar los recursos de los equilibradores de carga de los clústeres en su nombre y se utiliza cuando:

- Desea utilizar implementaciones azul/verde con Amazon ECS. El rol de infraestructura permite a Amazon ECS administrar los recursos del equilibrador de carga para las implementaciones.
- Es necesario Amazon ECS para crear, modificar o eliminar los recursos del equilibrador de carga, como los grupos de destino y los oyentes, durante las operaciones de implementación.

Cuando Amazon ECS asume este rol para tomar medidas en su nombre, los eventos estarán visibles en AWS CloudTrail. Si Amazon ECS utiliza el rol para administrar los recursos del equilibrador de carga para las implementaciones azul/verde, el registro de CloudTrail `roleSessionName` será `ECSNetworkingWithELB` o `ecs-service-scheduler`. Para usar este nombre para buscar eventos en la consola de CloudTrail, filtre por Nombre de usuario.

Amazon ECS proporciona una política administrada que contiene los permisos necesarios para la administración del equilibrador de carga. Para más información, consulte AmazonECSInfrastructureRolePolicyForLoadBalancers en la Guía de referencia de políticas administradas de AWS.

Creación del rol de infraestructura de Amazon ECS para los equilibradores de carga

Sustituya cada *`entrada del usuario`* por información propia.

1. Cree un archivo con el nombre `ecs-infrastructure-trust-policy.json`, que contenga la política de confianza que se va a utilizar para el rol de IAM. El archivo debe contener lo siguiente:

   JSON

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Sid": "AllowAccessToECSForInfrastructureManagement",
         "Effect": "Allow",
         "Principal": {
           "Service": "ecs.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
       }
     ]
   }
   ```

2. Utilice el siguiente comando de la AWS CLI para crear un rol con el nombre `ecsInfrastructureRoleForLoadBalancers` mediante la política de confianza que creó en el paso anterior.

   ```
   aws iam create-role \
         --role-name ecsInfrastructureRoleForLoadBalancers \
   ```

```
      --assume-role-policy-document file://ecs-infrastructure-trust-policy.json
```

3. Asocie la política `AmazonECSInfrastructureRolePolicyForLoadBalancers` administrada por AWS al rol `ecsInfrastructureRoleForLoadBalancers`.

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRoleForLoadBalancers \
      --policy-arn arn:aws:iam::aws:policy/
AmazonECSInfrastructureRolePolicyForLoadBalancers
```

También puede usar el flujo de trabajo de Política de confianza personalizada de la consola de IAM para crear el rol. Para obtener instrucciones, consulte Creating a role using custom trust policies (console) en la Guía del usuario de IAM.

> **Important**
>
> Si Amazon ECS utiliza el rol de infraestructura para administrar los recursos del equilibrador de carga para las implementaciones azul/verde, asegúrese de lo siguiente antes de eliminar o modificar el rol:
>
> - El rol no se elimina mientras las implementaciones activas están en curso.
> - La política de confianza del rol no se modifica para eliminar el acceso a Amazon ECS (`ecs.amazonaws.com`).
> - La política administrada `AmazonECSInfrastructureRolePolicyForLoadBalancers` no se elimina mientras las implementaciones activas están en curso.
>
> Eliminar o modificar el rol durante las implementaciones azul/verde activas puede provocar errores en la implementación y dejar los servicios en un estado incoherente.

Después de crear el archivo, debe conceder a su usuario permiso para transferir el rol a Amazon ECS.

Permiso para transferir el rol de infraestructura a Amazon ECS

Para utilizar un rol de IAM de infraestructura de ECS para los equilibradores de carga, debe conceder a su usuario permiso para transferir el rol a Amazon ECS. Adjunte el siguiente permiso

`iam:PassRole` a su usuario. Sustituya *`ecsInfrastructureRoleForLoadBalancers`* por el nombre del rol de infraestructura que haya creado.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource":
 ["arn:aws:iam::*:role/ecsInfrastructureRoleForLoadBalancers"],
            "Condition": {
                "StringEquals": {"iam:PassedToService": "ecs.amazonaws.com"}
            }
        }
    ]
}
```

Para obtener más información sobre `iam:Passrole` y la actualización de permisos de su usuario, consulte Granting a user permissions to pass a role to an AWS service y Changing permissions for an IAM user en la Guía del usuario de AWS Identity and Access Management.

Rol de IAM de CodeDeploy de Amazon ECS

Para poder utilizar el tipo de implementación azul/verde de CodeDeploy con Amazon ECS, el servicio de CodeDeploy necesita permisos para actualizar el servicio de Amazon ECS en su nombre. Estos permisos los proporciona el rol de IAM de CodeDeploy (`ecsCodeDeployRole`).

> **Note**
>
> Los usuarios también necesitan permisos para utilizar CodeDeploy; estos permisos se describen en Permisos de IAM necesarios.

Se proporcionan dos políticas administradas. Para obtener más información, consulte lo siguiente en la Guía de referencia de políticas administradas de AWS:

- AWSCodeDeployRoleForECS: concede a CodeDeploy permiso para actualizar cualquier recurso mediante la acción asociada.
- AWSCodeDeployRoleForECSLimited: concede a CodeDeploy más permisos limitados.

Creación del rol de CodeDeploy

Puede utilizar los siguientes procedimientos para crear un rol de CodeDeploy para Amazon ECS.

AWS Management Console

Creación del rol de servicio para CodeDeploy (consola de IAM)

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola de IAM, seleccione Roles y, a continuación, elija Crear rol.
3. En Tipo de entidad de confianza, elija Servicio de AWS.
4. En Servicio o caso de uso, seleccione CodeDeploy y, a continuación, seleccione el caso de uso CodeDeploy: ECS.
5. Elija Siguiente.
6. En la sección Asociar la política de permisos, asegúrese de que esté seleccionada la política AWSCodeDeployRoleForECS.
7. Elija Siguiente.
8. En Nombre del rol, ingrese ecsCodeDeployRole.
9. Revise el rol y, a continuación, elija Crear rol.

AWS CLI

Sustituya cada *`entrada del usuario`* por información propia.

1. Cree un archivo con el nombre `codedeploy-trust-policy.json`, que contenga la política de confianza que se va a utilizar para el rol de IAM de CodeDeploy.

   JSON

```
{
```

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

2. Cree un rol de IAM con el nombre `ecsCodedeployRole`, que utilice la política de confianza creada en el paso anterior.

```
aws iam create-role \
      --role-name ecsCodedeployRole \
      --assume-role-policy-document file://codedeploy-trust-policy.json
```

3. Adjunte la política administrada `AWSCodeDeployRoleForECS` o `AWSCodeDeployRoleForECSLimited` al rol `ecsTaskRole`.

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws:iam::aws:policy/AWSCodeDeployRoleForECS
```

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws:iam::aws:policy/AWSCodeDeployRoleForECSLimited
```

Cuando las tareas del servicio necesitan un rol de ejecución de tarea, debe agregar el permiso `iam:PassRole` correspondiente a cada rol de ejecución de tarea o anulación de rol de tarea al rol CodeDeploy como política.

Permisos de rol de ejecución de tareas

Cuando las tareas del servicio necesitan un rol de ejecución de tarea, debe agregar el permiso `iam:PassRole` correspondiente a cada rol de ejecución de tarea o anulación de rol de tarea al rol CodeDeploy como política. Para obtener más información, consulte Rol de IAM de ejecución de

tareas de Amazon ECS y Rol de IAM de tarea de Amazon ECS. A continuación, asocie esa política al rol CodeDeploy.

Creación de la política

AWS Management Console

Utilización del editor de política de JSON para la creación de una política

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, elija Políticas.

   Si es la primera vez que elige Políticas, aparecerá la página Welcome to Managed Policies (Bienvenido a políticas administradas). Elija Comenzar.
3. En la parte superior de la página, seleccione Crear política.
4. En la sección Editor de políticas, seleccione la opción JSON.
5. Ingrese el siguiente documento de política JSON:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>:role/
<ecsCodeDeployRole>"]
        }
    ]
}
```

6. Elija Siguiente.

> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual.

Para obtener más información, consulte Reestructuración de política en la Guía del usuario de IAM.

7. En la página Revisar y crear, introduzca el Nombre de la política y la Descripción (opcional) para la política que está creando. Revise los Permisos definidos en esta política para ver los permisos que concede la política.
8. Elija Crear política para guardar la nueva política.

Después de crear la política, adjunte la política al rol de CodeDeploy. Para obtener información acerca de cómo asociar la política al rol, consulte Actualización de los permisos de un rol en la Guía del usuario de AWS Identity and Access Management.

AWS CLI

Sustituya cada *entrada del usuario* por información propia.

1. Cree un archivo denominado `blue-green-iam-passrole.json` con el siguiente contenido.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                   "arn:arn:aws:iam::111122223333:role/ecs-code-deploy-
role"
            ]
        }
    ]
}
```

2. Utilice el siguiente comando de la para crear la política de IAM con el archivo de documento de política de JSON.

```
aws iam create-policy \
      --policy-name cdTaskExecutionPolicy \
```

```
      --policy-document file://blue-green-iam-passrole.json
```

3. Recupere el ARN de la política de IAM que creó mediante el siguiente comando.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`cdTaskExecutionPolicy`].Arn'
```

4. Utilice el siguiente comando para adjuntar la política al rol de IAM de CodeDeploy.

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/cdTaskExecutionPolicy
```

Rol de IAM de EventBridge de Amazon ECS

Para poder usar las tareas programadas de Amazon ECS con las reglas y los destinos de EventBridge, el servicio de EventBridge necesita varios permisos para ejecutar tareas de Amazon ECS en su nombre. Estos permisos los proporciona el rol de IAM de EventBridge (`ecsEventsRole`).

La política `AmazonEC2ContainerServiceEventsRole` se muestra a continuación.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:RunTask"],
            "Resource": ["*"]
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["*"],
            "Condition": {
                "StringLike": {"iam:PassedToService": "ecs-tasks.amazonaws.com"}
            }
        },
        {
            "Effect": "Allow",
```

```
            "Action": "ecs:TagResource",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:CreateAction": ["RunTask"]
                }
            }
        }
    ]
}
```

Si las tareas programadas requieren el uso del rol de ejecución de tareas, un rol de tarea o una cancelación del rol de tarea, debe agregar permisos `iam:PassRole` para cada rol de ejecución de tareas, rol de tarea o cancelación del rol de tarea al rol de IAM de EventBridge. Para obtener más información sobre el rol de ejecución de tareas, consulte Rol de IAM de ejecución de tareas de Amazon ECS.

> **Note**
>
> Especifique el ARN completo del rol de ejecución de tareas o el reemplazo del rol de tareas.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [

 "arn:aws:iam::111122223333:role/ecsTaskExecutionRole_or_TaskRole_name"
            ]
        }
    ]
}
```

Puede optar por dejar que AWS Management Console cree el rol de EventBridge por usted cuando configure una tarea programada. Para obtener más información, consulte Uso de Programador de Amazon EventBridge para programar tareas de Amazon ECS .

Creación del rol de EventBridge

Sustituya cada *entrada del usuario* por información propia.

1. Cree un archivo con el nombre `eventbridge-trust-policy.json`, que contenga la política de confianza que se va a utilizar para el rol de IAM. El archivo debe contener lo siguiente:

   JSON

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

2. Utilice el siguiente comando para crear un rol de IAM con el nombre `ecsEventsRole` mediante la política de confianza que se creó en el paso anterior.

```
aws iam create-role \
      --role-name ecsEventsRole \
      --assume-role-policy-document file://eventbridge-trust-policy.json
```

3. Adjunte la política administrada de AWS `AmazonEC2ContainerServiceEventsRole` al rol `ecsEventsRole` con el siguiente comando.

```
aws iam attach-role-policy \
      --role-name ecsEventsRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceEventsRole
```

También puede usar el flujo de trabajo de Política de confianza personalizada de la consola de IAM (https://console.aws.amazon.com/iam/) para crear el rol. Para obtener instrucciones, consulte Creating a role using custom trust policies (console) en la Guía del usuario de IAM.

Asociación de una política al rol **`ecsEventsRole`**

Puede utilizar los siguientes procedimientos para agregar permisos para el rol de ejecución de tareas al rol de IAM de EventBridge.

AWS Management Console

Utilización del editor de política de JSON para la creación de una política

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, elija Políticas.

   Si es la primera vez que elige Políticas, aparecerá la página Welcome to Managed Policies (Bienvenido a políticas administradas). Elija Comenzar.
3. En la parte superior de la página, seleccione Crear política.
4. En la sección Editor de políticas, seleccione la opción JSON.
5. Ingrese el siguiente documento de política JSON:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>:role/
<ecsTaskExecutionRole_or_TaskRole_name>"]
        }
    ]
}
```

6. Elija Siguiente.

> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de política en la Guía del usuario de IAM.

7. En la página Revisar y crear, introduzca el Nombre de la política y la Descripción (opcional) para la política que está creando. Revise los Permisos definidos en esta política para ver los permisos que concede la política.
8. Elija Crear política para guardar la nueva política.

Después de crear la política, adjúntela al rol de EventBridge. Para obtener información acerca de cómo asociar la política al rol, consulte Actualización de los permisos de un rol en la Guía del usuario de AWS Identity and Access Management.

AWS CLI

Sustituya cada *entrada del usuario* por información propia.

1. Cree un archivo denominado `ev-iam-passrole.json` con el siguiente contenido.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [

 "arn:aws:iam::111122223333:role/ecsTaskExecutionRole_or_TaskRole_name"
            ]
        }
    ]
}
```

2. Utilice el siguiente comando de la AWS CLI para crear la política de IAM con el archivo de documento de política de JSON.

```
aws iam create-policy \
      --policy-name eventsTaskExecutionPolicy \
      --policy-document file://ev-iam-passrole.json
```

3. Recupere el ARN de la política de IAM que creó mediante el siguiente comando.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`eventsTaskExecutionPolicy`].Arn'
```

4. Utilice el siguiente comando para adjuntar la política al rol de IAM de EventBridge mediante el ARN de la política.

```
aws iam attach-role-policy \
      --role-name ecsEventsRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/eventsTaskExecutionPolicy
```

Permisos necesarios para la consola de Amazon ECS

Siguiendo la práctica recomendada de concesión de privilegios mínimos, puede usar la política administrada `AmazonECS_FullAccess` como plantilla para crear su propia política personalizada. De esta forma, puede quitar permisos de la política administrada o agregar otros en función de sus requisitos específicos. Para obtener más información, consulte AmazonECS_FullAccess en Referencia de políticas administradas de AWS.

Permisos para crear roles de IAM

Las siguientes acciones requieren permisos adicionales para completar la operación:

- Registro de una instancia externa; para obtener más información, consulte Rol de IAM de Amazon ECS Anywhere
- Registro de una definición de tarea; para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS
- Creación de una regla de EventBridge para utilizarla en la programación de tareas; para obtener más información, consulte Rol de IAM de EventBridge de Amazon ECS

Puede añadir estos permisos creando un rol en IAM antes de usarlos en la consola de Amazon ECS. Si no crea los roles, la consola de Amazon ECS los creará en su nombre.

Permisos necesarios para registrar una instancia externa en un clúster

Necesita permisos adicionales cuando registra una instancia externa en un clúster y desea crear un nuevo rol de instancia externa (`ecsExternalInstanceRole`).

Se requieren los siguientes permisos adicionales:

- `iam`: permite a los usuarios principales crear y enumerar los roles de IAM y sus políticas asociadas.
- `ssm`: permite a los usuarios principales registrar la instancia externa con Systems Manager.

> **Note**
>
> Para elegir una `ecsExternalInstanceRole` existente, debe tener los permisos `iam:GetRole` y `iam:PassRole`.

La siguiente política contiene los permisos necesarios y limita las acciones al rol `ecsExternalInstanceRole`.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:CreateInstanceProfile",
              "iam:AddRoleToInstanceProfile",
              "iam:ListInstanceProfilesForRole",
              "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsExternalInstanceRole"
        },
        {
            "Effect": "Allow",
            "Action": ["iam:PassRole","ssm:CreateActivation"],
```

```
            "Resource": "arn:aws:iam::*:role/ecsExternalInstanceRole"
        }
    ]
}
```

Permisos necesarios para registrar una definición de tarea

Necesita permisos adicionales cuando registra una definición de tarea y desea crear un nuevo rol de ejecución de tareas (`ecsTaskExecutionRole`).

Se requieren los siguientes permisos adicionales:

- `iam`: permite a los usuarios principales crear y enumerar los roles de IAM y sus políticas asociadas.

> **Note**
>
> Para elegir una `ecsTaskExecutionRole` existente, debe tener el permiso `iam:GetRole`.

La siguiente política contiene los permisos necesarios y limita las acciones al rol `ecsTaskExecutionRole`.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsTaskExecutionRole"
        }
    ]
}
```

Permisos necesarios para crear una regla de EventBridge para las tareas programadas

Necesita permisos adicionales cuando programa una tarea y desea crear un nuevo rol de CloudWatch Events (`ecsEventsRole`).

Se requieren los siguientes permisos adicionales:

- `iam`: permite a los usuarios principales crear y enumerar los roles de IAM y sus políticas asociadas, y permite a Amazon ECS transferir el rol a otros servicios para que lo asuman.

> **Note**
>
> Para elegir una `ecsEventsRole` existente, debe tener los permisos `iam:GetRole` y `iam:PassRole`.

La siguiente política contiene los permisos necesarios y limita las acciones al rol `ecsEventsRole`.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:GetRole",
              "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsEventsRole"
        }
    ]
}
```

Permisos necesarios para ver las implementaciones de servicios

Si sigue la práctica recomendada de concesión de privilegios mínimos, debe agregar más permisos para ver las implementaciones de servicios en la consola.

Debe acceder a las siguientes acciones:

- ListServiceDeployments
- DescribeServiceDeployments
- DescribeServiceRevisions

Debe acceder a los siguientes recursos:

- Servicio
- Implementación de servicios
- Revisión de servicios

La siguiente política de ejemplo contiene los permisos necesarios y limita las acciones a un servicio especificado.

Sustituya `account`, `cluster-name` y `service-name` por sus valores.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "ecs:ListServiceDeployments",
              "ecs:DescribeServiceDeployments",
              "ecs:DescribeServiceRevisions"
            ],
            "Resource": [
             "arn:aws:ecs:us-east-1:123456789012:service/cluster-name/service-name",
             "arn:aws:ecs:us-east-1:123456789012:service-deployment/cluster-name/
service-name/*",
             "arn:aws:ecs:us-east-1:123456789012:service-revision/cluster-name/service-
name/*"
            ]
        }
    ]
}
```

Permisos necesarios para ver los eventos del ciclo de vida de Amazon ECS en Información de contenedores

Se requieren los siguientes permisos para ver los eventos del ciclo de vida: Agregue los siguientes permisos como una política en línea al rol. Para obtener más información, consulte Adición y eliminación de políticas de IAM.

- events:DescribeRule
- events:ListTargetsByRule
- logs:DescribeLogGroups

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:DescribeRule",
        "events:ListTargetsByRule",
        "logs:DescribeLogGroups"
      ],
      "Resource": "*"
    }
  ]
}
```

Permisos necesarios para habilitar los eventos del ciclo de vida de Amazon ECS en Información de contenedores

Los siguientes permisos son necesarios para configurar los eventos del ciclo de vida:

- events:PutRule
- events:PutTargets
- logs:CreateLogGroup

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutRule",
        "events:PutTargets",
        "logs:CreateLogGroup"
      ],
      "Resource": "*"
    }
  ]
}
```

Permisos necesarios para la consola de Amazon ECS con AWS CloudFormation

Antes de usar AWS Management Console para crear sus recursos, deberá asegurarse de tener los permisos de IAM correctos. Para obtener información sobre cómo configurar primero los permisos para la consola de Amazon ECS en general, consulte Permisos necesarios para la consola de Amazon ECS.

La consola de Amazon ECS se basa en AWS CloudFormation y requiere permisos de IAM adicionales en los siguientes casos:

- Creación de un clúster
- Crear un servicio
- Creación de un proveedor de capacidad

Puede crear una política para los permisos adicionales y, a continuación, adjuntarlas al rol de IAM que utilice para acceder a la consola. Para obtener más información, consulte Creación de políticas de IAM en la Guía del usuario de IAM.

Permisos necesarios para crear un clúster

Cuando crea un clúster en la consola, necesita permisos adicionales que le permitan administrar las pilas de AWS CloudFormation.

Se requieren los siguientes permisos adicionales:

- `cloudformation`: permite a los usuarios principales crear y administrar pilas de AWS CloudFormation. Esto se debe hacer cuando se crean clústeres de Amazon ECS mediante la AWS Management Console y posteriormente se administran.
- `ssm`: permite a AWS CloudFormation hacer referencia a la última AMI optimizada para Amazon ECS. Esto se debe hacer cuando se crean clústeres de Amazon ECS mediante el uso de la AWS Management Console.

La siguiente política contiene los permisos AWS CloudFormation necesarios y limita las acciones a los recursos creados en la consola de Amazon ECS.

```
{
  "Statement": [
      {
          "Effect": "Allow",
          "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStack*",
                "cloudformation:UpdateStack"
             ],
            "Resource": [
                "arn:*:cloudformation:*:*:stack/Infra-ECS-Cluster-*"
            ]
      },
      {
          "Effect": "Allow",
          "Action": "ssm:GetParameters",
          "Resource": [
            "arn:aws:ssm:*:*:parameter/aws/service/ecs/optimized-ami/amazon-linux-2*/
*",
            "arn:aws:ssm:*:*:parameter/aws/service/ecs/optimized-ami/amazon-
linux-2023*/*"
          ]
      }
   ]
}
```

Si no ha creado el rol de instancia de contenedor de Amazon ECS (`ecsInstanceRole`) y va a crear un clúster que utilice instancias de Amazon EC2, la consola creará el rol en su nombre.

Además, si utiliza grupos de escalado automático, necesitará permisos adicionales para que la consola pueda agregar etiquetas a los grupos de escalado automático cuando utilice la característica de escalado automático del clúster.

Se requieren los siguientes permisos adicionales:

- `autoscaling`: permite a la consola etiquetar el grupo de Amazon EC2 Auto Scaling. Esto se debe hacer cuando se administran grupos de Amazon EC2 Auto Scaling y se utiliza la característica de Auto Scaling de clústeres. La etiqueta es la etiqueta administrada por ECS que la consola agrega automáticamente al grupo para indicar que se creó en la consola.
- `iam`: permite a los usuarios principales enumerar los roles de IAM y sus políticas asociadas. Los usuarios principales también pueden enumerar perfiles de instancias disponibles para instancias de Amazon EC2.

La siguiente política contiene los permisos de IAM necesarios y limita las acciones al rol `ecsInstanceRole`.

Los permisos de Auto Scaling no están limitados.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:CreateInstanceProfile",
              "iam:AddRoleToInstanceProfile",
              "iam:ListInstanceProfilesForRole",
              "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsInstanceRole"
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:CreateOrUpdateTags",
            "Resource": "*"
        }
    ]
}
```

Permisos necesarios para crear un servicio

Cuando crea un servicio en la consola, necesita permisos adicionales que le permitan administrar las pilas de AWS CloudFormation. Se requieren los siguientes permisos adicionales:

- `cloudformation`: permite a los usuarios principales crear y administrar pilas de AWS CloudFormation. Esto se debe hacer cuando se crean servicios de Amazon ECS mediante la AWS Management Console y posteriormente se administran.

La siguiente política contiene los permisos necesarios y limita las acciones a los recursos creados en la consola de Amazon ECS.

```
{
  "Statement": [
      {
          "Effect": "Allow",
          "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStack*",
                "cloudformation:UpdateStack"
             ],
            "Resource": [
                "arn:*:cloudformation:*:*:stack/ECS-Console-V2-Service-*"
            ]
      }
   ]
}
```

Permisos necesarios para las funciones de Lambda en las implementaciones azul/verde de Amazon ECS

Al utilizar las funciones de Lambda como enlaces de ciclo de vida de las implementaciones en las implementaciones azul/verde de Amazon ECS, debe crear un rol de IAM con permisos específicos. Este rol permite a Amazon ECS invocar las funciones de Lambda en diversas etapas del ciclo de vida de la implementación.

Se requieren los siguientes permisos adicionales:

- `lambda:InvokeFunction`: permite a Amazon ECS invocar las funciones de Lambda configuradas como enlaces de ciclo de vida durante el proceso de implementación.

La siguiente política contiene los permisos necesarios para los enlaces de ciclo de vida de Lambda:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": "arn:aws:lambda:*:*:function:*"
        }
    ]
}
```

Para la política de confianza, debe permitir que el servicio asuma este rol:

JSON

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

Permisos de IAM necesarios para el escalado automático del servicio de Amazon ECS

Service Auto Scaling es posible gracias a la combinación de las API de Amazon ECS, CloudWatch y Application Auto Scaling. Los servicios se crean con Amazon ECS, las alarmas, con CloudWatch y las políticas de escalado, con Application Auto Scaling.

Además de los permisos estándar de IAM para la creación y actualización de los servicios, los siguientes permisos son necesarios para poder interactuar con la configuración de Service Auto Scaling, tal como se muestra en la siguiente política de ejemplo.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:*",
                "ecs:DescribeServices",
                "ecs:UpdateService",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarmHistory",
                "cloudwatch:DescribeAlarmsForMetric",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics",
                "cloudwatch:DisableAlarmActions",
                "cloudwatch:EnableAlarmActions",
                "iam:CreateServiceLinkedRole",
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Get*",
                "sns:List*"
            ],
            "Resource": ["*"]
        }
    ]
}
```

En las políticas de IAM Ejemplo de creación de servicios de Amazon ECS y Ejemplo de actualización de servicios de Amazon ECS de ejemplo, se muestran los permisos requeridos para utilizar Service Auto Scaling desde la AWS Management Console.

El servicio de Application Auto Scaling también necesita permiso para describir los servicios de Amazon ECS y las alarmas de CloudWatch, así como permisos para modificar el recuento deseado del servicio en su nombre. Los permisos `sns:` corresponden a las notificaciones que CloudWatch envía a un tema de Amazon SNS cuando se ha superado un umbral. Si usa el escalado automático para los servicios de Amazon ECS, se crea un rol vinculado a servicio con el nombre `AWSServiceRoleForApplicationAutoScaling_ECSService`. Este rol vinculado al servicio concede a Application Auto Scaling permiso para describir las alarmas de las políticas, monitorear el recuento de tareas actuales en ejecución del servicio y modificar el recuento deseado del servicio. El rol de Amazon ECS original administrado para Application Auto Scaling era `ecsAutoscaleRole`, pero ya no es necesario. El rol vinculado al servicio es el rol predeterminado para Auto Scaling de aplicaciones. Para obtener más información, consulte Roles vinculados a servicios de Application Auto Scaling en la Guía del usuario de Application Auto Scaling.

Si creó el rol de instancia de contenedor de Amazon ECS antes de que las métricas de CloudWatch estén disponibles para Amazon ECS, es posible que tenga que agregar el permiso `ecs:StartTelemetrySession`. Para obtener más información, consulte Consideraciones.

Conceder permisos para etiquetar recursos durante la creación

Las siguientes acciones de la API de creación de Amazon ECS mediante el agregado de etiquetas le permiten especificar etiquetas durante la creación del recurso. Si se especifican etiquetas en la acción de creación de recursos, AWS realiza una autorización adicional para verificar se asignen los permisos correctos para crear etiquetas.

- `CreateCapacityProvider`
- `CreateCluster`
- `CreateService`
- `CreateTaskSet`
- `RegisterContainerInstance`
- `RegisterTaskDefinition`
- `RunTask`
- `StartTask`

Puede utilizar etiquetas de recursos para implementar el control basado en atributos (ABAC). Para obtener más información, consulte the section called “Control del acceso a recursos de Amazon ECS mediante etiquetas de recursos” y Etiquetado de recursos.

Para permitir el etiquetado en el momento de la creación, cree o modifique una política que incluya tanto los permisos para usar la acción que crea el recurso, como `ecs:CreateCluster` o `ecs:RunTask`, como la acción `ecs:TagResource`.

En el ejemplo siguiente se muestra una política que permite a los usuarios crear clústeres y agregar etiquetas durante la creación del clúster. No se permite a los usuarios etiquetar ningún recurso (no pueden llamar directamente a la acción `ecs:TagResource`).

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ecs:CreateCluster"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ecs:TagResource"
      ],
      "Resource": "*",
      "Condition": {
         "StringEquals": {
                  "ecs:CreateAction": [
                      "CreateCluster",
                      "CreateCapacityProvider",
                      "CreateService",
                      "CreateTaskSet",
                      "RegisterContainerInstance",
                      "RegisterTaskDefinition",
                      "RunTask",
                      "StartTask"
                  ]
          }
       }
    }
  ]
```

```
}
```

La acción `ecs:TagResource` solo se evalúa si se aplican etiquetas durante la acción de creación de recursos. Por lo tanto, un usuario que tenga permisos para crear un recurso (suponiendo que no existan condiciones de etiquetado) no necesita permisos para utilizar la acción `ecs:TagResource` si no se especifica ninguna etiqueta en la solicitud. Sin embargo, si el usuario intenta crear un recurso con etiquetas, la solicitud dará un error si el usuario no tiene permisos para utilizar la acción `ecs:TagResource`.

Control de acceso de Amazon ECS para etiquetas específicas

Puede utilizar condiciones adicionales en el elemento `Condition` de las políticas de IAM para controlar las claves y los valores de etiqueta que se pueden aplicar a los recursos.

Las siguientes claves de condición se pueden utilizar con los ejemplos de la sección anterior:

- `aws:RequestTag`: indicar que una clave de etiqueta o una clave y valor de etiqueta determinados deben existir en una solicitud. También se pueden especificar otras etiquetas en la solicitud.
  - Debe utilizarse con el operador de condición `StringEquals` para aplicar la combinación de valor y clave de etiqueta específica; por ejemplo, para aplicar la etiqueta `cost-center=cc123`:

    ```
    "StringEquals": { "aws:RequestTag/cost-center": "cc123" }
    ```

  - Debe utilizarse con el operador de condición `StringLike` para aplicar una clave de etiqueta específica en la solicitud; por ejemplo, para aplicar la clave de etiqueta `purpose`:

    ```
    "StringLike": { "aws:RequestTag/purpose": "*" }
    ```

- `aws:TagKeys`: aplicar las claves de etiqueta que se usan en la solicitud.
  - Debe utilizarse con el modificador `ForAllValues` para aplicar claves de etiqueta específicas si estas se proporcionan en la solicitud (si se especifican etiquetas en la solicitud, solo se permiten claves de etiqueta específicas; no se permite ninguna etiqueta más). Por ejemplo, se permiten las claves de etiqueta `environment` o `cost-center`:

    ```
    "ForAllValues:StringEquals": { "aws:TagKeys": ["environment","cost-center"] }
    ```

  - Debe utilizarse con el modificador `ForAnyValue` para aplicar la presencia de como mínimo una de las claves de etiqueta especificadas en la solicitud. Por ejemplo, debe haber al menos una de las claves de etiqueta `environment` o `webserver` en la solicitud:

```
"ForAnyValue:StringEquals": { "aws:TagKeys": ["environment","webserver"] }
```

Estas claves de condición se pueden aplicar a las acciones que crean recursos y admiten el etiquetado, así como a las acciones `ecs:TagResource`. Para saber si una acción de la API de Amazon ECS admite el etiquetado, consulte Acciones, recursos y claves de condición para Amazon ECS.

Para obligar a los usuarios a especificar etiquetas cuando crean un recurso, debe utilizar la clave de condición `aws:RequestTag` o la clave de condición `aws:TagKeys` con el modificador `ForAnyValue` en la acción de creación del recurso. La acción `ecs:TagResource` no se evalúa si un usuario no especifica etiquetas para la acción de creación del recurso.

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.

Para obtener más información sobre las condiciones con varios valores, consulte Condiciones con varias claves de contexto o valores en la Guía del usuario de IAM.

Control del acceso a recursos de Amazon ECS mediante etiquetas de recursos

Cuando crea una política de IAM que concede a los usuarios permiso para utilizar recursos de Amazon ECS, puede incluir información de etiquetas en el elemento `Condition` de la política para controlar el acceso basado en etiquetas. Esto se conoce como control de acceso basado en atributos (ABAC). El ABAC le proporciona un mejor control sobre los recursos que un usuario puede modificar, utilizar o eliminar. Para obtener más información, consulte ¿Qué es ABAC para AWS?

Por ejemplo, puede crear una política que permita a los usuarios eliminar un clúster, pero que deniegue la acción si el clúster tiene la etiqueta `environment=production`. Para ello, utilice la clave de condición `aws:ResourceTag` para permitir o denegar el acceso al recurso en función de las etiquetas que están asociadas al recurso.

```
"StringEquals": { "aws:ResourceTag/environment": "production" }
```

Para saber si una acción de la API de Amazon ECS permite controlar el acceso mediante la clave de condición `aws:ResourceTag`, consulte Acciones, recursos y claves de condición para Amazon

ECS. Tenga en cuenta que las acciones `Describe` no admiten permisos de nivel de recursos, de forma que debe especificarlas en una instrucción aparte sin condiciones.

Para ver ejemplos de políticas de IAM, consulte Políticas de ejemplo de Amazon ECS .

Si permite o deniega a los usuarios acceso a recursos en función de etiquetas, debe considerar denegar explícitamente a los usuarios la posibilidad de agregar estas etiquetas o retirarlas de los mismos recursos. De lo contrario, es posible que un usuario eluda sus restricciones y obtenga acceso a un recurso modificando sus etiquetas.

Políticas de ejemplo de Amazon ECS

Puede utilizar políticas de IAM para conceder a los usuarios permisos para ver y trabajar con recursos específicos en la consola de Amazon ECS. Puede utilizar las políticas de ejemplo de la sección anterior; sin embargo, estas son específicas para solicitudes que se realizan con la AWS CLI o un AWS SDK.

Ejemplo: permitir a los usuarios eliminar un clúster de Amazon ECS en función de las etiquetas

La siguiente política permite a los usuarios eliminar clústeres cuando la etiqueta tiene un par clave/valor que diga “Propósito/Prueba”.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ecs:DeleteCluster"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:ecs:us-east-1:111122223333:cluster/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Purpose": "Testing"
                }
            }
        }
    ]
}
```

Solución de problemas de identidad y acceso de Amazon Elastic Container Service

Utilice la siguiente información para diagnosticar y solucionar los problemas habituales que pueden surgir cuando se trabaja con Amazon ECS e IAM.

Temas

- No tengo autorización para realizar una acción en Amazon ECS
- No tengo autorización para realizar la operación iam:PassRole
- Quiero permitir a personas externas a mi Cuenta de AWS el acceso a mis recursos de Amazon ECS
- Recursos adicionales de solución de problemas

No tengo autorización para realizar una acción en Amazon ECS

Si recibe un error que indica que no tiene autorización para realizar una acción, las políticas se deben actualizar para permitirle realizar la acción.

En el siguiente ejemplo, el error se produce cuando el usuario de IAM `mateojackson` intenta utilizar la consola para consultar los detalles acerca de un recurso ficticio *`my-example-widget`*, pero no tiene los permisos ficticios `ecs:`*`GetWidget`*.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 ecs:GetWidget on resource: my-example-widget
```

En este caso, la política del usuario `mateojackson` debe actualizarse para permitir el acceso al recurso *`my-example-widget`* mediante la acción `ecs:`*`GetWidget`*.

Si necesita ayuda, póngase en contacto con su gestionador de AWS. El gestionador es la persona que le proporcionó las credenciales de inicio de sesión.

No tengo autorización para realizar la operación iam:PassRole

Si recibe un error que indica que no tiene autorización para realizar la acción `iam:PassRole`, sus políticas deben actualizarse para permitirle pasar un rol a Amazon ECS.

Algunos Servicios de AWS le permiten transferir un rol existente a dicho servicio en lugar de crear un nuevo rol de servicio o uno vinculado al servicio. Para ello, debe tener permisos para transferir el rol al servicio.

En el siguiente ejemplo, el error se produce cuando un usuario de IAM con el nombre `marymajor` intenta utilizar la consola para realizar una acción en Amazon ECS. Sin embargo, la acción requiere que el servicio cuente con permisos que otorguen un rol de servicio. Mary no tiene permisos para transferir el rol al servicio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

En este caso, las políticas de Mary se deben actualizar para permitirle realizar la acción `iam:PassRole`.

Si necesita ayuda, póngase en contacto con su gestionador de AWS. El gestionador es la persona que le proporcionó las credenciales de inicio de sesión.

Quiero permitir a personas externas a mi Cuenta de AWS el acceso a mis recursos de Amazon ECS

Puede crear un rol que los usuarios de otras cuentas o las personas externas a la organización puedan utilizar para acceder a sus recursos. Puede especificar una persona de confianza para que asuma el rol. En el caso de los servicios que admitan las políticas basadas en recursos o las listas de control de acceso (ACL), puede utilizar dichas políticas para conceder a las personas acceso a sus recursos.

Para obtener más información, consulte lo siguiente:

- Para saber si Amazon ECS admite estas características, consulte Cómo funciona Amazon Elastic Container Service con IAM.
- Para obtener información acerca de cómo proporcionar acceso a los recursos de las Cuenta de AWS de su propiedad, consulte Proporcionar acceso a un usuario de IAM a otra Cuentas de AWS de la que es propietario en la Guía del usuario de IAM.
- Para obtener información acerca de cómo proporcionar acceso a tus recursos a Cuentas de AWS de terceros, consulta Proporcionar acceso a Cuentas de AWS que son propiedad de terceros en la Guía del usuario de IAM.

- Para obtener información sobre cómo proporcionar acceso mediante una federación de identidades, consulta Proporcionar acceso a usuarios autenticados externamente (identidad federada) en la Guía del usuario de IAM.
- Para conocer sobre la diferencia entre las políticas basadas en roles y en recursos para el acceso entre cuentas, consulte Acceso a recursos entre cuentas en IAM en la Guía del usuario de IAM.

Recursos adicionales de solución de problemas

En las páginas siguientes se proporciona información acerca de los códigos de error:

- Mensajes de error de las tareas detenidas de Amazon ECS
- Visualización de los mensajes de eventos del servicio de Amazon ECS

Prácticas recomendadas de IAM para Amazon ECS

Puede usar AWS Identity and Access Management (IAM) para administrar y controlar el acceso a sus servicios y recursos de AWS mediante políticas basadas en reglas con fines de autenticación y autorización. Más específicamente, a través de este servicio, usted controla el acceso a sus recursos de AWS mediante políticas que se aplican a los usuarios, grupos o roles. Entre estos tres, los usuarios son cuentas que pueden tener acceso a sus recursos. Además, un rol de IAM es un conjunto de permisos que puede asumir una identidad autenticada, que no está asociada a una identidad concreta ajena a IAM. Para obtener más información, consulte Información general sobre la administración del acceso de Amazon ECS: permisos y políticas.

Siga la política de acceso con privilegios mínimos

Cree políticas cuyo alcance permita a los usuarios realizar las tareas prescritas. Por ejemplo, si un desarrollador necesita detener una tarea periódicamente, cree una política que solo permita esa acción en particular. El ejemplo siguiente solo permite a un usuario detener una tarea que pertenece a una determinada `task_family` en un clúster con un nombre de recurso de Amazon (ARN) específico. Hacer referencia a un ARN en una condición también es un ejemplo del uso de permisos a nivel de recursos. Puede utilizar permisos a nivel de recursos para especificar el recurso al que desea que se aplique una acción.

JSON

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:StopTask"
            ],
            "Condition": {
                "ArnEquals": {
                       "ecs:cluster": "arn:aws:ecs:us-
east-1:111122223333:cluster/cluster_name"
                }
            },
            "Resource": [
                   "arn:aws:ecs:us-east-1:111122223333:task-
definition/task_family:*"
            ]
        }
    ]
}
```

Configuración de los recursos del clúster de modo que sean el límite administrativo

Las políticas que tienen un alcance demasiado limitado pueden provocar una proliferación de roles y aumentar la sobrecarga administrativa. En lugar de crear roles que se centren únicamente en tareas o servicios específicos, cree roles que se limiten a clústeres y utilice el clúster como límite administrativo principal.

Creación de canalizaciones automatizadas para aislar a los usuarios finales de la API

Puede limitar las acciones que los usuarios pueden utilizar mediante la creación de canalizaciones que empaqueten e implementen aplicaciones automáticamente en los clústeres de Amazon ECS. De este modo, se delega con efectividad en la canalización la tarea de crear, actualizar y eliminar tareas. Para obtener más información, consulte el Tutorial: Implementación estándar de Amazon ECS con CodePipeline en la Guía del usuario de AWS CodePipeline.

Utilizar condiciones de política para una capa de seguridad adicional

Cuando necesite una capa de seguridad adicional, agregue una condición a su política. Esto puede resultar útil si está realizando una operación privilegiada o si necesita restringir el conjunto de

acciones que se pueden realizar con recursos específicos. El siguiente ejemplo de política requiere una autorización multifactorial al eliminar un clúster.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:DeleteCluster"
      ],
      "Condition": {
        "Bool": {
          "aws:MultiFactorAuthPresent": "true"
        }
      },
    "Resource": ["*"]
    }
  ]
}
```

Las etiquetas aplicadas a los servicios se propagan a todas las tareas que forman parte de ese servicio. De este modo, puede crear roles que se limiten a los recursos de Amazon ECS con etiquetas específicas. En la siguiente política, una entidad principal de IAM inicia y detiene todas las tareas con una clave de etiqueta de `Department` y un valor de etiqueta de `Accounting`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:RunTask"
            ],
```

```
            "Resource": "arn:aws:ecs:arn:aws::111122223333:task/my-cluster/*",
            "Condition": {
                "StringEquals": {"ecs:ResourceTag/Department": "Accounting"}
            }
        }
    ]
}
```

Auditoría periódica del acceso a las API

Un usuario puede cambiar de rol. Después de cambiar de rol, es posible que los permisos que se le concedieron anteriormente dejen de aplicarse. Asegúrese de auditar quién tiene acceso a las API de Amazon ECS y de comprobar si ese acceso sigue estando justificado. Considere la posibilidad de integrar la IAM con una solución de administración del ciclo de vida del usuario que revoque automáticamente el acceso cuando un usuario abandona la organización. Para obtener más información, consulte las pautas de auditoría de seguridad de AWS en la Guía del usuario de AWS Identity and Access Management.

Registro y monitoreo en Amazon Elastic Container Service

El monitoreo es una parte importante a la hora de mantener la fiabilidad, la disponibilidad y el rendimiento de Amazon Elastic Container Service y las soluciones de AWS. Debe recopilar datos de monitoreo de todas las partes de su solución de AWS para que pueda depurar un error multipunto de una forma más fácil si se produce. AWS proporciona varias herramientas para monitorear sus recursos de Amazon ECS y responder ante posibles incidentes:

Alarmas de Amazon CloudWatch

Observe una sola métrica durante el periodo que especifique y realice una o varias acciones según el valor de la métrica relativo a un determinado umbral durante varios periodos de tiempo. La acción es una notificación enviada a un tema de Amazon Simple Notification Service (Amazon SNS) o a una política de Amazon EC2 Auto Scaling. Las alarmas de CloudWatch no invocan acciones tan solo por tener un estado determinado; es necesario que el estado haya cambiado y se mantenga durante un número específico de periodos. Para obtener más información, consulte Supervisión de Amazon ECS con CloudWatch.

En el caso de servicios con tareas que utilizan el tipo de lanzamiento de Fargate, puede usar las alarmas de CloudWatch para la reducción y el escalado horizontal de las tareas de su servicio

en función de métricas de CloudWatch como, por ejemplo, la utilización de CPU y memoria. Para obtener más información, consulte Escalado automático de su servicio de Amazon ECS.

En el caso de clústeres con tareas o servicios que utilizan el tipo de lanzamiento de EC2, puede usar las alarmas de CloudWatch para la reducción y el escalado horizontal de las instancias de contenedor en función de métricas de CloudWatch como, por ejemplo, la reserva de memoria del clúster.

Registros de Amazon CloudWatch

Puede monitorear, almacenar y obtener acceso a los archivos de registro desde los contenedores de las tareas de Amazon ECS mediante la especificación del controlador de registros `awslogs` en sus definiciones de tareas. Para obtener más información, consulte Uso del controlador de awslogs.

También puede monitorear, almacenar y obtener acceso a los archivos de registro del sistema operativo y del agente de contenedor de Amazon ECS desde las instancias de contenedor de Amazon ECS. Este método de acceso a los registros se puede utilizar para los contenedores que usan el tipo de lanzamiento EC2.

Eventos de Amazon CloudWatch

Seleccione los eventos y diríjalos a uno o varios flujos o funciones de destino para realizar cambios, capturar información de estado y aplicar medidas correctivas. Para obtener más información, consulte Automaticzación de las respuestas a los errores de Amazon ECS mediante EventBridge en esta guía y EventBridge es la evolución de los Eventos de Amazon CloudWatch en la Guía del usuario de Amazon EventBridge.

Registros de AWS CloudTrail

CloudTrail proporciona un registro de las acciones que realiza un usuario, rol o servicio de AWS en Amazon ECS. Mediante la información recopilada por CloudTrail, puede determinar la solicitud que se realizó a Amazon ECS, la dirección IP desde la que se realizó, quién la realizó, cuándo se realizó y otros detalles adicionales. Para obtener más información, consulte Registro de llamadas a la API de Amazon ECS mediante AWS CloudTrail.

AWS Trusted Advisor

Trusted Advisor aprovecha las prácticas recomendadas aprendidas al atender a cientos de miles de clientes de AWS. Trusted Advisor inspecciona su entorno de AWS y realiza recomendaciones cuando surge la oportunidad de ahorrar dinero, mejorar el rendimiento y la disponibilidad del sistema o ayudar a cerrar deficiencias de seguridad. Todos los clientes de AWS tienen acceso

a cinco comprobaciones de Trusted Advisor. Los clientes con un plan de soporte Business o Enterprise pueden ver todas las comprobaciones de Trusted Advisor.

Para obtener más información, consulte AWS Trusted Advisor en la Guía del usuario de Soporte.

AWS Compute Optimizer

AWS Compute Optimizer es un servicio que analiza las métricas de configuración y utilización de sus recursos de AWS. Informa si sus recursos son óptimos y genera recomendaciones de optimización para reducir el costo y mejorar el rendimiento de sus cargas de trabajo.

Para obtener más información, consulte Recomendaciones de AWS Compute Optimizer para Amazon ECS.

Otra parte importante del monitoreo de Amazon ECS implica el monitoreo manual de los elementos que no cubren las alarmas de CloudWatch. Los paneles de las consolas de CloudWatch, Trusted Advisor y otras consolas de AWS proporcionan una vista rápida del entorno de AWS. Le recomendamos que también compruebe los archivos de registro en sus instancias de contenedor y los contenedores en sus tareas.

Validación de conformidad para Amazon Elastic Container Service

Para saber si un Servicio de AWS está incluido en el ámbito de programas de conformidad específicos, consulte Servicios de Servicios de AWS en el ámbito del programa de conformidad y escoja el programa de conformidad que le interese. Para obtener información general, consulta Programas de conformidad de AWS.

Puedes descargar los informes de auditoría de terceros utilizando AWS Artifact. Para obtener más información, consulta Descarga de informes en AWS Artifact.

Su responsabilidad de conformidad al utilizar Servicios de AWS se determina en función de la sensibilidad de los datos, los objetivos de cumplimiento de su empresa y la legislación y los reglamentos correspondientes. AWS proporciona los siguientes recursos para ayudar con la conformidad:

- Cumplimiento de seguridad y gobernanza: en estas guías se explican las consideraciones de arquitectura y se proporcionan pasos para implementar las características de seguridad y cumplimiento.
- Referencia de servicios válidos de HIPAA: muestra una lista con los servicios válidos de HIPAA. No todos los Servicios de AWS son aptos para HIPAA.

- Recursos de conformidad de AWS: este conjunto de manuales y guías podría aplicarse a su sector y ubicación.
- Guías de cumplimiento para clientes de AWS: comprenda el modelo de responsabilidad compartida desde el punto de vista del cumplimiento. Las guías resumen las mejores prácticas para garantizar la seguridad de los Servicios de AWS y orientan los controles de seguridad en varios marcos (incluidos el Instituto Nacional de Estándares y Tecnología (NIST, por sus siglas en inglés), el Consejo de Estándares de Seguridad de la Industria de Tarjetas de Pago (PCI, por sus siglas en inglés) y la Organización Internacional de Normalización (ISO, por sus siglas en inglés)).
- Evaluación de recursos con reglas en la Guía para desarrolladores de AWS Config: el servicio AWS Config evalúa en qué medida las configuraciones de sus recursos cumplen las prácticas internas, las directrices del sector y las normativas.
- AWS Security Hub: este Servicio de AWS proporciona una visión completa de su estado de seguridad en AWS. Security Hub utiliza controles de seguridad para evaluar sus recursos de AWS y comprobar su cumplimiento con los estándares y las prácticas recomendadas del sector de la seguridad. Para obtener una lista de los servicios y controles compatibles, consulta la Referencia de controles de Security Hub.
- Amazon GuardDuty: este Servicio de AWS detecta posibles amenazas para sus Cuentas de AWS, cargas de tarea, contenedores y datos mediante la supervisión de su entorno para detectar actividades sospechosas y maliciosas. GuardDuty puedes ayudarlo a satisfacer varios requisitos de conformidad, como PCI DSS, cumpliendo los requisitos de detección de intrusos que exigen determinados marcos de conformidad.
- AWS Audit Manager: este servicio de Servicio de AWS le ayuda a auditar continuamente el uso de AWS con el fin de simplificar la forma en que administra el riesgo y la conformidad con las normativas y los estándares del sector.

Prácticas recomendadas sobre seguridad y conformidad de Amazon ECS

La responsabilidad de conformidad al utilizar Amazon ECS se determina en función de la confidencialidad de los datos, los objetivos de cumplimiento de la empresa y la legislación y normativa aplicables.

Normas de seguridad de datos del sector de pagos con tarjeta (PCI DSS)

Es importante que comprenda el flujo completo de datos de los titulares de tarjetas (CHD) en el entorno cuando se adhiera a la norma PCI DSS. El flujo de CHD determina la aplicabilidad de la norma PCI DSS, define los límites y los componentes de un entorno de datos de titulares

de tarjetas (CDE) y, por lo tanto, el alcance de una evaluación de la PCI DSS. Determinar con precisión el alcance de la norma PCI DSS es clave para definir la postura de seguridad y, en última instancia, para que la evaluación sea exitosa. Los clientes deben disponer de un procedimiento para determinar el alcance que garantice su integridad y detecte los cambios o desviaciones del alcance.

La naturaleza temporal de las aplicaciones en contenedores conlleva más dificultades a la hora de auditar las configuraciones. Como resultado, los clientes deben conocer todos los parámetros de configuración de los contenedores para garantizar que se cumplan los requisitos de conformidad en todas las fases del ciclo de vida del contenedor.

Para obtener más información sobre cómo lograr la conformidad con la norma PCI DSS en Amazon ECS, consulte los siguientes documentos técnicos.

- Diseño de arquitectura en Amazon ECS para cumplir con la norma PCI DSS

HIPAA (Ley de Portabilidad y Responsabilidad de Seguros Médicos de EE. UU)

El uso de Amazon ECS con cargas de trabajo que procesan protected health information (PHI, información médica protegida) no requiere ninguna configuración adicional. Amazon ECS actúa como un servicio de orquestación que coordina el lanzamiento de contenedores en Amazon EC2. No funciona con los datos de la carga de trabajo que se está orquestando ni sobre ellos. De conformidad con las normas de la HIPAA y el apéndice sobre socios comerciales de AWS, la PHI debe cifrarse tanto en tránsito como en reposo cuando se accede a ella desde contenedores lanzados con Amazon ECS.

Hay varios mecanismos de cifrado en reposo disponibles con cada opción de almacenamiento de AWS, como Amazon S3, Amazon EBS y AWS KMS. Puede implementar una red superpuesta (como VNS3 o Weave Net) para garantizar el cifrado completo de la PHI transferida entre contenedores o para proporcionar una capa de cifrado redundante. También debe utilizar el registro completo y dirigir todos los registros de contenedores a Amazon CloudWatch. Para obtener información sobre las prácticas recomendadas de seguridad de infraestructura, consulte Protección de la infraestructura en Pilar de seguridad del Marco de AWS Well-Architected.

AWS Security Hub

Utilice AWS Security Hub. Este Servicio de AWS proporciona una visión completa de su estado de seguridad en AWS. Security Hub utiliza controles de seguridad para evaluar sus recursos de AWS y comprobar su cumplimiento con los estándares y las prácticas recomendadas del sector de la

seguridad. Para obtener una lista de los servicios y controles compatibles, consulte la Referencia de controles de Security Hub.

Supervisión en tiempo de ejecución de Amazon GuardDuty con Amazon ECS

Amazon GuardDuty es un servicio de detección de amenazas que ayuda a proteger las cuentas, los contenedores, las cargas de trabajo y los datos de su entorno de AWS. Mediante modelos de machine learning (ML) y capacidades de detección de anomalías y amenazas, GuardDuty supervisa continuamente los diferentes orígenes de registro y la actividad en tiempo de ejecución para identificar y priorizar los posibles riesgos de seguridad y actividades maliciosas en su entorno.

Utilice la supervisión en tiempo de ejecución en GuardDuty para identificar comportamientos malintencionados o no autorizados. La supervisión en tiempo de ejecución protege las cargas de trabajo que se ejecutan en Fargate y EC2 mediante la supervisión continua de la actividad de registro y red de AWS para identificar comportamientos malintencionados o no autorizados. La supervisión en tiempo de ejecución utiliza un agente de seguridad de GuardDuty ligero y totalmente administrado que analiza el comportamiento en el host (como el acceso a los archivos, la ejecución de procesos y las conexiones de red). Esto abarca problemas como el escalado de privilegios, el uso de credenciales expuestas, la comunicación con direcciones IP malintencionadas, los dominios y la presencia de malware en las cargas de trabajo de contenedores e instancias de Amazon EC2. Para obtener más información, consulte GuardDuty Runtime Monitoring en la Guía del usuario de GuardDuty.

Recomendaciones de conformidad

Le recomendamos que se comunique con los propietarios de los programas de cumplimiento de su empresa desde el principio y utilice el Modelo de responsabilidad compartida de AWS para identificar la responsabilidad del control de cumplimiento, a fin de garantizar el éxito de los programas de cumplimiento pertinentes. Para obtener más información, consulte AWS shared responsibility model for Amazon ECS .

AWS Fargate Estándar de procesamiento de la información federal (FIPS, Federal Information Processing Standard 140)

Estándar de procesamiento de la información federal (FIPS) La norma FIPS-140 es un estándar del gobierno estadounidense y canadiense que especifica los requisitos de seguridad para los módulos criptográficos que protegen la información confidencial. La FIPS-140 define un conjunto de funciones criptográficas validadas que se pueden utilizar para cifrar los datos en tránsito y los datos en reposo.

Al activar la conformidad con la norma FIPS-140, puede ejecutar cargas de trabajo en Fargate de forma que cumpla con la norma FIPS-140. Para obtener más información acerca de la conformidad con el estándar FIPS-140, consulte Estándar federal de procesamiento de la información (FIPS) 140-3.

Consideraciones sobre FIPS-140 de AWS Fargate

Tenga en cuenta lo siguiente cuando utilice el cumplimiento con la norma FIPS-140 en Fargate:

- La conformidad con la norma FIPS-140 solo está disponible en las regiones AWS GovCloud (US).
- Fargate es compatible con la versión 140.3 del estándar FIPS-140
- La conformidad con la norma FIPS-140 se desactiva de forma predeterminada. Debe activarlo.
- Sus tareas deben utilizar la siguiente configuración para cumplir con la norma FIPS-140:
  - El `operatingSystemFamily` debe ser `LINUX`.
  - El `cpuArchitecture` debe ser `X86_64`.
  - La versión de la plataforma Fargate debe ser `1.4.0` o posterior.

Utilice la norma FIPS en Fargate.

Utilice el siguiente procedimiento para hacer uso de la conformidad con la norma FIPS-140 en Fargate.

1. Active la conformidad con la norma FIPS-140. Para obtener más información, consulte the section called “Conformidad de AWS Fargate con el Estándar Federal de Procesamiento de la Información(FIPS-140)”.
2. Puede utilizar ECS Exec para ejecutar el siguiente comando a fin de verificar el estado de conformidad con la norma FIPS-140 de un clúster.

   Sustituya *`cluster-name`* por el nombre del clúster, *`task-id`* por el ID o ARN de la tarea y *`container-name`* por el nombre del contenedor de la tarea en el que desea ejecutar el comando.

   Un valor devuelto de “1” indica que está utilizando la norma FIPS.

```
aws ecs execute-command \
    --cluster cluster-name \
    --task task-id \
```

```
    --container container-name \
    --interactive \
    --command "cat /proc/sys/crypto/fips_enabled"
```

Uso de CloudTrail para la auditoría de FIPS-140 de Fargate

CloudTrail se enciende en su cuenta de AWS cuando la crea. Cuando se produce una actividad relativa a la API y la consola en Amazon ECS, dicha actividad se registra en un evento de CloudTrail junto con los eventos de los demás servicios de AWS en Historial de eventos. Puede ver, buscar y descargar los últimos eventos de la cuenta de AWS. Para obtener más información, consulte Visualización de eventos con el historial de eventos de CloudTrail.

Para obtener un registro continuo de los eventos de la cuenta de AWS, incluidos los eventos de Amazon ECS, cree un registro de seguimiento, para que CloudTrail envíe los archivos de registro a un bucket de Amazon S3. De manera predeterminada, cuando se crea un registro de seguimiento en la consola, el registro de seguimiento se aplica a todas las regiones de . El registro de seguimiento registra los eventos de todas las regiones de la partición de AWS y envía los archivos de registro al bucket de Amazon S3 especificado. También es posible configurar otros servicios de AWS para analizar en profundidad y actuar en función de los datos de eventos recopilados en los registros de CloudTrail. Para obtener más información, consulte the section called “Registro de llamadas a la API de Amazon ECS mediante AWS CloudTrail”.

En el ejemplo siguiente, se muestra una entrada de registro de CloudTrail que ilustra la acción de API `PutAccountSettingDefault`:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAIV5AJI5LXF5EXAMPLE",
         "arn": "arn:aws:iam::123456789012:user/jdoe",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIPWIOFC3EXAMPLE",
    },
    "eventTime": "2023-03-01T21:45:18Z",
    "eventSource": "ecs.amazonaws.com",
    "eventName": "PutAccountSettingDefault",
    "awsRegion": "us-gov-east-1",
    "sourceIPAddress": "52.94.133.131",
```

```
    "userAgent": "aws-cli/2.9.8 Python/3.9.11 Windows/10 exe/AMD64 prompt/off command/
ecs.put-account-setting",
    "requestParameters": {
        "name": "fargateFIPSMode",
        "value": "enabled"
    },
    "responseElements": {
        "setting": {
            "name": "fargateFIPSMode",
            "value": "enabled",
            "principalArn": "arn:aws:iam::123456789012:user/jdoe"
        }
    },
    "requestID": "acdc731e-e506-447c-965d-f5f75EXAMPLE",
    "eventID": "6afced68-75cd-4d44-8076-0beEXAMPLE",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "ecs-fips.us-gov-east-1.amazonaws.com"
    }
}
```

Seguridad de la infraestructura de Amazon Elastic Container Service

Como servicio administrado, Amazon Elastic Container Service está protegido por la seguridad de la red global AWS. Para obtener información sobre los servicios de seguridad de AWS y sobre cómo AWS protege la infraestructura, consulte Seguridad en la nube de AWS. Para diseñar su entorno de AWS siguiendo las prácticas recomendadas de seguridad de infraestructura, consulte Protección de la infraestructura en Portal de seguridad de AWS Well-Architected Framework.

Puede utilizar llamadas a la API publicadas en AWS para acceder a Amazon ECS 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.

Además, las solicitudes deben estar firmadas mediante un ID de clave de acceso y una clave de acceso secreta que esté asociada a una entidad principal de IAM. También puedes utilizar AWS Security Token Service (AWS STS) para generar credenciales de seguridad temporales para firmar solicitudes.

Puede llamar a estas operaciones de la API desde cualquier ubicación de red. Amazon ECS admite políticas de acceso basadas en recursos, que pueden incluir restricciones en función de la dirección IP de origen, así que asegúrese de que las políticas tengan en cuenta la dirección IP de la ubicación de red. También puede utilizar políticas de Amazon ECS para controlar el acceso desde puntos de enlace específicos de Amazon Virtual Private Cloud o VPC específicas. Este proceso aísla con eficacia el acceso de red a un recurso de Amazon ECS determinado únicamente desde la VPC específica de la red de AWS. Para obtener más información, consulte Puntos de enlace de la VPC de interfaz de Amazon ECS (AWS PrivateLink).

Puntos de enlace de la VPC de interfaz de Amazon ECS (AWS PrivateLink)

Para mejorar la posición de seguridad de su VPC, configure Amazon ECS para que utilice un punto de enlace de la VPC de interfaz. Los puntos de conexión de interfaz utilizan la tecnología AWS PrivateLink, que le permite obtener acceso privado a las API de Amazon ECS mediante direcciones IP privadas. AWS PrivateLink restringe todo el tráfico de red entre su VPC y Amazon ECS a la red de Amazon. No necesita una gateway de Internet, un dispositivo NAT ni una gateway privada virtual.

Para obtener más información acerca de AWS PrivateLink y los puntos de conexión de VPC, consulte Puntos de conexión de VPC en la Guía del usuario de Amazon VPC.

Consideraciones

Las consideraciones sobre los puntos de conexión en las regiones se introdujeron a partir del 23 de diciembre de 2023.

Antes de configurar los puntos de enlace de la VPC de interfaz para Amazon ECS, debe tener en cuenta las consideraciones siguientes:

- Debe tener los siguientes puntos de conexión de VPC específicos de la región:

> **Note**
>
> Si no configura todos los puntos de conexión, el tráfico pasará por los puntos de conexión públicos, no por el punto de conexión de VPC.

- `com.amazonaws.region.ecs-agent`
- `com.amazonaws.region.ecs-telemetry`
- `com.amazonaws.region.ecs`

Por ejemplo, la región Oeste de Canadá (Calgary) (ca-west-1) necesita los siguientes puntos de conexión de VPC:

- `com.amazonaws.ca-west-1.ecs-agent`
- `com.amazonaws.ca-west-1.ecs-telemetry`
- `com.amazonaws.ca-west-1.ecs`

- Cuando utilice una plantilla para crear recursos de AWS en la nueva región y la plantilla se haya copiado de una región introducida antes del 23 de diciembre de 2023, en función de la región de la que se haya copiado, lleve a cabo una de las siguientes operaciones.

  Por ejemplo, la región de origen de la copia es Este de EE. UU. (Norte de Virginia) (us-east-1). La región de destino de la copia es Oeste de Canadá (Calgary) (ca-west-1).

Configuración	Acción
La región de origen de la copia no tiene ningún punto de conexión de VPC.	Cree los tres puntos de conexión de VPC para la nueva región (por ejemplo, `com.amazonaws.ca-west-1.ecs-agent`).
La región de origen de la copia contiene puntos de conexión de VPC específicos de la región.	a. Cree los tres puntos de conexión de VPC para la nueva región (por ejemplo, `com.amazonaws.ca-west-1.ecs-agent`).

Configuración	Acción	
	b. Elimine los tres puntos de conexión de VPC de la región de origen de la copia (por ejemplo, `com.amazonaws.us-east-1.ecs-agent`).	

Consideraciones para los puntos de conexión de VPC de Amazon ECS del tipo de lanzamiento de Fargate

Cuando haya un punto de conexión de VPC para `ecr.dkr` y `ecr.api` en la misma VPC en la que se implementa una tarea de Fargate, utilizará el punto de conexión de VPC. Si no hay ningún punto de conexión de VPC, utilizará la interfaz de Fargate.

Antes de configurar los puntos de enlace de la VPC de interfaz para Amazon ECS, debe tener en cuenta las consideraciones siguientes:

- Las tareas que utilizan el tipo de lanzamiento de Fargate no requieren puntos de enlace de la VPC de interfaz para Amazon ECS, pero es posible que necesite los puntos de enlace de la VPC de interfaz de Amazon ECR, Secrets Manager o Amazon CloudWatch Logs que se describen en los siguientes puntos.
  - Para permitir que sus tareas extraigan imágenes privadas de Amazon ECR, debe crear los puntos de enlace de la VPC de interfaz de Amazon ECR. Para obtener más información, consulte Puntos de enlace de la VPC de interfaz (AWS PrivateLink) en la Guía del usuario de Amazon Elastic Container Registry.

> **Important**
>
> Si configura Amazon ECR para utilizar un punto de enlace de la VPC de interfaz, puede crear un rol de ejecución de tareas que incluya claves de condición para restringir el acceso a una VPC o punto de enlace de la VPC específicos. Para obtener más información, consulte Las tareas de Fargate que extraen imágenes de Amazon ECR a través de permisos de puntos de conexión de interfaz.

  - Para permitir que sus tareas extraigan información confidencial de Secrets Manager, debe crear los puntos de enlace de la VPC de interfaz de Secrets Manager. Para obtener más información, consulte Utilización de Secrets Manager con puntos de enlace de la VPC en la Guía del usuario de AWS Secrets Manager.
  - Si la VPC no tiene un gateway de Internet y sus tareas utilizan el controlador de registros `awslogs` para enviar información de registro a CloudWatch Logs, debe crear un punto de enlace de la VPC de interfaz de CloudWatch Logs. Para obtener más información, consulte Utilización de CloudWatch Logs con puntos de enlace de la VPC de interfaz en la Guía del usuario de Amazon CloudWatch Logs.
- Los puntos de enlace de la VPC no admiten las solicitudes entre regiones. Asegúrese de crear el punto de enlace en la misma región en la que tiene previsto enviar llamadas a la API de Amazon ECS. Por ejemplo, supongamos que desea ejecutar tareas en Este de EE. UU. (Norte de Virginia). Entonces, debe crear el punto de conexión de VPC de Amazon ECS en la región Este de EE. UU. (Norte de Virginia). Un punto de conexión de VPC de Amazon ECS creado en cualquier otra región no puede ejecutar tareas en Este de EE. UU. (Norte de Virginia).
- Los puntos de conexión de VPC solo admiten DNS proporcionadas por Amazon a través de Amazon Route 53. Si desea utilizar su propio DNS, puede utilizar el enrutamiento de DNS condicional. Para obtener más información, consulte Conjuntos de opciones de DHCP en la Guía del usuario de Amazon VPC.
- El grupo de seguridad asociado al punto de conexión de la VPC debe permitir las conexiones entrantes en el puerto TCP 443 desde la subred privada de la VPC.
- La administración de Service Connect del proxy de Envoy utiliza el punto de conexión de VPC `com.amazonaws.`*`region`*`.ecs-agent`. Cuando no utiliza los puntos de conexión de VPC, la administración de Service Connect del proxy de Envoy utiliza el punto de conexión `ecs-sc` de esa región. Para obtener una lista de los puntos de conexión de Amazon ECS en cada región, consulte puntos de conexión y cuotas de Amazon ECS.

Consideraciones para los puntos de conexión de VPC de Amazon ECS del tipo de lanzamiento de EC2

Antes de configurar los puntos de enlace de la VPC de interfaz para Amazon ECS, debe tener en cuenta las consideraciones siguientes:

- Las tareas que utilizan el tipo de lanzamiento de EC2 requieren que las instancias de contenedor en las que se lanzan ejecuten la versión `1.25.1` del agente de contenedor de Amazon ECS o una

posterior. Para obtener más información, consulte Administración de instancias de contenedor de Linux de Amazon ECS.

- Para permitir que sus tareas extraigan información confidencial de Secrets Manager, debe crear los puntos de enlace de la VPC de interfaz de Secrets Manager. Para obtener más información, consulte Utilización de Secrets Manager con puntos de enlace de la VPC en la Guía del usuario de AWS Secrets Manager.
- Si la VPC no tiene un gateway de Internet y sus tareas utilizan el controlador de registros `awslogs` para enviar información de registro a CloudWatch Logs, debe crear un punto de enlace de la VPC de interfaz de CloudWatch Logs. Para obtener más información, consulte Utilización de CloudWatch Logs con puntos de enlace de la VPC de interfaz en la Guía del usuario de Amazon CloudWatch Logs.
- Los puntos de enlace de la VPC no admiten las solicitudes entre regiones. Asegúrese de crear el punto de enlace en la misma región en la que tiene previsto enviar llamadas a la API de Amazon ECS. Por ejemplo, supongamos que desea ejecutar tareas en Este de EE. UU. (Norte de Virginia). Entonces, debe crear el punto de conexión de VPC de Amazon ECS en la región Este de EE. UU. (Norte de Virginia). Un punto de conexión de VPC de Amazon ECS creado en cualquier otra región no puede ejecutar tareas en Este de EE. UU. (Norte de Virginia).
- Los puntos de conexión de VPC solo admiten DNS proporcionadas por Amazon a través de Amazon Route 53. Si desea utilizar su propio DNS, puede utilizar el enrutamiento de DNS condicional. Para obtener más información, consulte Conjuntos de opciones de DHCP en la Guía del usuario de Amazon VPC.
- El grupo de seguridad asociado al punto de conexión de la VPC debe permitir las conexiones entrantes en el puerto TCP 443 desde la subred privada de la VPC.

Comprensión de los patrones de nomenclatura de los puntos de conexión de Amazon ECS

Es importante comprender que el agente de Amazon ECS puede realizar solicitudes a los puntos de conexión con sufijos numerados, como los siguientes:

- `ecs-a-1.region.amazonaws.com`, `ecs-a-2.region.amazonaws.com`, etc. para los puntos de conexión del agente.
- `ecs-t-1.region.amazonaws.com`, `ecs-t-2.region.amazonaws.com`, etc. para los puntos de conexión de telemetría.

Este comportamiento se produce porque el agente de Amazon ECS usa la API DiscoverPollEndpoint para determinar dinámicamente a qué punto de conexión específico conectarse. Si sus puntos de conexión de VPC no gestionan correctamente estas variaciones de puntos de conexión numerados, el agente volverá a utilizar puntos de conexión públicos, incluso si configuró puntos de conexión de VPC para los nombres base.

El rol de la API DiscoverPollEndpoint

El agente de Amazon ECS utiliza la API DiscoverPollEndpoint para descubrir el punto de conexión adecuado para sondear las tareas. Cuando el agente llama a esta API, recibe una URL de punto de conexión específica que puede incluir un sufijo numerado. Para garantizar que los puntos de conexión de la VPC funcionen correctamente, la configuración de la red debe permitir que el agente haga lo siguiente:

1. Acceder a la API DiscoverPollEndpoint.
2. Conectarse a las URL del punto de conexión devueltas, incluidas las que tienen sufijos numerados.

Si está solucionando problemas de conectividad de puntos de conexión de VPC, verifique que su agente pueda acceder tanto a los puntos de conexión base como a cualquier variación numerada que pueda devolver la API DiscoverPollEndpoint.

Creación de puntos de enlace de la VPC para Amazon ECS

Para crear el punto de conexión de VPC para el servicio de Amazon ECS, utilice el procedimiento Acceso a un servicio de AWS mediante un punto de conexión de interfaz en la Guía del usuario de Amazon VPC a fin de generar los siguientes puntos de conexión. Si dispone de instancias de contenedor dentro de la VPC, debe crear los puntos de enlace en el orden en que aparecen en la lista. Si tiene previsto crear sus instancias de contenedor después de que se cree el punto de enlace de la VPC, el orden no importa.

> **Note**
>
> Si no configura todos los puntos de conexión, el tráfico pasará por los puntos de conexión públicos, no por el punto de conexión de VPC.
> Al crear puntos de conexión, Amazon ECS también crea un nombre de DNS privado para el punto de conexión. Por ejemplo, `ecs-a.region.amazonaws.com` para ecs-agent y `ecs-t.region.amazonaws.com` para ecs-telemetry.

- `com.amazonaws.region.ecs-agent`
- `com.amazonaws.region.ecs-telemetry`
- `com.amazonaws.region.ecs`

> **Note**
>
> La *región* representa el identificador de región de una región de AWS compatible con Amazon ECS, como `us-east-2` para la región EE. UU. Este (Ohio).

El punto de conexión `ecs-agent` usa la API `ecs:poll` y el punto de conexión `ecs-telemetry` usa las API `ecs:poll` y `ecs:StartTelemetrySession`.

Si tiene tareas existentes que utilizan el tipo de lanzamiento de EC2, una vez creados los puntos de enlace de la VPC, cada instancia de contenedor necesita obtener la nueva configuración. Para que esto ocurra, debe reiniciar cada instancia de contenedor o reiniciar el agente de contenedor de Amazon ECS en cada instancia de contenedor. Para reiniciar el agente de contenedor, haga lo siguiente.

Para reiniciar el agente de contenedor de Amazon ECS

1. Inicie sesión en su instancia de contenedor mediante SSH.
2. Detenga el agente de contenedor de .

   ```
   sudo docker stop ecs-agent
   ```

3. Inicie el agente de contenedor.

   ```
   sudo docker start ecs-agent
   ```

Una vez creados los puntos de enlace de la VPC y después de que el agente de contenedor de Amazon ECS se haya reiniciado en cada instancia de contenedor, todas las tareas recién lanzadas obtendrán la nueva configuración.

Creación de una política de puntos de enlace de la VPC para Amazon ECS

Puede asociar una política de puntos de enlace con su punto de enlace de la VPC que controla el acceso a Amazon ECS. La política especifica la siguiente información:

- La entidad principal que puede realizar acciones.
- Las acciones que se pueden realizar.
- Los recursos en los que se pueden llevar a cabo las acciones.

Para más información, consulte Control del acceso a los servicios con puntos de enlace de la VPC en la Guía del usuario de Amazon VPC.

Ejemplo: política de punto de enlace de la VPC para acciones de Amazon ECS

A continuación, se muestra un ejemplo de una política de punto de enlace para Amazon ECS. Cuando se asocia a un punto de conexión, esta política concede acceso a permisos para crear y enumerar clústeres. Las acciones `CreateCluster` y `ListClusters` no aceptan ningún recurso, por tanto la definición de recursos se define en * para todos los recursos.

```
{
   "Statement":[
    {
      "Principal":"*",
      "Effect": "Allow",
      "Action": [
        "ecs:CreateCluster",
        "ecs:ListClusters"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

AWS shared responsibility model for Amazon ECS

Los asuntos relacionados con la seguridad y la conformidad son una responsabilidad compartida entre AWS y el cliente. Este modelo compartido puede ayudar a aliviar la carga operativa del cliente,

ya que AWS opera, administra y controla los componentes, desde el sistema operativo host y la capa de virtualización hasta la seguridad física de las instalaciones en las que operan los servicios. Por otra parte, el cliente asume la responsabilidad y la administración del sistema operativo invitado (incluidas las actualizaciones y los parches de seguridad), de otros softwares de aplicaciones asociadas y de la configuración del firewall del grupo de seguridad que ofrece AWS. Los clientes deben pensar detenidamente en los servicios que eligen, ya que las responsabilidades varían en función de los servicios que utilicen, la integración de estos en su entorno de TI, y la legislación y los reglamentos aplicables. La naturaleza de esta responsabilidad compartida también ofrece la flexibilidad y la posibilidad de que el cliente pueda controlar la implementación.

Tipo de lanzamiento de Fargate

En la siguiente ilustración se muestra el modelo de responsabilidad compartida para el tipo de lanzamiento de Fargate. Fargate ejecuta cada carga de trabajo en un entorno virtualizado de hardware aislado. Como resultado, cada tarea tiene capacidad de infraestructura exclusiva. Las cargas de trabajo en contenedores que se ejecutan en Fargate no comparten sistema operativo, kernel de Linux, interfaz de red, almacenamiento efímero, CPU ni memoria con otras tareas. Cuando se usa Fargate, los clientes no son responsables de proteger la infraestructura informática en la que se ejecutan sus contenedores. Fargate proporcionará y reparará la infraestructura en la que se ejecutan las cargas de trabajo de los clientes. Para obtener más información, consulte Retirada y mantenimiento de tareas para AWS Fargate en Amazon ECS .

Usted es responsable de administrar los siguientes recursos:

- La configuración de red, que incluye la VPC, las NACL, los grupos de seguridad y las tablas de enrutamiento.
- Cifrado de almacenamiento de clientes y servicios. Para obtener más información, consulte Opciones de almacenamiento para las tareas de Amazon ECS.
- Imágenes de contenedor. Para obtener más información, consulte Prácticas recomendadas de seguridad para las tareas y contenedores de Amazon ECS.
- Permisos de IAM para las aplicaciones mediante el rol de tareas. Para obtener más información, consulte Rol de IAM de tarea de Amazon ECS.

Tipo de lanzamiento de EC2

En la siguiente ilustración se muestra el modelo de responsabilidad compartida para el tipo de lanzamiento de EC2. Cuando se ejecutan tareas en instancias de EC2, usted es el responsable del mantenimiento de las instancias de EC2, además de los siguientes recursos:

- El agente de Amazon ECS.
- La AMI de la instancia de EC2, que incluye la reparación y el endurecimiento.
- La configuración de red, que incluye la VPC, las NACL, los grupos de seguridad y las tablas de enrutamiento.
- Cifrado de almacenamiento de clientes y servicios. Para obtener más información, consulte Opciones de almacenamiento para las tareas de Amazon ECS.
- Imágenes de contenedor. Para obtener más información, consulte Prácticas recomendadas de seguridad para las tareas y contenedores de Amazon ECS.
- Permisos de IAM para las aplicaciones mediante el rol de tareas. Para obtener más información, consulte Rol de IAM de tarea de Amazon ECS.

Prácticas recomendadas de seguridad de red para Amazon ECS

La seguridad de la red es un tema amplio que abarca varios subtemas. Estos incluyen el cifrado en tránsito, la segmentación y el aislamiento de la red, los firewalls, el enrutamiento del tráfico y la observabilidad.

Cifrado en tránsito

El cifrado del tráfico de la red evita que los usuarios no autorizados intercepten y lean los datos cuando esos se transmiten en una red. Con Amazon ECS, el cifrado de red se puede implementar de cualquiera de las siguientes formas.

- Uso de instancias de Nitro:

  De forma predeterminada, el tráfico se cifra automáticamente entre los siguientes tipos de instancia de Nitro: C5n, G4, I3en, M5dn, M5n, P3dn, R5dn y R5n. El tráfico no se cifra cuando se enruta a través de una puerta de enlace de tránsito, un equilibrador de carga o un intermediario similar.

  - Cifrado en tránsito
  - Anuncios de novedades de 2019

  - Esta charla de re:Inforce 2019
- Utilizar el protocolo de indicación del nombre del servidor (SNI) con el equilibrador de carga de aplicación:

  El equilibrador de carga de aplicación (ALB) y el equilibrador de carga de red (NLB) admiten la indicación de nombre de servidor (SNI). Al usar la SNI, puede colocar varias aplicaciones seguras en un solo oyente. Para eso, cada oyente cuenta con su propio certificado TLS. Le recomendamos que aprovisione los certificados para el equilibrador de cargas mediante AWS Certificate Manager (ACM) y, a continuación, los agregue a la lista de certificados del oyente. El equilibrador de carga de AWS utiliza un algoritmo de selección de certificados inteligentes con SNI. Si el nombre de host proporcionado por un cliente coincide con un único certificado en la lista de certificados, el equilibrador de carga selecciona este certificado. Si un nombre de host proporcionado por un cliente coincide con varios certificados de la lista, el equilibrador de carga selecciona un certificado que el cliente puede admitir. Los ejemplos incluyen un certificado autofirmado o un certificado generado a través del ACM.

  - SNI con un equilibrador de carga de aplicación
  - SNI con un equilibrador de carga de red
- Cifrado de extremo a extremo con certificados TLS:

  Esto implica implementar un certificado TLS con la tarea. Puede ser un certificado autofirmado o un certificado de una entidad de certificación de confianza. Puede obtener el certificado haciendo referencia a un secreto del certificado. De lo contrario, puede optar por ejecutar un contenedor que emita una solicitud de firma de certificado (CSR) a ACM y, a continuación, monte el secreto resultante en un volumen compartido.

  - Mantener la seguridad de la capa de transporte hasta sus contenedores mediante el equilibrador de carga de red con Amazon ECS, parte 1
  - Mantener la seguridad de la capa de transporte (TLS) hasta el contenedor, parte 2: Uso de AWS Private Certificate Authority

Integración en red de las tareas

Las siguientes recomendaciones tienen en cuenta el funcionamiento de Amazon ECS. Amazon ECS no utiliza una red superpuesta. En su lugar, las tareas se configuran para funcionar en distintos modos de red. Por ejemplo, las tareas que están configuradas para usar el modo `bridge` adquieren una dirección IP no enrutable de una red Docker que se ejecuta en cada host. Las tareas que están configuradas para usar el modo de red `awsvpc` adquieren una dirección IP de la subred del host.

Las tareas que se configuran con la red `host` utilizan la interfaz de red del host. `awsvpc` es el modo de red preferido. Esto se debe a que es el único modo que puede utilizar para asignar grupos de seguridad a las tareas. También es el único modo disponible para las tareas de AWS Fargate en Amazon ECS.

Grupos de seguridad para tareas

Se recomienda configurar las tareas para que utilicen el modo de red `awsvpc`. Tras configurar la tarea para utilizar este modo, el agente de Amazon ECS aprovisiona y adjunta automáticamente una interfaz de red elástica (ENI) a la tarea. Cuando se aprovisiona la ENI, la tarea se inscribe en un grupo de seguridad de AWS. Un grupo de seguridad funciona como un firewall virtual que puede utilizar para controlar el tráfico entrante y saliente.

Si utiliza un firewall personalizado con tareas o servicios, agregue una regla de salida para permitir el tráfico en los puntos de conexión de administración de agentes de Amazon ECS (“`ecs-a-*.`*`region`*`.amazonaws.com`”), los puntos de conexión de telemetría (“`ecs-t-*.`*`region`*`.amazonaws.com`”) y los puntos de conexión de administración de Service Connect Envoy (“`ecs-sc.`*`region`*`.api.aws`”).

AWS PrivateLink y Amazon ECS

AWS PrivateLink es una tecnología de red que permite crear puntos de conexión privados para distintos servicios de AWS, incluido Amazon ECS. Los puntos de conexión son necesarios en entornos aislados en los que no hay una puerta de enlace de Internet (IGW) conectada a la VPC de Amazon ni rutas alternativas a Internet. El uso de AWS PrivateLink garantiza que las llamadas al servicio de Amazon ECS permanezcan dentro de la VPC de Amazon y no atraviesen Internet. Para obtener instrucciones sobre cómo crear puntos de conexión de AWS PrivateLink para Amazon ECS y otros servicios relacionados, consulte Puntos de conexión de VPC de Amazon en la interfaz de Amazon ECS.

> **Important**
>
> Las tareas de AWS Fargate no requieren un punto de conexión de AWS PrivateLink para Amazon ECS.

Tanto Amazon ECR como Amazon ECS admiten políticas de puntos de conexión. Estas políticas le permiten restringir el acceso a las API de un servicio. Por ejemplo, podría crear una política de punto

de conexión para Amazon ECR que solo permita enviar imágenes a los registros de determinadas cuentas de AWS. Una política como esta podría utilizarse para evitar que los datos se sustraigan a través de imágenes de contenedores y, al mismo tiempo, permitir a los usuarios acceder a registros autorizados de Amazon ECR. Para obtener más información, consulte Políticas de puntos de conexión de VPC.

La siguiente política permite a todas las entidades principales de AWS de su cuenta realizar todas las acciones únicamente contra los repositorios de Amazon ECR:

```
{
  "Statement": [
    {
      "Sid": "LimitECRAccess",
      "Principal": "*",
      "Action": "*",
      "Effect": "Allow",
      "Resource": "arn:aws:ecr:region:account_id:repository/*"
    },
  ]
}
```

Puede mejorarlo aún más estableciendo una condición que utilice la nueva propiedad de `PrincipalOrgID`. Esto evita que una entidad principal de IAM que no forma parte de su AWS Organizations inserte y extraiga imágenes. Para obtener más información, consulte aws:PrincipalOrgID.

Recomendamos aplicar la misma política tanto a los puntos de conexión `com.amazonaws.`*`region`*`.ecr.dkr` como a los `com.amazonaws.`*`region`*`.ecr.api`.

Configuración del agente de contenedor

El archivo de configuración del agente de contenedor de Amazon ECS incluye distintas variables de entorno relacionadas con la seguridad de la red. `ECS_AWSVPC_BLOCK_IMDS` y `ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST` se utilizan para bloquear el acceso de una tarea a los metadatos de Amazon EC2. `HTTP_PROXY` se utiliza para configurar el agente para que se dirija a través de un proxy HTTP para conectarse a Internet. Para obtener instrucciones sobre cómo configurar el agente y el tiempo de ejecución de Docker para que se enruten a través de un proxy, consulte Configuración del proxy HTTP.

> **Important**
>
> Estos ajustes no están disponibles cuando utilice AWS Fargate.

Recomendaciones de seguridad de red

Le recomendamos que haga lo siguiente al configurar la VPC de Amazon, los equilibradores de carga y la red.

Uso del cifrado de red cuando corresponda con Amazon ECS

Debe utilizar el cifrado de red cuando corresponda. Algunos programas de conformidad, como el PCI DSS, exigen que se cifren los datos en tránsito si estos contienen datos del titular de la tarjeta. Si su carga de trabajo tiene requisitos similares, configure el cifrado de red.

Los navegadores modernos advierten a los usuarios cuando se conectan a sitios inseguros. Si su servicio se presenta mediante un equilibrador de carga público, utilice TLS/SSL para cifrar el tráfico del navegador del cliente hacia el equilibrador de carga y, si es necesario, vuelva a cifrarlo al backend.

Uso del modo de red `awsvpc` y los grupos de seguridad para controlar el tráfico entre tareas y otros recursos en Amazon ECS

Debe utilizar el modo de red y los grupos de seguridad `awsvpc` cuando necesite controlar el tráfico entre tareas o entre tareas y otros recursos de la red. Si su servicio está basado en un ALB, utilice grupos de seguridad para permitir únicamente el tráfico entrante desde otros recursos de red que utilicen el mismo grupo de seguridad que el ALB. Si su aplicación está basada en un NLB, configure el grupo de seguridad de la tarea para que solo permita el tráfico entrante del rango CIDR de la VPC de Amazon y las direcciones IP estáticas asignadas al NLB.

Los grupos de seguridad también se deben usar para controlar el tráfico entre las tareas y otros recursos de la VPC de Amazon, como las bases de datos de Amazon RDS.

Creación de clústeres de Amazon ECS en VPC de Amazon independientes cuando es necesario aislar estrictamente el tráfico de red

Debe crear clústeres en VPC de Amazon independientes cuando sea necesario aislar estrictamente el tráfico de red. Evite ejecutar cargas de trabajo que tengan requisitos de seguridad estrictos en

clústeres con cargas de trabajo que no tengan que cumplir dichos requisitos. Cuando sea obligatorio aislar la red de forma estricta, cree clústeres en VPC de Amazon independientes y exponga los servicios de forma selectiva a otras VPC de Amazon mediante puntos de conexión de VPC de Amazon. Para obtener más información, consulte Puntos de enlace de la VPC.

Configuración de los puntos de conexión de AWS PrivateLink cuando esté justificado para Amazon ECS

Debe configurar los puntos de conexión de AWS PrivateLink cuando esté justificado. Si su política de seguridad le impide adjuntar una puerta de enlace de Internet (IGW) a sus VPC de Amazon, configure los puntos de conexión de AWS PrivateLink para Amazon ECS y otros servicios, como Amazon ECR, AWS Secrets Manager y Amazon CloudWatch.

Uso de los registros de flujo de VPC de Amazon para analizar el tráfico hacia y desde las tareas de larga ejecución en Amazon ECS

Debe utilizar los registros de flujo de VPC de Amazon para analizar el tráfico hacia y desde las tareas de larga ejecución. Las tareas que utilizan el modo de red `awsvpc` tienen su propia ENI. De este modo, puede supervisar el tráfico que va y viene de tareas individuales mediante registros de flujo de VPC de Amazon. Una actualización reciente de los registros de flujo de VPC de Amazon (v3) enriquece los registros con metadatos de tráfico, incluidos el ID de VPC, el ID de subred y el ID de instancia. Estos metadatos se pueden utilizar para ayudar a enfocar una investigación. Para obtener más información, consulte Registros de flujo de VPC de Amazon.

> **Note**
>
> Debido a la naturaleza temporal de los contenedores, es posible que los registros de flujo no siempre sean una forma eficaz de analizar los patrones de tráfico entre diferentes contenedores o entre contenedores y otros recursos de la red.

Prácticas recomendadas de seguridad para las tareas y contenedores de Amazon ECS

Debe considerar la imagen del contenedor como su primera línea de defensa contra un ataque. Una imagen insegura y mal construida puede permitir que un atacante escape de los límites del contenedor y acceder al host. Debe hacer lo siguiente para reducir el riesgo de que esto ocurra.

Al configurar las tareas y los contenedores, recomendamos que realice las siguientes acciones.

Cree imágenes minimalistas o utilice imágenes sin distribución.

Comience por eliminar todos los archivos binarios extraños de la imagen del contenedor. Si utiliza una imagen desconocida de la galería pública de Amazon ECR, inspeccione la imagen para consultar el contenido de cada una de las capas del contenedor. Para ello, puede utilizar una aplicación como Dive.

Como alternativa, puede usar imágenes sin distribución que solo incluyan su aplicación y sus dependencias de tiempo de ejecución. Estas no contienen administradores ni intérprete de comandos de paquetes. Las imágenes sin distribución mejoran la “relación señal-ruido de los escáneres y reducen el esfuerzo de establecer la procedencia a lo que se necesita”. Para obtener más información, consulte la documentación sobre sin distribución en GitHub.

Docker cuenta con un mecanismo para crear imágenes a partir de una imagen reservada y minimalista conocida como scratch. Para obtener más información, consulte Crear una imagen principal simple mediante scratch en la documentación de Docker. Con lenguajes como Go, puede crear un binario enlazado estático y hacer referencia a él en el Dockerfile. En los siguientes ejemplos, se muestra cómo hacerlo:

```
############################
# STEP 1 build executable binary
############################
FROM golang:alpine AS builder
# Install git.
# Git is required for fetching the dependencies.
RUN apk update && apk add --no-cache git
WORKDIR $GOPATH/src/mypackage/myapp/
COPY . .
# Fetch dependencies.
# Using go get.
RUN go get -d -v
# Build the binary.
RUN go build -o /go/bin/hello
############################
# STEP 2 build a small image
############################
FROM scratch
# Copy our static executable.
COPY --from=builder /go/bin/hello /go/bin/hello
# Run the hello binary.
```

```
ENTRYPOINT ["/go/bin/hello"]
This creates a container image that consists of your application and nothing else,
 making it extremely secure.
```

El ejemplo anterior también ilustra una compilación de varias etapas. Estos tipos de compilaciones son atractivos desde el punto de vista de la seguridad porque se pueden utilizar para minimizar el tamaño de la imagen final que se envía al registro del contenedor. Las imágenes de contenedores sin herramientas de compilación ni otros binarios desconocidos mejoran la seguridad al reducir la superficie expuesta a ataques de la imagen. Para obtener más información sobre las compilaciones de varias etapas, consulte Compilaciones de varias etapas en la documentación de Docker.

Escanear las imágenes para detectar vulnerabilidades

Al igual que las máquinas virtuales homólogas, las imágenes de contenedores pueden contener binarios y bibliotecas de aplicaciones con vulnerabilidades o, con el tiempo, pueden surgir vulnerabilidades en ellas. La mejor forma de protegerse contra las vulnerabilidades es escanear las imágenes con regularidad con un escáner de imágenes.

Amazon ECR proporciona dos versiones del escaneo básico que utilizan la base de datos de vulnerabilidades y exposiciones comunes (CVE):

- Análisis básico nativo de AWS: utiliza tecnología nativa de AWS, que ahora está disponible de forma general y cuyo uso está recomendado. Este análisis básico mejorado está diseñado para ofrecer a los clientes mejores resultados de análisis y detección de vulnerabilidades en un amplio conjunto de sistemas operativos populares. Esto permite a los clientes reforzar aún más la seguridad de las imágenes de sus contenedores. Todos los registros de clientes nuevos están incluidos en esta versión mejorada de forma predeterminada.
- Análisis básico de Clair: la versión de análisis básico anterior, que utiliza el proyecto Clair de código abierto. Para obtener más información acerca de Clair, consulte Clair en GitHub.

Tanto el análisis básico nativo de AWS como el de Clair están disponibles en todas las regiones que aparecen en Servicios de AWS por región, excepto las que se agregaron después de septiembre de 2024. Debido a que la compatibilidad con Clair está obsoleta, Clair no será compatible en las nuevas regiones a medida que se vayan agregando y dejará de ser compatible en todas las regiones a partir del 1 de octubre de 2025.

Amazon ECR utiliza la gravedad de CVE del origen de distribución ascendente si está disponible. De lo contrario, se utiliza la puntuación del sistema de clasificación de vulnerabilidades comunes

(CVSS). La puntuación CVSS se puede utilizar para obtener la calificación de gravedad de vulnerabilidad de NVD. Para obtener más información, consulte NVD Vulnerability Severity Ratings.

También puede ver los resultados de un escaneo desde la consola de Amazon ECR o llamando a la API DescribeImageScanFindings. Las imágenes con una vulnerabilidad `HIGH` o `CRITICAL` deben eliminarse o reconstruirse. Si una imagen que se ha implementado presenta una vulnerabilidad, debe sustituirse lo antes posible.

La versión 2.3.6.0 o posterior de Docker Desktop Edge puede escanear imágenes locales. Los escaneos se basan en Snyk, un servicio de seguridad de aplicaciones. Cuando se descubren vulnerabilidades, Snyk identifica las capas y las dependencias en las que se encuentra la vulnerabilidad en el Dockerfile. También recomienda alternativas seguras, como usar una imagen base más delgada con menos vulnerabilidades o actualizar un paquete concreto a una versión más reciente. Al utilizar el escaneo de Docker, los desarrolladores pueden resolver posibles problemas de seguridad antes de enviar sus imágenes al registro.

- En Automatizar la conformidad de imagen con Amazon ECR y AWS Security Hub, se explica cómo visualizar la información sobre vulnerabilidades de Amazon ECR en AWS Security Hub y cómo automatizar la corrección bloqueando el acceso a las imágenes vulnerables.

Eliminar los permisos especiales de sus imágenes

Los marcadores de derechos de acceso `setuid` y `setgid` permiten ejecutar un ejecutable con los permisos del propietario o del grupo del ejecutable. Elimine todos los binarios con estos derechos de acceso de la imagen, ya que estos binarios se pueden usar para aumentar los privilegios. Considere la posibilidad de eliminar todos los intérpretes de comandos y utilidades similares a `nc` y `curl` que puedan usarse con fines malintencionados. Puede encontrar los archivos con derechos de acceso `setuid` y `setgid` con el siguiente comando.

```
find / -perm /6000 -type f -exec ls -ld {} \;
```

Para eliminar los permisos especiales de estos archivos, agregue la siguiente directiva a la imagen del contenedor.

```
RUN find / -xdev -perm /6000 -type f -exec chmod a-s {} \; || true
```

Crear un conjunto de imágenes mantenidas

En lugar de permitir que los desarrolladores creen sus propias imágenes, cree un conjunto de imágenes revisadas para las diferentes pilas de aplicaciones de la organización. De este modo, los desarrolladores pueden prescindir de aprender cómo componer Dockerfiles y concentrarse en escribir código. A medida que los cambios se van integrando en el código base, una canalización de CI/CD puede compilar automáticamente el activo y, a continuación, almacenarlo en un repositorio de artefactos. Y, por último, copie el artefacto en la imagen correspondiente antes de subirlo a un registro de Docker, como Amazon ECR. Como mínimo, debe crear un conjunto de imágenes base a partir de las cuales los desarrolladores puedan crear sus propios Dockerfiles. Debe evitar extraer imágenes de Docker Hub. No siempre se sabe lo que hay en la imagen y aproximadamente una quinta parte de las 1000 imágenes principales tienen vulnerabilidades. Puede encontrar una lista de esas imágenes y sus vulnerabilidades en https://vulnerablecontainers.org/.

Escanear los paquetes y las bibliotecas de aplicaciones para detectar vulnerabilidades

El uso de bibliotecas de código abierto ahora es común. Al igual que ocurre con los sistemas operativos y los paquetes de sistemas operativos, estas bibliotecas pueden tener vulnerabilidades. Como parte del ciclo de vida del desarrollo, estas bibliotecas deben escanearse y actualizarse cuando se encuentren vulnerabilidades críticas.

Docker Desktop realiza escaneos locales con Snyk. También se puede utilizar para encontrar vulnerabilidades y posibles problemas de licencia en bibliotecas de código abierto. Se puede integrar directamente en los flujos de trabajo de los desarrolladores, lo que permite mitigar los riesgos que plantean las bibliotecas de código abierto. Para obtener más información, consulte los temas siguientes:

- En Herramientas de seguridad de aplicaciones de código abierto, se incluye una lista de herramientas para detectar vulnerabilidades en las aplicaciones.

Realizar un análisis de código estático

Debe realizar un análisis de código estático antes de crear una imagen de contenedor. Se realiza en función del código fuente y se utiliza para identificar errores de codificación y códigos que un agente malintencionado podría aprovechar, como las inserciones de errores. Puede usar Amazon Inspector.

Para obtener más información, consulte Escaneo de imágenes de contenedor de Amazon ECR con Amazon Inspector en la Guía del usuario de Amazon Inspector.

Ejecutar contenedores como usuario no raíz

Debe ejecutar contenedores como usuario no raíz. De forma predeterminada, los contenedores se ejecutan como usuario `root`, a menos que la directiva `USER` esté incluida en el Dockerfile. Las capacidades predeterminadas de Linux asignadas por Docker restringen las acciones que se pueden ejecutar como `root`, pero solo de manera marginal. Por ejemplo, un contenedor que se ejecuta como `root` aún no puede acceder a los dispositivos.

Como parte de su canalización de CI/CD, debe utilizar Dockerfiles para buscar la directiva `USER` y fallar en la compilación si no existe. Para obtener más información, consulte los temas siguientes:

- Dockerfile-lint es herramienta de código abierto de RedHat que se puede utilizar para comprobar si el archivo se ajusta a las prácticas recomendadas.
- Hadolint es otra herramienta para crear imágenes de Docker que se ajusten a las prácticas recomendadas.

Utilizar un sistema de archivos raíz de solo lectura

Debe utilizar un sistema de archivos raíz de solo lectura. De forma predeterminada, se puede escribir en el sistema de archivos raíz de un contenedor. Cuando configuras un contenedor con un sistema de archivos raíz `RO` (de solo lectura), se obliga a definir explícitamente dónde se pueden persistir los datos. Esto reduce la superficie expuesta a ataques, ya que no se puede escribir en el sistema de archivos del contenedor a menos que se concedan permisos específicos.

> **Note**
>
> Tener un sistema de archivos raíz de solo lectura puede provocar problemas con algunos paquetes del sistema operativo que esperan poder escribir en el sistema de archivos. Si planea utilizar sistemas de archivos raíz de solo lectura, pruébelos minuciosamente de antemano.

Configurar tareas con límites de CPU y memoria (Amazon EC2)

Debe configurar las tareas con límites de CPU y memoria para minimizar los siguientes riesgos. Los límites de recursos de una tarea establecen un límite superior para la cantidad de CPU y memoria que pueden reservar todos los contenedores de una tarea. Si no se establece ningún límite, las tareas tienen acceso a la CPU y la memoria del host. Esto puede provocar problemas en los que las tareas implementadas en un host compartido pueden privar a otras tareas de recursos del sistema.

> **Note**
>
> En cuanto a las tareas de AWS Fargate, Amazon ECS requiere que especifique los límites de CPU y memoria, ya que utiliza estos valores a efectos de facturación. Una tarea que acapara todos los recursos del sistema no es un problema para Amazon ECS Fargate, ya que cada tarea se ejecuta en su propia instancia dedicada. Si no especifica un límite de memoria, Amazon ECS asigna un mínimo de 4 MB a cada contenedor. Del mismo modo, si no se establece ningún límite de CPU para la tarea, el agente de contenedor de Amazon ECS le asigna un mínimo de 2 CPU.

Utilizar etiquetas inmutables con Amazon ECR

Con Amazon ECR, puede y debe utilizar imágenes configuradas con etiquetas inmutables. Esto evita que se envíe una versión alterada o actualizada de una imagen a su repositorio de imágenes con una etiqueta idéntica. De este modo, se evita que un atacante coloque una versión comprometida de una imagen sobre la suya con la misma etiqueta. Al usar etiquetas inmutables, se obliga de forma efectiva a colocar una nueva imagen con una etiqueta diferente en cada cambio.

Evitar ejecutar contenedores con privilegios (Amazon EC2)

Debe evitar ejecutar contenedores con privilegios. En segundo plano, los contenedores ejecutados como `privileged` se ejecutan con privilegios ampliados en el host. Esto significa que el contenedor hereda todas las capacidades de Linux asignadas `root` en el host. Su uso debe estar estrictamente restringido o prohibido. Recomendamos configurar la variable de entorno del agente de contenedores de Amazon ECS en `ECS_DISABLE_PRIVILEGED` a `true` para evitar que los contenedores se ejecuten como `privileged` en determinados hosts si `privileged` no es necesario. Como alternativa, puede utilizar AWS Lambda para escanear las definiciones de las tareas para utilizar el parámetro `privileged`.

> **Note**
>
> Ejecutar un contenedor como `privileged` no es compatible con Amazon ECS en AWS Fargate.

Eliminar del contenedor las capacidades innecesarias de Linux

A continuación, se presenta una lista de las capacidades de Linux predeterminadas asignadas a los contenedores de Docker. Para obtener más información sobre cada capacidad, consulte Descripción general de las capacidades de Linux.

```
CAP_CHOWN, CAP_DAC_OVERRIDE, CAP_FOWNER, CAP_FSETID, CAP_KILL,
CAP_SETGID, CAP_SETUID, CAP_SETPCAP, CAP_NET_BIND_SERVICE,
CAP_NET_RAW, CAP_SYS_CHROOT, CAP_MKNOD, CAP_AUDIT_WRITE,
CAP_SETFCAP
```

Si un contenedor no requiere todas las capacidades del kernel de Docker enumeradas anteriormente, considere eliminarlas del contenedor. Para obtener más información sobre cada capacidad del kernel de Docker, consulte KernelCapabilities. Puede saber qué capacidades se están utilizando siguiendo los siguientes pasos:

- Instale el paquete de sistema operativo libcap-ng y ejecute la utilidad `pscap` para enumerar las capacidades que utiliza cada proceso.
- También puede utilizar capsh para descifrar qué capacidades utiliza un proceso.

Utilizar una clave administrada por el cliente (CMK) para cifrar imágenes enviadas a Amazon ECR

Debe utilizar una clave administrada por el cliente (CMK) para cifrar imágenes enviadas a Amazon ECR. Las imágenes que se envían a Amazon ECR se cifran automáticamente en reposo con una clave administrada AWS Key Management Service (AWS KMS). Si prefiere usar su propia clave, Amazon ECR ahora admite el cifrado AWS KMS con claves administradas por el cliente (CMK). Antes de habilitar el cifrado del lado del servidor con una CMK, consulte las consideraciones que se indican en la documentación sobre el cifrado en reposo.

Tutoriales para Amazon ECS

En los siguientes tutoriales, aprenderá a realizar tareas comunes mediante Amazon ECS.

Puede utilizar cualquiera de los siguientes tutoriales para obtener más información sobre cómo empezar a utilizar Amazon ECS.

Información general del tutorial	Más información
Introducción a Amazon ECS en Fargate.	Obtenga información sobre cómo crear una tarea de Linux de Amazon ECS para el tipo de lanzamiento de Fargate.
Introducción a los contenedores de Windows en Fargate.	Obtenga información sobre cómo crear una tarea de Windows de Amazon ECS para el tipo de lanzamiento de Fargate.
Introducción a los contenedores Windows para el tipo de lanzamiento de EC2.	Obtenga información sobre cómo crear una tarea de Windows de Amazon ECS para el tipo de lanzamiento de EC2.

Puede utilizar cualquiera de los siguientes tutoriales para implementar tareas en Amazon ECS mediante la AWS CLI.

Información general del tutorial	Más información
Cree una tarea de Linux para el tipo de lanzamiento de Fargate.	Creación de una tarea de Linux de Amazon ECS para

Información general del tutorial	Más información	
	el tipo de lanzamiento de Fargate con la AWS CLI	
Cree una tarea de Windows para el tipo de lanzamiento de Fargate.	Creación de una tarea de Amazon ECS de Windows para el tipo de lanzamiento de Fargate con la AWS CLI	
Cree una tarea de Linux para el tipo de lanzamiento de EC2.	Creación de una tarea de Amazon ECS para el tipo de lanzamiento de EC2 con la AWS CLI	

Puede utilizar cualquiera de los siguientes tutoriales para obtener más información sobre la supervisión y el registro.

Información general del tutorial	Más información	
Configure una función de Lambda simple que escuche los eventos de tareas y los escriba en un flujo de registro de Registros de CloudWatch.	Configuración de Amazon ECS para escuchar los eventos de Eventos de CloudWatch	
Configure una regla de eventos de Amazon EventBridge que solo capture eventos de tareas en los que la tarea dejó de ejecutarse porque uno de sus contenedores esenciales finalizó.	Envío de alertas de Amazon Simple Notification Service para eventos de tareas detenidas de Amazon ECS	

Información general del tutorial	Más información	
Concatene mensajes de registro que originalmente pertenecen a un contexto, pero que se dividieron en varios registros o líneas de registro.	Concatenación de mensajes de registro de Amazon ECS de seguimiento de pila o de varias líneas	
Implemente contenedores de Fluent Bit en sus instancias de Windows que se ejecutan en Amazon ECS para transmitir los registros generados por las tareas de Windows a Amazon CloudWatch a fin de obtener un registro centralizado.	Implementación de Fluent Bit en contenedores de Amazon ECS para Windows	

Puede utilizar cualquiera de los siguientes tutoriales para obtener más información sobre cómo utilizar la autenticación de Active Directory con una cuenta de servicio administrada de grupo en Amazon ECS.

Información general del tutorial	Más información	
Utilice una cuenta de servicio administrada de grupo con contenedores de Linux en EC2.	Uso de gMSA para contenedores de EC2 Linux en Amazon ECS	
Utilice una cuenta de servicio administrada de grupo con contenedores de Windows en EC2.	Obtenga información sobre cómo utilizar gMSA para contenedores de EC2 para Windows en Amazon ECS.	

Información general del tutorial	Más información	
Utilice una cuenta de servicio administrada de grupo con contenedores de Linux en Fargate.	Uso de gMSA para contenedores de Linux en Fargate	
Cree una tarea que ejecute un contenedor de Windows que tenga credenciales para acceder a Active Directory con una cuenta de servicio administrada de grupo sin dominio.	Uso de contenedores de Amazon ECS para Windows con gMSA sin dominio mediante la AWS CLI	

Creación de una tarea de Linux de Amazon ECS para el tipo de lanzamiento de Fargate con la AWS CLI

Los siguientes pasos le ayudan a configurar un clúster, registrar una definición de tarea, ejecutar una tarea Linux y llevar a cabo otros escenarios comunes en Amazon ECS con la AWS CLI. Utilice la versión más reciente de la AWS CLI. Para obtener más información acerca de cómo actualizar a la versión más reciente, consulte Instalación o actualización de la versión más reciente de la AWS CLI.

> **Note**
>
> Puede utilizar puntos de conexión de servicio de doble pila para interactuar con Amazon ECS desde la AWS CLI, los SDK y la API de Amazon ECS a través de IPv4 e IPv6. Para obtener más información, consulte Uso de puntos de conexión de doble pila en Amazon ECS.

Temas

- Requisitos previos
- Paso 1: Crear un clúster
- Paso 2: Registrar una definición de tarea Linux

- Paso 3: Mostrar la lista de definiciones de tareas
- Paso 4: Crear un servicio
- Paso 5: Mostrar la lista de los servicios
- Paso 6: Describir el servicio en ejecución
- Paso 7: Prueba
- Paso 8: eliminación

Requisitos previos

En este tutorial se supone que los siguientes requisitos previos se han realizado.

- La última versión de la AWS CLI está instalada y configurada. Para obtener información acerca de cómo instalar o actualizar la AWS CLI, consulte Instalación o actualización de la versión más reciente de la AWS CLI.
- Se han completado los pasos que se indican en Configuración para utilizar Amazon ECS.
- Su usuario de IAM dispone de los permisos requeridos que se especifican en la política de IAM AmazonECS_FullAccess de ejemplo.
- Tiene una VPC y un grupo de seguridad creados para utilizarlos. En este tutorial se utiliza una imagen de contenedor alojada en Amazon ECR Public por lo que su tarea debe tener acceso a Internet. Para que su tarea tenga una ruta a Internet, utilice una de las siguientes opciones.
  - Utilice una subred privada con una gateway NAT que tenga una dirección IP elástica.
  - Utilice una subred pública y asigne una dirección IP pública a la tarea.

  Para obtener más información, consulte the section called “Creación de una nube virtual privada”.

  Para obtener información sobre las reglas y los grupos de seguridad, consulte Default security groups for your VPCs (Grupos de seguridad predeterminados para sus VPC) y Example rules (Reglas de ejemplo) en la Guía del usuario de Amazon Virtual Private Cloud.
- Si sigue este tutorial utilizando una subred privada, puede utilizar Amazon ECS Exec para interactuar directamente con el contenedor y probar la implementación. Necesitará crear un rol de IAM en una tarea para usar ECS Exec. Para obtener más información acerca del rol de IAM de la tarea y otros requisitos previos, consulte Supervisión de los contenedores de Amazon ECS con Amazon ECS Exec.

- (Opcional) AWS CloudShell es una herramienta que proporciona a los clientes una línea de comandos sin necesidad de crear su propia instancia de EC2. Para obtener más información, consulte ¿Qué es AWS CloudShell? en la Guía del usuario de AWS CloudShell.

Paso 1: Crear un clúster

De forma predeterminada, la cuenta recibe un clúster `default`.

> **Note**
>
> El beneficio de utilizar el clúster `default` que se le facilita es que no tiene que especificar la opción `--cluster` *`cluster_name`* en los comandos siguientes. Si crea su propio clúster no predeterminado, tiene que especificar `--cluster` *`cluster_name`* para cada comando que pretenda utilizar con dicho clúster.

Cree su propio clúster con un nombre único con el comando siguiente:

```
aws ecs create-cluster --cluster-name fargate-cluster
```

Salida:

```
{
    "cluster": {
        "status": "ACTIVE",
        "defaultCapacityProviderStrategy": [],
        "statistics": [],
        "capacityProviders": [],
        "tags": [],
        "clusterName": "fargate-cluster",
        "settings": [
            {
                "name": "containerInsights",
                "value": "disabled"
            }
        ],
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
```

```
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster"
    }
}
```

Paso 2: Registrar una definición de tarea Linux

Antes de poder ejecutar una tarea en su clúster de ECS, debe registrar una definición de tareas. Las definiciones de tareas son listas de contenedores agrupadas. El siguiente ejemplo es una definición de tarea simple que crea una aplicación web PHP usando la imagen del contenedor httpd alojada en Docker Hub. Para obtener más información acerca de los parámetros de definición de tareas disponibles, consulte Definiciones de tareas de Amazon ECS. Para este tutorial, el `taskRoleArn` solo es necesario si va a implementar la tarea en una subred privada y desea probar la implementación. Reemplace el `taskRoleArn` por el rol de tarea de IAM que creó para utilizar ECS Exec, como se menciona en Requisitos previos.

```
 {
        "family": "sample-fargate",
        "networkMode": "awsvpc",
        "taskRoleArn": "arn:aws:iam::aws_account_id:role/execCommandRole",
        "containerDefinitions": [
            {
                "name": "fargate-app",
                "image": "public.ecr.aws/docker/library/httpd:latest",
                "portMappings": [
                    {
                        "containerPort": 80,
                        "hostPort": 80,
                        "protocol": "tcp"
                    }
                ],
                "essential": true,
                "entryPoint": [
                    "sh",
                    "-c"
                ],
                "command": [
                    "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
 ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && httpd-
foreground\""
```

```
                ]
            }
        ],
        "requiresCompatibilities": [
            "FARGATE"
        ],
        "cpu": "256",
        "memory": "512"
}
```

Guarde el JSON de definición de tarea como un archivo y páselo con la opción `--cli-input-json file://path_to_file.json`.

Para utilizar un archivo JSON para definiciones de contenedor:

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/fargate-task.json
```

El comando register-task-definition devuelve una descripción de la definición de tarea después de realizar su registro.

Paso 3: Mostrar la lista de definiciones de tareas

Puede enumerar las definiciones de tareas para su cuenta en cualquier momento con el comando list-task-definitions. La salida de este comando muestra los valores `family` y `revision`, que puede utilizar conjuntamente al llamar a run-task o start-task.

```
aws ecs list-task-definitions
```

Salida:

```
{
    "taskDefinitionArns": [
        "arn:aws:ecs:region:aws_account_id:task-definition/sample-fargate:1"
    ]
}
```

Paso 4: Crear un servicio

Una vez que se haya registrado una tarea para la cuenta, puede crear un servicio para la tarea registradas en su clúster. En este ejemplo, se crea un servicio con una instancia de la definición de

tarea `sample-fargate:1` que se ejecuta en el clúster. La tarea requiere una ruta a Internet, por lo que hay dos maneras de lograr esto. Una forma es utilizar una subred privada configurada con una gateway NAT con una dirección IP elástica en una subred pública. Otra forma es utilizar una subred pública y asignar una dirección IP pública a su tarea. A continuación, ofrecemos ambos ejemplos.

Ejemplo de uso de una subred privada. La opción `enable-execute-command` es necesaria para utilizar Amazon ECS Exec.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service --
task-definition sample-fargate:1 --desired-count 1 --launch-type "FARGATE" --network-
configuration "awsvpcConfiguration={subnets=[subnet-abcd1234],securityGroups=[sg-
abcd1234]}" --enable-execute-command
```

Ejemplo de uso de una subred pública.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service --
task-definition sample-fargate:1 --desired-count 1 --launch-type "FARGATE" --network-
configuration "awsvpcConfiguration={subnets=[subnet-abcd1234],securityGroups=[sg-
abcd1234],assignPublicIp=ENABLED}"
```

El comando create-service devuelve una descripción de la definición de tarea después de realizar su registro.

Paso 5: Mostrar la lista de los servicios

Enumere los servicios de su clúster. Debe aparecer el servicio que ha creado en la sección anterior. Puede tomar el nombre de servicio o el ARN completo que se devuelve desde este comando y utilizarlo para describir el servicio más adelante.

```
aws ecs list-services --cluster fargate-cluster
```

Salida:

```
{
    "serviceArns": [
        "arn:aws:ecs:region:aws_account_id:service/fargate-cluster/fargate-service"
    ]
}
```

Paso 6: Describir el servicio en ejecución

Describa el servicio utilizado el nombre del servicio recuperado antes para obtener información adicional sobre la tarea.

```
aws ecs describe-services --cluster fargate-cluster --services fargate-service
```

Si tiene éxito, devolverá una descripción de los errores del servicio y los servicios. Por ejemplo, en la sección `services`, encontrará información sobre implementaciones, como el estado de las tareas en ejecución o pendientes. También puede encontrar información sobre la definición de tarea, la configuración de la red y los eventos con marca temporal. En la sección de errores, encontrará información sobre los errores, si los hay, asociados a la llamada. Para solucionar problemas, consulte Mensajes de eventos de servicio. Para obtener más información acerca de la descripción del servicio, consulte Describir servicios.

```
{
    "services": [
        {
            "networkConfiguration": {
                "awsvpcConfiguration": {
                    "subnets": [
                        "subnet-abcd1234"
                    ],
                    "securityGroups": [
                        "sg-abcd1234"
                    ],
                    "assignPublicIp": "ENABLED"
                }
            },
            "launchType": "FARGATE",
            "enableECSManagedTags": false,
            "loadBalancers": [],
            "deploymentController": {
                "type": "ECS"
            },
            "desiredCount": 1,
            "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster",
            "serviceArn": "arn:aws:ecs:region:aws_account_id:service/fargate-service",
            "deploymentConfiguration": {
                "maximumPercent": 200,
                "minimumHealthyPercent": 100
```

```
            },
            "createdAt": 1692283199.771,
            "schedulingStrategy": "REPLICA",
            "placementConstraints": [],
            "deployments": [
                {
                    "status": "PRIMARY",
                    "networkConfiguration": {
                        "awsvpcConfiguration": {
                            "subnets": [
                                "subnet-abcd1234"
                            ],
                            "securityGroups": [
                                "sg-abcd1234"
                            ],
                            "assignPublicIp": "ENABLED"
                        }
                    },
                    "pendingCount": 0,
                    "launchType": "FARGATE",
                    "createdAt": 1692283199.771,
                    "desiredCount": 1,
                    "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-
definition/sample-fargate:1",
                    "updatedAt": 1692283199.771,
                    "platformVersion": "1.4.0",
                    "id": "ecs-svc/9223370526043414679",
                    "runningCount": 0
                }
            ],
            "serviceName": "fargate-service",
            "events": [
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 53c0de40-ea3b-489f-a352-623bf1235f08) (task d0aec985-901b-488f-9fb4-61b991b332a3).",
                    "id": "92b8443e-67fb-4886-880c-07e73383ea83",
                    "createdAt": 1510811841.408
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 b4911bee-7203-4113-99d4-e89ba457c626) (task cc5853e3-6e2d-4678-8312-74f8a7d76474).",
                    "id": "d85c6ec6-a693-43b3-904a-a997e1fc844d",
                    "createdAt": 1510811601.938
                },
```

```
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 cba86182-52bf-42d7-9df8-b744699e6cfc) (task f4c1ad74-a5c6-4620-90cf-2aff118df5fc).",
                    "id": "095703e1-0ca3-4379-a7c8-c0f1b8b95ace",
                    "createdAt": 1510811364.691
                }
            ],
            "runningCount": 0,
            "status": "ACTIVE",
            "serviceRegistries": [],
            "pendingCount": 0,
            "createdBy": "arn:aws:iam::aws_account_id:user/user_name",
            "platformVersion": "LATEST",
            "placementStrategy": [],
            "propagateTags": "NONE",
            "roleArn": "arn:aws:iam::aws_account_id:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
            "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-definition/
sample-fargate:1"
        }
    ],
    "failures": []
}
```

Paso 7: Prueba

Prueba de tarea implementada mediante una subred pública

Describa la tarea del servicio para que pueda obtener la interfaz de red elástica (ENI) para la tarea.

Primero, obtenga el ARN de la tarea.

```
aws ecs list-tasks --cluster fargate-cluster --service fargate-service
```

El resultado contiene el ARN de la tarea.

```
{
    "taskArns": [
        "arn:aws:ecs:us-east-1:123456789012:task/fargate-service/EXAMPLE
    ]
}
```

Describa la tarea y busque el ID de ENI. Utilice el ARN de tareas para el parámetro `tasks`.

```
aws ecs describe-tasks --cluster fargate-cluster --tasks arn:aws:ecs:us-
east-1:123456789012:task/service/EXAMPLE
```

La información de datos adjuntos se muestra en la salida.

```
{
    "tasks": [
        {
            "attachments": [
                {
                    "id": "d9e7735a-16aa-4128-bc7a-b2d5115029e9",
                    "type": "ElasticNetworkInterface",
                    "status": "ATTACHED",
                    "details": [
                        {
                            "name": "subnetId",
                            "value": "subnetabcd1234"
                        },
                        {
                            "name": "networkInterfaceId",
                            "value": "eni-0fa40520aeEXAMPLE"
                        },
                    ]
                }
…
}
```

Describa el ENI para obtener la dirección IP pública.

```
aws ec2 describe-network-interfaces --network-interface-id  eni-0fa40520aeEXAMPLE
```

La dirección IP pública se encuentra en la salida.

```
{
    "NetworkInterfaces": [
        {
            "Association": {
                "IpOwnerId": "amazon",
                "PublicDnsName": "ec2-34-229-42-222.compute-1.amazonaws.com",
                "PublicIp": "198.51.100.2"
```

```
            },
…
}
```

Ingrese dicha dirección IP pública en el navegador web. Debe ver una página web en la que se muestre la aplicación Amazon ECS de muestra.

Prueba de tarea implementada mediante una subred privada

Describa la tarea y ubique los `managedAgents` para comprobar que el `ExecuteCommandAgent` se está ejecutando. Tenga en cuenta la `privateIPv4Address` para un uso posterior.

```
aws ecs describe-tasks --cluster fargate-cluster --tasks arn:aws:ecs:us-
east-1:123456789012:task/fargate-service/EXAMPLE
```

La información administrada del agente se enumera en la salida.

```
{
     "tasks": [
        {
            "attachments": [
                {
                    "id": "d9e7735a-16aa-4128-bc7a-b2d5115029e9",
                    "type": "ElasticNetworkInterface",
                    "status": "ATTACHED",
                    "details": [
                        {
                            "name": "subnetId",
                            "value": "subnetabcd1234"
                        },
                        {
                            "name": "networkInterfaceId",
                            "value": "eni-0fa40520aeEXAMPLE"
                        },
                        {
                            "name": "privateIPv4Address",
                            "value": "10.0.143.156"
                        }
                    ]
                }
            ],
     ...
```

```
     "containers": [
         {
         ...
        "managedAgents": [
                        {
                            "lastStartedAt": "2023-08-01T16:10:13.002000+00:00",
                            "name": "ExecuteCommandAgent",
                            "lastStatus": "RUNNING"
                        }
                ],
        ...
    }
```

Después de comprobar que el `ExecuteCommandAgent` se está ejecutando, puede ejecutar el siguiente comando para ejecutar un intérprete de comandos interactivo en el contenedor de la tarea.

```
aws ecs execute-command --cluster fargate-cluster \
    --task  arn:aws:ecs:us-east-1:123456789012:task/fargate-service/EXAMPLE  \
    --container  fargate-app \
    --interactive \
    --command "/bin/sh"
```

Después de ejecutar el intérprete de comandos interactivo, ejecute los siguientes comandos para instalar la cURL.

```
apt update
```

```
apt install curl
```

Tras instalar la cURL, ejecute el siguiente comando con la dirección IP privada que obtuvo anteriormente.

```
curl 10.0.143.156
```

Debería ver el equivalente en HTML de la página web de la aplicación de ejemplo de Amazon ECS.

```
<html>
    <head>
     <title>Amazon ECS Sample App</title>
     <style>body {margin-top: 40px; background-color: #333;} </style>
```

```
    </head>
      <body>
      <div style=color:white;text-align:center>
      <h1>Amazon ECS Sample App</h1>
      <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p>
      </div>
      </body>
</html>
```

Paso 8: eliminación

Cuando termine este tutorial, debe limpiar los recursos asociados para evitar incurrir en cargos generados por recursos sin utilizar.

Elimine el servicio.

```
aws ecs delete-service --cluster fargate-cluster --service fargate-service --force
```

Eliminar el clúster.

```
aws ecs delete-cluster --cluster fargate-cluster
```

Creación de una tarea de Amazon ECS de Windows para el tipo de lanzamiento de Fargate con la AWS CLI

Los siguientes pasos le ayudan a configurar un clúster, registrar una definición de tarea, ejecutar una tarea de Windows y llevar a cabo otras actividades comunes en Amazon ECS con la AWS CLI. Asegúrese de que utiliza la versión más reciente de la AWS CLI. Para obtener más información acerca de cómo actualizar a la versión más reciente, consulte Instalación o actualización de la versión más reciente de la AWS CLI.

> **Note**
>
> Puede utilizar puntos de conexión de servicio de doble pila para interactuar con Amazon ECS desde la AWS CLI, los SDK y la API de Amazon ECS a través de IPv4 e IPv6. Para obtener más información, consulte Uso de puntos de conexión de doble pila en Amazon ECS.

Temas

- Requisitos previos
- Paso 1: Crear un clúster
- Paso 2: Registrar una definición de tareas de Windows
- Paso 3: Mostrar la lista de definiciones de tareas
- Paso 4: Crear un servicio
- Paso 5: Mostrar la lista de los servicios
- Paso 6: Describir el servicio en ejecución
- Paso 7: Eliminación

Requisitos previos

En este tutorial se supone que los siguientes requisitos previos se han realizado.

- La última versión de la AWS CLI está instalada y configurada. Para obtener más información acerca de cómo instalar la AWS CLI o actualizarla, consulte Instalación o actualización de la versión más reciente de la AWS CLI.
- Se han completado los pasos que se indican en Configuración para utilizar Amazon ECS.
- Su usuario de IAM dispone de los permisos requeridos que se especifican en la política de IAM AmazonECS_FullAccess de ejemplo.
- Tiene una VPC y un grupo de seguridad creados para utilizarlos. En este tutorial se utiliza una imagen de contenedor alojada en Docker Hub por lo que su tarea debe tener acceso a Internet. Para que su tarea tenga una ruta a Internet, utilice una de las siguientes opciones.
  - Utilice una subred privada con una gateway NAT que tenga una dirección IP elástica.
  - Utilice una subred pública y asigne una dirección IP pública a la tarea.

  Para obtener más información, consulte the section called “Creación de una nube virtual privada”.

  Para obtener información sobre las reglas y los grupos de seguridad, consulte Default security groups for your VPCs (Grupos de seguridad predeterminados para sus VPC) y Example rules (Reglas de ejemplo) en la Guía del usuario de Amazon Virtual Private Cloud.
- (Opcional) AWS CloudShell es una herramienta que proporciona a los clientes una línea de comandos sin necesidad de crear su propia instancia de EC2. Para obtener más información, consulte ¿Qué es AWS CloudShell? en la Guía del usuario de AWS CloudShell.

Paso 1: Crear un clúster

De forma predeterminada, la cuenta recibe un clúster `default`.

> **Note**
>
> El beneficio de utilizar el clúster `default` que se le facilita es que no tiene que especificar la opción `--cluster` *`cluster_name`* en los comandos siguientes. Si crea su propio clúster no predeterminado, tiene que especificar `--cluster` *`cluster_name`* para cada comando que pretenda utilizar con dicho clúster.

Cree su propio clúster con un nombre único con el comando siguiente:

```
aws ecs create-cluster --cluster-name fargate-cluster
```

Salida:

```
{
    "cluster": {
        "status": "ACTIVE",
        "statistics": [],
        "clusterName": "fargate-cluster",
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster"
    }
}
```

Paso 2: Registrar una definición de tareas de Windows

Antes de poder ejecutar una tarea de Windows en su clúster de Amazon ECS, debe registrar una definición de tareas. Las definiciones de tareas son listas de contenedores agrupadas. El siguiente ejemplo es una sencilla definición de tarea que crea una aplicación web. Para obtener más información acerca de los parámetros de definición de tareas disponibles, consulte Definiciones de tareas de Amazon ECS.

```
{
```

```
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

El ejemplo anterior de JSON se puede transferir a la AWS CLI de dos formas: puede guardar la definición de tareas JSON como un archivo y transferirlo con la opción `--cli-input-json file://`*`path_to_file.json`*.

Para utilizar un archivo JSON para definiciones de contenedor:

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/fargate-task.json
```

El comando register-task-definition devuelve una descripción de la definición de tarea después de realizar su registro.

Paso 3: Mostrar la lista de definiciones de tareas

Puede enumerar las definiciones de tareas para su cuenta en cualquier momento con el comando list-task-definitions. La salida de este comando muestra los valores `family` y `revision`, que puede utilizar conjuntamente al llamar a run-task o start-task.

```
aws ecs list-task-definitions
```

Salida:

```
{
    "taskDefinitionArns": [
        "arn:aws:ecs:region:aws_account_id:task-definition/sample-fargate-windows:1"
    ]
}
```

Paso 4: Crear un servicio

Una vez que se haya registrado una tarea para la cuenta, puede crear un servicio para la tarea registradas en su clúster. En este ejemplo, se crea un servicio con una instancia de la definición de tarea `sample-fargate:1` que se ejecuta en el clúster. La tarea requiere una ruta a Internet, por lo que hay dos maneras de lograr esto. Una forma es utilizar una subred privada configurada con una gateway NAT con una dirección IP elástica en una subred pública. Otra forma es utilizar una subred pública y asignar una dirección IP pública a su tarea. A continuación, ofrecemos ambos ejemplos.

Ejemplo de uso de una subred privada.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service
 --task-definition sample-fargate-windows:1 --desired-count 1 --launch-type
 "FARGATE" --network-configuration "awsvpcConfiguration={subnets=[subnet-
abcd1234],securityGroups=[sg-abcd1234]}"
```

Ejemplo de uso de una subred pública.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service
 --task-definition sample-fargate-windows:1 --desired-count 1 --launch-type
```

```
 "FARGATE" --network-configuration "awsvpcConfiguration={subnets=[subnet-
abcd1234],securityGroups=[sg-abcd1234],assignPublicIp=ENABLED}"
```

El comando create-service devuelve una descripción de la definición de tarea después de realizar su registro.

Paso 5: Mostrar la lista de los servicios

Enumere los servicios de su clúster. Debe aparecer el servicio que ha creado en la sección anterior. Puede tomar el nombre de servicio o el ARN completo que se devuelve desde este comando y utilizarlo para describir el servicio más adelante.

```
aws ecs list-services --cluster fargate-cluster
```

Salida:

```
{
    "serviceArns": [
        "arn:aws:ecs:region:aws_account_id:service/fargate-service"
    ]
}
```

Paso 6: Describir el servicio en ejecución

Describa el servicio utilizado el nombre del servicio recuperado antes para obtener información adicional sobre la tarea.

```
aws ecs describe-services --cluster fargate-cluster --services fargate-service
```

Si tiene éxito, devolverá una descripción de los errores del servicio y los servicios. Por ejemplo, en la sección de servicios, encontrará información sobre implementaciones, como el estado de las tareas en ejecución o pendientes. También puede encontrar información sobre la definición de tarea, la configuración de la red y los eventos con marca temporal. En la sección de errores, encontrará información sobre los errores, si los hay, asociados a la llamada. Para solucionar problemas, consulte Mensajes de eventos de servicio. Para obtener más información acerca de la descripción del servicio, consulte Describir servicios.

```
{
    "services": [
        {
```

```
            "status": "ACTIVE",
            "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-definition/
sample-fargate-windows:1",
            "pendingCount": 2,
            "launchType": "FARGATE",
            "loadBalancers": [],
            "roleArn": "arn:aws:iam::aws_account_id:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
            "placementConstraints": [],
            "createdAt": 1510811361.128,
            "desiredCount": 2,
            "networkConfiguration": {
                "awsvpcConfiguration": {
                    "subnets": [
                        "subnet-abcd1234"
                    ],
                    "securityGroups": [
                        "sg-abcd1234"
                    ],
                    "assignPublicIp": "DISABLED"
                }
            },
            "platformVersion": "LATEST",
            "serviceName": "fargate-service",
            "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster",
            "serviceArn": "arn:aws:ecs:region:aws_account_id:service/fargate-service",
            "deploymentConfiguration": {
                "maximumPercent": 200,
                "minimumHealthyPercent": 100
            },
            "deployments": [
                {
                    "status": "PRIMARY",
                    "networkConfiguration": {
                        "awsvpcConfiguration": {
                            "subnets": [
                                "subnet-abcd1234"
                            ],
                            "securityGroups": [
                                "sg-abcd1234"
                            ],
                            "assignPublicIp": "DISABLED"
                        }
                    },
```

```
                    "pendingCount": 2,
                    "launchType": "FARGATE",
                    "createdAt": 1510811361.128,
                    "desiredCount": 2,
                    "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-
definition/sample-fargate-windows:1",
                    "updatedAt": 1510811361.128,
                    "platformVersion": "0.0.1",
                    "id": "ecs-svc/9223370526043414679",
                    "runningCount": 0
                }
            ],
            "events": [
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 53c0de40-ea3b-489f-a352-623bf1235f08) (task d0aec985-901b-488f-9fb4-61b991b332a3).",
                    "id": "92b8443e-67fb-4886-880c-07e73383ea83",
                    "createdAt": 1510811841.408
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 b4911bee-7203-4113-99d4-e89ba457c626) (task cc5853e3-6e2d-4678-8312-74f8a7d76474).",
                    "id": "d85c6ec6-a693-43b3-904a-a997e1fc844d",
                    "createdAt": 1510811601.938
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 cba86182-52bf-42d7-9df8-b744699e6cfc) (task f4c1ad74-a5c6-4620-90cf-2aff118df5fc).",
                    "id": "095703e1-0ca3-4379-a7c8-c0f1b8b95ace",
                    "createdAt": 1510811364.691
                }
            ],
            "runningCount": 0,
            "placementStrategy": []
        }
    ],
    "failures": []
}
```

Paso 7: Eliminación

Cuando termine este tutorial, debe limpiar los recursos asociados para evitar incurrir en cargos generados por recursos sin utilizar.

Elimine el servicio.

```
aws ecs delete-service --cluster fargate-cluster --service fargate-service --force
```

Eliminar el clúster.

```
aws ecs delete-cluster --cluster fargate-cluster
```

Creación de una tarea de Amazon ECS para el tipo de lanzamiento de EC2 con la AWS CLI

Los siguientes pasos le ayudan a configurar un clúster, registrar una definición de tarea, ejecutar una tarea y llevar a cabo otros escenarios comunes en Amazon ECS con la AWS CLI. Utilice la versión más reciente de la AWS CLI. Para obtener más información acerca de cómo actualizar a la versión más reciente, consulte Instalación o actualización de la versión más reciente de la AWS CLI.

> **Note**
>
> Puede utilizar puntos de conexión de servicio de doble pila para interactuar con Amazon ECS desde la AWS CLI, los SDK y la API de Amazon ECS a través de IPv4 e IPv6. Para obtener más información, consulte Uso de puntos de conexión de doble pila en Amazon ECS.

Temas

- Requisitos previos
- Creación de un clúster
- Lanzamiento de una instancia de contenedor de AMI para Amazon ECS
- Enumeración de instancias de contenedor
- Descripción de la instancia de contenedor
- Registro de una definición de tareas
- Enumeración de definiciones de tareas
- Crear un servicio
- Enumeración de servicios
- Descripción de los servicios

- Descripción de la tarea en ejecución
- Prueba del servidor web
- Eliminar recursos

Requisitos previos

En este tutorial se supone que los siguientes requisitos previos se han completado:

- La última versión de la AWS CLI está instalada y configurada. Para obtener más información acerca de cómo instalar la AWS CLI o actualizarla, consulte Instalación o actualización de la versión más reciente de la AWS CLI.
- Se han completado los pasos que se indican en Configuración para utilizar Amazon ECS.
- Su usuario de IAM dispone de los permisos requeridos que se especifican en la política de IAM AmazonECS_FullAccess de ejemplo.
- Se ha creado un rol de IAM de instancia de contenedor para usarlo. Para obtener más información, consulte Rol de IAM de instancia de contenedor de Amazon ECS.
- Se ha creado una VPC para usarla. Para obtener más información, consulte the section called “Creación de una nube virtual privada”.
- (Opcional) AWS CloudShell es una herramienta que proporciona a los clientes una línea de comandos sin necesidad de crear su propia instancia de EC2. Para obtener más información, consulte ¿Qué es AWS CloudShell? en la Guía del usuario de AWS CloudShell.

Creación de un clúster

De forma predeterminada, su cuenta recibe un clúster `default` donde lanzar su primera instancia de contenedor.

> **Note**
>
> El beneficio de utilizar el clúster `default` que se le facilita es que no tiene que especificar la opción `--cluster` *`cluster_name`* en los comandos siguientes. Si crea su propio clúster no predeterminado, tiene que especificar `--cluster` *`cluster_name`* para cada comando que pretenda utilizar con dicho clúster.

Cree su propio clúster con un nombre único con el comando siguiente:

```
aws ecs create-cluster --cluster-name MyCluster
```

Salida:

```
{
    "cluster": {
        "clusterName": "MyCluster",
        "status": "ACTIVE",
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/MyCluster"
    }
}
```

Lanzamiento de una instancia de contenedor de AMI para Amazon ECS

Las instancias de contenedor son instancias de EC2 que ejecutan el agente de contenedor de Amazon ECS y que se han registrado en un clúster. En esta sección, lanzará una instancia EC2 con la AMI optimizada para ECS.

Lanzamiento de una instancia de contenedor con la AWS CLI

1. Recupere el último ID de AMI de Amazon Linux 2 optimizado para ECS para su Región de AWS mediante el siguiente comando. Este comando utiliza el almacén de parámetros AWS Systems Manager para obtener el último ID de AMI optimizado para ECS. La AMI incluye el agente de contenedor de Amazon ECS y el tiempo de ejecución de Docker preinstalados.

   ```
   aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
   recommended --query 'Parameters[0].Value' --output text | jq -r '.image_id'
   ```

   Salida:

   ```
   ami-abcd1234
   ```

2. Cree un grupo de seguridad que permita el acceso SSH para administrar la instancia de contenedor y el acceso HTTP para el servidor web.

   ```
   aws ec2 create-security-group --group-name ecs-tutorial-sg --description "ECS
    tutorial security group"
   ```

   Salida:

```
{
    "GroupId": "sg-abcd1234"
}
```

3. Agregue una regla de entrada al grupo de seguridad mediante la ejecución del siguiente comando.

```
aws ec2 authorize-security-group-ingress --group-id sg-abcd1234 --protocol tcp --
port 80 --cidr 0.0.0.0/0
```

Salida:

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-efgh5678",
            "GroupId": "sg-abcd1234",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
```

El grupo de seguridad ahora permite el acceso SSH desde el rango de IP especificado y el acceso HTTP desde cualquier lugar. En un entorno de producción, debe restringir el acceso SSH a su dirección IP específica y plantearse la limitación del acceso HTTP según sea necesario.

4. Cree un par de claves EC2 para el acceso SSH a su instancia de contenedor.

```
aws ec2 create-key-pair --key-name ecs-tutorial-key --query 'KeyMaterial' --output
 text > ecs-tutorial-key.pem
chmod 400 ecs-tutorial-key.pem
```

La clave privada se guarda en la máquina local con los permisos adecuados para el acceso SSH.

5. Lance una instancia EC2 mediante la AMI optimizada para ECS y configúrela para que se una a su clúster.

```
aws ec2 run-instances --image-id ami-abcd1234 --instance-type t3.micro --key-
name ecs-tutorial-key --security-group-ids sg-abcd1234 --iam-instance-profile
 Name=ecsInstanceRole --user-data '#!/bin/bash
echo ECS_CLUSTER=MyCluster >> /etc/ecs/ecs.config'
{
    "Instances": [
        {
            "InstanceId": "i-abcd1234",
            "ImageId": "ami-abcd1234",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "PrivateDnsName": "",
            "PublicDnsName": "",
            "StateReason": {
                "Code": "pending",
                "Message": "pending"
            },
            "InstanceType": "t3.micro",
            "KeyName": "ecs-tutorial-key",
            "LaunchTime": "2025-01-13T10:30:00.000Z"
        }
    ]
}
```

El script de datos de usuario configura el agente de Amazon ECS para que registre la instancia en su `MyCluster`. La instancia usa el rol de IAM `ecsInstanceRole`, que proporciona los permisos necesarios para el agente.

Enumeración de instancias de contenedor

A los pocos minutos de lanzar la instancia de contenedor, el agente de Amazon ECS registra la instancia con su clúster predeterminado. Puede mostrar la lista de las instancias de contenedor de un clúster ejecutando el comando siguiente:

```
aws ecs list-container-instances --cluster default
```

Salida:

```
{
    "containerInstanceArns": [
        "arn:aws:ecs:us-east-1:aws_account_id:container-instance/container_instance_ID"
    ]
}
```

Descripción de la instancia de contenedor

Una vez que tenga el ARN o el ID de una instancia de contenedor, puede utilizar el comando describe-container-instances para obtener información valiosa sobre la instancia, como por ejemplo, los recursos registrados y restantes de CPU y de memoria.

```
aws ecs describe-container-instances --cluster MyCluster --container-
instances container_instance_ID
```

Salida:

```
{
    "failures": [],
    "containerInstances": [
        {
            "status": "ACTIVE",
            "registeredResources": [
                {
                    "integerValue": 1024,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "CPU",
                    "doubleValue": 0.0
                },
                {
```

```
                    "integerValue": 995,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "MEMORY",
                    "doubleValue": 0.0
                },
                {
                    "name": "PORTS",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [
                        "22",
                        "2376",
                        "2375",
                        "51678"
                    ],
                    "type": "STRINGSET",
                    "integerValue": 0
                },
                {
                    "name": "PORTS_UDP",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [],
                    "type": "STRINGSET",
                    "integerValue": 0
                }
            ],
            "ec2InstanceId": "instance_id",
            "agentConnected": true,
            "containerInstanceArn": "arn:aws:ecs:us-west-2:aws_account_id:container-
instance/container_instance_ID",
            "pendingTasksCount": 0,
            "remainingResources": [
                {
                    "integerValue": 1024,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "CPU",
                    "doubleValue": 0.0
                },
                {
                    "integerValue": 995,
                    "longValue": 0,
```

```
                "type": "INTEGER",
                "name": "MEMORY",
                "doubleValue": 0.0
            },
            {
                "name": "PORTS",
                "longValue": 0,
                "doubleValue": 0.0,
                "stringSetValue": [
                    "22",
                    "2376",
                    "2375",
                    "51678"
                ],
                "type": "STRINGSET",
                "integerValue": 0
            },
            {
                "name": "PORTS_UDP",
                "longValue": 0,
                "doubleValue": 0.0,
                "stringSetValue": [],
                "type": "STRINGSET",
                "integerValue": 0
            }
        ],
        "runningTasksCount": 0,
        "attributes": [
            {
                "name": "com.amazonaws.ecs.capability.privileged-container"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
            },
            {
                "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
            },
            {
                "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
            },
            {
```

```
                    "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
                }
            ],
            "versionInfo": {
                "agentVersion": "1.5.0",
                "agentHash": "b197edd",
                "dockerVersion": "DockerVersion: 1.7.1"
            }
        }
    ]
}
```

También puede buscar el ID de instancia de Amazon EC2 que puede utilizar para monitorizar la instancia en la consola de Amazon EC2 o con el comando aws ec2 describe-instances --instance-id ***instance_id***.

Registro de una definición de tareas

Antes de poder ejecutar una tarea en su clúster de Amazon ECS, debe registrar una definición de tareas. Las definiciones de tareas son listas de contenedores agrupadas. El ejemplo siguiente es una definición de tareas sencilla que utiliza una imagen `nginx`. Para obtener más información acerca de los parámetros de definición de tareas disponibles, consulte Definiciones de tareas de Amazon ECS.

```
{
    "family": "nginx-task",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/ecs-sample-image/amazon-ecs-sample:latest",
            "cpu": 256,
            "memory": 512,
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "requiresCompatibilities": ["EC2"],
```

```
    "networkMode": "bridge"
}
```

El ejemplo anterior de JSON se puede transferir a la AWS CLI de dos formas: puede guardar la definición de tareas JSON como un archivo y transferirlo con la opción `--cli-input-json file://path_to_file.json`. O bien, puede eludir las comillas en JSON y transferir las definiciones de contenedor JSON en la línea de comandos. Si elige transferir las definiciones de contenedor en la línea de comandos, el comando requiere adicionalmente un parámetro `--family` que se utiliza para mantener varias versiones de la definición de tareas asociadas entre sí.

Para utilizar un archivo JSON para definiciones de contenedor:

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/nginx.json
```

register-task-definition devuelve una descripción de la definición de tarea después de completar su registro.

```
{
    "taskDefinition": {
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/nginx-
task:1",
        "family": "nginx-task",
        "revision": 1,
        "status": "ACTIVE",
        "containerDefinitions": [
            {
                "name": "nginx",
                "image": "public.ecr.aws/docker/library/nginx:latest",
                "cpu": 256,
                "memory": 512,
                "essential": true,
                "portMappings": [
                    {
                        "containerPort": 80,
                        "hostPort": 80,
                        "protocol": "tcp"
                    }
                ],
                "environment": [],
                "mountPoints": [],
                "volumesFrom": []
            }
```

```
        ],
        "volumes": [],
        "networkMode": "bridge",
        "compatibilities": [
            "EC2"
        ],
        "requiresCompatibilities": [
            "EC2"
        ]
    }
}
```

Enumeración de definiciones de tareas

Puede enumerar las definiciones de tareas para su cuenta en cualquier momento con el comando list-task-definitions. La salida de este comando muestra los valores `family` y `revision`, que se pueden utilizar conjuntamente al llamar a create-service.

```
aws ecs list-task-definitions
```

Salida:

```
{
    "taskDefinitionArns": [
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/sleep360:1",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/sleep360:2",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/nginx-task:1",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:3",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:4",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:5",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:6"
    ]
}
```

Crear un servicio

Después de haber registrado una tarea para la cuenta y haber lanzado una instancia de contenedor que esté registrada en su clúster, puede crear un servicio de Amazon ECS que ejecute y mantenga el número deseado de tareas simultáneamente mediante la definición de tarea que se haya registrado. Para este ejemplo, coloque una instancia única de la definición de tarea `nginx:1` en su clúster predeterminado.

```
aws ecs create-service --cluster default --service-name nginx-service --task-
definition nginx-task:1 --desired-count 1
```

Salida:

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:aws_account_id:service/MyCluster/nginx-
service",
        "serviceName": "nginx-service",
        "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/MyCluster",
        "taskDefinition": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/nginx-
task:1",
        "desiredCount": 1,
        "runningCount": 0,
        "pendingCount": 0,
        "launchType": "EC2",
        "status": "ACTIVE",
        "createdAt": "2025-01-13T10:45:00.000Z"
    }
}
```

Enumeración de servicios

Enumere los servicios de su clúster. Debe aparecer el servicio que ha creado en la sección anterior. Puede tomar nota del ID de servicio o el ARN completo que se devuelve desde este comando y utilizarlo para describir el servicio más adelante.

```
aws ecs list-services --cluster MyCluster
```

Salida:

```
{
    "taskArns": [
        "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID"
    ]
}
```

Descripción de los servicios

Describa el servicio con el siguiente comando para obtener más información sobre el servicio.

```
aws ecs describe-services --cluster MyCluster --services nginx-service
```

Salida:

```
{
    "services": [
        {
            "serviceArn": "arn:aws:ecs:us-east-1:aws_account_id:service/MyCluster/
nginx-service",
            "serviceName": "nginx-service",
            "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/MyCluster",
            "taskDefinition": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/
nginx-task:1",
            "desiredCount": 1,
            "runningCount": 1,
            "pendingCount": 0,
            "launchType": "EC2",
            "status": "ACTIVE",
            "createdAt": "2025-01-13T10:45:00.000Z",
            "events": [
                {
                    "id": "abcd1234-5678-90ab-cdef-1234567890ab",
                    "createdAt": "2025-01-13T10:45:30.000Z",
                    "message": "(service nginx-service) has started 1 tasks: (task
 abcd1234-5678-90ab-cdef-1234567890ab)."
                }
            ]
        }
    ]
}
```

Descripción de la tarea en ejecución

Después de describir el servicio, ejecute el siguiente comando para obtener más información sobre la tarea que se está ejecutando como parte de su servicio.

```
aws ecs list-tasks --cluster MyCluster --service-name nginx-service
```

Salida:

```
{
```

```
    "tasks": [
        {
            "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/MyCluster/
abcd1234-5678-90ab-cdef-1234567890ab",
            "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/MyCluster",
            "taskDefinitionArn": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/
nginx-task:1",
            "containerInstanceArn": "arn:aws:ecs:us-east-1:aws_account_id:container-
instance/MyCluster/abcd1234-5678-90ab-cdef-1234567890ab",
            "lastStatus": "RUNNING",
            "desiredStatus": "RUNNING",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-east-1:aws_account_id:container/
MyCluster/abcd1234-5678-90ab-cdef-1234567890ab/abcd1234-5678-90ab-cdef-1234567890ab",
                    "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/MyCluster/
abcd1234-5678-90ab-cdef-1234567890ab",
                    "name": "nginx",
                    "lastStatus": "RUNNING",
                    "networkBindings": [
                        {
                            "bindIP": "0.0.0.0",
                            "containerPort": 80,
                            "hostPort": 80,
                            "protocol": "tcp"
                        }
                    ]
                }
            ],
            "createdAt": "2025-01-13T10:45:00.000Z",
            "startedAt": "2025-01-13T10:45:30.000Z"
        }
    ]
}
```

Prueba del servidor web

Prueba del servidor web

1. Obtenga la dirección IP pública de su instancia de contenedor con la ejecución del siguiente comando.

```
aws ec2 describe-instances --instance-ids i-abcd1234 --query
 'Reservations[0].Instances[0].PublicIpAddress' --output text
```

Salida:

```
203.0.113.25
```

2. Después de recuperar la dirección IP, ejecute el siguiente comando `curl` con la dirección IP.

```
curl http://203.0.113.25
```

Salida:

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
...
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you can see this page, the nginx web server is successfully installed and
 working.</p>
...
</body>
</html>
```

La página de bienvenida de nginx confirma que el servicio se está ejecutando correctamente y que es accesible desde Internet.

Eliminar recursos

Para dejar de incurrir en cargos, debe limpiar los recursos que se crearon en este tutorial.

Cómo limpiar los recursos

1. Actualice el servicio para que no tener tarea deseadas y, a continuación, elimínelo.

```
aws ecs update-service --cluster MyCluster --service nginx-service --desired-count
 0
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:123456789012:service/MyCluster/nginx-
service",
        "serviceName": "nginx-service",
        "desiredCount": 0,
        "runningCount": 1,
        "pendingCount": 0,
        "status": "ACTIVE"
    }
}
```

2. Espere a que se detengan las tareas en ejecución y, a continuación, elimine el servicio.

```
aws ecs delete-service --cluster MyCluster --service nginx-service
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:123456789012:service/MyCluster/nginx-
service",
        "serviceName": "nginx-service",
        "status": "DRAINING"
    }
}
```

3. Finalice la instancia de contenedor que creó.

```
aws ec2 terminate-instances --instance-ids i-abcd1234
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-abcd1234",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
```

```
    ]
}
```

4. Limpie el grupo de seguridad y el par de claves que creó.

```
aws ec2 delete-security-group --group-id sg-abcd1234
aws ec2 delete-key-pair --key-name ecs-tutorial-key
rm ecs-tutorial-key.pem
```

5. Elimine el clúster de Amazon ECS.

```
aws ecs delete-cluster --cluster MyCluster
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
        "clusterName": "MyCluster",
        "status": "INACTIVE"
    }
}
```

Configuración de Amazon ECS para escuchar los eventos de Eventos de CloudWatch

Obtenga información sobre cómo configurar una función de Lambda simple que escuche eventos de tareas y los escriba en un flujo de registro de Registros de CloudWatch.

Requisito previo: configurar un clúster de prueba

Si no dispone de un clúster en ejecución del que capturar eventos, siga los pasos en the section called “Creación de un clúster para el tipo de lanzamiento de Fargate” para crear uno. Al final de este tutorial, se ejecuta una tarea en este clúster para comprobar que la función de Lambda se ha configurado correctamente.

Paso 1: Crear la función de Lambda

En este procedimiento, se crea una función de Lambda sencilla que servirá como destino para los mensajes de la secuencia de eventos de Amazon ECS.

1. Abra la consola de AWS Lambda en https://console.aws.amazon.com/lambda/.

2. Elija Crear función.
3. En la pantalla Author from scratch, haga lo siguiente:
   a. Para Name (Nombre), escriba un valor.
   b. En Runtime (Tiempo de ejecución), elija su versión de Python, por ejemplo, Python 3.9.
   c. Para Role (Rol), elija Create a new role with basic Lambda permissions (Crear un nuevo rol con permisos básicos de Lambda)
4. Seleccione Crear función.
5. En la sección Código de función, edite el código de muestra de tal modo que coincida con el siguiente ejemplo:

```
import json

def lambda_handler(event, context):
    if event["source"] != "aws.ecs":
       raise ValueError("Function only supports input from events with a source
 type of: aws.ecs")

    print('Here is the event:')
    print(json.dumps(event))
```

   Se trata de una función Python 3.9 sencilla que imprime el evento que envía Amazon ECS. Si se configura todo correctamente, al final de este tutorial verá los detalles de los eventos que aparecerán en la secuencia de registros de CloudWatch Logs asociada a esta función de Lambda.
6. Seleccione Save.

Paso 2: Registrar una regla de eventos

A continuación, se crea una regla de eventos de CloudWatch Events que captura eventos de tareas procedentes de los clústeres de Amazon ECS. Esta regla captura todos los eventos procedentes de todos los clústeres dentro de la cuenta donde está definido. Los propios mensajes de tareas contienen información acerca del origen de evento, incluido el clúster en el que reside, que puede usar para filtrar y ordenar eventos mediante programación.

> **Note**
>
> Cuando se utiliza la AWS Management Console para crear una regla de eventos, la consola agrega automáticamente los permisos de IAM necesarios para conceder permiso a CloudWatch Events para ejecutar la función de Lambda. Si crea una regla de eventos utilizando la AWS CLI, tiene que otorgar este permiso explícitamente. Para obtener más información, consulte Eventos en Amazon EventBridge y Patrones de eventos de Amazon EventBridge en la Guía del usuario de Amazon EventBridge.

Para dirigir eventos a la función Lambda

1. Abra la consola de CloudWatch en https://console.aws.amazon.com/cloudwatch/.
2. En el panel de navegación, elija Events (Eventos), Rules (Reglas), Create rule (Crear regla).
3. En Event Source, seleccione ECS como el origen de los eventos. De forma predeterminada, la regla se aplica a todos los eventos de Amazon ECS de todos los grupos de Amazon ECS. Como alternativa, puede seleccionar eventos específicos o un grupo de Amazon ECS específico.
4. En Targets (Destinos), elija Add target (Agregar destino); en Target type (Tipo de destino), elija Lambda function (Función de Lambda) y, a continuación, seleccione la función de Lambda.
5. Seleccione Configurar los detalles.
6. Para Rule definition, escriba un nombre y la descripción para su regla y seleccione Create rule.

Paso 3: Cree una definición de tarea

Cree una definición de tarea.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, elija Task Definitions.
3. Elija Create new Task Definition (Crear nueva definición de tarea) y Create new revision with JSON (Crear nueva revisión con JSON).
4. Copie y pegue la siguiente definición de tarea de ejemplo en el cuadro y, a continuación, elija Save (Guardar).

```
{
   "containerDefinitions": [
      {
```

```
        "command": [
           "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</
title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
        ],
        "entryPoint": [
           "sh",
           "-c"
        ],
        "essential": true,
        "image": "public.ecr.aws/docker/library/httpd:2.4",
        "logConfiguration": {
           "logDriver": "awslogs",
           "options": {
              "awslogs-group" : "/ecs/fargate-task-definition",
              "awslogs-region": "us-east-1",
              "awslogs-stream-prefix": "ecs"
           }
        },
        "name": "sample-fargate-app",
        "portMappings": [
           {
              "containerPort": 80,
              "hostPort": 80,
              "protocol": "tcp"
           }
        ]
     }
  ],
  "cpu": "256",
  "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
  "family": "fargate-task-definition",
  "memory": "512",
  "networkMode": "awsvpc",
  "runtimePlatform": {
       "operatingSystemFamily": "LINUX"
   },
  "requiresCompatibilities": [
      "FARGATE"
   ]
```

```
}
```

5. Seleccione Crear.

Paso 4: Probar la regla

Por último, se crea una regla de eventos de CloudWatch Events que captura los eventos de tareas procedentes de los clústeres de Amazon ECS. Esta regla captura todos los eventos procedentes de todos los clústeres dentro de la cuenta donde está definido. Los propios mensajes de tareas contienen información acerca del origen de evento, incluido el clúster en el que reside, que puede usar para filtrar y ordenar eventos mediante programación.

Para probar la regla

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. Elija Task definitions (Definiciones de tareas).
3. Elija console-sample-app-static y, a continuación, elija Deploy(Implementar), Run new task (Ejecutar nueva tarea).
4. En Cluster (Clúster), elija default (predeterminado) y, a continuación, elija Deploy (Implementar).
5. Abra la consola de CloudWatch en https://console.aws.amazon.com/cloudwatch/.
6. En el panel de navegación, elija Logs (Registros) y seleccione el grupo de registros para la función de Lambda (por ejemplo, /aws/lambda/*my-function*).
7. Seleccione una secuencia de registro para ver los datos de los eventos.

Envío de alertas de Amazon Simple Notification Service para eventos de tareas detenidas de Amazon ECS

Configure una regla de eventos de Amazon EventBridge que solo capture eventos de tareas en los que la tarea dejó de ejecutarse porque uno de sus contenedores esenciales finalizó. El evento solo envía eventos de tareas con una propiedad `stoppedReason` específica al tema de Amazon SNS designado.

Requisito previo: configurar un clúster de prueba

Si no dispone de un clúster en ejecución de donde capturar eventos, siga los pasos que se describen en Introducción a la consola mediante contenedores de Linux en AWS Fargate para crear uno. Al

final de este tutorial, ejecutará una tarea en este clúster para comprobar que el tema de Amazon SNS y la regla de EventBridge se han configurado correctamente.

Requisito previo: configurar los permisos para Amazon SNS

Para permitir que EventBridge publique en un tema de Amazon SNS, utilice los comandos aws sns get-topic-attributes y aws sns set-topic-attributes.

Para obtener más información sobre cómo agregar el permiso, consulte Permisos de Amazon SNS en la Guía para desarrolladores de Amazon Simple Notification Service.

Agregue los siguientes permisos:

```
{
  "Sid": "PublishEventsToMyTopic",
  "Effect": "Allow",
  "Principal": {
     "Service": "events.amazonaws.com"
  },
  "Action": "sns: Publish",
  "Resource": "arn:aws:sns:region:account-id:TaskStoppedAlert",
}
```

Paso 1: Crear y suscribirse a un tema de Amazon SNS

Para este tutorial, se configura un tema de Amazon SNS para utilizarse como destino de eventos para la nueva regla de eventos.

Para obtener información acerca de cómo crear un tema de Amazon SNS y suscribirse a él, consulte Introducción a Amazon SNS en la Guía para desarrolladores de Amazon Simple Notification Service y utilice la siguiente tabla para determinar qué opciones seleccionar.

Opción	Valor
Tipo	Estándar
Nombre	TaskStoppedAlert
Protocolo	Correo electrónico

Opción	Valor
Punto de conexión	Una dirección de correo electrónico a la que actualmente tiene acceso

Paso 2: Registrar una regla de eventos

A continuación, registre una regla de eventos que capture solo eventos de tarea parada de tareas con contenedores parados.

Para obtener información sobre cómo crear y suscribirse a un tema de Amazon SNS, consulte Create a rule in Amazon EventBridge (Crear una regla en Amazon EventBridge) en la Guía del usuario de Amazon EventBridge y utilice la siguiente tabla para determinar qué opciones seleccionar.

Opción	Valor
Tipo de regla	Regla con un patrón de evento
Origen del evento	Eventos de AWS o eventos de socios de EventBridge
Patrón del evento	Patrón personalizado (editor JSON)
Patrón del evento	`{ "source":[ "aws.ecs" ], "detail-type":[ "ECS Task State Change" ], "detail":{ "lastStatus":[ "STOPPED" ], "stoppedReason":[`

Opción	Valor
	`"Essentia l container in task exited" ] } }`
Tipo de objetivo	Servicio de AWS
Destino	Tema de SNS
Tema	TaskStopPedAlert (El tema que ha creado en el paso 1)

Paso 3: Pruebe la regla

Compruebe que la regla funciona mediante la ejecución de una tarea que se cierra poco después de iniciarse. Si la regla de eventos está configurada correctamente, recibirá un mensaje de correo electrónico en unos minutos con el texto del evento. Si tiene una definición de tarea existente que puede satisfacer los requisitos de la regla, ejecute una tarea con ella. Si no lo hace, los siguientes pasos le guiarán a través del registro de una definición de tarea de Fargate y su ejecución.

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, elija Task Definitions (Definiciones de tareas).
3. Elija Create new task definition (Crear nueva definición de tarea) y Create new task definition with JSON (Crear nueva definición de tarea con JSON).
4. En el cuadro del editor JSON, edite el archivo JSON y copie lo siguiente en el editor.

```
{
   "containerDefinitions":[
      {
         "command":[
            "sh",
            "-c",
            "sleep 5"
         ],
         "essential":true,
```

```
            "image":"public.ecr.aws/amazonlinux/amazonlinux:latest",
            "name":"test-sleep"
        }
    ],
    "cpu":"256",
    "executionRoleArn":"arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "family":"fargate-task-definition",
    "memory":"512",
    "networkMode":"awsvpc",
    "requiresCompatibilities":[
        "FARGATE"
    ]
}
```

5. Seleccione Crear.

Para ejecutar una tarea a través de la consola

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En la página Clústeres, seleccione el clúster que creó en los requisitos previos.
3. En la pestaña Tasks (Tareas), elija Run new task (Ejecutar nueva tarea).
4. En Application type (Tipo de aplicación), elija Task (Tarea).
5. En Definición de tareas, elija fargate-task-definition.
6. En Desired tasks (Tareas deseadas), ingrese el número de tareas que se lanzarán.
7. Seleccione Crear.

Concatenación de mensajes de registro de Amazon ECS de seguimiento de pila o de varias líneas

Empezando por AWS para Fluent Bit versión 2.22.0, se incluye un filtro multilínea. El filtro multilínea ayuda a concatenar mensajes de registro que originalmente pertenecen a un contexto, pero que se dividieron en varios registros o líneas de registro. Para obtener más información acerca del filtro multilínea, consulte la documentación de Fluent Bit.

Los siguientes son algunos ejemplos comunes de mensajes de registro divididos:

- Seguimiento de pila.

- Aplicaciones que imprimen registros en varias líneas.
- Mensajes de registro que se dividieron porque eran más largos que el tamaño máximo de búfer en tiempo de ejecución especificado. Puede concatenar mensajes de registro divididos por el tiempo de ejecución del contenedor siguiendo el ejemplo de GitHub: Ejemplo de FireLens: Concatenar registros de contenedores parciales/divididos.

Permisos de IAM necesarios

Dispone de los permisos de IAM necesarios para que el agente de contenedor extraiga las imágenes de contenedor de Amazon ECR y para que el contenedor dirija los registros a Registros de CloudWatch.

Para estos permisos, debe tener los siguientes roles:

- Un rol de IAM de tareas.
- Un rol de IAM de ejecución de tareas.

Utilización del editor de política de JSON para la creación de una política

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, elija Políticas.

   Si es la primera vez que elige Políticas, aparecerá la página Welcome to Managed Policies (Bienvenido a políticas administradas). Elija Comenzar.
3. En la parte superior de la página, seleccione Crear política.
4. En la sección Editor de políticas, seleccione la opción JSON.
5. Ingrese el siguiente documento de política JSON:

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": [
   "logs:CreateLogStream",
   "logs:CreateLogGroup",
   "logs:PutLogEvents"
```

```
  ],
  "Resource": "*"
 }]
}
```

6. Elija Siguiente.

> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de política en la Guía del usuario de IAM.

7. En la página Revisar y crear, introduzca el Nombre de la política y la Descripción (opcional) para la política que está creando. Revise los Permisos definidos en esta política para ver los permisos que concede la política.
8. Elija Crear política para guardar la nueva política.

Determine cuándo utilizar la configuración de registro multilínea

En los siguientes ejemplos, se muestran fragmentos de registro de ejemplo que se ven en la consola de Registros de CloudWatch con la configuración de registro predeterminada. Puede fijarse en la línea que empieza por `log` para determinar si necesita el filtro multilínea. Cuando el contexto es el mismo, puede utilizar la configuración de registro de varias líneas. En este ejemplo, el contexto es “com.myproject.model.MyProject”.

```
2022-09-20T15:47:56:595-05-00                           {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-
app", "source=": "stdout", "log": ": "     at com.myproject.modele.
(MyProject.badMethod.java:22)",
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
      "log": ": "     at com.myproject.model.MyProject.badMethod(MyProject.java:22)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE",
```

```
      "ecs_task_definition": "firelense-example-multiline:3"
     }
```

```
2022-09-20T15:47:56:595-05-00                            {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-app", "stdout",
 "log": ": "     at com.myproject.modele.(MyProject.oneMoreMethod.java:18)",
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
      "log": ": "     at
 com.myproject.model.MyProject.oneMoreMethod(MyProject.java:18)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE,
      "ecs_task_definition": "firelense-example-multiline:3"
     }
```

Después de utilizar la configuración del registro multilínea, la salida tendrá un aspecto similar al del ejemplo siguiente.

```
2022-09-20T15:47:56:595-05-00                            {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-app",
 "stdout",...
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
      "log:    "September 20, 2022 06:41:48 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!\n
    at com.myproject.module.MyProject.badMethod(MyProject.java:22)\n    at
    at com.myproject.model.MyProject.oneMoreMethod(MyProject.java:18)
 com.myproject.module.MyProject.main(MyProject.java:6)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE",
      "ecs_task_definition": "firelense-example-multiline:2"
     }
```

Opciones de análisis y concatenación

Para analizar registros y concatenar líneas que se dividieron debido a las líneas nuevas, puede utilizar cualquiera de estas dos opciones.

- Utilice su propio archivo analizador que contenga las reglas para analizar y concatenar líneas que pertenecen al mismo mensaje.
- Utilizar un analizador integrado de Fluent Bit. Para ver una lista de los idiomas admitidos en los analizadores integrados de Fluent Bit, consulte la documentación de Fluent Bit.

El siguiente tutorial lo guiará por los pasos de cada caso de uso. En los pasos, se muestra cómo concatenar varias líneas y enviar los registros a Amazon CloudWatch. Puede especificar otro destino para los registros.

Ejemplo: utilizar un analizador que cree

En este ejemplo, completará los siguientes pasos:

1. Cree y cargue la imagen en un contenedor de Fluent Bit.
2. Cree y cargue la imagen de una aplicación multilínea de demostración que se ejecuta, falla y genera un seguimiento de pila multilínea.
3. Cree la definición de tarea y ejecute la tarea.
4. Consulte los registros para verificar que los mensajes que abarcan varias líneas aparezcan concatenados.

Crear y cargar la imagen en un contenedor de Fluent Bit

En esta imagen, se incluirá el archivo analizador en el que se especifica la expresión regular y un archivo de configuración que hace referencia al archivo analizador.

1. Cree una carpeta con el nombre `FluentBitDockerImage`.
2. Dentro de la carpeta, cree un archivo analizador que contenga las reglas para analizar el registro y concatenar líneas que pertenecen al mismo mensaje.

   a. Pegue los siguientes contenidos en el archivo analizador:

```
[MULTILINE_PARSER]
```

```
    name          multiline-regex-test
    type          regex
    flush_timeout 1000
    #
    # Regex rules for multiline parsing
    # ---------------------------------
    #
    # configuration hints:
    #
    #  - first state always has the name: start_state
    #  - every field in the rule must be inside double quotes
    #
    # rules |   state name  | regex pattern                  | next state
    # ------|---------------|--------------------------------------------
    rule      "start_state"   "/(Dec \d+ \d+\:\d+\:\d+)(.*)/"  "cont"
    rule      "cont"          "/^\s+at.*/"                     "cont"
```

A medida que personaliza el patrón de expresiones regulares, recomendamos utilizar un editor de expresiones regulares para probar la expresión.

b. Guarde el archivo como `parsers_multiline.conf`.

3. En la carpeta `FluentBitDockerImage`, cree un archivo de configuración personalizado que haga referencia al archivo analizador que creó en el paso anterior.

   Para obtener más información sobre el archivo de configuración personalizado, consulte Especificación de un archivo de configuración personalizado en la Guía para desarrolladores de Amazon Elastic Container Service

   a. Pegue los siguientes contenidos en el archivo:

```
[SERVICE]
    flush                1
    log_level            info
    parsers_file         /parsers_multiline.conf

[FILTER]
    name                 multiline
    match                *
    multiline.key_content log
    multiline.parser     multiline-regex-test
```

> **Note**
>
> Debe utilizar la ruta absoluta del analizador.

b. Guarde el archivo como `extra.conf`.

4. En la carpeta `FluentBitDockerImage`, cree el archivo de Dockerfile con la imagen de Fluent Bit, el analizador y los archivos de configuración que ha creado.

   a. Pegue los siguientes contenidos en el archivo:

```
FROM public.ecr.aws/aws-observability/aws-for-fluent-bit:latest

ADD parsers_multiline.conf /parsers_multiline.conf
ADD extra.conf /extra.conf
```

   b. Guarde el archivo como `Dockerfile`.

5. Con el archivo de Dockerfile, cree una imagen de Fluent Bit personalizada con el analizador y los archivos de configuración personalizados incluidos.

> **Note**
>
> Puede colocar el archivo analizador y el archivo de configuración en cualquier parte de la imagen de Docker, excepto en `/fluent-bit/etc/fluent-bit.conf`, ya que FireLens utiliza esta ruta de archivo.

   a. Cree la imagen: `docker build -t fluent-bit-multiline-image.`

      Donde: `fluent-bit-multiline-image` es el nombre de la imagen de este ejemplo.

   b. Compruebe que la imagen se haya creado correctamente: `docker images –filter reference=fluent-bit-multiline-image`

      Si la acción se realiza correctamente, el resultado muestra la imagen y la etiqueta `latest`.

6. Cargue la imagen de Fluent Bit personalizada en Amazon Elastic Container Registry.

   a. Cree un repositorio de Amazon ECR para almacenar la imagen: `aws ecr create-repository --repository-name fluent-bit-multiline-repo --region us-east-1`

Donde: `fluent-bit-multiline-repo` es el nombre del repositorio y `us-east-1` es la región de este ejemplo.

El resultado proporciona los detalles del nuevo repositorio.

b. Etiquete la imagen con el valor `repositoryUri` del resultado anterior: `docker tag` *`fluent-bit-multiline-image repositoryUri`*

   Ejemplo: `docker tag fluent-bit-multiline-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

c. Ejecute la imagen de Docker para verificar que se haya ejecutado correctamente: `docker images –filter reference=`*`repositoryUri`*

   En el resultado, el nombre del repositorio cambia de fluent-bit-multiline-repo a `repositoryUri`.

d. Autentíquese en Amazon ECR ejecutando el comando `aws ecr get-login-password` y especifique el ID del registro en el que desea efectuar la autenticación: `aws ecr get-login-password | docker login --username AWS --password-stdin` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com`

   Ejemplo: `ecr get-login-password | docker login --username AWS --password-stdin xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com`

   Aparece un mensaje de inicio de sesión correcto.

e. Envíe la imagen a Amazon ECR: `docker push` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`repository name`*

   Ejemplo: `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

Crear y cargar la imagen para una aplicación multilínea de demostración

En esta imagen, se incluirá un archivo de script de Python que ejecuta la aplicación y un archivo de registro de ejemplo.

Cuando ejecuta la tarea, la aplicación simula las ejecuciones y, a continuación, falla y crea un seguimiento de pila.

1. Cree una carpeta denominada `multiline-app`: `mkdir multiline-app`
2. Cree un archivo de script de Python.
   a. En la carpeta `multiline-app`, cree un archivo y asígnele el nombre `main.py`.
   b. Pegue los siguientes contenidos en el archivo:

```
import os
import time
file1 = open('/test.log', 'r')
Lines = file1.readlines()

count = 0

for i in range(10):
    print("app running normally...")
    time.sleep(1)

# Strips the newline character
for line in Lines:
    count += 1
    print(line.rstrip())
print(count)
print("app terminated.")
```

   c. Guarde el archivo `main.py`.
3. Cree un archivo de registro de ejemplo.
   a. En la carpeta `multiline-app`, cree un archivo y asígnele el nombre `test.log`.
   b. Pegue los siguientes contenidos en el archivo:

```
single line...
Dec 14 06:41:08 Exception in thread "main" java.lang.RuntimeException:
 Something has gone wrong, aborting!
    at com.myproject.module.MyProject.badMethod(MyProject.java:22)
    at com.myproject.module.MyProject.oneMoreMethod(MyProject.java:18)
    at com.myproject.module.MyProject.anotherMethod(MyProject.java:14)
    at com.myproject.module.MyProject.someMethod(MyProject.java:10)
    at com.myproject.module.MyProject.main(MyProject.java:6)
another line...
```

c. Guarde el archivo `test.log`.

4. En la carpeta `multiline-app`, cree el archivo de Dockerfile.

   a. Pegue los siguientes contenidos en el archivo:

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
ADD test.log /test.log

RUN yum upgrade -y && yum install -y python3

WORKDIR /usr/local/bin

COPY main.py .

CMD ["python3", "main.py"]
```

   b. Guarde el archivo `Dockerfile`.

5. Con el archivo de Dockerfile, cree una imagen.

   a. Cree la imagen: `docker build -t multiline-app-image`

      Donde: `multiline-app-image` es el nombre de la imagen de este ejemplo.

   b. Compruebe que la imagen se haya creado correctamente: `docker images –filter reference=multiline-app-image`

      Si la acción se realiza correctamente, el resultado muestra la imagen y la etiqueta `latest`.

6. Cargue la imagen en Amazon Elastic Container Registry.

   a. Cree un repositorio de Amazon ECR para almacenar la imagen: `aws ecr create-repository --repository-name multiline-app-repo --region us-east-1`

      Donde: `multiline-app-repo` es el nombre del repositorio y `us-east-1` es la región de este ejemplo.

      El resultado proporciona los detalles del nuevo repositorio. Tome nota del valor `repositoryUri`, ya que lo necesitará en los siguientes pasos.

   b. Etiquete la imagen con el valor `repositoryUri` del resultado anterior: `docker tag` *`multiline-app-image repositoryUri`*

Ejemplo: `docker tag multiline-app-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

c. Ejecute la imagen de Docker para verificar que se haya ejecutado correctamente: `docker images –filter reference=repositoryUri`

   En el resultado, el nombre del repositorio cambia de `multiline-app-repo` al valor `repositoryUri`.

d. Envíe la imagen a Amazon ECR: `docker push aws_account_id.dkr.ecr.region.amazonaws.com/repository name`

   Ejemplo: `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

Crear la definición de tarea y ejecutar la tarea

1. Cree un archivo de definición de tarea con el nombre de archivo `multiline-task-definition.json`.
2. Pegue los siguientes contenidos en el archivo `multiline-task-definition.json`:

```
{
    "family": "firelens-example-multiline",
    "taskRoleArn": "task role ARN,
    "executionRoleArn": "execution role ARN",
    "containerDefinitions": [
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-
multiline-image:latest",
            "name": "log_router",
            "firelensConfiguration": {
                "type": "fluentbit",
                "options": {
                    "config-file-type": "file",
                    "config-file-value": "/extra.conf"
                }
            },
            "memoryReservation": 50
        },
        {
```

```
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/multiline-app-
image:latest",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "cloudwatch_logs",
                    "region": "us-east-1",
                    "log_group_name": "multiline-test/application",
                    "auto_create_group": "true",
                    "log_stream_prefix": "multiline-"
                }
            },
            "memoryReservation": 100
        }
    ],
    "requiresCompatibilities": ["FARGATE"],
    "networkMode": "awsvpc",
    "cpu": "256",
    "memory": "512"
}
```

Sustituya lo siguiente en la definición de tarea `multiline-task-definition.json`:

a. *`task role ARN`*

   Para encontrar el ARN del rol de tarea, vaya a la consola de IAM. Elija Roles y busque el rol de tarea `ecs-task-role-for-firelens` que ha creado. Elija el rol y copie el ARN que aparece en la sección Summary (Resumen).

b. *`execution role ARN`*

   Para encontrar el ARN del rol de ejecución, vaya a la consola de IAM. Elija Roles y busque el rol `ecsTaskExecutionRole`. Elija el rol y copie el ARN que aparece en la sección Summary (Resumen).

c. *`aws_account_id`*

   Para encontrar su `aws_account_id`, inicie sesión en la AWS Management Console. Elija su nombre de usuario en la parte superior derecha y copie su ID de cuenta.

d. *`us-east-1`*

Sustituya la región si es necesario.

3. Registre el archivo de definición de tarea: `aws ecs register-task-definition --cli-input-json file://multiline-task-definition.json --region region`
4. Abra la consola en https://console.aws.amazon.com/ecs/v2.
5. En el panel de navegación, elija Task Definitions (Definiciones de tarea) y, a continuación, seleccione la familia `firelens-example-multiline` porque hemos registrado la definición de tarea en esta familia en la primera línea de la definición de tarea anterior.
6. Elija la versión más reciente.
7. Elija Implementar y Ejecutar tarea.
8. En la página Ejecutar tarea, en Clúster, elija el clúster y, a continuación, en Redes, para Subredes, elija las subredes disponibles para la tarea.
9. Seleccione Crear.

Verificar que los mensajes de registro multilínea en Amazon CloudWatch aparezcan concatenados

1. Abra la consola de CloudWatch en https://console.aws.amazon.com/cloudwatch/.
2. En el panel de navegación, amplíe Logs (Registros) y elija Log groups (Grupos de registros).
3. Elija el grupo de registros `multiline-test/applicatio`.
4. Elija el registro. Vea los mensajes. Las líneas que coinciden con las reglas del archivo analizador se concatenan y aparecen como un solo mensaje.

   En el siguiente fragmento de registro, se muestran las líneas concatenadas en un único evento de seguimiento de pila Java:

```
{
    "container_id": "xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "log": "Dec 14 06:41:08 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!\n
 at com.myproject.module.MyProject.badMethod(MyProject.java:22)\n    at
 com.myproject.module.MyProject.oneMoreMethod(MyProject.java:18)\n
 at com.myproject.module.MyProject.anotherMethod(MyProject.java:14)\n
 at com.myproject.module.MyProject.someMethod(MyProject.java:10)\n    at
 com.myproject.module.MyProject.main(MyProject.java:6)",
    "ecs_cluster": "default",
```

```
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:2"
}
```

El siguiente fragmento de registro muestra cómo aparece el mismo mensaje con una sola línea si se ejecuta un contenedor de Amazon ECS que no está configurado para concatenar mensajes de registro de varias líneas.

```
{
    "log": "Dec 14 06:41:08 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!",
    "container_id": "xxxxxx-xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:3"
}
```

Ejemplo: utilizar un analizador integrado de Fluent Bit

En este ejemplo, completará los siguientes pasos:

1. Cree y cargue la imagen en un contenedor de Fluent Bit.
2. Cree y cargue la imagen de una aplicación multilínea de demostración que se ejecuta, falla y genera un seguimiento de pila multilínea.
3. Cree la definición de tarea y ejecute la tarea.
4. Consulte los registros para verificar que los mensajes que abarcan varias líneas aparezcan concatenados.

Crear y cargar la imagen en un contenedor de Fluent Bit

En esta imagen, se incluirá un archivo de configuración que hace referencia al analizador de Fluent Bit.

1. Cree una carpeta con el nombre `FluentBitDockerImage`.
2. En la carpeta `FluentBitDockerImage`, cree un archivo de configuración personalizado que haga referencia al archivo analizador integrado de Fluent Bit.

Para obtener más información sobre el archivo de configuración personalizado, consulte Especificación de un archivo de configuración personalizado en la Guía para desarrolladores de Amazon Elastic Container Service

a. Pegue los siguientes contenidos en el archivo:

```
[FILTER]
    name                  multiline
    match                 *
    multiline.key_content log
    multiline.parser      go
```

b. Guarde el archivo como `extra.conf`.

3. En la carpeta `FluentBitDockerImage`, cree el archivo de Dockerfile con la imagen de Fluent Bit, el analizador y los archivos de configuración que ha creado.

a. Pegue los siguientes contenidos en el archivo:

```
FROM public.ecr.aws/aws-observability/aws-for-fluent-bit:latest
ADD extra.conf /extra.conf
```

b. Guarde el archivo como `Dockerfile`.

4. Con el archivo de Dockerfile, cree una imagen de Fluent Bit personalizada con el archivo de configuración personalizado incluido.

> **Note**
>
> Puede colocar el archivo de configuración en cualquier parte de la imagen de Docker, excepto en `/fluent-bit/etc/fluent-bit.conf`, ya que FireLens utiliza esta ruta de archivo.

a. Cree la imagen: `docker build -t fluent-bit-multiline-image.`

Donde: `fluent-bit-multiline-image` es el nombre de la imagen de este ejemplo.

b. Compruebe que la imagen se haya creado correctamente: `docker images –filter reference=fluent-bit-multiline-image`

Si la acción se realiza correctamente, el resultado muestra la imagen y la etiqueta `latest`.

5. Cargue la imagen de Fluent Bit personalizada en Amazon Elastic Container Registry.

   a. Cree un repositorio de Amazon ECR para almacenar la imagen: `aws ecr create-repository --repository-name fluent-bit-multiline-repo --region us-east-1`

      Donde: `fluent-bit-multiline-repo` es el nombre del repositorio y `us-east-1` es la región de este ejemplo.

      El resultado proporciona los detalles del nuevo repositorio.

   b. Etiquete la imagen con el valor `repositoryUri` del resultado anterior: `docker tag` *`fluent-bit-multiline-image repositoryUri`*

      Ejemplo: `docker tag fluent-bit-multiline-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

   c. Ejecute la imagen de Docker para verificar que se haya ejecutado correctamente: `docker images –filter reference=`*`repositoryUri`*

      En el resultado, el nombre del repositorio cambia de fluent-bit-multiline-repo a `repositoryUri`.

   d. Autentíquese en Amazon ECR ejecutando el comando `aws ecr get-login-password` y especifique el ID del registro en el que desea efectuar la autenticación: `aws ecr get-login-password | docker login --username AWS --password-stdin` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com`

      Ejemplo: `ecr get-login-password | docker login --username AWS --password-stdin xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com`

      Aparece un mensaje de inicio de sesión correcto.

   e. Envíe la imagen a Amazon ECR: `docker push` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`repository name`*

      Ejemplo: `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

Crear y cargar la imagen para una aplicación multilínea de demostración

En esta imagen, se incluirá un archivo de script de Python que ejecuta la aplicación y un archivo de registro de ejemplo.

1. Cree una carpeta denominada `multiline-app`: `mkdir multiline-app`
2. Cree un archivo de script de Python.
   a. En la carpeta `multiline-app`, cree un archivo y asígnele el nombre `main.py`.
   b. Pegue los siguientes contenidos en el archivo:

```
import os
import time
file1 = open('/test.log', 'r')
Lines = file1.readlines()

count = 0

for i in range(10):
    print("app running normally...")
    time.sleep(1)

# Strips the newline character
for line in Lines:
    count += 1
    print(line.rstrip())
print(count)
print("app terminated.")
```

   c. Guarde el archivo `main.py`.
3. Cree un archivo de registro de ejemplo.
   a. En la carpeta `multiline-app`, cree un archivo y asígnele el nombre `test.log`.
   b. Pegue los siguientes contenidos en el archivo:

```
panic: my panic

goroutine 4 [running]:
panic(0x45cb40, 0x47ad70)
```

```
  /usr/local/go/src/runtime/panic.go:542 +0x46c fp=0xc42003f7b8 sp=0xc42003f710
 pc=0x422f7c
main.main.func1(0xc420024120)
  foo.go:6 +0x39 fp=0xc42003f7d8 sp=0xc42003f7b8 pc=0x451339
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003f7e0
 sp=0xc42003f7d8 pc=0x44b4d1
created by main.main
  foo.go:5 +0x58

goroutine 1 [chan receive]:
runtime.gopark(0x4739b8, 0xc420024178, 0x46fcd7, 0xc, 0xc420028e17, 0x3)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc420053e30 sp=0xc420053e00
 pc=0x42503c
runtime.goparkunlock(0xc420024178, 0x46fcd7, 0xc, 0x1000f010040c217, 0x3)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc420053e70 sp=0xc420053e30
 pc=0x42512e
runtime.chanrecv(0xc420024120, 0x0, 0xc420053f01, 0x4512d8)
  /usr/local/go/src/runtime/chan.go:506 +0x304 fp=0xc420053f20 sp=0xc420053e70
 pc=0x4046b4
runtime.chanrecv1(0xc420024120, 0x0)
  /usr/local/go/src/runtime/chan.go:388 +0x2b fp=0xc420053f50 sp=0xc420053f20
 pc=0x40439b
main.main()
  foo.go:9 +0x6f fp=0xc420053f80 sp=0xc420053f50 pc=0x4512ef
runtime.main()
  /usr/local/go/src/runtime/proc.go:185 +0x20d fp=0xc420053fe0 sp=0xc420053f80
 pc=0x424bad
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc420053fe8
 sp=0xc420053fe0 pc=0x44b4d1

goroutine 2 [force gc (idle)]:
runtime.gopark(0x4739b8, 0x4ad720, 0x47001e, 0xf, 0x14, 0x1)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003e768 sp=0xc42003e738
 pc=0x42503c
runtime.goparkunlock(0x4ad720, 0x47001e, 0xf, 0xc420000114, 0x1)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003e7a8 sp=0xc42003e768
 pc=0x42512e
runtime.forcegchelper()
  /usr/local/go/src/runtime/proc.go:238 +0xcc fp=0xc42003e7e0 sp=0xc42003e7a8
 pc=0x424e5c
runtime.goexit()
```

```
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003e7e8
 sp=0xc42003e7e0 pc=0x44b4d1
created by runtime.init.4
  /usr/local/go/src/runtime/proc.go:227 +0x35

goroutine 3 [GC sweep wait]:
runtime.gopark(0x4739b8, 0x4ad7e0, 0x46fdd2, 0xd, 0x419914, 0x1)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003ef60 sp=0xc42003ef30
 pc=0x42503c
runtime.goparkunlock(0x4ad7e0, 0x46fdd2, 0xd, 0x14, 0x1)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003efa0 sp=0xc42003ef60
 pc=0x42512e
runtime.bgsweep(0xc42001e150)
  /usr/local/go/src/runtime/mgcsweep.go:52 +0xa3 fp=0xc42003efd8
 sp=0xc42003efa0 pc=0x419973
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003efe0
 sp=0xc42003efd8 pc=0x44b4d1
created by runtime.gcenable
  /usr/local/go/src/runtime/mgc.go:216 +0x58
one more line, no multiline
```

c. Guarde el archivo `test.log`.

4. En la carpeta `multiline-app`, cree el archivo de Dockerfile.

   a. Pegue los siguientes contenidos en el archivo:

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
ADD test.log /test.log

RUN yum upgrade -y && yum install -y python3

WORKDIR /usr/local/bin

COPY main.py .

CMD ["python3", "main.py"]
```

   b. Guarde el archivo `Dockerfile`.

5. Con el archivo de Dockerfile, cree una imagen.

   a. Cree la imagen: `docker build -t multiline-app-image`

Donde: `multiline-app-image` es el nombre de la imagen de este ejemplo.

b. Compruebe que la imagen se haya creado correctamente: `docker images –filter reference=multiline-app-image`

Si la acción se realiza correctamente, el resultado muestra la imagen y la etiqueta `latest`.

6. Cargue la imagen en Amazon Elastic Container Registry.

a. Cree un repositorio de Amazon ECR para almacenar la imagen: `aws ecr create-repository --repository-name multiline-app-repo --region us-east-1`

Donde: `multiline-app-repo` es el nombre del repositorio y `us-east-1` es la región de este ejemplo.

El resultado proporciona los detalles del nuevo repositorio. Tome nota del valor `repositoryUri`, ya que lo necesitará en los siguientes pasos.

b. Etiquete la imagen con el valor `repositoryUri` del resultado anterior: `docker tag` *`multiline-app-image repositoryUri`*

Ejemplo: `docker tag multiline-app-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

c. Ejecute la imagen de Docker para verificar que se haya ejecutado correctamente: `docker images –filter reference=`*`repositoryUri`*

En el resultado, el nombre del repositorio cambia de `multiline-app-repo` al valor `repositoryUri`.

d. Envíe la imagen a Amazon ECR: `docker push` *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`repository name`*

Ejemplo: `docker push` *`xxxxxxxxxxxx`*`.dkr.ecr.`*`us-east-1`*`.amazonaws.com/multiline-app-repo`

Crear la definición de tarea y ejecutar la tarea

1. Cree un archivo de definición de tarea con el nombre de archivo `multiline-task-definition.json`.
2. Pegue los siguientes contenidos en el archivo `multiline-task-definition.json`:

```
{
    "family": "firelens-example-multiline",
    "taskRoleArn": "task role ARN,
    "executionRoleArn": "execution role ARN",
    "containerDefinitions": [
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-
multiline-image:latest",
            "name": "log_router",
            "firelensConfiguration": {
                "type": "fluentbit",
                "options": {
                    "config-file-type": "file",
                    "config-file-value": "/extra.conf"
                }
            },
            "memoryReservation": 50
        },
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/multiline-app-
image:latest",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "cloudwatch_logs",
                    "region": "us-east-1",
                    "log_group_name": "multiline-test/application",
                    "auto_create_group": "true",
                    "log_stream_prefix": "multiline-"
                }
            },
            "memoryReservation": 100
        }
    ],
    "requiresCompatibilities": ["FARGATE"],
    "networkMode": "awsvpc",
    "cpu": "256",
    "memory": "512"
}
```

Sustituya lo siguiente en la definición de tarea `multiline-task-definition.json`:

a. *`task role ARN`*

   Para encontrar el ARN del rol de tarea, vaya a la consola de IAM. Elija Roles y busque el rol de tarea `ecs-task-role-for-firelens` que ha creado. Elija el rol y copie el ARN que aparece en la sección Summary (Resumen).

b. *`execution role ARN`*

   Para encontrar el ARN del rol de ejecución, vaya a la consola de IAM. Elija Roles y busque el rol `ecsTaskExecutionRole`. Elija el rol y copie el ARN que aparece en la sección Summary (Resumen).

c. *`aws_account_id`*

   Para encontrar su `aws_account_id`, inicie sesión en la AWS Management Console. Elija su nombre de usuario en la parte superior derecha y copie su ID de cuenta.

d. *`us-east-1`*

   Sustituya la región si es necesario.

3. Registre el archivo de definición de tarea: `aws ecs register-task-definition --cli-input-json file://multiline-task-definition.json --region us-east-1`
4. Abra la consola en https://console.aws.amazon.com/ecs/v2.
5. En el panel de navegación, elija Task Definitions (Definiciones de tarea) y, a continuación, seleccione la familia `firelens-example-multiline` porque hemos registrado la definición de tarea en esta familia en la primera línea de la definición de tarea anterior.
6. Elija la versión más reciente.
7. Elija Implementar y Ejecutar tarea.
8. En la página Ejecutar tarea, en Clúster, elija el clúster y, a continuación, en Redes, para Subredes, elija las subredes disponibles para la tarea.
9. Seleccione Crear.

Verificar que los mensajes de registro multilínea en Amazon CloudWatch aparezcan concatenados

1. Abra la consola de CloudWatch en https://console.aws.amazon.com/cloudwatch/.
2. En el panel de navegación, amplíe Logs (Registros) y elija Log groups (Grupos de registros).

3. Elija el grupo de registros `multiline-test/applicatio`.
4. Elija el registro y vea los mensajes. Las líneas que coinciden con las reglas del archivo analizador se concatenan y aparecen como un solo mensaje.

   En el siguiente fragmento de registro, se muestra un seguimiento de pila Go que está concatenado en un único evento:

```
{
    "log": "panic: my panic\n\ngoroutine 4 [running]:\npanic(0x45cb40,
 0x47ad70)\n  /usr/local/go/src/runtime/panic.go:542 +0x46c fp=0xc42003f7b8
 sp=0xc42003f710 pc=0x422f7c\nmain.main.func1(0xc420024120)\n  foo.go:6
 +0x39 fp=0xc42003f7d8 sp=0xc42003f7b8 pc=0x451339\nruntime.goexit()\n  /usr/
local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003f7e0 sp=0xc42003f7d8
 pc=0x44b4d1\ncreated by main.main\n  foo.go:5 +0x58\n\ngoroutine 1 [chan receive]:
\nruntime.gopark(0x4739b8, 0xc420024178, 0x46fcd7, 0xc, 0xc420028e17, 0x3)\n  /usr/
local/go/src/runtime/proc.go:280 +0x12c fp=0xc420053e30 sp=0xc420053e00 pc=0x42503c
\nruntime.goparkunlock(0xc420024178, 0x46fcd7, 0xc, 0x1000f010040c217, 0x3)\n
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc420053e70 sp=0xc420053e30
 pc=0x42512e\nruntime.chanrecv(0xc420024120, 0x0, 0xc420053f01, 0x4512d8)\n
  /usr/local/go/src/runtime/chan.go:506 +0x304 fp=0xc420053f20 sp=0xc420053e70
 pc=0x4046b4\nruntime.chanrecv1(0xc420024120, 0x0)\n  /usr/local/go/src/runtime/
chan.go:388 +0x2b fp=0xc420053f50 sp=0xc420053f20 pc=0x40439b\nmain.main()\n
 foo.go:9 +0x6f fp=0xc420053f80 sp=0xc420053f50 pc=0x4512ef\nruntime.main()\n
  /usr/local/go/src/runtime/proc.go:185 +0x20d fp=0xc420053fe0 sp=0xc420053f80
 pc=0x424bad\nruntime.goexit()\n  /usr/local/go/src/runtime/asm_amd64.s:2337
 +0x1 fp=0xc420053fe8 sp=0xc420053fe0 pc=0x44b4d1\n\ngoroutine 2 [force gc
 (idle)]:\nruntime.gopark(0x4739b8, 0x4ad720, 0x47001e, 0xf, 0x14, 0x1)\n  /
usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003e768 sp=0xc42003e738
 pc=0x42503c\nruntime.goparkunlock(0x4ad720, 0x47001e, 0xf, 0xc420000114, 0x1)\n
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003e7a8 sp=0xc42003e768
 pc=0x42512e\nruntime.forcegchelper()\n  /usr/local/go/src/runtime/proc.go:238
 +0xcc fp=0xc42003e7e0 sp=0xc42003e7a8 pc=0x424e5c\nruntime.goexit()\n  /usr/
local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003e7e8 sp=0xc42003e7e0
 pc=0x44b4d1\ncreated by runtime.init.4\n  /usr/local/go/src/runtime/proc.go:227
 +0x35\n\ngoroutine 3 [GC sweep wait]:\nruntime.gopark(0x4739b8, 0x4ad7e0,
 0x46fdd2, 0xd, 0x419914, 0x1)\n  /usr/local/go/src/runtime/proc.go:280 +0x12c
 fp=0xc42003ef60 sp=0xc42003ef30 pc=0x42503c\nruntime.goparkunlock(0x4ad7e0,
 0x46fdd2, 0xd, 0x14, 0x1)\n  /usr/local/go/src/runtime/proc.go:286 +0x5e
 fp=0xc42003efa0 sp=0xc42003ef60 pc=0x42512e\nruntime.bgsweep(0xc42001e150)\n
  /usr/local/go/src/runtime/mgcsweep.go:52 +0xa3 fp=0xc42003efd8 sp=0xc42003efa0
 pc=0x419973\nruntime.goexit()\n  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1
 fp=0xc42003efe0 sp=0xc42003efd8 pc=0x44b4d1\ncreated by runtime.gcenable\n  /usr/
local/go/src/runtime/mgc.go:216 +0x58",
```

```
    "container_id": "xxxxxx-xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:2"
}
```

En el siguiente fragmento de registro, se muestra cómo aparece el mismo evento si ejecuta un contenedor ECS que no está configurado para concatenar mensajes de registro multilínea. El campo de registro contiene una sola línea.

```
{
    "log": "panic: my panic",
    "container_id": "xxxxxx-xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:3"
```

> **Note**
>
> Si los registros van a los archivos de registro en lugar de al resultado estándar, recomendamos especificar `multiline.parser` y los parámetros de configuración `multiline.key_content` en el complemento de entrada de cola en lugar del filtro.

Implementación de Fluent Bit en contenedores de Amazon ECS para Windows

Fluent Bit es un procesador y enrutador de registros rápido y flexible, que es compatible con varios sistemas operativos. Se puede usar para enrutar registros a varios destinos de AWS, como Registros de Amazon CloudWatch, Firehose Amazon S3 y Amazon OpenSearch Service. Fluent Bit admite soluciones de socios comunes, como Datadog, Splunk y servidores HTTP personalizados. Para obtener más información acerca de Fluent Bit, consulte el sitio web de Fluent Bit .

La imagen de AWS para Bit Fluent está disponible en la galería pública y en un repositorio de Amazon ECR en la mayoría de las regiones para lograr alta disponibilidad. Para obtener más información, consulte aws-for-fluent-bit en el sitio web de GitHub.

Este tutorial explica cómo implementar contenedores de Fluent Bit en sus instancias de Windows que se ejecutan en Amazon ECS para transmitir los registros generados por las tareas de Windows a Amazon CloudWatch para un registro centralizado.

Este tutorial utiliza el siguiente enfoque:

- Fluent Bit funciona como un servicio mediante la estrategia de programación de Daemon. Esta estrategia garantiza que siempre se ejecute una única instancia de Fluent Bit en las instancias de contenedor del clúster.
  - Escucha en el puerto 24224 mediante el complemento de entrada directa.
  - Exponga el puerto 24224 al host para que el tiempo de ejecución de Docker pueda enviar registros a Fluent Bit mediante este puerto expuesto.
  - Tiene una configuración que permite a Fluent Bit enviar los registros a destinos especificados.
- Inicie todos los demás contenedores de tareas de Amazon ECS con el controlador de registro fluentd. Para obtener más información, consulte Controlador de registro Fluentd en el sitio web de documentación de Docker.
  - Docker se conecta al socket TCP 24224 en localhost dentro del espacio de nombres del host.
  - El agente de Amazon ECS agrega etiquetas a los contenedores que incluyen el nombre del clúster, el nombre de familia y el número de revisión de la definición de tarea, el ARN de la tarea y el nombre del contenedor. La misma información se agrega al archivo de registro mediante la opción de etiquetas del controlador de registro fluentd de Docker. Para obtener más información, consulte labels, labels-regex, env y env-regex en el sitio web de documentación de Docker.
  - Como la opción `async` del controlador de registro fluentd está configurada en `true`, Docker almacena en búfer los registros hasta que se reinicie el contenedor de Fluent Bit. Puede aumentar el límite del búfer si configura la opción fluentd-buffer-limit. Para obtener más información, consulte fluentd-buffer-limit en el sitio web de documentación de Docker.

El flujo de trabajo es el siguiente:

- El contenedor de Fluent Bit se inicia y escucha en el puerto 24224, el cual está expuesto al host.
- Fluent Bit usa las credenciales del rol de IAM de la tarea especificadas en la definición de tarea.

- Otras tareas ejecutadas en la misma instancia utilizan el controlador de registro fluentd de Docker para conectarse al contenedor de Fluent Bit en el puerto 24224.
- Cuando los contenedores de la aplicación generan registros, el tiempo de ejecución de Docker los etiqueta, agrega metadatos adicionales especificados en las etiquetas y luego los reenvía al puerto 24224 del espacio de nombres del host.
- Fluent Bit recibe el archivo de registro en el puerto 24224 porque está expuesto al espacio de nombres del host.
- Fluent Bit lleva a cabo su procesamiento interno y enruta los registros según lo especificado.

Este tutorial utiliza la configuración predeterminada de Fluent Bit de CloudWatch, que hace lo siguiente:

- Crea un nuevo grupo de registro para cada clúster y familia de definiciones de tarea.
- Crea un nuevo flujo de registro para cada contenedor de tareas del grupo de registro generado anteriormente cada vez que se lanza una nueva tarea. Cada flujo se marcará con el identificador de tarea al que pertenece el contenedor.
- Agrega metadatos adicionales en cada entrada del registro, como el nombre del clúster, el ARN de la tarea, el nombre del contenedor de la tarea y la familia y el número de revisión de definición de la tarea.

  Por ejemplo, si tiene `task_1` con `container_1` y `container_2`, además de `ask_2` con `container_3`, entonces los siguientes son los flujos de registro de CloudWatch:

  - `/aws/ecs/windows.ecs_task_1`

    `task-out.`*`TASK_ID`*`.container_1`

    `task-out.`*`TASK_ID`*`.container_2`
  - `/aws/ecs/windows.ecs_task_2`

    `task-out.`*`TASK_ID`*`.container_3`

> **Note**
>
> Puede utilizar puntos de conexión de servicio de doble pila para interactuar con Amazon ECS desde la AWS CLI, los SDK y la API de Amazon ECS a través de IPv4 e IPv6. Para obtener más información, consulte Uso de puntos de conexión de doble pila en Amazon ECS.

Pasos

- Requisitos previos
- Paso 1: Crear roles de acceso de IAM
- Paso 2: Crear una instancia de contenedor de Amazon ECS para Windows
- Paso 3: Configurar Fluent Bit
- Paso 4: Registrar una definición de tarea de Fluent Bit para Windows que dirija los registros a CloudWatch
- Paso 5: Ejecutar la definición de tarea ecs-windows-fluent-bit como un servicio de Amazon ECS mediante la estrategia de programación de daemon
- Paso 6: Registrar una definición de tarea de Windows que genere los registros
- Paso 7: Ejecutar la definición de tarea windows-app-task
- Paso 8: Verificar los registros en CloudWatch
- Paso 9: limpiar

Requisitos previos

En este tutorial se supone que los siguientes requisitos previos se han completado:

- La última versión de la AWS CLI está instalada y configurada. Para obtener más información, consulte Instalación o actualización de la versión más reciente de la AWS CLI.
- La imagen del contenedor `aws-for-fluent-bit` está disponible para los siguientes sistemas operativos Windows:
  - Windows Server 2019 Core
  - Windows Server 2019 Full
  - Windows Server 2022 Core
  - Windows Server 2022 Full

- Se han completado los pasos que se indican en Configuración para utilizar Amazon ECS.
- Tiene un clúster. En este tutorial, el nombre del clúster es FluentBit-cluster.
- Tiene una VPC con una subred pública donde se lanzará la instancia de EC2. Puede utilizar la VPC predeterminada. También puede usar una subred privada que permita que los puntos de conexión de Amazon CloudWatch lleguen a la subred. Para obtener más información sobre los puntos de conexión de Amazon CloudWatch, consulte Cuotas y puntos de conexión de Amazon CloudWatch en la Referencia general de AWS. Para obtener información acerca de cómo usar el asistente de Amazon VPC para crear una VPC, consulte the section called “Creación de una nube virtual privada”.

Paso 1: Crear roles de acceso de IAM

Cree los roles de IAM de Amazon ECS.

1. Cree el rol de la instancia de contenedor de Amazon ECS denominado “ecsInstanceRole”. Para obtener más información, consulte Rol de IAM de instancia de contenedor de Amazon ECS.
2. Cree un rol de IAM para la tarea de Fluent Bit denominado `fluentTaskRole`. Para obtener más información, consulte the section called “Rol de IAM para la tarea”.

   Los contenedores de tareas asumen los permisos de IAM concedidos en este rol de IAM. Para permitir que Fluent Bit envíe registros a CloudWatch, debe adjuntar los siguientes permisos al rol de IAM de la tarea.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream",
            "logs:CreateLogGroup",
            "logs:DescribeLogStreams",
            "logs:PutLogEvents"
        ],
        "Resource": "*"
    }
```

```
    ]
}
```

3. Asocie la política de al rol.

   a. Guarde el contenido anterior en un archivo denominado `fluent-bit-policy.json`.

   b. Ejecute el siguiente comando para asociar la política insertada al rol de IAM `fluentTaskRole`.

```
aws iam put-role-policy --role-name fluentTaskRole --policy-name
 fluentTaskPolicy --policy-document file://fluent-bit-policy.json
```

Paso 2: Crear una instancia de contenedor de Amazon ECS para Windows

Cree una instancia de contenedor de Amazon ECS para Windows.

Para crear una instancia de Amazon ECS

1. Utilice el comando `aws ssm get-parameters` para recuperar el ID de la AMI de la región que aloja la VPC. Para obtener más información, consulte Recuperación de los metadatos de la AMI optimizada para Amazon ECS.
2. Utilice la consola de Amazon EC2 para lanzar la instancia.

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

   b. En la barra de navegación, seleccione la región a utilizar.

   c. En el panel de EC2, elija Launch Instance (Lanzar instancia).

   d. En Name (Nombre), escriba un nombre único.

   e. En Application and OS Images (Amazon Machine Image) (Imágenes de aplicaciones y SO [Amazon Machine Image]), elija la AMI que recuperó en el primer paso.

   f. Para Tipo de instancia, elija `t3.xlarge`.

   g. En Key pair (login) (Par de claves [inicio de sesión]), elija un par de claves.

   h. En Network settings (Configuración de red), para Security groups (Grupo de seguridad), seleccione un grupo de seguridad existente o cree uno nuevo.

   i. En Network settings (Configuración de red), para Auto-assign Public IP (Asignar automáticamente una IP pública), selecciona Enable (Activar).

j. En Advanced details (Detalles avanzados), en IAM instance profile (Perfil de instancia de IAM), elija ecsInstanceRole.

k. Configure su instancia de contenedor de Amazon ECS con los siguientes datos de usuario. En Advanced details (Detalles avanzados), pegue el siguiente script en el campo User data (Datos de usuario), reemplazando *cluster_name* con el nombre de su clúster.

```
<powershell>
Import-Module ECSTools
Initialize-ECSAgent -Cluster cluster-name -EnableTaskENI -EnableTaskIAMRole -
LoggingDrivers '["awslogs","fluentd"]'
</powershell>
```

l. Cuando esté listo, seleccione el campo de confirmación y después elija Launch Instances.

m. Verá una página de confirmación que indicará que la instancia se está lanzando. Elija View instances para cerrar la página de confirmación y volver a la consola.

Paso 3: Configurar Fluent Bit

Puede utilizar la siguiente configuración predeterminada proporcionada por AWS para empezar rápidamente:

- Amazon CloudWatch, el cual se basa en el complemento Fluent Bit para Amazon CloudWatch en el Fluent Bit Official Manual (Manual oficial de Fluent Bit).

También puede usar otras configuraciones predeterminadas de AWS. Para obtener más información, consulte Overriding the entrypoint for the Windows image (Anulación del punto de entrada de la imagen de Windows) en `aws-for-fluent-bit` en el sitio web de GitHub.

La configuración predeterminada de Fluent Bit de Amazon CloudWatch se muestra a continuación.

Reemplace las siguientes variables:

- *region* con la región a la que desea enviar los registros de Amazon CloudWatch.

```
[SERVICE]
    Flush               5
    Log_Level           info
    Daemon              off
```

```
[INPUT]
    Name                forward
    Listen              0.0.0.0
    Port                24224
    Buffer_Chunk_Size   1M
    Buffer_Max_Size     6M
    Tag_Prefix          ecs.

# Amazon ECS agent adds the following log keys as labels to the docker container.
# We would use fluentd logging driver to add these to log record while sending it to
 Fluent Bit.
[FILTER]
    Name                modify
    Match               ecs.*
    Rename              com.amazonaws.ecs.cluster ecs_cluster
    Rename              com.amazonaws.ecs.container-name ecs_container_name
    Rename              com.amazonaws.ecs.task-arn ecs_task_arn
    Rename              com.amazonaws.ecs.task-definition-family
 ecs_task_definition_family
    Rename              com.amazonaws.ecs.task-definition-version
 ecs_task_definition_version

[FILTER]
    Name                rewrite_tag
    Match               ecs.*
    Rule                $ecs_task_arn ^([a-z-:0-9]+)/([a-zA-Z0-9-_]+)/([a-z0-9]+)$
 out.$3.$ecs_container_name false
    Emitter_Name        re_emitted

[OUTPUT]
    Name                cloudwatch_logs
    Match               out.*
    region              region
    log_group_name      fallback-group
    log_group_template  /aws/ecs/$ecs_cluster.$ecs_task_definition_family
    log_stream_prefix   task-
    auto_create_group   On
```

Cada registro que entra en Fluent Bit tiene una etiqueta que usted especifica o se genera automáticamente cuando no proporcione una. Las etiquetas se pueden usar para enrutar distintos registros a diferentes destinos. Para obtener información adicional, consulte Tag (Etiqueta) en el Fluent Bit Official Manual (Manual oficial de Fluent Bit).

La configuración de Fluent Bit descrita anteriormente tiene las siguientes propiedades:

- El complemento de entrada directa escucha el tráfico entrante en el puerto TCP 24224.
- Cada entrada de registro recibida en ese puerto tiene una etiqueta que el complemento de entrada directa modifica para poner un prefijo de cadena `ecs.` al registro.
- La canalización interna de Fluent Bit enruta la entrada de registro para modificar el filtro mediante la expresión regular Match. Este filtro reemplaza las claves del archivo de registro JSON por el formato que Fluent Bit puede utilizar.
- A continuación, el filtro rewrite_tag utiliza la entrada de registro modificada. Este filtro cambia la etiqueta del archivo de registro al formato out.*`TASK_ID`*.*`CONTAINER_NAME`*.
- La nueva etiqueta se enrutará al complemento de salida cloudwatch_logs, el cual crea los grupos de registro y las transmisiones tal como se describió anteriormente mediante las opciones `log_group_template` y `log_stream_prefix` del complemento de salida de CloudWatch. Para obtener información adicional, consulte Configuration parameters (Parámetros de configuración) en el Fluent Bit Official Manual (Manual oficial de Fluent Bit).

Paso 4: Registrar una definición de tarea de Fluent Bit para Windows que dirija los registros a CloudWatch

Registre una definición de tarea de Fluent Bit para Windows que dirija los registros a CloudWatch.

> **Note**
>
> Esta definición de tarea expone el puerto de contenedor de Fluent Bit 24224 al puerto de host 24224. Compruebe que este puerto no esté abierto en el grupo de seguridad de la instancia de EC2 para impedir el acceso desde el exterior.

Para registrar una definición de tareas

1. Cree un archivo denominado `fluent-bit.json` con el siguiente contenido.

   Reemplace las siguientes variables:

   - *`task-iam-role`* con el nombre de recurso de Amazon (ARN) del rol de IAM de su tarea
   - *`region`* con la región en la que se ejecuta la tarea

```
{
  "family": "ecs-windows-fluent-bit",
  "taskRoleArn": "task-iam-role",
  "containerDefinitions": [
    {
      "name": "fluent-bit",
      "image": "public.ecr.aws/aws-observability/aws-for-fluent-
bit:windowsservercore-latest",
      "cpu": 512,
      "portMappings": [
        {
          "hostPort": 24224,
          "containerPort": 24224,
          "protocol": "tcp"
        }
      ],
      "entryPoint": [
        "Powershell",
        "-Command"
      ],
      "command": [
        "C:\\entrypoint.ps1 -ConfigFile C:\\ecs_windows_forward_daemon\
\cloudwatch.conf"
      ],
      "environment": [
        {
          "name": "AWS_REGION",
          "value": "region"
        }
      ],
      "memory": 512,
      "essential": true,
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/fluent-bit-logs",
          "awslogs-region": "region",
          "awslogs-stream-prefix": "flb",
          "awslogs-create-group": "true"
        }
      }
    }
```

```
  ],
  "memory": "512",
  "cpu": "512"
}
```

2. Ejecute el siguiente comando para registrar la definición de tarea.

```
aws ecs register-task-definition --cli-input-json file://fluent-bit.json --
region region
```

Puede enumerar las definiciones de tarea para su cuenta con el comando `list-task-definitions`. El resultado muestra los valores de familia y revisión que puede usar junto con `run-task` o `start-task`.

Paso 5: Ejecutar la definición de tarea **`ecs-windows-fluent-bit`** como un servicio de Amazon ECS mediante la estrategia de programación de daemon

Después de registrar una definición de tarea para su cuenta, puede ejecutar una tarea en el clúster. En este tutorial, se ejecuta una instancia de la definición de tarea `ecs-windows-fluent-bit:1` en el clúster `FluentBit-cluster`. Ejecute la tarea en un servicio que utilice la estrategia de programación de daemon, lo que garantizará que siempre se ejecute una sola instancia de Fluent Bit en cada una de las instancias de contenedor.

Para ejecutar una tarea

1. Ejecute el siguiente comando para iniciar la definición de tarea `ecs-windows-fluent-bit:1` (registrada en el paso anterior) como servicio.

> **Note**
>
> Esta definición de tareas utiliza el controlador de registro `awslogs`; la instancia de contenedor debe tener los permisos necesarios.

Reemplace las siguientes variables:

- *`region`* con la región en la que se ejecuta el servicio

```
aws ecs create-service \
    --cluster FluentBit-cluster \
    --service-name FluentBitForwardDaemonService \
    --task-definition ecs-windows-fluent-bit:1 \
    --launch-type EC2 \
    --scheduling-strategy DAEMON \
    --region region
```

2. Ejecute el siguiente comando para enumerar las tareas.

   Reemplace las siguientes variables:

   - *region* con la región en la que se ejecutan las tareas de servicio

```
aws ecs list-tasks --cluster FluentBit-cluster --region region
```

Paso 6: Registrar una definición de tarea de Windows que genere los registros

Registre una definición de tarea que genere los registros. Esta definición de tarea implementa una imagen de contenedor de Windows que escribirá un número incremental por segundo en `stdout`.

La definición de tarea utiliza el controlador de registro fluentd que se conecta al puerto 24224, el cual escucha el complemento de Fluent Bit. El agente de Amazon ECS etiqueta a cada contenedor de Amazon ECS con etiquetas que incluyen el nombre del clúster, el ARN de la tarea, el nombre de la familia y el número de revisión de la definición de tarea y el nombre del contenedor de la tarea. Estas etiquetas de clave-valor se pasan a Fluent Bit.

> **Note**
>
> Esta tarea utiliza el modo de red `default`. Sin embargo, también puede usar el modo de red `awsvpc` con la tarea.

Para registrar una definición de tareas

1. Cree un archivo denominado `windows-app-task.json` con el siguiente contenido.

```
{
  "family": "windows-app-task",
  "containerDefinitions": [
    {
      "name": "sample-container",
      "image": "mcr.microsoft.com/windows/servercore:ltsc2019",
      "cpu": 512,
      "memory": 512,
      "essential": true,
      "entryPoint": [
        "Powershell",
        "-Command"
      ],
      "command": [
        "$count=1;while(1) { Write-Host $count; sleep 1; $count=$count+1;}"
      ],
      "logConfiguration": {
        "logDriver": "fluentd",
        "options": {
          "fluentd-address": "localhost:24224",
          "tag": "{{ index .ContainerLabels \"com.amazonaws.ecs.task-definition-
family\" }}",
          "fluentd-async": "true",
          "labels": "com.amazonaws.ecs.cluster,com.amazonaws.ecs.container-
name,com.amazonaws.ecs.task-arn,com.amazonaws.ecs.task-definition-
family,com.amazonaws.ecs.task-definition-version"
        }
      }
    }
  ],
  "memory": "512",
  "cpu": "512"
}
```

2. Ejecute el siguiente comando para registrar la definición de tarea.

   Reemplace las siguientes variables:

   - *region* con la región en la que se ejecuta la tarea

```
aws ecs register-task-definition --cli-input-json file://windows-app-task.json --
region region
```

Puede enumerar las definiciones de tarea para su cuenta con el comando `list-task-definitions`. El resultado muestra los valores de familia y revisión que puede usar junto con `run-task` o `start-task`.

Paso 7: Ejecutar la definición de tarea **`windows-app-task`**

Después de registrar la definición de tarea `windows-app-task`, ejecútela en el clúster `FluentBit-cluster`.

Para ejecutar una tarea

1. Ejecute la definición de tarea `windows-app-task:1` registrada en el paso anterior.

   Reemplace las siguientes variables:

   - *`region`* con la región en la que se ejecuta la tarea

   ```
   aws ecs run-task --cluster FluentBit-cluster --task-definition windows-app-task:1
    --count 2 --region region
   ```

2. Ejecute el siguiente comando para enumerar las tareas.

   ```
   aws ecs list-tasks --cluster FluentBit-cluster
   ```

Paso 8: Verificar los registros en CloudWatch

Para verificar la configuración de Fluent Bit, compruebe los siguientes grupos de registro en la consola de CloudWatch:

- `/ecs/fluent-bit-logs`: este es el grupo de registro que corresponde al contenedor de daemon de Fluent Bit que se ejecuta en la instancia de contenedor.

- `/aws/ecs/FluentBit-cluster.windows-app-task`: este es el grupo de registro que corresponde a todas las tareas lanzadas para la familia de definición de tarea `windows-app-task` dentro del clúster `FluentBit-cluster`.

  `task-out.`*`FIRST_TASK_ID`*`.sample-container`: este flujo de registro contiene todos los registros generados por la primera instancia de la tarea en el contenedor de tareas del contenedor de muestras.

  `task-out.`*`SECOND_TASK_ID`*`.sample-container`: este flujo de registro contiene todos los registros generados por la segunda instancia de la tarea en el contenedor de tareas del contenedor de muestras.

El flujo de registro `task-out.`*`TASK_ID`*`.sample-container` tiene campos similares a los siguientes:

```
{
    "source": "stdout",
    "ecs_task_arn": "arn:aws:ecs:region:0123456789012:task/FluentBit-
cluster/13EXAMPLE",
    "container_name": "/ecs-windows-app-task-1-sample-container-cEXAMPLE",
    "ecs_cluster": "FluentBit-cluster",
    "ecs_container_name": "sample-container",
    "ecs_task_definition_version": "1",
    "container_id": "61f5e6EXAMPLE",
    "log": "10",
    "ecs_task_definition_family": "windows-app-task"
}
```

Para verificar la configuración de Fluent Bit

1. Abra la consola de CloudWatch en https://console.aws.amazon.com/cloudwatch/.
2. En el panel de navegación, seleccione Grupos de registro. Asegúrese de que está en la región donde ha implementado Fluent Bit en los contenedores.

   En la lista de grupos de registro de la Región de AWS, debería ver lo siguiente:

   - `/ecs/fluent-bit-logs`
   - `/aws/ecs/FluentBit-cluster.windows-app-task`

Si ve estos grupos de registro, la configuración de Fluent Bit es correcta.

Paso 9: limpiar

Cuando termine este tutorial, debe limpiar los recursos asociados para evitar incurrir en cargos por recursos que no está utilizando.

Para borrar los recursos del tutorial, realice el siguiente procedimiento:

1. Detenga la tarea `windows-simple-task` y la tarea `ecs-fluent-bit`. Para obtener más información, consulte the section called “Detención de una tarea”.
2. Ejecute el siguiente comando para eliminar el grupo de registro `/ecs/fluent-bit-logs`. Para obtener más información sobre la eliminación de grupos de registro, consulte delete-log-group en la Referencia de la AWS Command Line Interface.

```
aws logs delete-log-group --log-group-name /ecs/fluent-bit-logs
aws logs delete-log-group --log-group-name /aws/ecs/FluentBit-cluster.windows-app-
task
```

3. Ejecute el siguiente comando para finalizar la instancia.

```
aws ec2 terminate-instances --instance-ids instance-id
```

4. Ejecute los siguientes comandos para eliminar los roles de IAM.

```
aws iam delete-role --role-name ecsInstanceRole
aws iam delete-role --role-name fluentTaskRole
```

5. Ejecute el siguiente comando para eliminar el clúster de Amazon ECS.

```
aws ecs delete-cluster --cluster FluentBit-cluster
```

Uso de gMSA para contenedores de EC2 Linux en Amazon ECS

Amazon ECS admite la autenticación de Active Directory para contenedores de Linux en EC2 con un tipo especial de cuenta de servicio denominada cuenta de servicio administrada de grupo (gMSA).

Las aplicaciones de red basadas en Linux, como las aplicaciones .NET Core, pueden utilizar Active Directory para facilitar la administración de autorizaciones y autenticación entre usuarios y servicios. Puede utilizar esta característica diseñando aplicaciones que se integren con Active Directory y se ejecuten en servidores unidos a un dominio. Sin embargo, dado que los contenedores de Linux no se pueden unir a un dominio, debe configurar un contenedor de Linux para que se ejecute con gMSA.

Un contenedor de Linux que se ejecuta con gMSA depende del daemon `credentials-fetcher` que se ejecuta en la instancia de Amazon EC2 del host del contenedor. Es decir, el daemon recupera las credenciales de la gMSA del controlador de dominio de Active Directory y, a continuación, las transfiere a la instancia de contenedor. Para obtener más información sobre las cuentas de servicio, consulte Crear gMSAs para contenedores de Windows en el sitio web de Microsoft Learn.

Consideraciones

Tenga en cuenta lo siguiente antes de usar una gMSA para contenedores de Linux:

- Si sus contenedores se ejecutan en EC2, puede utilizar gMSA para contenedores de Windows y de Linux. Para obtener información sobre cómo utilizar el contenedor gMSA de Linux en Fargate, consulte Uso de gMSA para contenedores de Linux en Fargate.
- Puede que necesite una computadora con Windows unida al dominio para cumplir los requisitos previos. Por ejemplo, puede que necesite una computadora con Windows que esté unida al dominio para crear la gMSA en Active Directory con PowerShell. Las herramientas de PowerShell en Active Directory RSAT solo están disponibles para Windows. Para obtener más información, consulte Instalar las herramientas de administración de Active Directory.
- Puede elegir entre gMSA sin dominio o unir cada instancia a un único dominio. Al usar una gMSA sin dominio, la instancia de contenedor no se une al dominio, las demás aplicaciones de la instancia no pueden utilizar las credenciales para acceder al dominio y las tareas que unen diferentes dominios se pueden ejecutar en la misma instancia.

  A continuación, seleccione el almacenamiento de datos para las CredSpec y, de forma opcional, para las credenciales de usuario de Active Directory para gMSA sin dominio.

  Amazon ECS utiliza un archivo de especificaciones de credenciales de Active Directory (CredSpec). Este archivo contiene los metadatos de gMSA utilizados para propagar el contexto de la cuenta de gMSA al contenedor. Genera el archivo de CredSpec y, a continuación, lo almacena en una de las opciones de almacenamiento de CredSpec de la siguiente tabla, específica del sistema operativo de las instancias de contenedor. Para usar el método sin dominio, en una sección opcional del archivo de CredSpec se pueden especificar las credenciales de una de las

opciones de almacenamiento domainless user credentials de la siguiente tabla, específicas del sistema operativo de las instancias de contenedor.

Ubicación de almacenamiento	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	credenciales de usuario sin dominio	credenciales de usuario sin dominio
Parameter Store de Amazon EC2 Systems Manager	CredSpec	CredSpec, credenciales de usuario sin dominio
Archivo local	N/A	CredSpec

Requisitos previos

Antes de utilizar la característica de gMSA para contenedores de Linux con Amazon ECS, asegúrese de completar lo siguiente:

- Configure un dominio de Active Directory con los recursos a los que desea que accedan sus contenedores. Amazon ECS admite las siguientes configuraciones:
  - Un AWS Directory Service Active Directory. AWS Directory Service es un Active Directory administrado por AWS y alojado en Amazon EC2. Para obtener más información, consulte Introducción a Microsoft AD administrado por AWS en la Guía de administración de AWS Directory Service.
  - Un Active Directory en las instalaciones. Debe asegurarse de que la instancia de contenedor de Linux de Amazon ECS pueda unirse al dominio. Para obtener más información, consulte AWS Direct Connect.
- Tiene una cuenta de gMSA existente en Active Directory. Para obtener más información, consulte Uso de gMSA para contenedores de EC2 Linux en Amazon ECS.
- Instaló y está ejecutando el `credentials-fetcher` daemon en una instancia de contenedor Linux de Amazon ECS. También agregó un conjunto inicial de credenciales al `credentials-fetcher` daemon para autenticarse en Active Directory.

> **Note**
>
> El `credentials-fetcher` daemon solo está disponible para Amazon Linux 2023 y Fedora 37 y versiones posteriores. El daemon no está disponible para Amazon Linux 2. Para obtener más información, consulte aws/credentials-fetcher en GitHub.

- Configuró las credenciales para que el `credentials-fetcher` daemon se autentique en Active Directory. Las credenciales deben ser miembros del grupo de seguridad de Active Directory que tenga acceso a la cuenta de gMSA. Hay varias opciones en Decida si quiere unir las instancias al dominio o usar gMSA sin dominio..
- Agregó los permisos necesarios de IAM. Los permisos necesarios dependen de los métodos que elija para las credenciales iniciales y para almacenar la especificación de las credenciales:
  - Si utiliza las credenciales iniciales sin dominio de gMSA, se requieren permisos de IAM para AWS Secrets Manager en el rol de ejecución de tareas.
  - Si almacena la especificación de credenciales en SSM Parameter Store, se requieren permisos de IAM para Parameter Store de Amazon EC2 Systems Manager en el rol de ejecución de la tarea.
  - Si almacena la especificación de credenciales en Amazon S3, se requieren permisos de IAM para Amazon Simple Storage Service en el rol de ejecución de tareas.

Configuración de contenedores de Linux compatibles con gMSA en Amazon ECS

Preparar la infraestructura

Los siguientes pasos son consideraciones y la configuración que se realizan una vez. Después de completar estos pasos, puede automatizar la creación de instancias de contenedores para reutilizar esta configuración.

Decida cómo se proporcionan las credenciales iniciales y configure los datos de usuario de EC2 en una plantilla de lanzamiento de EC2 reutilizable para instalar el `credentials-fetcher` daemon.

1. Decida si quiere unir las instancias al dominio o usar gMSA sin dominio.
   - Unir instancias de EC2 al dominio de Active Directory

- Una las instancias por datos de usuario.

  Agregue los pasos para unir el dominio de Active Directory a sus datos de usuario de EC2 en una plantilla de lanzamiento de EC2. Varios grupos de Amazon EC2 Auto Scaling pueden utilizar la misma plantilla de lanzamiento.

  Puede seguir estos pasos Unirse a un dominio FreeIPA o a Active Directory en los documentos de Fedora.

- Hacer que un usuario de Active Directory sea una gMSA sin dominio

  El `credentials-fetcher` daemon tiene una característica que se denomina gMSA sin dominio. Esta característica requiere un dominio, pero no es necesario unir la instancia EC2 al dominio. Al usar una gMSA sin dominio, la instancia de contenedor no se une al dominio, las demás aplicaciones de la instancia no pueden utilizar las credenciales para acceder al dominio y las tareas que unen diferentes dominios se pueden ejecutar en la misma instancia. En su lugar, debe proporcionar el nombre de un secreto en AWS Secrets Manager en el archivo de CredSpec. El secreto debe contener un nombre de usuario, una contraseña y el dominio para iniciar sesión.

  Esta característica es compatible y se puede utilizar con contenedores de Linux y Windows.

  Esta característica es similar a la característica gMSA support for non-domain-joined container hosts. Para obtener más información sobre la característica de Windows, consulte Arquitectura y mejoras de gMSA en el sitio web de Microsoft Learn.

  a. Cree un usuario en el dominio de Active Directory. El usuario de Active Directory debe tener permiso para acceder a las cuentas de servicio de gMSA que utilice en las tareas.

  b. Cree un secreto en AWS Secrets Manager después de crear el usuario en Active Directory. Para obtener más información, consulte Crear un secreto de AWS Secrets Manager.

  c. Introduzca el nombre de usuario, la contraseña y el dominio en pares clave-valor de JSON denominados `username`, `password` y `domainName`, respectivamente.

  ```
  {"username":"username","password":"passw0rd", "domainName":"example.com"}
  ```

  d. Agregue la configuración al archivo de CredSpec de la cuenta de servicio. La `HostAccountConfig` adicional contiene el nombre de recurso de Amazon (ARN) del secreto de Secrets Manager.

En Windows, el `PluginGUID` debe coincidir con el GUID del siguiente fragmento de ejemplo. En Linux, se omite el `PluginGUID`. Reemplace `MySecret` con un ejemplo del nombre de recurso de Amazon (ARN) de su secreto.

```
    "ActiveDirectoryConfig": {
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
```

e. La característica de gMSA sin dominio necesita permisos adicionales en el rol de ejecución de tareas. Siga el paso (Opcional) secreto de gMSA sin dominio.

2. Configurar las instancias e instalar el **`credentials-fetcher`** daemon

Puede instalar el `credentials-fetcher` daemon con un script de datos de usuario en su plantilla de lanzamiento de EC2. Los siguientes ejemplos muestran dos tipos de datos de usuario, `cloud-config` YAML o script de bash. Estos ejemplos son para Amazon Linux 2023 (AL2023). Reemplace `MyCluster` por el nombre del clúster de Amazon ECS al que desea que se unan estas instancias.

- **`cloud-config`** YAML

```
Content-Type: text/cloud-config
package_reboot_if_required: true
packages:
  # prerequisites
  - dotnet
  - realmd
  - oddjob
  - oddjob-mkhomedir
  - sssd
  - adcli
  - krb5-workstation
  - samba-common-tools
  # https://github.com/aws/credentials-fetcher gMSA credentials management for
 containers
  - credentials-fetcher
```

```
write_files:
# configure the ECS Agent to join your cluster.
# replace MyCluster with the name of your cluster.
- path: /etc/ecs/ecs.config
  owner: root:root
  permissions: '0644'
  content: |
    ECS_CLUSTER=MyCluster
    ECS_GMSA_SUPPORTED=true
runcmd:
# start the credentials-fetcher daemon and if it succeeded, make it start after
 every reboot
- "systemctl start credentials-fetcher"
- "systemctl is-active credentials-fetcher && systemctl enable credentials-
fetcher"
```

- bash script

  Si se siente más cómodo con scripts de bash y tiene distintas variables en las que escribir `/etc/ecs/ecs.config`, utilice el siguiente formato de `heredoc`. Este formato escribe todo entre las líneas que comienzan por cat y `EOF` en el archivo de configuración.

```
#!/usr/bin/env bash
set -euxo pipefail

# prerequisites
timeout 30 dnf install -y dotnet realmd oddjob oddjob-mkhomedir sssd adcli
 krb5-workstation samba-common-tools
# install https://github.com/aws/credentials-fetcher gMSA credentials
 management for containers
timeout 30 dnf install -y credentials-fetcher

# start credentials-fetcher
systemctl start credentials-fetcher
systemctl is-active credentials-fetcher && systemctl enable credentials-fetcher

cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_GMSA_SUPPORTED=true
EOF
```

Hay variables de configuración opcionales para el `credentials-fetcher` daemon que puede configurar en `/etc/ecs/ecs.config`. Se recomienda configurar las variables de los datos de usuario en el bloque YAML o en `heredoc`, de forma similar a los ejemplos anteriores. De este modo, se evitan los problemas de configuración parcial que pueden producirse al editar un archivo varias veces. Para obtener más información sobre la configuración del agente de ECS, consulte Agente de contenedor de Amazon ECS en GitHub.

- De forma opcional, puede usar la variable `CREDENTIALS_FETCHER_HOST` si cambia la configuración del `credentials-fetcher` daemon para mover el socket a otra ubicación.

Configuración de permisos y secretos

Realice los siguientes pasos una vez para cada aplicación y cada definición de tarea. Recomendamos utilizar la práctica recomendada de concesión de privilegios mínimos y reducir los permisos que se utilizan en la política. De esta forma, cada tarea solo puede leer los secretos que necesita.

1. (Opcional) secreto de gMSA sin dominio

   Si utiliza el método sin dominio en el que la instancia no está unida al dominio, siga este paso.

   También debe agregar los siguientes permisos como una política en línea al rol de IAM de ejecución de tareas. De ese modo, el `credentials-fetcher` daemon tendrá acceso al secreto de Secrets Manager. Reemplace el ejemplo de `MySecret` con el nombre de recurso de Amazon (ARN) de su secreto en la lista de `Resource`.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                   "arn:aws:ssm:arn:aws: 111122223333:secret:MySecret"
            ]
```

```
        }
    ]
}
```

> **Note**
>
> Si utiliza su propia clave de KMS para cifrar el secreto, debe agregar los permisos necesarios a este rol y agregar este rol a la política de claves de AWS KMS.

2. Decidir si utilizar SSM Parameter Store o S3 para almacenar el CredSpec

   Amazon ECS admite las siguientes formas de hacer referencia a la ruta de archivo en el campo `credentialSpecs` de una definición de tarea.

   Si une las instancias a un solo dominio, utilice el prefijo `credentialspec:` al principio del ARN de la cadena. Si utiliza la gMSA sin dominio, utilice `credentialspecdomainless:`.

   Para obtener más información sobre CredSpec, consulte Archivo de especificaciones de credenciales.

   - Bucket de Amazon S3

     Agregue la especificación de credenciales a un bucket de Amazon S3. A continuación, haga referencia al nombre de recurso de Amazon (ARN) del bucket de Amazon S3 en el campo `credentialSpecs` de la definición de tareas.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/
${ObjectName}"
            ],
            ...
        }
    ],
    ...
```

```
}
```

Para que las tareas tengan acceso al bucket de S3, agregue los siguientes permisos como una política en línea al rol de IAM de ejecución de tareas de Amazon ECS.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/{object}"
            ]
        }
    ]
}
```

- Parámetro de Parameter Store de SSM

  Agregue la especificación de credenciales a un parámetro de SSM Parameter Store. A continuación, haga referencia al nombre de recurso de Amazon (ARN) del parámetro de SSM Parameter Store en el campo `credentialSpecs` de la definición de tareas.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:ssm:aws-
region:111122223333:parameter/parameter_name"
```

```
            ],
            ...
        }
    ],
    ...
}
```

Para que las tareas tengan acceso al parámetro de SSM Parameter Store, agregue los siguientes permisos como una política en línea al rol de IAM de ejecución de tareas de Amazon ECS.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": [
            "arn:aws:ssm:arn:aws::111122223333:parameter/parameter_name"
            ]
        }
    ]
}
```

Archivo de especificaciones de credenciales

Amazon ECS utiliza un archivo de especificaciones de credenciales de Active Directory (CredSpec). Este archivo contiene los metadatos de gMSA utilizados para propagar el contexto de la cuenta de gMSA al contenedor de Linux. Genere las CredSpec y haga referencia a ellas en el campo `credentialSpecs` de la definición de tareas. El archivo de CredSpec no contiene ningún secreto.

A continuación se muestra un ejemplo de un archivo CredSpec.

```
{
    "CmsPlugins": [
        "ActiveDirectory"
```

```
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "example.com",
        "DnsName": "example.com",
        "NetBiosName": "example"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "example.com"
            }
        ],
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
    }
}
```

Creación de un CredSpec

Cree unas CredSpec utilizando el módulo PowerShell de CredSpec en una computadora con Windows que esté unida al dominio. Siga los pasos que se indican en Crear una especificación de credenciales en el sitio web de Microsoft Learn.

Uso de gMSA para contenedores de Linux en Fargate

Amazon ECS admite la autenticación de Active Directory para contenedores de Linux en Fargate con un tipo especial de cuenta de servicio denominada cuenta de servicio administrada de grupo (gMSA).

Las aplicaciones de red basadas en Linux, como las aplicaciones .NET Core, pueden utilizar Active Directory para facilitar la administración de autorizaciones y autenticación entre usuarios y servicios. Puede utilizar esta característica diseñando aplicaciones que se integren con Active Directory y se

ejecuten en servidores unidos a un dominio. Sin embargo, dado que los contenedores de Linux no se pueden unir a un dominio, debe configurar un contenedor de Linux para que se ejecute con gMSA.

Consideraciones

Tenga en cuenta lo siguiente antes de usar gMSA para contenedores de Linux en Fargate:

- Debe ejecutar la versión 1.4 de la plataforma o una posterior.
- Puede que necesite una computadora con Windows unida al dominio para cumplir los requisitos previos. Por ejemplo, puede que necesite una computadora con Windows que esté unida al dominio para crear la gMSA en Active Directory con PowerShell. Las herramientas de PowerShell en Active Directory RSAT solo están disponibles para Windows. Para obtener más información, consulte Instalar las herramientas de administración de Active Directory.
- Debe usar gMSA sin dominio.

  Amazon ECS utiliza un archivo de especificaciones de credenciales de Active Directory (CredSpec). Este archivo contiene los metadatos de gMSA utilizados para propagar el contexto de la cuenta de gMSA al contenedor. Debe generar el archivo CredSpec y, a continuación, almacenarlo en un bucket de Amazon S3.
- Una tarea solo puede admitir un Active Directory.

Requisitos previos

Antes de utilizar la característica de gMSA para contenedores de Linux con Amazon ECS, asegúrese de completar lo siguiente:

- Configure un dominio de Active Directory con los recursos a los que desea que accedan sus contenedores. Amazon ECS admite las siguientes configuraciones:
  - Un AWS Directory Service Active Directory. AWS Directory Service es un Active Directory administrado por AWS y alojado en Amazon EC2. Para obtener más información, consulte Introducción a Microsoft AD administrado por AWS en la Guía de administración de AWS Directory Service.
  - Un Active Directory en las instalaciones. Debe asegurarse de que la instancia de contenedor de Linux de Amazon ECS pueda unirse al dominio. Para obtener más información, consulte AWS Direct Connect.

- Debe tener una cuenta de gMSA existente en Active Directory y un usuario que tenga permiso para acceder a la cuenta de servicio gMSA. Para obtener más información, consulte Hacer que un usuario de Active Directory sea una gMSA sin dominio.
- Tiene un bucket de Amazon S3. Para obtener más información, consulte Crear un bucket en la Guía del usuario de Amazon S3.

Configuración de contenedores de Linux compatibles con gMSA en Amazon ECS

Preparar la infraestructura

Los siguientes pasos son consideraciones y la configuración que se realizan una vez.

- Hacer que un usuario de Active Directory sea una gMSA sin dominio

  Cuando se utiliza gMSA sin dominio, el contenedor no está unido al dominio. Otras aplicaciones que se ejecutan en el contenedor no pueden utilizar las credenciales para acceder al dominio. Las tareas que utilizan un dominio diferente se pueden ejecutar en el mismo contenedor. Usted proporciona el nombre de un secreto en AWS Secrets Manager en el archivo CredSpec. El secreto debe contener un nombre de usuario, una contraseña y el dominio para iniciar sesión.

  Esta característica es similar a la característica gMSA support for non-domain-joined container hosts. Para obtener más información sobre la característica de Windows, consulte Arquitectura y mejoras de gMSA en el sitio web de Microsoft Learn.

  a. Configure un usuario en el dominio de Active Directory. El usuario de Active Directory debe tener permiso para acceder a la cuenta de servicio de gMSA que utilice en las tareas.
  b. Tiene una VPC y subredes que pueden resolver el nombre de dominio de Active Directory. Configure las opciones de VPC con DHCP con el nombre de dominio que apunta al nombre del servicio de Active Directory. Para obtener más información sobre cómo configurar opciones de DHCP para una VPC, consulte Trabajar con los conjuntos de opciones de DHCP en la Guía del usuario de Amazon Virtual Private Cloud.
  c. Cree un secreto en AWS Secrets Manager.
  d. Cree el archivo de especificación de credenciales.

Configuración de permisos y secretos

Siga los siguientes pasos una vez por cada aplicación y definición de tarea. Recomendamos utilizar la práctica recomendada de concesión de privilegios mínimos y reducir los permisos que se utilizan en la política. De esta forma, cada tarea solo puede leer los secretos que necesita.

1. Cree un usuario en el dominio de Active Directory. El usuario de Active Directory debe tener permiso para acceder a las cuentas de servicio de gMSA que utilice en las tareas.
2. Cree un secreto en AWS Secrets Manager después de crear el usuario en Active Directory. Para obtener más información, consulte Crear un secreto de AWS Secrets Manager.
3. Introduzca el nombre de usuario, la contraseña y el dominio en pares clave-valor de JSON denominados `username`, `password` y `domainName`, respectivamente.

```
{"username":"username","password":"passw0rd", "domainName":"example.com"}
```

4. También debe agregar los siguientes permisos como una política en línea al rol de IAM de ejecución de tareas. De ese modo, el `credentials-fetcher` daemon tendrá acceso al secreto de Secrets Manager. Reemplace el ejemplo de `MySecret` con el nombre de recurso de Amazon (ARN) de su secreto en la lista de `Resource`.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
            "arn:aws:secretsmanager:arn:aws::111122223333:secret:MySecret"
            ]
        }
    ]
}
```

> **Note**
>
> Si utiliza su propia clave de KMS para cifrar el secreto, debe agregar los permisos necesarios a este rol y agregar este rol a la política de claves de AWS KMS.

5. Agregue la especificación de credenciales a un bucket de Amazon S3. A continuación, haga referencia al nombre de recurso de Amazon (ARN) del bucket de Amazon S3 en el campo `credentialSpecs` de la definición de tareas.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/${ObjectName}"
            ],
            ...
        }
    ],
    ...
}
```

Para que las tareas tengan acceso al bucket de S3, agregue los siguientes permisos como una política en línea al rol de IAM de ejecución de tareas de Amazon ECS.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListObjects"
```

```
            ],
            "Resource": [
                "arn:aws:s3:::{bucket_name}",
                "arn:aws:s3:::{bucket_name}/{object}"
            ]
        }
    ]
}
```

Archivo de especificaciones de credenciales

Amazon ECS utiliza un archivo de especificaciones de credenciales de Active Directory (CredSpec). Este archivo contiene los metadatos de gMSA utilizados para propagar el contexto de la cuenta de gMSA al contenedor de Linux. Genere las CredSpec y haga referencia a ellas en el campo `credentialSpecs` de la definición de tareas. El archivo de CredSpec no contiene ningún secreto.

A continuación se muestra un ejemplo de un archivo CredSpec.

```
{
    "CmsPlugins": [
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "example.com",
        "DnsName": "example.com",
        "NetBiosName": "example"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "example.com"
            }
        ],
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
```

```
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
    }
}
```

Creación de CredSpec y cómo cargarlo en Amazon S3

Cree unas CredSpec utilizando el módulo PowerShell de CredSpec en una computadora con Windows que esté unida al dominio. Siga los pasos que se indican en Crear una especificación de credenciales en el sitio web de Microsoft Learn.

Tras crear el archivo de especificación de credenciales, cárguelo a un bucket de Amazon S3. Copie el archivo CredSpec en la computadora o el entorno en el que esté ejecutando los comandos de la AWS CLI.

Ejecute el siguiente comando de la AWS CLI para cargar el CredSpec a Amazon S3. Reemplace *amzn-s3-demo-bucket* con el nombre de su bucket de Amazon S3. Puede almacenar el archivo como un objeto en cualquier bucket y ubicación, pero debe permitir el acceso a ese bucket y ubicación en la política que asocie al rol de ejecución de la tarea.

Para PowerShell, use el comando siguiente:

```
$ Write-S3Object -BucketName "amzn-s3-demo-bucket" -Key "ecs-domainless-gmsa-credspec"
 -File "gmsa-cred-spec.json"
```

El siguiente comando de la AWS CLI utiliza caracteres de continuación con barra invertida que utilizan los intérpretes de comandos compatibles y `sh`.

```
$ aws s3 cp gmsa-cred-spec.json \
s3://amzn-s3-demo-bucket/ecs-domainless-gmsa-credspec
```

Uso de contenedores de Amazon ECS para Windows con gMSA sin dominio mediante la AWS CLI

En el siguiente tutorial, se muestra cómo crear una tarea de Amazon ECS que ejecute un contenedor de Windows que tenga credenciales para acceder a Active Directory con la AWS CLI. Al usar una

gMSA sin dominio, la instancia de contenedor no se une al dominio, las demás aplicaciones de la instancia no pueden utilizar las credenciales para acceder al dominio y las tareas que unen diferentes dominios se pueden ejecutar en la misma instancia.

Temas

- Requisitos previos
- Paso 1: Crear y configurar la cuenta de gMSA en los servicios de dominio de Active Directory (AD DS)
- Paso 2: Cargar credenciales a Secrets Manager
- Paso 3: Modifique el JSON CredSpec para incluir información de gMSA sin dominio
- Paso 4: Cargar CredSpec a Amazon S3
- Paso 5: (Opcional) crear un clúster de Amazon ECS
- Paso 6: crear un rol de IAM para instancias de contenedor
- Paso 7: Crear un rol de ejecución de tarea personalizado
- Paso 8: crear un rol de tarea para Amazon ECS Exec
- Paso 9: Registrar una definición de tarea que utilice gMSA sin dominio
- Paso 10: Registrar una instancia de contenedor de Windows en el clúster
- Paso 11: Verificar la instancia de contenedor
- Paso 12: Ejecutar una tarea de Windows
- Paso 13: Comprobar que el contenedor tenga credenciales de gMSA
- Paso 14: Limpiar
- Depuración de gMSA sin dominio de Amazon ECS para contenedores de Windows

Requisitos previos

En este tutorial se supone que los siguientes requisitos previos se han completado:

- Se han completado los pasos que se indican en Configuración para utilizar Amazon ECS.
- Su usuario de IAM dispone de los permisos requeridos que se especifican en la política de IAM AmazonECS_FullAccess de ejemplo.
- La última versión de la AWS CLI está instalada y configurada. Para obtener más información acerca de cómo instalar o actualizar la AWS CLI, consulte Instalación de la AWS Command Line Interface.

> **Note**
>
> Puede utilizar puntos de conexión de servicio de doble pila para interactuar con Amazon ECS desde la AWS CLI, los SDK y la API de Amazon ECS a través de IPv4 e IPv6. Para obtener más información, consulte Uso de puntos de conexión de doble pila en Amazon ECS.

- Configure un dominio de Active Directory con los recursos a los que desea que accedan sus contenedores. Amazon ECS admite las siguientes configuraciones:
  - Un AWS Directory Service Active Directory. AWS Directory Service es un Active Directory administrado por AWS y alojado en Amazon EC2. Para obtener más información, consulte Introducción a Microsoft AD administrado por AWS en la Guía de administración de AWS Directory Service.
  - Un Active Directory en las instalaciones. Debe asegurarse de que la instancia de contenedor de Linux de Amazon ECS pueda unirse al dominio. Para obtener más información, consulte AWS Direct Connect.
- Tiene una VPC y subredes que pueden resolver el nombre de dominio de Active Directory.
- Puede elegir entre gMSA sin dominio o unir cada instancia a un único dominio. Al usar una gMSA sin dominio, la instancia de contenedor no se une al dominio, las demás aplicaciones de la instancia no pueden utilizar las credenciales para acceder al dominio y las tareas que unen diferentes dominios se pueden ejecutar en la misma instancia.

  A continuación, seleccione el almacenamiento de datos para las CredSpec y, de forma opcional, para las credenciales de usuario de Active Directory para gMSA sin dominio.

  Amazon ECS utiliza un archivo de especificaciones de credenciales de Active Directory (CredSpec). Este archivo contiene los metadatos de gMSA utilizados para propagar el contexto de la cuenta de gMSA al contenedor. Genera el archivo de CredSpec y, a continuación, lo almacena en una de las opciones de almacenamiento de CredSpec de la siguiente tabla, específica del sistema operativo de las instancias de contenedor. Para usar el método sin dominio, en una sección opcional del archivo de CredSpec se pueden especificar las credenciales de una de las opciones de almacenamiento domainless user credentials de la siguiente tabla, específicas del sistema operativo de las instancias de contenedor.

Ubicación de almacenamiento	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	credenciales de usuario sin dominio	credenciales de usuario sin dominio
Parameter Store de Amazon EC2 Systems Manager	CredSpec	CredSpec, credenciales de usuario sin dominio
Archivo local	N/A	CredSpec

- (Opcional) AWS CloudShell es una herramienta que proporciona a los clientes una línea de comandos sin necesidad de crear su propia instancia de EC2. Para obtener más información, consulte ¿Qué es AWS CloudShell? en la Guía del usuario de AWS CloudShell.

Paso 1: Crear y configurar la cuenta de gMSA en los servicios de dominio de Active Directory (AD DS)

Cree y configure una cuenta de gMSA en el dominio de Active Directory.

> **Note**
>
> Este paso crea dos cuentas independientes: una cuenta de servicio administrado por grupos (gMSA) que proporciona la identidad para los contenedores y una cuenta de usuario normal que se utiliza para la autenticación de dominio. Estas cuentas tienen propósitos distintos y deben tener nombres distintos.

1. Generar una clave raíz del servicio de distribución de claves

   > **Note**
   >
   > Si utiliza AWS Directory Service, puede omitir este paso.

La clave raíz de KDS y los permisos de gMSA se configuran con su Microsoft AD administrado por AWS.

Si aún no ha creado una cuenta de servicio de gMSA en su dominio, primero tendrá que generar una clave raíz del Servicio de distribución de claves (KDS). El KDS es responsable de crear, rotar y divulgar la contraseña de gMSA a los hosts autorizados. Cuando `ccg.exe` necesita recuperar las credenciales de gMSA, se pone en contacto con KDS para recuperar la contraseña actual.

Para comprobar si la clave raíz de KDS ya se ha creado, ejecute el siguiente cmdlet de PowerShell con privilegios de administrador de dominio en un controlador de dominio mediante el módulo PowerShell de `ActiveDirectory`. Para obtener más información sobre el módulo, consulte el Módulo de ActiveDirectory en el sitio web de Microsoft Learn.

```
PS C:\> Get-KdsRootKey
```

Si el comando devuelve un ID de clave, puede omitir el resto de este paso. De lo contrario, cree la clave raíz de KDS mediante el siguiente comando:

```
PS C:\> Add-KdsRootKey -EffectiveImmediately
```

Aunque el argumento `EffectiveImmediately` al comando implica que la clave entra en vigor inmediatamente, debe esperar 10 horas antes de que la clave raíz de KDS se replique y esté disponible para su uso en todos los controladores de dominio.

2. Crear la cuenta de gMSA

   Para crear la cuenta de gMSA y permitir que `ccg.exe` recupere la contraseña de gMSA, ejecute los siguientes comandos de PowerShell desde un servidor o cliente de Windows con acceso al dominio. Sustituya `ExampleAccount` por el nombre que desee para la cuenta gMSA y sustituya `example-domain` por el nombre de su dominio de Active Directory (por ejemplo, si su dominio es `contoso.com`, utilice `contoso`).

   a.
   ```
   PS C:\> Install-WindowsFeature RSAT-AD-PowerShell
   ```

   b.
   ```
   PS C:\> New-ADGroup -Name "ExampleAccount Authorized Hosts" -SamAccountName
    "ExampleAccountHosts" -GroupScope DomainLocal
   ```

c.
```
PS C:\> New-ADServiceAccount -Name "ExampleAccount" -DnsHostName "example-domain" -ServicePrincipalNames "host/ExampleAccount", "host/example-domain" -PrincipalsAllowedToRetrieveManagedPassword "ExampleAccountHosts"
```

d. Cree un usuario con una contraseña permanente que no caduque. Estas credenciales se almacenan en AWS Secrets Manager y utilizan en cada tarea para unirse al dominio. Se trata de una cuenta de usuario independiente de la cuenta gMSA creada anteriormente. Sustituya `ExampleServiceUser` por el nombre que desee para esta cuenta de usuario de servicio.

```
PS C:\> New-ADUser -Name "ExampleServiceUser" -AccountPassword (ConvertTo-SecureString -AsPlainText "Test123" -Force) -Enabled 1 -PasswordNeverExpires 1
```

e.
```
PS C:\> Add-ADGroupMember -Identity "ExampleAccountHosts" -Members
 "ExampleServiceUser"
```

f. Instale el módulo de PowerShell para crear objetos de CredSpec en Active Directory y generar el JSON CredSpec.

```
PS C:\> Install-PackageProvider -Name NuGet -Force
```

```
PS C:\> Install-Module CredentialSpec
```

g.
```
PS C:\> New-CredentialSpec -AccountName ExampleAccount
```

3. Copie la salida JSON del comando anterior en un archivo llamado `gmsa-cred-spec.json`. Este es el archivo CredSpec. Se usa en el paso 3, Paso 3: Modifique el JSON CredSpec para incluir información de gMSA sin dominio.

Paso 2: Cargar credenciales a Secrets Manager

Copie las credenciales de Active Directory en un sistema de almacenamiento de credenciales seguro para que cada tarea las recupere. Este es el método de gMSA sin dominio. Al usar una gMSA sin dominio, la instancia de contenedor no se une al dominio, las demás aplicaciones de la instancia no pueden utilizar las credenciales para acceder al dominio y las tareas que unen diferentes dominios se pueden ejecutar en la misma instancia.

En este paso se utiliza la AWS CLI. Puede ejecutar estos comandos en AWS CloudShell en el intérprete de comandos predeterminado, que es `bash`.

- Ejecute el siguiente comando de AWS CLI y sustituya el nombre de usuario, la contraseña y el nombre de dominio por los correspondientes a su entorno. Utilice el nombre de la cuenta de usuario del servicio (no el nombre de la cuenta gMSA) como nombre de usuario. Guardar el ARN del secreto para usarlo en el siguiente paso, Paso 3: Modifique el JSON CredSpec para incluir información de gMSA sin dominio

  El siguiente comando utiliza caracteres de continuación con barra invertida que utilizan los intérpretes de comandos compatibles y `sh`. No se admite este comando con PowerShell. Debe modificar el comando para usarlo con PowerShell.

```
$ aws secretsmanager create-secret \
--name gmsa-plugin-input \
--description "Amazon ECS - gMSA Portable Identity." \
--secret-string "{\"username\":\"ExampleServiceUser\",\"password\":\"Test123\",
\"domainName\":\"contoso.com\"}"
```

Paso 3: Modifique el JSON CredSpec para incluir información de gMSA sin dominio

Antes de subir las CredSpec a una de las opciones de almacenamiento, agregue información a las CredSpec con el ARN del secreto en Secrets Manager del paso anterior. Para obtener más información, consulte Configuración de especificaciones de credenciales adicionales para el caso de uso de hosts de contenedores no unidos a un dominio en el sitio web de Microsoft Learn.

1. Agregue la siguiente información al archivo de CredSpec que se encuentra dentro del `ActiveDirectoryConfig`. Reemplace el ARN con el secreto de Secrets Manager del paso anterior.

   Tenga en cuenta que el valor de `PluginGUID` debe coincidir con el GUID del siguiente fragmento de ejemplo y es obligatorio.

```
"HostAccountConfig": {
      "PortableCcgVersion": "1",
      "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
```

```
      "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
```

También puede utilizar un secreto en SSM Parameter Store mediante el ARN en este formato: `\"arn:aws:ssm:aws-region:111122223333:parameter/gmsa-plugin-input\"`.

2. Tras modificar el archivo de CredSpec, tendrá un aspecto semejante al de este ejemplo:

```
{
  "CmsPlugins": [
    "ActiveDirectory"
  ],
  "DomainJoinConfig": {
    "Sid": "S-1-5-21-4066351383-705263209-1606769140",
    "MachineAccountName": "ExampleAccount",
    "Guid": "ac822f13-583e-49f7-aa7b-284f9a8c97b6",
    "DnsTreeName": "example-domain",
    "DnsName": "example-domain",
    "NetBiosName": "example-domain"
  },
  "ActiveDirectoryConfig": {
    "GroupManagedServiceAccounts": [
      {
        "Name": "ExampleAccount",
        "Scope": "example-domain"
      },
      {
        "Name": "ExampleAccount",
        "Scope": "example-domain"
      }
    ],
    "HostAccountConfig": {
      "PortableCcgVersion": "1",
      "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
      "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
  }
}
```

Paso 4: Cargar CredSpec a Amazon S3

En este paso se utiliza la AWS CLI. Puede ejecutar estos comandos en AWS CloudShell en el intérprete de comandos predeterminado, que es `bash`.

1. Copie el archivo CredSpec en la computadora o el entorno en el que esté ejecutando los comandos de la AWS CLI.
2. Ejecute el siguiente comando de la AWS CLI para cargar el CredSpec a Amazon S3. Reemplace `MyBucket` con el nombre de su bucket de Amazon S3. Puede almacenar el archivo como un objeto en cualquier bucket y ubicación, pero debe permitir el acceso a ese bucket y ubicación en la política que asocie al rol de ejecución de la tarea.

   El siguiente comando utiliza caracteres de continuación con barra invertida que utilizan los intérpretes de comandos compatibles y `sh`. No se admite este comando con PowerShell. Debe modificar el comando para usarlo con PowerShell.

   ```
   $ aws s3 cp gmsa-cred-spec.json \
   s3://MyBucket/ecs-domainless-gmsa-credspec
   ```

Paso 5: (Opcional) crear un clúster de Amazon ECS

De forma predeterminada, su cuenta tiene un clúster de Amazon ECS denominado `default`. Este clúster se utiliza de forma predeterminada en AWS CLI, los SDK y AWS CloudFormation. Puede utilizar clústeres adicionales para agrupar y organizar las tareas y la infraestructura, y asignar valores predeterminados para algunas configuraciones.

Puede crear un clúster con la AWS Management Console, la AWS CLI, los SDK o AWS CloudFormation. Los ajustes y la configuración del clúster no afectan gMSA.

En este paso se utiliza la AWS CLI. Puede ejecutar estos comandos en AWS CloudShell en el intérprete de comandos predeterminado, que es `bash`.

```
$ aws ecs create-cluster --cluster-name windows-domainless-gmsa-cluster
```

> **Important**
>
> Si decide crear su propio clúster, debe especificar `--cluster clusterName` para cada comando que pretenda utilizar con dicho clúster.

Paso 6: crear un rol de IAM para instancias de contenedor

Una instancia de contenedor es una computadora host para ejecutar contenedores en tareas de ECS, por ejemplo, instancias de Amazon EC2. Cada instancia de contenedor se registra en un clúster de Amazon ECS. Antes de lanzar instancias de Amazon EC2 y registrarlas en un clúster, debe crear un rol de IAM para que lo utilicen las instancias de contenedor.

Para crear el rol de instancia de contenedor, consulte Rol de IAM de instancia de contenedor de Amazon ECS. La opción predeterminada `ecsInstanceRole` tiene permisos suficientes para completar este tutorial.

Paso 7: Crear un rol de ejecución de tarea personalizado

Amazon ECS puede usar un rol de IAM diferente para los permisos necesarios para iniciar cada tarea, en lugar del rol de instancia de contenedor. Este rol se denomina rol de ejecución de tareas. Recomendamos crear un rol de ejecución de tareas con solo los permisos necesarios para que ECS ejecute la tarea, también conocidos como permisos con privilegios mínimos. Para obtener más información sobre el principio de privilegios mínimos, consulte SEC03-BP02 Conceder acceso con privilegios mínimos en el AWS Well-Architected Framework.

1. Para crear un rol de ejecución de tareas, consulte Creación del rol de de ejecución de tareas. Los permisos predeterminados permiten a la instancia de contenedor extraer imágenes de contenedores de Amazon Elastic Container Registry y `stdout` y `stderr` de sus aplicaciones para registrarlas en registros de Amazon CloudWatch.

   Como el rol necesita permisos personalizados para este tutorial, puede asignarle un nombre diferente al de `ecsTaskExecutionRole`. Este tutorial utiliza `ecsTaskExecutionRole` en otros pasos.
2. Agregue los siguientes permisos creando una política personalizada, ya sea una política en línea que solo exista para este rol o una política que pueda reutilizar. Sustituya el ARN con el `Resource` en la primera declaración con el bucket y la ubicación de Amazon S3, y el segundo `Resource` con el ARN del secreto de Secrets Manager.

Si cifra el secreto en Secrets Manager con una clave personalizada, también debe permitir `kms:Decrypt` para la clave.

Si utiliza SSM Parameter Store en lugar de Secrets Manager, debe permitir `ssm:GetParameter` para el parámetro, en lugar de `secretsmanager:GetSecretValue`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::MyBucket/ecs-domainless-gmsa-credspec/
gmsa-cred-spec.json"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource":
 "arn:aws:secretsmanager:arn:aws:: 111122223333:secret:gmsa-plugin-AbCdEf"
        }
    ]
}
```

Paso 8: crear un rol de tarea para Amazon ECS Exec

En este tutorial, se utiliza Amazon ECS Exec para verificar la funcionalidad mediante la ejecución de un comando dentro de una tarea en ejecución. Para usar ECS Exec, el servicio o la tarea debe activar ECS Exec y el rol de la tarea (pero no el rol de ejecución de la tarea) debe tener permisos de `ssmmessages`. Para obtener información sobre la política de IAM necesaria, consulte Permisos de ECS Exec.

En este paso se utiliza la AWS CLI. Puede ejecutar estos comandos en AWS CloudShell en el intérprete de comandos predeterminado, que es `bash`.

Para crear un rol de tarea mediante la AWS CLI, siga estos pasos.

1. Cree un archivo denominado `ecs-tasks-trust-policy.json` con el siguiente contenido:

   JSON

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

2. Crear un rol de IAM. Puede reemplazar el nombre `ecs-exec-demo-task-role`, pero consérvelo para los siguientes pasos.

   El siguiente comando utiliza caracteres de continuación con barra invertida que utilizan los intérpretes de comandos compatibles y `sh`. No se admite este comando con PowerShell. Debe modificar el comando para usarlo con PowerShell.

   ```
   $ aws iam create-role --role-name ecs-exec-demo-task-role \
   --assume-role-policy-document file://ecs-tasks-trust-policy.json
   ```

   Puede eliminar el archivo `ecs-tasks-trust-policy.json`.

3. Cree un archivo denominado `ecs-exec-demo-task-role-policy.json` con el siguiente contenido:

   JSON

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

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssmmessages:CreateControlChannel",
                "ssmmessages:CreateDataChannel",
                "ssmmessages:OpenControlChannel",
                "ssmmessages:OpenDataChannel"
            ],
            "Resource": "*"
        }
    ]
}
```

4. Cree una política de IAM y asóciela al rol del paso anterior.

   El siguiente comando utiliza caracteres de continuación con barra invertida que utilizan los intérpretes de comandos compatibles y `sh`. No se admite este comando con PowerShell. Debe modificar el comando para usarlo con PowerShell.

```
$ aws iam put-role-policy \
    --role-name ecs-exec-demo-task-role \
    --policy-name ecs-exec-demo-task-role-policy \
    --policy-document file://ecs-exec-demo-task-role-policy.json
```

   Puede eliminar el archivo `ecs-exec-demo-task-role-policy.json`.

Paso 9: Registrar una definición de tarea que utilice gMSA sin dominio

En este paso se utiliza la AWS CLI. Puede ejecutar estos comandos en AWS CloudShell en el intérprete de comandos predeterminado, que es `bash`.

1. Cree un archivo denominado `windows-gmsa-domainless-task-def.json` con el siguiente contenido:

```
{
  "family": "windows-gmsa-domainless-task",
  "containerDefinitions": [
    {
      "name": "windows_sample_app",
```

```
      "image": "mcr.microsoft.com/windows/servercore/iis",
      "cpu": 1024,
      "memory": 1024,
      "essential": true,
      "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::ecs-domainless-gmsa-
credspec/gmsa-cred-spec.json"
      ],
      "entryPoint": [
        "powershell",
        "-Command"
      ],
      "command": [
        "New-Item -Path C:\\inetpub\\wwwroot\\index.html -ItemType file -Value
 '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>' -Force ; C:\
\ServiceMonitor.exe w3svc"
      ],
      "portMappings": [
        {
          "protocol": "tcp",
          "containerPort": 80,
          "hostPort": 8080
        }
      ]
    }
  ],
  "taskRoleArn": "arn:aws:iam::111122223333:role/ecs-exec-demo-task-role",
  "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole"
}
```

2. Registre la definición de tarea ejecutando el siguiente comando:

   El siguiente comando utiliza caracteres de continuación con barra invertida que utilizan los intérpretes de comandos compatibles y `sh`. No se admite este comando con PowerShell. Debe modificar el comando para usarlo con PowerShell.

```
$ aws ecs register-task-definition \
--cli-input-json file://windows-gmsa-domainless-task-def.json
```

Paso 10: Registrar una instancia de contenedor de Windows en el clúster

Lance una instancia de Amazon EC2 para Windows y ejecute el agente de contenedor de ECS para registrarla como instancia de contenedor en el clúster. ECS ejecuta las tareas en las instancias de contenedor que están registradas en el clúster en el que se inician las tareas.

1. Para lanzar una instancia de Amazon EC2 para Windows que esté configurada para Amazon ECS en la AWS Management Console, consulte Lanzamiento de una instancia de contenedor de Windows de Amazon ECS. Deténgase en el paso correspondiente a los datos de usuario.
2. Para gMSA, los datos del usuario deben configurar la variable de entorno `ECS_GMSA_SUPPORTED` antes de iniciar el agente de contenedor de ECS.

   En el caso de ECS Exec, el agente debe empezar con el argumento `-EnableTaskIAMRole`.

   Para proteger el rol de IAM de la instancia impidiendo que las tareas lleguen al servicio web IMDS de EC2 para recuperar las credenciales del rol, agregue el argumento `-AwsvpcBlockIMDS`. Esto solo se aplica a las tareas que utilizan el modo de red de `awsvpc`.

   ```
   <powershell>
   [Environment]::SetEnvironmentVariable("ECS_GMSA_SUPPORTED", $TRUE, "Machine")
   Import-Module ECSTools
   Initialize-ECSAgent -Cluster windows-domainless-gmsa-cluster -EnableTaskIAMRole -
   AwsvpcBlockIMDS
   </powershell>
   ```

3. Revise un resumen de la configuración de su instancia en el panel Summary (Resumen); cuando haya terminado, elija Launch instance.

Paso 11: Verificar la instancia de contenedor

Puede comprobar que hay una instancia de contenedor en el clúster mediante la AWS Management Console. Sin embargo, gMSA necesita otras características que se indican como atributos. Estos atributos no están visibles en la AWS Management Console, por lo que en este tutorial se utiliza la AWS CLI.

En este paso se utiliza la AWS CLI. Puede ejecutar estos comandos en AWS CloudShell en el intérprete de comandos predeterminado, que es `bash`.

1. Haga una lista de las instancias de contenedor del clúster. Las instancias de contenedor tienen un ID diferente del ID de la instancia de EC2.

```
$ aws ecs list-container-instances
```

Salida:

```
{
    "containerInstanceArns": [
        "arn:aws:ecs:aws-region:111122223333:container-
instance/default/MyContainerInstanceID"
    ]
}
```

Por ejemplo, `526bd5d0ced448a788768334e79010fd` es un ID de instancia de contenedor válido.

2. Utilice el ID de instancia de contenedor del paso anterior para obtener los detalles de la instancia de contenedor. Reemplace `MyContainerInstanceID` por el ID.

El siguiente comando utiliza caracteres de continuación con barra invertida que utilizan los intérpretes de comandos compatibles y `sh`. No se admite este comando con PowerShell. Debe modificar el comando para usarlo con PowerShell.

```
$ aws ecs describe-container-instances \
     ----container-instances MyContainerInstanceID
```

Tenga en cuenta que el resultado es muy largo.

3. Compruebe que la lista de `attributes` tenga un objeto con la clave llamado `name` y un valor `ecs.capability.gmsa-domainless`. A continuación, se muestra un ejemplo del objeto.

Salida:

```
{
    "name": "ecs.capability.gmsa-domainless"
}
```

Paso 12: Ejecutar una tarea de Windows

Ejecute una tarea de Amazon ECS. Si solo hay una instancia de contenedor en el clúster, puede usar `run-task`. Si hay muchas instancias de contenedor diferentes, puede ser más fácil utilizar `start-task` y especificar el ID de la instancia de contenedor en la que ejecutará la tarea, antes que agregar restricciones de ubicación a la definición de la tarea para controlar en qué tipo de instancia de contenedor se va a ejecutar la tarea.

En este paso se utiliza la AWS CLI. Puede ejecutar estos comandos en AWS CloudShell en el intérprete de comandos predeterminado, que es `bash`.

1. El siguiente comando utiliza caracteres de continuación con barra invertida que utilizan los intérpretes de comandos compatibles y `sh`. No se admite este comando con PowerShell. Debe modificar el comando para usarlo con PowerShell.

   ```
   aws ecs run-task --task-definition windows-gmsa-domainless-task \
       --enable-execute-command --cluster windows-domainless-gmsa-cluster
   ```

   Tenga en cuenta el ID de la tarea que devuelve el comando.

2. Ejecute el siguiente comando para comprobar que la tarea ha comenzado. Este comando espera y no devuelve el indicador del intérprete de comandos hasta que se inicie la tarea. Sustituya `MyTaskID` por el ID de la tarea del paso anterior.

   ```
   $ aws ecs wait tasks-running --task MyTaskID
   ```

Paso 13: Comprobar que el contenedor tenga credenciales de gMSA

Compruebe que el contenedor de la tarea tenga un token Kerberos. gMSA

En este paso se utiliza la AWS CLI. Puede ejecutar estos comandos en AWS CloudShell en el intérprete de comandos predeterminado, que es `bash`.

1. El siguiente comando utiliza caracteres de continuación con barra invertida que utilizan los intérpretes de comandos compatibles y `sh`. No se admite este comando con PowerShell. Debe modificar el comando para usarlo con PowerShell.

```
$ aws ecs execute-command \
--task MyTaskID \
--container windows_sample_app \
--interactive \
--command powershell.exe
```

El resultado será un indicador de PowerShell.

2. Ejecute el siguiente comando en la terminal de PowerShell dentro del contenedor.

```
PS C:\> klist get ExampleAccount$
```

En la salida, tenga en cuenta que `Principal` es la que creó anteriormente.

Paso 14: Limpiar

Cuando termine este tutorial, debe limpiar los recursos asociados para evitar incurrir en cargos generados por recursos sin utilizar.

En este paso se utiliza la AWS CLI. Puede ejecutar estos comandos en AWS CloudShell en el intérprete de comandos predeterminado, que es `bash`.

1. Detenga la tarea. Sustituya `MyTaskID` por el ID de la tarea del paso 12, Paso 12: Ejecutar una tarea de Windows.

```
$ aws ecs stop-task --task MyTaskID
```

2. Termine la instancia de Amazon EC2. Después, la instancia de contenedor del clúster se eliminará automáticamente al cabo de una hora.

   Puede buscar y terminar la instancia con la consola de Amazon EC2. O bien, puede ejecutar el siguiente comando. Para ejecutar el comando, busque el ID de instancia de EC2 en el resultado del comando `aws ecs describe-container-instances` del paso 1, Paso 11: Verificar la instancia de contenedor. El ID i-10a64379 es un ejemplo de un ID de instancia de EC2.

```
$ aws ec2 terminate-instances --instance-ids MyInstanceID
```

3. Elimine el archivo CredSpec en Amazon S3. Reemplace `MyBucket` con el nombre de su bucket de Amazon S3.

```
$ aws s3api delete-object --bucket MyBucket --key ecs-domainless-gmsa-credspec/
gmsa-cred-spec.json
```

4. Elimine el secreto de Secrets Manager. Si utilizó SSM Parameter Store en su lugar, elimine el parámetro.

   El siguiente comando utiliza caracteres de continuación con barra invertida que utilizan los intérpretes de comandos compatibles y `sh`. No se admite este comando con PowerShell. Debe modificar el comando para usarlo con PowerShell.

```
$ aws secretsmanager delete-secret --secret-id gmsa-plugin-input \
     --force-delete-without-recovery
```

5. Anule el registro y elimine la definición de tarea. Al anular el registro de la definición de tarea, se marca como inactiva para que no se pueda utilizar para iniciar nuevas tareas. A continuación, puede eliminar la definición de tarea.

   a. Anule el registro de la definición de la tarea especificando la versión. ECS crea automáticamente versiones de las definiciones de tareas, numeradas a partir del 1. Se hace referencia a las versiones en el mismo formato que las etiquetas de las imágenes de los contenedores, por ejemplo `:1`.

```
$ aws ecs deregister-task-definition --task-definition windows-gmsa-domainless-
task:1
```

   b. Elimine la definición de tareas.

```
$ aws ecs delete-task-definitions --task-definition windows-gmsa-domainless-
task:1
```

6. (Opcional) Elimine el clúster de ECS si creó un clúster.

```
$ aws ecs delete-cluster --cluster windows-domainless-gmsa-cluster
```

Depuración de gMSA sin dominio de Amazon ECS para contenedores de Windows

Estado de tarea de Amazon ECS

ECS intenta iniciar una tarea exactamente una vez. Cualquier tarea que tenga un problema se detiene y se establece en el estado `STOPPED`. Hay dos tipos comunes de problemas con las tareas. En primer lugar, las tareas que no se pudieron iniciar. En segundo lugar, las tareas en las que la aplicación se ha detenido dentro de uno de los contenedores. En la AWS Management Console, observe el campo Motivo de la detención de la tarea y busque el motivo por el que se detuvo la tarea. En la AWS CLI, describa la tarea y observe el `stoppedReason`. Para ver los pasos a seguir en laAWS Management Console y AWS CLI, consulte Visualización de los errores de las tareas detenidas de Amazon ECS.

Eventos de Windows

Los eventos de Windows para gMSA que se encuentran en contenedores se registran en el archivo de registro de `Microsoft-Windows-Containers-CCG` y se encuentran en el Visor de eventos, en la sección Aplicaciones y servicios de `Logs\Microsoft\Windows\Containers-CCG\Admin`. Para obtener más consejos de depuración, consulte Solución de problemas de gMSA para contenedores de Windows en el sitio web de Microsoft Learn.

Complemento de gMSA para agente de ECS

El registro del complemento de gMSA para el agente de ECS en la instancia de contenedor de Windows se encuentra en el siguiente directorio, `C:/ProgramData/Amazon/gmsa-plugin/`. Consulte este registro para comprobar si las credenciales de usuario sin dominio se descargaron de la ubicación de almacenamiento, como Secrets Manager, y si el formato de la credencial se leyó correctamente.

Obtenga información sobre cómo utilizar gMSA para contenedores de EC2 para Windows en Amazon ECS.

Amazon ECS admite la autenticación de Active Directory para contenedores de Windows a través de un tipo especial de cuenta de servicio denominada cuenta de servicio administrada de grupo (gMSA).

Las aplicaciones de red basadas en Windows, como las aplicaciones .NET, suelen utilizar Active Directory para facilitar la gestión de autorizaciones y la autenticación entre usuarios y servicios. Normalmente, los desarrolladores diseñan sus aplicaciones para que se integren con Active Directory y se ejecuten en servidores unidos a dominios para este propósito. Dado que los contenedores de Windows no se pueden unir a un dominio, debe configurar un contenedor de Windows para que se ejecute con gMSA.

Un contenedor de Windows que se ejecuta con gMSA depende de su instancia de Amazon EC2 host para recuperar las credenciales de gMSA del controlador de dominio de Active Directory y proporcionárselas a la instancia de contenedor. Para obtener más información, consulte Crear GMSA para contenedores de Windows.

> **Note**
>
> Esta característica no es compatible con los contenedores de Windows de Fargate.

Temas

- Consideraciones
- Requisitos previos
- Configuración de gMSA para contenedores de Windows en Amazon ECS

Consideraciones

Cuando se utilizan gMSA para contenedores de Windows, se debe tener en cuenta lo siguiente:

- Al utilizar la AMI de Windows Server 2016 Full optimizada para Amazon ECS para las instancias de contenedor, el nombre de host del contenedor debe ser el mismo que el nombre de cuenta de gMSA definido en el archivo de especificación de credenciales. Para especificar un nombre de host para un contenedor, utilice el parámetro `hostname` de definición de contenedor. Para obtener más información, consulte Configuración de red.
- Puede elegir entre gMSA sin dominio o unir cada instancia a un único dominio. Al usar una gMSA sin dominio, la instancia de contenedor no se une al dominio, las demás aplicaciones de la instancia no pueden utilizar las credenciales para acceder al dominio y las tareas que unen diferentes dominios se pueden ejecutar en la misma instancia.

  A continuación, seleccione el almacenamiento de datos para las CredSpec y, de forma opcional, para las credenciales de usuario de Active Directory para gMSA sin dominio.

  Amazon ECS utiliza un archivo de especificaciones de credenciales de Active Directory (CredSpec). Este archivo contiene los metadatos de gMSA utilizados para propagar el contexto de la cuenta de gMSA al contenedor. Genera el archivo de CredSpec y, a continuación, lo almacena en una de las opciones de almacenamiento de CredSpec de la siguiente tabla, específica del sistema operativo de las instancias de contenedor. Para usar el método sin dominio, en una

sección opcional del archivo de CredSpec se pueden especificar las credenciales de una de las opciones de almacenamiento domainless user credentials de la siguiente tabla, específicas del sistema operativo de las instancias de contenedor.

Ubicación de almacenamiento	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	credenciales de usuario sin dominio	credenciales de usuario sin dominio
Parameter Store de Amazon EC2 Systems Manager	CredSpec	CredSpec, credenciales de usuario sin dominio
Archivo local	N/A	CredSpec

Requisitos previos

Antes de utilizar la característica de gMSA para contenedores de Windows con Amazon ECS, asegúrese de completar lo siguiente:

- Configure un dominio de Active Directory con los recursos a los que desea que accedan sus contenedores. Amazon ECS admite las siguientes configuraciones:
  - Un AWS Directory Service Active Directory. AWS Directory Service es un Active Directory administrado por AWS y alojado en Amazon EC2. Para obtener más información, consulte Introducción a Microsoft AD administrado por AWS en la Guía de administración de AWS Directory Service.
  - Un Active Directory en las instalaciones. Debe asegurarse de que la instancia de contenedor de Linux de Amazon ECS pueda unirse al dominio. Para obtener más información, consulte AWS Direct Connect.
- Tiene una cuenta de gMSA existente en Active Directory. Para obtener más información, consulte Crear GMSA para contenedores de Windows.

- Eligió utilizar gMSA sin dominio o la instancia de contenedor de Windows de Amazon ECS en la que se aloja la tarea de Amazon ECS que debe ser un dominio unido a Active Directory y ser miembro del grupo de seguridad de Active Directory que tenga acceso a la cuenta de gMSA.

  Al usar una gMSA sin dominio, la instancia de contenedor no se une al dominio, las demás aplicaciones de la instancia no pueden utilizar las credenciales para acceder al dominio y las tareas que unen diferentes dominios se pueden ejecutar en la misma instancia.
- Agregó los permisos necesarios de IAM. Los permisos necesarios dependen de los métodos que elija para las credenciales iniciales y para almacenar la especificación de las credenciales:
  - Si utiliza las gMSA sin dominio para credenciales iniciales, se requieren permisos de IAM para AWS Secrets Manager en el rol de instancias de Amazon EC2.
  - Si almacena la especificación de credenciales en SSM Parameter Store, se requieren permisos de IAM para Parameter Store de Amazon EC2 Systems Manager en el rol de ejecución de la tarea.
  - Si almacena la especificación de credenciales en Amazon S3, se requieren permisos de IAM para Amazon Simple Storage Service en el rol de ejecución de tareas.

Configuración de gMSA para contenedores de Windows en Amazon ECS

Para configurar gMSA para contenedores de Windows en Amazon ECS, puede seguir el tutorial completo que incluye la configuración de los requisitos previos Uso de contenedores de Amazon ECS para Windows con gMSA sin dominio mediante la AWS CLI.

En las siguientes secciones, se describe la configuración de CredSpec en detalle.

Temas

- Ejemplo CredSpec
- Configuración de gMSA sin dominio
- Cómo hacer referencia a un archivo de especificación de credenciales en una definición de tareas

Ejemplo CredSpec

Amazon ECS utiliza un archivo de especificación de credenciales que contiene los metadatos de gMSA utilizados para propagar el contexto de la cuenta de gMSA al contenedor de Windows. Puede generar el archivo de especificación de credenciales y hacer referencia a él en el campo

`credentialSpec` de la definición de tareas. El archivo de especificación de credenciales no contiene ningún secreto.

A continuación se muestra un ejemplo de archivo de especificación de credenciales:

```
{
    "CmsPlugins": [
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "contoso.com",
        "DnsName": "contoso.com",
        "NetBiosName": "contoso"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "contoso.com"
            }
        ]
    }
}
```

Configuración de gMSA sin dominio

Recomendamos gMSA sin dominio en lugar de unir las instancias de contenedor a un solo dominio. Al usar una gMSA sin dominio, la instancia de contenedor no se une al dominio, las demás aplicaciones de la instancia no pueden utilizar las credenciales para acceder al dominio y las tareas que unen diferentes dominios se pueden ejecutar en la misma instancia.

1. Antes de subir las CredSpec a una de las opciones de almacenamiento, agregue información a las CredSpec con el ARN del secreto en Secrets Manager o en SSM Parameter Store. Para obtener más información, consulte Configuración de especificaciones de credenciales adicionales para el caso de uso de hosts de contenedores no unidos a un dominio en el sitio web de Microsoft Learn.

   Formato de credenciales de gMSA sin dominio

El siguiente es el formato JSON para las credenciales de gMSA sin dominio de Active Directory. Almacene las credenciales en Secrets Manager o SSM Parameter Store.

```
{
    "username":"WebApp01",
    "password":"Test123!",
    "domainName":"contoso.com"
}
```

2. Agregue la siguiente información al archivo de CredSpec que se encuentra dentro del `ActiveDirectoryConfig`. Sustituya el ARN por el secreto en Secrets Manager o SSM Parameter Store.

   Tenga en cuenta que el valor de `PluginGUID` debe coincidir con el GUID del siguiente fragmento de ejemplo y es obligatorio.

```
    "HostAccountConfig": {
        "PortableCcgVersion": "1",
        "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
        "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
```

   También puede utilizar un secreto en SSM Parameter Store mediante el ARN en este formato: `\"arn:aws:ssm:aws-region:111122223333:parameter/gmsa-plugin-input\"`.

3. Tras modificar el archivo de CredSpec, tendrá un aspecto semejante al de este ejemplo:

```
{
  "CmsPlugins": [
    "ActiveDirectory"
  ],
  "DomainJoinConfig": {
    "Sid": "S-1-5-21-4066351383-705263209-1606769140",
    "MachineAccountName": "WebApp01",
    "Guid": "ac822f13-583e-49f7-aa7b-284f9a8c97b6",
    "DnsTreeName": "contoso",
    "DnsName": "contoso",
    "NetBiosName": "contoso"
  },
```

```
  "ActiveDirectoryConfig": {
    "GroupManagedServiceAccounts": [
      {
        "Name": "WebApp01",
        "Scope": "contoso"
      },
      {
        "Name": "WebApp01",
        "Scope": "contoso"
      }
    ],
    "HostAccountConfig": {
      "PortableCcgVersion": "1",
      "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
      "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
  }
}
```

Cómo hacer referencia a un archivo de especificación de credenciales en una definición de tareas

Amazon ECS admite las siguientes formas de hacer referencia a la ruta de archivo en el campo `credentialSpecs` de una definición de tarea. Para cada una de estas opciones, puede proporcionar `credentialspec:` o `domainlesscredentialspec:`, en función de si va a unir las instancias de contenedor a un único dominio o si va a utilizar gMSA sin dominio, respectivamente.

Bucket de Amazon S3

Agregue la especificación de credenciales a un bucket de Amazon S3 y, a continuación, haga referencia al nombre de recurso de Amazon (ARN) del bucket de Amazon S3 en el campo `credentialSpecs` de la definición de tareas.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
```

```
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/${ObjectName}"
            ],
            ...
        }
    ],
    ...
}
```

También debe agregar los siguientes permisos como una política en línea al rol de IAM de ejecución de tareas de Amazon ECS, de modo que las tareas puedan obtener acceso al bucket de Amazon S3.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::{bucket_name}",
                "arn:aws:s3:::{bucket_name}/{object}"
            ]
        }
    ]
}
```

Parámetro de Parameter Store de SSM

Agregue la especificación de credenciales a un parámetro de Parameter Store de SSM. A continuación, haga referencia al nombre de recurso de Amazon (ARN) del parámetro de Parameter Store de SSM en el campo `credentialSpecs` de la definición de tareas.

```
{
    "family": "",
    "executionRoleArn": "",
```

```
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [

 "credentialspecdomainless:arn:aws:ssm:region:111122223333:parameter/parameter_name"
            ],
            ...
        }
    ],
    ...
}
```

También debe agregar los siguientes permisos como una política en línea al rol de IAM de ejecución de tareas de Amazon ECS, para que las tareas puedan obtener acceso al parámetro de Parameter Store de SSM.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": [
            "arn:aws:ssm:arn:aws::111122223333:parameter/parameter_name"
            ]
        }
    ]
}
```

Archivo local

Con los detalles de especificación de credenciales en un archivo local, haga referencia a la ruta del archivo en el campo `credentialSpecs` de la definición de tareas. La ruta del archivo al que se

hace referencia debe ser relativa al directorio `C:\ProgramData\Docker\CredentialSpecs` y usar la barra invertida (“\”) como separador de la ruta del archivo.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspec:file://CredentialSpecDir\CredentialSpecFile.json"
            ],
            ...
        }
    ],
    ...
}
```

Uso del Generador de imágenes de EC2 para crear AMI personalizadas optimizadas para Amazon ECS

AWS recomienda utilizar las AMI optimizadas para Amazon ECS, ya que están preconfiguradas con estos requisitos y recomendaciones para ejecutar sus cargas de trabajo de contenedor. Es posible que en ocasiones necesite personalizar la AMI para agregar software adicional. Puede utilizar el Generador de imágenes de EC2 para la creación, administración e implementación de sus imágenes de servidor. Retiene la propiedad de las imágenes personalizadas creadas en su cuenta. Puede utilizar las canalizaciones del Generador de imágenes de EC2 para automatizar las actualizaciones y los parches del sistema para sus imágenes o puede utilizar un comando independiente para crear una imagen con los recursos de configuración definidos.

Usted crea una receta para su imagen. La receta incluye una imagen principal y cualquier componente adicional. También crea una canalización que distribuye su AMI personalizada.

Usted crea una receta para su imagen. Una receta de imagen de Generador de Imágenes es un documento que define la imagen base y los componentes que se aplicarán a la imagen base para producir la configuración deseada de la imagen AMI de salida. También crea una canalización que distribuye su AMI personalizada. Para obtener más información, consulte How EC2 Image Builder works en la Guía del usuario del Generador de imágenes de EC2.

Se recomienda que utilice una de las siguientes AMI optimizadas para Amazon ECS como “imagen principal” en el Generador de imágenes de EC2:

- Linux
  - AL2023 x86 optimizada para Amazon ECS
  - AMI de Amazon Linux 2023 (arm64) optimizada para Amazon ECS
  - AMI de Amazon Linux 2 con kernel 5 optimizada para Amazon ECS
  - AMI de Amazon Linux 2 x86 optimizada para Amazon ECS
- Windows
  - AMI de Windows 2022 Full x86 optimizada para Amazon ECS
  - AMI de Windows 2022 Core x86 optimizada para Amazon ECS
  - AMI de Windows 2019 Full x86 optimizada para Amazon ECS
  - AMI de Windows 2019 Core x86 optimizada para Amazon ECS
  - AMI de Windows 2016 Full x86 optimizada para Amazon ECS

Se recomienda que seleccione la opción “Usar la versión más reciente disponible del SO”. La canalización utilizará el control de versiones semántico para la imagen principal, lo que ayuda a detectar las actualizaciones de dependencias en los trabajos programados automáticamente. Para más información, consulte Semantic versioning en la Guía de usuario del Generador de imágenes de EC2.

AWS actualiza periódicamente las imágenes de AMI optimizadas para Amazon ECS con parches de seguridad y la nueva versión del agente de contenedores. Cuando utiliza un ID de AMI como imagen principal en la receta de la imagen, debe comprobar periódicamente si hay actualizaciones en la imagen principal. Si las hay, debe crear una nueva versión de su receta con la AMI actualizada. Esto garantiza que las imágenes personalizadas incorporen la última versión de la imagen principal. Para obtener información sobre cómo crear un flujo de trabajo para actualizar automáticamente las instancias de EC2 de su clúster de Amazon ECS con las AMI recién creadas, consulte How to create an AMI hardening pipeline and automate updates to your ECS instance fleet.

También puede especificar el nombre de recurso de Amazon (ARN) de una imagen principal que se publique mediante una canalización administrada del Generador de imágenes de EC2. Amazon publica de forma habitual imágenes AMI optimizadas para Amazon ECS a través de canalizaciones administradas. Estas imágenes son de acceso público. Debe tener los permisos correctos para acceder a la imagen. Cuando utiliza un ARN de imagen en lugar de una AMI en su receta del

Generador de imágenes, su canalización utiliza automáticamente la versión más reciente de la imagen principal cada vez que se ejecuta. Este enfoque elimina la necesidad de crear manualmente nuevas versiones de recetas para cada actualización.

Uso del ARN de imagen con la infraestructura como código (IaC)

Puede configurar la receta mediante la consola del Generador de imágenes de EC2 o infraestructura como código (por ejemplo, AWS CloudFormation) o el AWS SDK. Al especificar una imagen principal en la receta, puede especificar un ID de AMI de EC2, un ARN de imagen del Generador de imágenes, un ID de producto de AWS Marketplace o una imagen de contenedor. AWS publica los ID de AMI y los ARN de imagen del Generador de imágenes de las AMI optimizadas para Amazon ECS públicamente. El siguiente es el formato del ARN para la imagen:

```
arn:${Partition}:imagebuilder:${Region}:${Account}:image/${ImageName}/${ImageVersion}
```

La `ImageVersion` tiene el siguiente formato. Sustituya los valores *`principal`*, *`secundario`* y *`parche`* por los valores más recientes.

```
<major>.<minor>.<patch>
```

Puede reemplazar los valores `major`, `minor` y `patch` con valores específicos o puede utilizar el ARN sin versión de una imagen para que su canalización permanezca actualizada con la última versión de la imagen principal. Un ARN sin versiones utiliza el formato comodín “x.x.x” para representar la versión de la imagen. Este enfoque permite que el servicio del Generador de imágenes se resuelva automáticamente en la versión más reciente de la imagen. El uso de un ARN sin versiones garantiza que su referencia siempre apunte a la imagen más reciente disponible, lo que agiliza el proceso de mantener las imágenes actualizadas en su implementación. Al crear una receta con la consola, el Generador de imágenes de EC2 identifica automáticamente el ARN de la imagen principal. Cuando utilice IaC para crear la receta, debe identificar el ARN y seleccionar la versión de imagen deseada o utilizar el ARN sin versión para indicar la última imagen disponible. Se recomienda que cree un script automático para filtrar y mostrar solo las imágenes que se ajusten a sus criterios. El siguiente script de Python muestra cómo se puede recuperar una lista de AMI optimizadas para Amazon ECS.

El script acepta dos argumentos opcionales: `owner` y `platform`, con los valores predeterminados de “Amazon” y “Windows”, respectivamente. Los valores válidos para el argumento propietario son: `Self`, `Shared`, `Amazon` y `ThirdParty`. Los valores válidos para el argumento plataforma son

`Windows` y `Linux`. Por ejemplo, si ejecuta el script con el argumento `owner` establecido en `Amazon` y `platform` establecido en `Linux`, el script genera una lista de imágenes publicadas por Amazon, que incluye las imágenes optimizadas para Amazon ECS.

```
import boto3
import argparse

def list_images(owner, platform):
    # Create a Boto3 session
    session = boto3.Session()

    # Create an EC2 Image Builder client
    client = session.client('imagebuilder')

    # Define the initial request parameters
    request_params = {
        'owner': owner,
        'filters': [
            {
                'name': 'platform',
                'values': [platform]
            }
        ]
    }

    # Initialize the results list
    all_images = []

    # Get the initial response with the first page of results
    response = client.list_images(**request_params)

    # Extract images from the response
    all_images.extend(response['imageVersionList'])

    # While 'nextToken' is present, continue paginating
    while 'nextToken' in response and response['nextToken']:
        # Update the token for the next request
        request_params['nextToken'] = response['nextToken']

        # Get the next page of results
        response = client.list_images(**request_params)

        # Extract images from the response
```

```
        all_images.extend(response['imageVersionList'])

    return all_images

def main():
    # Initialize the parser
    parser = argparse.ArgumentParser(description="List AWS images based on owner and
 platform")

    # Add the parameters/arguments
    parser.add_argument("--owner", default="Amazon", help="The owner of the images.
 Default is 'Amazon'.")
    parser.add_argument("--platform", default="Windows", help="The platform type of the
 images. Default is 'Windows'.")

    # Parse the arguments
    args = parser.parse_args()

    # Retrieve all images based on the provided owner and platform
    images = list_images(args.owner, args.platform)

    # Print the details of the images
    for image in images:
        print(f"Name: {image['name']}, Version: {image['version']}, ARN:
 {image['arn']}")

if __name__ == "__main__":
    main()
```

Uso del ARN de la imagen con AWS CloudFormation

Una receta de imagen del Generador de imágenes es un esquema que especifica la imagen principal y los componentes necesarios para lograr la configuración deseada de la imagen de salida. Usted utiliza el recurso `AWS::ImageBuilder::ImageRecipe`. Establezca el valor `ParentImage` en el ARN de la imagen. Utilice el ARN sin versiones de la imagen que desee para asegurarse de que su canalización utilice siempre la versión más reciente de la imagen. Por ejemplo, `arn:aws:imagebuilder:us-east-1:aws:image/amazon-linux-2023-ecs-optimized-x86/x.x.x`. La siguiente definición de recurso `AWS::ImageBuilder::ImageRecipe` utiliza un ARN de imagen administrada por Amazon:

```
ECSRecipe:
    Type: AWS::ImageBuilder::ImageRecipe
```

```
    Properties:
      Name: MyRecipe
      Version: '1.0.0'
      Components:
        - ComponentArn: [<The component arns of the image recipe>]
      ParentImage: "arn:aws:imagebuilder:us-east-1:aws:image/amazon-linux-2023-ecs-
optimized-x86/x.x.x"
```

Para más información sobre el recurso `AWS::ImageBuilder::ImageRecipe`, consulte la Guía del usuario de AWS CloudFormation.

Puede automatizar la creación de nuevas imágenes en su canalización configurando la propiedad `Schedule` del recurso `AWS::ImageBuilder::ImagePipeline`. La programación incluye una condición de inicio y una expresión cron. Para obtener más información, consulte `AWS::ImageBuilder::ImagePipeline` en la Guía del usuario de AWS CloudFormation.

En el siguiente ejemplo de `AWS::ImageBuilder::ImagePipeline`, la canalización ejecuta una compilación todos los días a las 10:00 h, Hora Universal Coordinada (UTC). Defina los siguientes valores de `Schedule`:

- Establece `PipelineExecutionStartCondition` en `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE`. La compilación se inicia solo si se actualizan los recursos dependientes, como la imagen principal o los componentes, que utilizan el comodín “x” en sus versiones semánticas. Esto garantiza que la compilación incorpore las actualizaciones más recientes de esos recursos.
- Establezca ScheduleExpression en la expresión cron `(0 10 * * ? *)`.

```
ECSPipeline:
    Type: AWS::ImageBuilder::ImagePipeline
    Properties:
      Name: my-pipeline
      ImageRecipeArn: <arn of the recipe you created in previous step>
      InfrastructureConfigurationArn: <ARN of the infrastructure configuration
 associated with this image pipeline>
      Schedule:
        PipelineExecutionStartCondition:
 EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE
        ScheduleExpression: 'cron(0 10 * * ? *)'
```

Uso del ARN de la imagen con Terraform

El enfoque para especificar la imagen principal y la programación de su canalización en Terraform se alinea con las de AWS CloudFormation. Usted utiliza el recurso `aws_imagebuilder_image_recipe`. Establezca el valor `parent_image` en el ARN de la imagen. Utilice el ARN sin versiones de la imagen que desee para asegurarse de que su canalización utilice siempre la versión más reciente de la imagen. Para más información, consulte `aws_imagebuilder_image_recipe` en la documentación de Terraform.

En el bloque de configuración de programación de la `aws_imagebuilder_image_pipeline resource`, establezca el valor del argumento `schedule_expression` en una expresión cron de su elección para especificar la frecuencia con la que se ejecuta la canalización y establezca `pipeline_execution_start_condition` en `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE`. Para más información, consulte `aws_imagebuilder_image_pipeline` en la documentación de Terraform.

Uso de contenedores de aprendizaje profundo de AWS en Amazon ECS

AWS Deep Learning Containers proporciona un conjunto de imágenes de Docker para formar y trabajar con modelos en TensorFlow y en Apache MXNet (incubación) en Amazon ECS. Deep Learning Containers proporciona entornos optimizados con bibliotecas de TensorFlow, CUDA de NVIDIA (para instancias de GPU) y MKL de Intel (para instancias de CPU). Las imágenes de contenedor de Deep Learning Containers están disponibles en Amazon ECR para referenciarlas en las definiciones de tareas de Amazon ECS. Puede utilizar Deep Learning Containers junto con Amazon Elastic Inference para reducir los costes de inferencia.

Para empezar a utilizar contenedores de aprendizaje profundo sin Elastic Inference en Amazon ECS, consulte Configuración de Amazon ECS en la Guía para desarrolladores de AWS Deep Learning AMIs.

Cuotas de servicio de Amazon ECS

Cuando cree su Cuenta de AWS, estableceremos cuotas predeterminadas (también denominadas “límites”) en sus recursos de AWS por regiones. Si intenta superar la cuota de un recurso, se producirá un error en la solicitud.

La consola de Service Quotas es una ubicación central donde puede ver y administrar sus cuotas de servicios de AWS y solicitar un aumento de cuota para muchos de los recursos que utiliza. Utilice la información sobre las cuotas que proporcionamos para administrar sus recursos. Planifique la solicitud del aumento de las cuotas antes del momento en que lo necesite.

Para obtener más información acerca de las cuotas de Amazon ECS y Fargate, consulte Puntos de conexión y cuotas de Amazon ECS en la Referencia general de Amazon Web Services. Para obtener información sobre la limitación de API en la API de Amazon ECS, consulte Limitación de solicitudes para la API de Amazon ECS.

Administración de Amazon ECS y cuotas de servicio de AWS Fargate en la AWS Management Console

Amazon ECS se ha integrado a Service Quotas, un servicio de AWS que le permite consultar y administrar sus cuotas desde una ubicación central. Para obtener más información, consulte ¿Qué es Service Quotas? en la Guía del usuario de Service Quotas.

Con Service Quotas, resulta más sencillo buscar el valor de las cuotas de servicio de Amazon ECS.

AWS Management Console

Para consultar las cuotas de servicio de Amazon ECS y Fargate mediante la AWS Management Console

1. Abra la consola de Service Quotas en https://console.aws.amazon.com/servicequotas/.
2. En el panel de navegación, elija Servicios de AWS.
3. En la lista de servicios de AWS, busque y seleccione Amazon Elastic Container Service (Amazon ECS) o AWS Fargate.

   En la lista Service Quotas, puede ver el nombre de la Service Quota, el valor aplicado (si está disponible), la cuota predeterminada de AWS y si el valor de cuota es ajustable.

4. Para ver información adicional sobre una cuota de servicio, como, por ejemplo, la descripción, elija el nombre de cuota.
5. (Opcional) Para solicitar un aumento de cuota, seleccione la cuota que desea aumentar, seleccione Solicitar aumento de cuota, escriba o seleccione la información necesaria y seleccione Solicitar.

Para trabajar más con cuotas de servicio mediante la AWS Management Console, consulte la Guía del usuario de Service Quotas. Para solicitar un aumento de cuota, consulte Solicitud de un aumento de cuota en la Guía de usuario de Service Quotas.

AWS CLI

Para consultar las cuotas de servicio de Amazon ECS y Fargate mediante la AWS CLI

Ejecute el siguiente comando para consultar las cuotas predeterminadas de Amazon ECS.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code ecs \
    --output table
```

Ejecute el siguiente comando para consultar las cuotas predeterminadas de Fargate.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code fargate \
    --output table
```

Ejecute el siguiente comando para consultar las cuotas de Fargate aplicadas.

```
aws service-quotas list-service-quotas \
    --service-code fargate
```

> **Note**
>
> Amazon ECS no admite cuotas aplicadas.

Para obtener más información, acerca de cómo trabajar con las cuotas de servicio mediante la AWS CLI, consulte la Referencia de comandos de la AWS CLI de Service Quotas. Para solicitar un aumento de cuota, consulte el `request-service-quota-increase` comando en la Referencia de comandos de la AWS CLI.

Gestión de las cuotas de servicio de Amazon ECS y los límites de las API

Amazon ECS está integrado con varios Servicios de AWS, como Elastic Load Balancing, AWS Cloud Map y Amazon EC2. Con esta estrecha integración, Amazon ECS incluye varias características, como el equilibrio de carga de servicios, Service Connect, la creación de redes de tareas y el escalado automático de clústeres. Amazon ECS y los demás Servicios de AWS con los que se integra mantienen las cuotas de servicio y los límites de velocidad de las API para garantizar un rendimiento y una utilización uniformes. Estas cuotas de servicio también evitan el aprovisionamiento accidental de más recursos de los necesarios y protegen contra acciones malintencionadas que podrían aumentar su factura.

Si se familiariza con las cuotas de servicio y los límites de velocidad de las API de AWS, podrá planificar el escalado de sus cargas de trabajo sin preocuparse por una degradación inesperada del rendimiento. Para obtener más información, consulte Request throttling for the Amazon ECS API.

Al escalar sus cargas de trabajo en Amazon ECS, le recomendamos que tenga en cuenta la siguiente cuota de servicio.

- AWS Fargate tiene cuotas que limitan el número de tareas que se ejecutan simultáneamente en cada Región de AWS. Existen cuotas para las tareas de spot de Fargate y bajo demanda en Amazon ECS. Cada cuota de servicio también incluye todos los pods de Amazon EKS que ejecute en Fargate.
- Para las tareas que se ejecutan en instancias de Amazon EC2, el número máximo de instancias de Amazon EC2 que puede registrar para cada clúster es de 5000. Si utiliza el escalado automático de clústeres de Amazon ECS con un proveedor de capacidad del grupo de escalado automático o si administra las instancias de Amazon EC2 para su clúster por su cuenta, esta cuota podría convertirse en un obstáculo para la implementación. Si necesita más capacidad, puede crear más clústeres o solicitar un aumento de la cuota de servicio.
- Si utiliza el escalado automático de clústeres de Amazon ECS con un proveedor de capacidad del grupo de escalado automático, al escalar sus servicios tenga en cuenta la cuota de `Tasks in`

`the PROVISIONING state per cluster`. Esta cuota es la cantidad máxima de tareas en el estado `PROVISIONING` para cada clúster para las que los proveedores de capacidad pueden aumentar la capacidad. Si lanza un gran número de tareas al mismo tiempo, puede cumplir fácilmente con esta cuota. Un ejemplo es si implementa simultáneamente decenas de servicios, cada uno con cientos de tareas. Cuando esto sucede, el proveedor de capacidad debe lanzar nuevas instancias de contenedor para colocar las tareas cuando el clúster no tiene suficiente capacidad. Mientras el proveedor de capacidad lanza instancias de Amazon EC2 adicionales, es probable que el programador de servicios de Amazon ECS continúe lanzando tareas en paralelo. Sin embargo, esta actividad podría verse limitada debido a la capacidad insuficiente del clúster. El programador de servicios de Amazon ECS implementa una estrategia de limitación exponencial y de retroceso para volver a intentar la ubicación de las tareas a medida que se lanzan nuevas instancias de contenedores. Como resultado, es posible que los tiempos de implementación o escalado horizontal sean más lentos. Para evitar esta situación, puede planificar las implementaciones de sus servicios en una de las siguientes opciones. Implemente una gran cantidad de tareas que no requieran aumentar la capacidad del clúster o mantenga la capacidad sobrante del clúster para el lanzamiento de nuevas tareas.

Además de tener en cuenta la cuota de servicio de Amazon ECS al escalar sus cargas de trabajo, considere también la cuota de servicio para los demás Servicios de AWS que estén integrados con Amazon ECS.

Elastic Load Balancing

Puede configurar los servicios de Amazon ECS para que utilicen Elastic Load Balancing a fin de distribuir el tráfico de manera uniforme entre las tareas. Para obtener más información y las prácticas recomendadas sobre cómo seleccionar un equilibrador de carga, consulte Uso del equilibrador de carga para distribuir el tráfico de servicio de Amazon ECS.

Cuotas de servicio de Elastic Load Balancing

Cuando escale sus cargas de trabajo, tenga en cuenta las siguientes cuotas de servicio de Elastic Load Balancing. La mayoría de las cuotas de servicio de Elastic Load Balancing son ajustables y se puede solicitar un aumento en la consola de Service Quotas.

Equilibrador de carga de aplicación

Cuando utilice un equilibrador de carga de aplicación, en función de su caso de uso, es posible que tenga que solicitar un aumento de cuota para:

- La cuota de `Targets per Application Load Balancer`, que es el número de destinos detrás de su equilibrador de carga de aplicación.
- La cuota de `Targets per Target Group per Region`, que es el número de destinos detrás de sus grupos de destino.

Para obtener más información, consulte Quotas for your Application Load Balancers en la Guía del usuario de los equilibradores de carga de aplicaciones.

Network Load Balancer

Existen limitaciones más estrictas en cuanto a la cantidad de destinos que puede registrar con un equilibrador de carga de red. Cuando utilice un equilibrador de carga de red, a menudo querrá habilitar la compatibilidad entre zonas, lo que conlleva limitaciones de escalado adicionales en `Targets per Availability Zone Per Network Load Balancer` el número máximo de destinos por zona de disponibilidad para cada equilibrador de carga de red. Para obtener más información, consulte Quotas for your Network Load Balancers en la Guía del usuario para equilibradores de carga de red.

Limitación de la API de Elastic Load Balancing

Al configurar un servicio de Amazon ECS para usar un equilibrador de carga, se deben aprobar las comprobaciones de estado del grupo de destino antes de que se considere que el servicio está en buen estado. Para llevar a cabo estas comprobaciones de estado, Amazon ECS invoca las operaciones de la API de Elastic Load Balancing en su nombre. Si tiene una gran cantidad de servicios configurados con equilibradores de carga en su cuenta, podría ralentizar las implementaciones de servicios debido a una posible limitación específica de las operaciones de las API `RegisterTarget`, `DeregisterTarget` y `DescribeTargetHealth` de Elastic Load Balancing. Cuando se produce una limitación, se producen errores de limitación en los mensajes de eventos del servicio de Amazon ECS.

Si experimenta una limitación de la API de AWS Cloud Map, puede contactar con Soporte para obtener orientación sobre cómo aumentar sus límites de limitación de la API de AWS Cloud Map. Para obtener más información acerca de la supervisión y la solución de problemas tales como errores de limitación, consulte Gestión de los problemas de limitación de Amazon ECS.

Interfaces de red elásticas

Cuando sus tareas utilizan el modo de red `awsvpc`, Amazon ECS proporciona una interfaz de red elástica (ENI) única para cada tarea. Cuando sus servicios de Amazon ECS utilizan un equilibrador

de carga de Elastic Load Balancing, estas interfaces de red también se registran como destinos para el grupo de destino correspondiente definido en el servicio.

Cuotas de servicio de la interfaz de red elástica

Cuando ejecuta tareas que utilizan el modo de red `awsvpc`, se adjunta una interfaz de red elástica única a cada tarea. Si es necesario acceder a esas tareas a través de Internet, asigne una dirección IP pública a la interfaz de red elástica de esas tareas. Cuando escale sus cargas de trabajo de Amazon ECS, tenga en cuenta estas dos cuotas importantes:

- La cuota de `Network interfaces per Region`, que es el número máximo de interfaces de red en una Región de AWS para su cuenta.
- La cuota de `Elastic IP addresses per Region`, que es el número máximo de direcciones IP elásticas en una Región de AWS.

Estas dos cuotas de servicio son ajustables y puede solicitar un aumento de estas desde la consola de Service Quotas. Para obtener más información, consulte Amazon VPC service quotas en la Guía del usuario de Amazon Virtual Private Cloud.

En cuanto a las cargas de trabajo de Amazon ECS que se alojan en instancias de Amazon EC2, al ejecutar tareas que utilizan el modo de red `awsvpc`, tenga en cuenta la cuota de servicio `Maximum network interfaces` (el número máximo de instancias de red para cada instancia de Amazon EC2). Esta cuota limita la cantidad de tareas que puede colocar en una instancia. No puede ajustar la cuota y no está disponible en la consola de Service Quotas. Para obtener más información, consulte Direcciones IP por interfaz de red por tipo de instancia en la Guía del usuario de Amazon EC2.

Aunque no se puede cambiar el número de interfaces de red que pueden adjuntarse a una instancia de Amazon EC2, se puede utilizar la característica de enlace troncal de interfaz de red elástica para aumentar el número de interfaces de red disponibles. Por ejemplo, de forma predeterminada, una instancia `c5.large` puede tener hasta tres interfaces de red. La interfaz de red principal de la instancia cuenta como una. Por lo tanto, puede adjuntar dos interfaces de red adicionales a la instancia. Dado que cada tarea que utiliza el modo de red `awsvpc` requiere una interfaz de red, normalmente solo puede ejecutar dos de esas tareas en este tipo de instancia. Esto puede provocar una infrautilización de la capacidad del clúster. Si habilita el enlace troncal de la interfaz de red elástica, puede aumentar la densidad de la interfaz de red para colocar un mayor número de tareas en cada instancia. Con el enlace troncal activado, una instancia `c5.large` puede tener hasta 12 interfaces de red. La instancia tiene la interfaz de red principal y Amazon ECS crea y asocia una interfaz de red “troncal” a la instancia. Como resultado, con esta configuración, puede ejecutar

10 tareas en la instancia en lugar de las dos tareas predeterminadas. Para obtener más información, consulte Aumento de las interfaces de red de instancias de contenedor de Linux de Amazon ECS.

Limitación de la API de interfaz de red elástica

Cuando ejecuta tareas que utilizan el modo de red `awsvpc`, Amazon ECS se basa en las siguientes API de Amazon EC2. Cada una de estas API tiene diferentes limitadores de API. Para obtener información, consulte Request throttling for the Amazon EC2 API en la Referencia de la API de Amazon EC2.

- CreateNetworkInterface
- AttachNetworkInterface
- DetachNetworkInterface
- DeleteNetworkInterface
- DescribeNetworkInterfaces
- DescribeVpcs
- DescribeSubnets
- DescribeSecurityGroups
- DescribeInstances

Si las llamadas a la API de Amazon EC2 se limitan durante los flujos de trabajo de aprovisionamiento de la interfaz de red elástica, el programador de servicios de Amazon ECS lo volverá a intentar automáticamente con retrasos exponenciales. En ocasiones, estas retiradas automáticas pueden provocar un retraso en el lanzamiento de las tareas, lo que se traduce en velocidades de implementación más lentas. Cuando se produzca una limitación de la API, verá el mensaje `Operations are being throttled. Will try again later.` en los mensajes de eventos del servicio. Si experimenta limitaciones constantes de la API de Amazon EC2, puede ponerse en contacto con Soporte para obtener orientación sobre cómo aumentar sus límites de limitación de la API. Para obtener más información acerca de la supervisión y la solución de problemas tales como errores de limitación, consulte Handling throttling issues.

AWS Cloud Map

La detección de servicios de Amazon ECS y Service Connect utilizan las API de AWS Cloud Map para administrar los espacios de nombres de sus servicios de Amazon ECS. Si sus servicios tienen un gran número de tareas, tenga en cuenta las siguientes recomendaciones.

Service Quotas de AWS Cloud Map

Cuando los servicios de Amazon ECS están configurados para usar la detección de servicios o Service Connect, la cuota `Tasks per service`, que es el número máximo de tareas para el servicio, se ve afectada por la cuota de servicio `Instances per service` de AWS Cloud Map, que es el número máximo de instancias para ese servicio. En concreto, la cuota de servicio de AWS Cloud Map reduce la cantidad de tareas que puede ejecutar hasta un máximo de 1000 instancias por servicio. No se puede cambiar la cuota AWS Cloud Map. Para más información, consulte Service Quotas de AWS Cloud Map.

Limitación de API de AWS Cloud Map

Amazon ECS llama a las API `ListInstances`, `GetInstancesHealthStatus`, `RegisterInstance` y `DeregisterInstance` de AWS Cloud Map en su nombre. Ayudan en la detección de servicios y en la ejecución de comprobaciones de estado al lanzar una tarea. Cuando se implementan varios servicios que utilizan la detección de servicios con una gran cantidad de tareas al mismo tiempo, esto puede provocar que se superen los límites de limitación de las API de AWS Cloud Map. Cuando esto suceda, es probable que vea el siguiente mensaje: `Operations are being throttled. Will try again later` en los mensajes de eventos del servicio de Amazon ECS y una velocidad de implementación y lanzamiento de tareas más lenta. AWS Cloud Map no documenta los límites de limitación de estas API. Si experimenta limitaciones, puede ponerse en contacto con Soporte para obtener orientación sobre cómo aumentar sus límites de limitación de las API. Para obtener más recomendaciones acerca de la supervisión y la solución de problemas tales como errores de limitación, consulte Gestión de los problemas de limitación de Amazon ECS.

Referencia de la API de Amazon ECS

Además de la AWS Management Console y la AWS Command Line Interface (AWS CLI), Amazon ECS también proporciona una API. Puede utilizar la API para automatizar las tareas de administración de recursos de Amazon ECS.

- Para obtener una lista de las operaciones de la API por recurso de Amazon ECS, consulte Acciones por recurso de Amazon ECS.
- Para ver una lista de acciones de la API ordenada alfabéticamente, consulte el tema relacionado con las acciones.
- Para ver una lista de tipos de datos ordenada alfabéticamente, consulte el tema relacionado con los Tipos de datos.
- Para ver una lista de parámetros de consulta comunes, consulte el tema relacionado con los Parámetros comunes.
- Para ver las descripciones de los códigos de error, consulte el tema relacionado con los Errores comunes.

Para obtener más información sobre la AWS CLI, consulte la Referencia de AWS Command Line Interface para Amazon ECS.

Historial de documentos

En la siguiente tabla, se describen las principales actualizaciones y las nuevas características de la Guía para desarrolladores de Amazon Elastic Container Service. Actualizamos la documentación con frecuencia para dar respuesta a los comentarios que se nos envía.

Cambio	Descripción	Fecha
Agregue permisos a la política administrada AmazonEC2ContainerServiceforEC2Role.	La política de IAM administrada `AmazonEC2ContainerServiceforEC2Role` se actualizó para incluir el permiso `ecs:ListTagsForResource`. Este permiso permite al agente de Amazon ECS recuperar las etiquetas de tareas y de instancias de contenedores a través del punto de conexión de metadatos de la tarea (`${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`). Para más información, consulte AmazonEC2ContainerServiceforEC2Role.	4 de agosto de 2025
Agregue permisos a AmazonECSInfrastructureRolePolicyForLoadBalancers.	La política de IAM administrada `AmazonECSInfrastructureRolePolicyForLoadBalancers` se actualizó con nuevos permisos para describir los grupos de destino, registrarlos y anular el registro de estos.	25 de julio de 2025
Compatibilidad con la actualización de un controlador de implementación de servicios	Amazon ECS admite la actualización de un controlador de implementación de servicios, lo que le permite cambiar entre distintos tipos de implementación para un servicio existente. Para más información, consulte Actualización de los parámetros de servicio de Amazon ECS.	15 de julio de 2025
Implementaciones azul/verde nativas de Amazon ECS	Amazon ECS admite las implementaciones azul/verde nativas, lo que le permite validar las nuevas revisiones de servicios antes de dirigir el tráfico de producción	15 de julio de 2025

Cambio	Descripción	Fecha
	hacia ellas. Para más información, consulte Amazon ECS blue/green deployments.	
Agregue la nueva política administrada AmazonECSInfrastructureRolePolicyForLoadBalancers.	Esta política proporciona acceso a otros recursos de servicio de AWS necesarios para administrar los equilibradores de carga asociados a las cargas de trabajo de Amazon ECS. Para más información, consulte AmazonECSInfrastructureRolePolicyForLoadBalancers.	15 de julio de 2025
Soporte para el manifiesto v2 de índices SOCI	Los clientes de Fargate que deseen utilizar SOCI solo podrán usar el manifiesto v2 de índices SOCI. Los clientes que hayan utilizado anteriormente SOCI en Fargate pueden seguir utilizando el manifiesto v1 de índices SOCI; sin embargo, les recomendamos encarecidamente que migren al manifiesto v2 de índices SOCI. Para obtener más información, consulte Carga diferida de imágenes de contenedores mediante Seekable OCI (SOCI).	2 de julio de 2025
Se agregó el ID de la tarea en los eventos de acción de servicio generados debido a fallos de estado.	Para obtener más información, consulte los mensajes de eventos de servicio que no están en buen estado de Amazon ECS.	30 de junio de 2025
Actualice al modo de controlador de registro predeterminado.	El modo de registro predeterminado de Amazon ECS se ha actualizado de `blocking` a `non-blocking` . Puede anular este modo predeterminado al especificar un `mode` en la `logConfiguration` de la definición de su contenedor o configurar su propio modo predeterminado al establecer la configuración de la cuenta `defaultLogDriverMode` mediante la consola de Amazon ECS o la AWS CLI. Para obtener más información, consulte Modo de controlador de registro predeterminado.	25 de junio de 2025

Cambio	Descripción	Fecha
Agregue permisos a AmazonECSServiceRolePolicy.	La política de IAM administrada `AmazonECSServiceRolePolicy` se actualizó con nuevos permisos AWS Cloud Map que permiten a Amazon ECS actualizar los atributos de los servicios de AWS Cloud Map que administra Amazon ECS.	24 de junio de 2025
Compatibilidad para la AMI de Windows Server 2025 Full optimizada para Amazon ECS y la AMI de Windows Server 2025 Core optimizada para Amazon ECS.	Para obtener más información, consulte las AMI de Windows optimizadas para Amazon ECS.	20 de junio de 2025
Compatibilidad para actualizar una estrategia de proveedor de capacidad de servicio.	Ahora puede actualizar una estrategia de proveedor de capacidad de servicio. Para obtener más información, consulte Update service parameters.	12 de junio de 2025
Adición de permisos a AmazonECSInfrastructureRolePolicyForVolumes	Se actualizó la política `AmazonECSInfrastructureRolePolicyForVolumes` para agregar el permiso `ec2:DescribeInstances`. El permiso ayuda a prevenir la colisión de nombres de dispositivos para volúmenes de Amazon EBS que están asociados a tareas de Amazon ECS que se ejecutan en la misma instancia de contenedor.	2 de junio de 2025
Compatibilidad con la tasa de inicialización de volúmenes para volúmenes de Amazon EBS adjuntos a tareas de Amazon ECS.	Ahora puede configurar un valor de `volumeInitializationRate` para determinar la velocidad, en MiB/s, a la que se obtienen los datos de una instantánea de un volumen de Amazon EBS existente para crear nuevos volúmenes que se adjuntarán a las tareas de Amazon ECS. Para obtener más información, consulte volumeInitializationRate.	13 de mayo de 2025

Cambio	Descripción	Fecha
Compatibilidad para detener una implementación de servicio.	Ahora puede detener la implementación de un servicio. Para obtener más información, consulte Detención de implementaciones de servicios de Amazon ECS.	2 de mayo de 2025
Compatibilidad con puntos de conexión de doble pila de Amazon ECS.	Ahora puede utilizar puntos de conexión de doble pila para interactuar con Amazon ECS. Los puntos de conexión de doble pila admiten solicitudes a Amazon ECS tanto a través del Protocolo de Internet versión 4 (IPv4) como del Protocolo de Internet versión 6 (IPv6). Para obtener más información, consulte Puntos de conexión de doble pila en Amazon ECS.	30 de abril de 2025
Soporte para configurar un modo de controlador de registro predeterminado en la cuenta en una Región de AWS.	Ahora puede configurar un modo de entrega de los mensajes de registro predeterminado de un contenedor al controlador de registro que especifique en su `logConfiguration` con la configuración de la cuenta de `defaultLogDriverMode`. Para obtener más información, consulte Modo de controlador de registro predeterminado.	17 de abril de 2025
Soporte para migrar un servicio con un formato de ARN corto al formato de ARN largo.	Ahora puede migrar un servicio con un formato de ARN corto al paquete de formato de ARN largo. Para obtener más información, consulte Migración de un ARN de servicio corto de Amazon ECS a un ARN largo.	13 de febrero de 2025
Compatibilidad con tres tipos de inyección de errores en Amazon ECS y Fargate.	Ahora puede utilizar la inyección de errores de puerto blackhole, latencia de red y pérdida de paquetes de red con Amazon ECS y Fargate. Para obtener más información, consulte Cómo utilizar AWS Fault Injection Service para probar las cargas de trabajo de Amazon ECS.	19 de diciembre de 2024

Cambio	Descripción	Fecha
Compatibilidad con Información de contenedores de CloudWatch con capacidad de observabilidad mejorada.	Puede utilizar Información de contenedores de CloudWatch con capacidad de observación mejorada. Para obtener más información, consulte Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada.	2 de diciembre de 2024
Compatibilidad con el escalado predictivo.	Puede utilizar el escalado predictivo para anticipar las necesidades futuras y aumentar proactivamente las tareas del servicio según sea necesario. Para obtener más información, consulte Uso de patrones históricos para escalar los servicios de CloudWatch con escalado predictivo.	21 de noviembre de 2024
Compatibilidad con el reequilibrio de servicios.	Ahora Amazon ECS puede equilibre automáticamente las tareas de servicio entre las zonas de disponibilidad. Para obtener más información, consulte Equilibrio de un servicio de Amazon ECS entre zonas de disponibilidad.	19 de noviembre de 2024
Compatibilidad para configurar la conversión de las etiquetas de imagen de los contenedores en resúmenes de imágenes.	Ahora puede configurar la conversión de las etiquetas de imagen de los contenedores en resúmenes de cada contenedor de las tareas que forman parte de un servicio. Para obtener más información, consulte Resolución de imagen de contenedor y `versionConsistency`.	19 de noviembre de 2024
Adición de una nueva política de IAM `AmazonECSInfrastructureRolePolicyForVpcLattice`	La política `AmazonECSInfrastructureRolePolicyForVpcLattice` proporciona acceso a otros recursos de servicio de AWS necesarios para administrar en su nombre la característica de VPC Lattice en las cargas de trabajo de Amazon ECS.	18 de noviembre de 2024

Cambio	Descripción	Fecha
VPC Lattice ya está disponible para los servicios de Amazon ECS.	Ahora puede usar VPC Lattice para estandarizar la conectividad, la seguridad y la observabilidad de un servicio a otro. Para obtener más información, consulte Uso de Amazon VPC Lattice para conectar, observar y proteger los servicios de Amazon ECS.	18 de noviembre de 2024
Compatibilidad con el historial de servicios mediante implement aciones y revisiones de servicios.	Puede utilizar las implementaciones y las revisione s de los servicios para ver el historial de implement aciones. Para obtener más información, consulte Visualización del historial de servicios mediante las implementaciones de servicios de Amazon ECS y Revisiones de servicios de Amazon ECS.	30 de octubre de 2024
Compatibilidad para volúmenes de Amazon EBS para el tipo de inicialización de EC2 con el sistema operativo Windows.	Amazon ECS admite los volúmenes de Amazon EBS adjuntos a las tareas que se ejecutan en instancia s de EC2 con el sistema operativo Windows. Para obtener más información, consulte Uso de volúmenes de Amazon EBS con Amazon ECS.	24 de octubre de 2024
Agregue permisos a AmazonECSInfrastru ctureRolePolicyFor Volumes.	La política `AmazonECSInfrastructureRole PolicyForVolumes` se ha actualizado para permitir a los clientes crear un volumen de Amazon EBS a partir de una instantánea. Para obtener más información, consulte AmazonECSInfrastructureRole PolicyForVolumes.	10 de octubre de 2024
Fargate Spot está disponible para Linux ARM.	Se agregó soporte para Fargate Spot en Linux ARM en Amazon ECS.	6 de septiembre de 2024

Cambio	Descripción	Fecha
Política de reinicio de contenedores para contenedores individua les en tareas de Amazon ECS.	Puede habilitar las políticas de reinicio para los contenedores esenciales y no esenciales definidas en la definición de su tarea para superar los errores transitorios con mayor rapidez y mantener la disponibi lidad de las tareas. Para obtener más información, consulte Reinicio de contenedores individuales en tareas de Amazon ECS con políticas de reinicio de contenedores.	15 de agosto de 2024
Permisos agregados de the section called "AmazonECS_FullAcc ess".	La política `AmazonECS_FullAccess` se actualizó para añadir permisos `iam:PassRole` para los roles de IAM para una función denominada `ecsInfras tructureRole`. Este es el rol de IAM predeterm inado que crea AWS Management Console destinado a usarse como un rol de infraestructura de ECS que permite a Amazon ECS administrar los volúmenes de Amazon EBS adjuntos a las tareas de ECS.	13 de agosto de 2024
Se agregó compatibi lidad con Amazon Linux 2023, CentOS Stream 9, Fedora 40 y Ubuntu 24 para ECS Anywhere	Se agregó compatibilidad para los sistemas operativo s Amazon Linux 2023, CentOS Stream 9, Fedora 40 y Ubuntu 24 a ECS Anywhere. Para obtener más información, consulte Sistemas operativos y arquitect uras de sistemas compatibles.	23 de julio de 2024
Resolución de imagen de contenedor para los servicios que utilizan el tipo de implementación de actualización continua (ECS).	Para garantizar que todas las tareas de un servicio que utiliza el controlador de implementación de actualizaciones continuas utilicen la misma imagen de contenedor, Amazon ECS convierte los nombres de las imágenes de contenedor y cualquier etiqueta de imagen especificada en la definición de la tarea en resúmenes de imágenes de contenedor. Para obtener más información, consulte resolución de imagen de contenedor.	10 de julio de 2024

Cambio	Descripción	Fecha
Se agregó compatibilidad con ECS Anywhere para Debian 11 y Debian 12	Se agregó compatibilidad con los sistemas operativos Debian 11 y Debian 12 para ECS Anywhere. Para obtener más información, consulte Sistemas operativos y arquitecturas de sistemas compatibles.	28 de marzo de 2024
Compatibilidad con gMSA para contenedores de Linux en Fargate	Amazon ECS admite la autenticación de Active Directory para contenedores de Linux en Fargate con un tipo especial de cuenta de servicio denominada cuenta de servicio administrada de grupo (gMSA). Para obtener más información, consulte Using gMSA for Linux containers on Fargate.	5 de marzo de 2024
Se agregaron métricas de CloudWatch para los volúmenes de Amazon EBS adjuntos a las tareas	A partir de ahora, Amazon ECS publica las métricas de CloudWatch para la utilización del almacenamiento de Amazon EBS para las tareas que tienen un volumen de Amazon EBS adjunto. Para obtener más información, consulte Amazon ECS CloudWatch metrics.	8 de febrero de 2024
TLS de Service Connect	Ahora puede utilizar TLS con Service Connect.	22 de enero de 2024
Política administrada de TLS de Service Connect	Se agregó la nueva política AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity.	22 de enero de 2024
Actualización de la configuración del tiempo de espera de Service Connect	La configuración del tiempo de espera de Service Connect ahora se puede actualizar e incluye dos parámetros opcionales: `idleTimeout` y `perRequestTimeout` .	22 de enero de 2024
Drenaje de instancias administradas por Amazon ECS	Puede utilizar el drenaje de instancias administradas por Amazon ECS para facilitar la terminación correcta de las instancias de Amazon ECS.	19 de enero de 2024

Cambio	Descripción	Fecha
Se agregó compatibilidad con Ubuntu 22 para ECS Anywhere	Se agregó compatibilidad con el sistema operativo Ubuntu 22 para ECS Anywhere. Para obtener más información, consulte Sistemas operativos y arquitecturas de sistemas compatibles.	16 de enero de 2024
Adición de la política de IAM `AmazonECSInfrastructureRolePolicyForVolumes`	Se agregó la `AmazonECSInfrastructureolePolicyForVolumes`. La política concede los permisos que Amazon ECS necesita para hacer llamadas a la API AWS con el fin de administrar los volúmenes de Amazon EBS asociados a las cargas de trabajo de Amazon ECS.	11 de enero de 2024
Volumen de datos de Amazon EBS para una tarea de Amazon ECS	Puede configurar un volumen de datos de Amazon EBS por tarea durante la implementación para adjuntarlo a tareas independientes de Amazon ECS o tareas administradas por un servicio de ECS. La configuración de un volumen en el momento de la implementación le permite crear definiciones de tareas reutilizables que no se limitan a tipos o ajustes de volumen específicos. Los volúmenes de Amazon EBS proporcionan almacenamiento en bloques rentable, altamente disponible, duradero y de alto rendimiento para cargas de trabajo en contenedores con uso intensivo de datos.	11 de enero de 2024
La consola clásica de Amazon ECS llegó al final de su vida útil	La consola de Amazon ECS llegó al final de su vida útil.	4 de diciembre de 2023
Se actualizó la política de	La política de IAM administrada AmazonECSServiceRolePolicy se actualizó con los nuevos permisos `events` y los permisos adicionales `autoscaling` y `autoscaling-plans`.	4 de diciembre de 2023

Cambio	Descripción	Fecha
Compatibilidad con la supervisión en tiempo de ejecución	Puede usar la supervisión en tiempo de ejecución para supervisar sus cargas de trabajo de Amazon ECS e identificar comportamientos malintencionados o no autorizados. Para obtener más información, consulte Runtime Monitoring.	26 de noviembre de 2023
Política actualizada	La política de IAM administrada de `AmazonECSServiceRolePolicy` se actualizó para permitir el acceso a la API de AWS Cloud Map `DiscoverInstancesRevision`.	4 de octubre de 2023
Configuración de retirada de tareas AWS Fargate	Puede configurar el período de espera antes de que se retiren las tareas de Fargate. Para obtener más información, consulte mantenimiento de tareas de AWS Fargate.	5 de septiembre de 2023
Otros parámetros de definición de tareas en AWS Fargate	AWS Fargate agrega soporte para `pidMode` y `systemControls` en la versión de la plataforma Linux `1.4.0`. Para obtener más información, consulte Definiciones de tareas.	9 de agosto de 2023
Rediseño de la página de definición de tareas de la consola de Amazon ECS	La página de definición de tareas de la consola de Amazon ECS se ha rediseñado y contiene opciones adicionales. Para obtener más información, consulte Crear una definición de tarea con la consola.	26 de julio de 2023
Fargate admite la carga diferida con los índices Seekable OCI	AWS Fargate presenta los índices de Seekable OCI (SOCI). Con los SOCI, los contenedores solo tardan unos segundos en extraer la imagen antes de empezar, lo que proporciona tiempo para configurar el entorno y crear instancias de la aplicación mientras la imagen se descarga en segundo plano. Para obtener más información, consulte Carga diferida de imágenes de contenedores mediante Seekable OCI (SOCI).	17 de julio de 2023

Cambio	Descripción	Fecha
Soporte mejorado para gMSA en Linux y Windows	La definición de la tarea tiene un nuevo campo de `credentialSpecs` para gMSA para Linux y Windows. Se ha agregado un nuevo tutorial completo sobre gMSA sin dominio en Windows. Consulte Tutorial: Uso de contenedores de Windows con gMSA sin dominio mediante AWS CLI. Para obtener más información, consulte Uso de gMSA para contenedores de Linux y Uso de gMSA para contenedores de Windows.	14 de julio de 2023
Documentación mejorada de las versiones del agente de ECS	Se ha actualizado la documentación de las versiones del agente de Amazon ECS. Recomendamos que utilice la versión `v20.10.13` o una posterior de Docker con la versión más reciente del agente de contenedores de Amazon ECS. Las versiones publicadas y los cambios del agente están disponibles en GitHub. Para obtener más información, consulte Versiones del agente de contenedores de Linux en Amazon ECS.	20 de junio de 2023
Disponibilidad regional actualizada para la compatibilidad con Fargate ARM64	Se ha actualizado la disponibilidad regional para la compatibilidad con Fargate ARM64. Para obtener más información, consulte Consideraciones.	19 de junio de 2023
Mejorar la documentación del escalado automático de clústeres	La documentación para el escalado de Amazon EC2 Auto Scaling de Amazon ECS presenta mejoras significativas en cuanto a precisión y legibilidad. Para obtener más información, consulte Escalado automático de clústeres de Amazon ECS.	4 de mayo de 2023

Cambio	Descripción	Fecha
Autorización de etiquetado para la creación de recursos.	Los usuarios deben tener permisos para realizar las acciones que crean el recurso, como `ecsCreateCluster` . Al crear un recurso y especificar etiquetas para ese recurso, AWS realiza una autorización adicional para comprobar que hay permisos para crear etiquetas. Para obtener más información, consulte Autorización de etiquetado y Otorgar permiso para etiquetar recursos en el momento de su creación.	18 de abril de 2023
Compatibilidad con gMSA para contenedores de Linux en EC2	Puede usar gMSA para autenticarse en los contenedores de Active Directory para Linux en EC2. Para obtener más información, consulte Utilizar gMSA para contenedores de Linux.	14 de abril de 2023
Compatibilidad del almacenamiento efímero para los contenedores de Windows en AWS Fargate	Puede utilizar el almacenamiento efímero para los contenedores de Windows en AWS Fargate. Para obtener más información, consulte Almacenamiento de tareas de Fargate.	14 de abril de 2023
Compatibilidad de AWS Cost Management con datos de CUR a nivel de tarea	Puede activar el uso de recursos y costos a nivel de tarea en los informes de costo y uso. Esto agrega datos de asignación de costos divididos para las tareas que se ejecutan en AWS Fargate y EC2. Para obtener más información, consulte Informes sobre costo y uso de nivel de tarea.	12 de abril de 2023
AMI de Amazon Linux 2023 optimizada para Amazon ECS	Puede implementar cargas de trabajo en la AMI de Amazon Linux 2023 optimizada para Amazon ECS. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.	10 de abril de 2023

Cambio	Descripción	Fecha
AWS Fargate Estándar Federal de Procesamiento de la Información (FIPS) 140	Puede implementar cargas de trabajo en Amazon ECS en AWS Fargate de modo que presta conformidad con el Estándar Federal de Procesamiento de Información (FIPS) 140. Para obtener más información, consulte AWS Fargate Estándar de procesamiento de la información federal (FIPS, Federal Information Processing Standard 140).	10 de abril de 2023
Eliminación de definición de tareas	Puede eliminar una definición de tareas con la consola de Amazon ECS, el SDK y la AWS CLI. Para obtener más información, consulte Eliminar una revisión de la definición de tareas mediante la consola y Definiciones de tareas.	24 de febrero de 2023
Recomendaciones del servicio de AWS Fargate en Compute Optimizer	AWS Compute Optimizer genera recomendaciones sobre el tamaño de las tareas y los contenedores en función de la utilización de las tareas en ejecución en los servicios de Amazon ECS en AWS Fargate. Para obtener más información, consulte Visualización de recomendaciones para servicios de Amazon ECS en Fargate.	27 de enero de 2023
Consola de Amazon ECS	La nueva consola de Amazon ECS ahora es la consola predeterminada. Para obtener más información, consulte Nueva consola de Amazon ECS.	19 de enero de 2023
Política de IAM `AmazonECS_FullAccess` actualizada	La política de IAM `AmazonECS_FullAccess` se actualiza para incluir permisos para agregar etiquetas a los equilibradores de carga durante la creación. Para obtener más información, consulte AmazonECS_FullAccess.	4 de enero de 2023

Cambio	Descripción	Fecha
Uso de alarmas de CloudWatch para detectar errores en la implementación de servicios de Amazon ECS	Puede configurar Amazon ECS para que defina la implementación como fallida cuando detecte que una alarma de CloudWatch específica ha pasado al estado ALARM (ALARMA). Para obtener más información, consulte the section called “Implementaciones de actualización continua”.	19 de diciembre de 2022
Compatibilidad con la asignación de puertos de contenedores	Puede definir el rango de números de puerto en el contenedor que está vinculado al rango de puertos de host asignado de manera dinámica. Para obtener más información, consulte the section called “Mapeos de puertos”.	15 de diciembre de 2022
Disponibilidad general de Amazon ECS Service Connect	Esta función agrega la detección de servicios y la malla de servicios que se controla mediante implement aciones de servicios de Amazon ECS. Para obtener más información, consulte the section called “ Service Connect”.	27 de noviembre de 2022
La nueva experiencia de consola de Amazon ECS para definiciones de tareas está actualizada.	La nueva experiencia de consola de Amazon ECS ahora contiene un editor de JSON para las definicio nes de tareas. Para obtener más información, consulte the section called “Creación de una definición de tareas con la consola”.	27 de octubre de 2022
La nueva experiencia de consola de Amazon ECS para definiciones de tareas está actualizada.	La nueva experiencia de consola de Amazon ECS ahora contiene un editor de JSON para las definicio nes de tareas. Para obtener más información, consulte the section called “Creación de una definición de tareas con la consola”.	27 de octubre de 2022

Cambio	Descripción	Fecha
La nueva experiencia de la consola de Amazon ECS está actualizada.	La nueva experiencia de consola de Amazon ECS se actualizó con parámetros adicionales de servicios y tareas. Para obtener más información, consulte the section called “Creación de una implementación de actualización continua” y the section called “Ejecución de una aplicación como tarea”.	7 de octubre de 2022
Nueva información en la versión 4 del punto de conexión de metadatos de tareas	La versión 4 del punto de conexión de metadatos de tareas incluye ahora el identificador de la VPC y el nombre del servicio. Para obtener más información, consulte the section called “Versión 4 del punto de enlace de metadatos de tareas”.	7 de octubre de 2022
Nuevos tamaños de definición de tareas	Amazon ECS en Fargate ahora admite los tamaños de tarea de 8 vCPU y 16 vCPU. Para obtener más información, consulte the section called “Tamaño de tarea”.	16 de septiembre de 2022
Páginas CLI de ECS archivadas	La documentación de la CLI de ECS se archivó. Le recomendamos que utilice Copilot de AWS para sus necesidades de herramientas de línea de comandos. Para obtener más información, consulte Creación de recursos de Amazon ECS mediante la interfaz de la línea de comandos de AWS Copilot.	15 de septiembre de 2022
Nuevas cuotas de Fargate	Fargate pasa de las cuotas basadas en el recuento de tareas a las cuotas basadas en vCPU. Para obtener más información, consulte Puntos de conexión y cuotas de Amazon ECS.	8 de septiembre de 2022
Compatibilidad con los grupos de calentamiento para Amazon EC2 Auto Scaling.	Ahora puede utilizar grupos de calentamiento de Amazon EC2 Auto Scaling para escalar horizonta lmente las aplicaciones de forma más rápida y ahorrar costos. Para obtener más información, consulte Configuración de instancias preinicializadas para el grupo de escalado automático de Amazon ECS.	23 de marzo de 2022

Cambio	Descripción	Fecha
Compatibilidad con instancias de Windows en ECS Anywhere.	ECS Anywhere ahora admite instancias de Windows. Para obtener más información, consulte Clústeres de Amazon ECS para el tipo de lanzamiento externo.	3 de marzo de 2022
Se ha agregado soporte ECS Exec para instancias externas	Ahora se admite ECS Exec para instancias externas. Para obtener más información, consulte Supervisión de los contenedores de Amazon ECS con ECS Exec.	24 de enero de 2022
Actualización de la nueva experiencia de consola de Amazon ECS	La nueva experiencia de consola de Amazon ECS permite crear y eliminar un clúster, actualizar una definición de tarea y anular el registro de una definición de tarea. Para obtener más información, consulte Creación de un clúster de Amazon ECS para el tipo de lanzamiento de Fargate, Eliminación de un clúster de Amazon ECS, Actualización de una definición de tareas de Amazon ECS mediante la consola y Anulación del registro de la revisión de una definición de tareas de Amazon ECS mediante la consola.	8 de diciembre de 2021
Actualización de la nueva experiencia de consola de Amazon ECS	La nueva experiencia de consola de Amazon ECS admite la creación de una definición de tarea. Para obtener más información, consulte Creación de una definición de tareas de Amazon ECS mediante la consola.	23 de noviembre de 2021
Amazon ECS admite la arquitectura ARM de 64 bits para Linux.	Amazon ECS admite la arquitectura de CPU ARM de 64 bits para el sistema operativo Linux. Para obtener más información, consulte the section called “Definiciones de tareas para cargas de trabajo de ARM de 64 bits”.	23 de noviembre de 2021
Compatibilidad con Amazon ECS para la opción de log-driver-buffer-limit de fluentd	Amazon ECS admite la opción `log-driver-buffer-limit` fluentd. Para obtener más información, consulte Envío de registros de Amazon ECS a un servicio de AWS o AWS Partner.	22 de noviembre de 2021

Cambio	Descripción	Fecha
Script de compilación de la AMI de Linux optimizada para Amazon ECS	Amazon ECS ha establecido en código abierto los scripts de compilación que se utilizan para crear las variantes de Linux de la AMI optimizada para Amazon ECS. Para obtener más información, consulte Script de compilación de la AMI de Linux optimizada para Amazon ECS.	19 de noviembre de 2021
Estado de instancias de contenedor	Amazon ECS agrega soporte para la supervisión del estado de las instancias de contenedores. Para obtener más información, consulte Supervisión del estado de la instancia de contenedor de Amazon ECS.	10 de noviembre de 2021
Compatibilidad con Windows Amazon ECS Exec	Amazon ECS Exec admite Windows. Para obtener más información, consulte Supervisión de los contenedores de Amazon ECS con ECS Exec.	1 de noviembre de 2021
Compatibilidad con contenedores Windows en Fargate.	Amazon ECS admite contenedores Windows en Fargate. Para obtener más información, consulte Registro de cambios de la versión de la plataforma de Fargate Windows.	28 de octubre de 2021
Compatibilidad de GPU para instancias externas en Amazon ECS Anywhere	Amazon ECS admite la especificación de los requisitos de GPU en la definición de tareas para las tareas ejecutadas en instancias externas. Para obtener más información, consulte Definiciones de tareas de Amazon ECS para cargas de trabajo de GPU y Registro de una instancia externa en un clúster de Amazon ECS.	8 de octubre de 2021
Compatibilidad con el modo de red `awsvpc` en Windows	Amazon ECS admite el modo de red `awsvpc` en Windows. Para obtener más información, consulte Asignación de una interfaz de red para una tarea de Amazon ECS.	15 de julio de 2021

Cambio	Descripción	Fecha
Disponibilidad general de Bottlerocket	Amazon ECS admite que se proporcione una variante de la AMI optimizada para Amazon ECS del sistema operativo Bottlerocket como AMI. Para obtener más información, consulte AMI Bottlerocket optimizadas para Amazon ECS.	30 de junio de 2021
Actualización de tareas programadas de Amazon ECS	Amazon EventBridge agregó compatibilidad con parámetros adicionales al crear reglas que desencade nan tareas programadas de Amazon ECS.	25 de junio de 2021
Políticas administradas por AWS para Amazon ECS	Amazon ECS agregó documentación de políticas administradas por AWS para roles vinculados al servicio. Para obtener más información, consulte Políticas administradas por AWS para Amazon Elastic Container Service.	8 de junio de 2021
Introducción al AWS CDK	Se agregó una guía de introducción para usar el AWS CDK con Amazon ECS. Para obtener más informaci ón, consulte Creación de recursos de Amazon ECS con AWS CDK.	27 de mayo de 2021
Amazon ECS Anywhere	Amazon ECS ahora admite que se registre un servidor ubicado en las instalaciones o una máquina virtual (VM) en el clúster. Para obtener más información, consulte Clústeres de Amazon ECS para el tipo de lanzamiento externo.	25 de mayo de 2021
AMI de Windows Server 20H2 Core optimizada para Amazon ECS	Amazon ECS ha agregado compatibilidad con una nueva variante de AMI de Windows optimizada para Amazon ECS para Windows Server 20H2 Core. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.	19 de abril de 2021

Cambio	Descripción	Fecha
Amazon ECS Exec	Amazon ECS ha lanzado una nueva herramienta de depuración llamada ECS Exec. Para obtener más información, consulte Supervisión de los contenedores de Amazon ECS con ECS Exec.	15 de marzo de 2021
Admite la política de puntos de enlace de la VPC	Amazon ECS ahora admite las políticas de punto de enlace de la VPC Para obtener más información, consulte Creación de una política de puntos de enlace de la VPC para Amazon ECS.	11 de enero de 2021
Nueva experiencia de consola	Amazon ECS ha lanzado una nueva experiencia de consola que admite la creación o actualización de un servicio o la ejecución de una tarea independiente. Para obtener más información, consulte Creación de una implementación de actualización continua de Amazon ECS y Ejecución de una aplicación como tarea de Amazon ECS.	28 de diciembre de 2020
Actualización de proveedores de capacidad	Amazon ECS ahora admite la actualización de un proveedor de capacidad de grupo de Auto Scaling.	23 de noviembre de 2020
ECS ahora admite Amazon FSx for Windows File Server para tareas de Windows	Amazon ECS agregó compatibilidad con la especific ación de volúmenes de Amazon FSx for Windows File Server en las definiciones de tareas de Windows. Para obtener más información, consulte Uso de volúmenes de FSx para Windows File Server con Amazon ECS.	11 de noviembre de 2020
Se ha agregado compatibilidad con VPC en modo de pila doble	Amazon ECS ha agregado compatibilidad con la utilización de una VPC en modo de pila doble con tareas que utilizan el modo de red `awsvpc`, que proporciona compatibilidad con las direcciones IPv6. Para obtener más información, consulte Utilización de una VPC en modo de pila doble.	5 de noviembre de 2020

Cambio	Descripción	Fecha
Actualización del punto de enlace de metadatos de tareas v4	Amazon ECS agregó metadatos adicionales a los resultados del punto de enlace de metadatos de tareas v4. Para obtener más información, consulte Versión 4 del punto de conexión de metadatos de tareas de Amazon ECS.	5 de noviembre de 2020
Admite Local Zones y zonas de Wavelength	Amazon ECS ahora admite cargas de trabajo en Local Zones y zonas de Wavelength. Para obtener más información, consulte Aplicaciones de Amazon ECS en subredes compartidas, zonas locales y zonas de Wavelength.	4 de septiembre de 2020
Variante de la AMI de Bottlerocket para Amazon ECS	Bottlerocket es un sistema operativo de código abierto basado en Linux que está diseñado específicamente por AWS para ejecutar contenedores. Se proporcio na una variante de la AMI optimizada para Amazon ECS del sistema operativo Bottlerocket, que se puede utilizar al lanzar instancias de contenedor de Amazon ECS. Para obtener más información, consulte AMI Bottlerocket optimizadas para Amazon ECS.	31 de agosto de 2020
Se actualizó la versión 4 del punto de enlace de metadatos de tareas para las estadísticas de velocidad de red	Se actualizó la versión 4 del punto de enlace de metadatos de tareas para proporcionar estadísticas de velocidad de red para las tareas de Amazon ECS que utilizan los modos de red `awsvpc` o `bridge` alojados en instancias de Amazon EC2 que ejecutan al menos la versión `1.43.0` del agente de contenedor. Para obtener más información, consulte Versión 4 del punto de conexión de metadatos de tareas de Amazon ECS.	10 de agosto de 2020
Métricas de uso de Fargate	AWS Fargate brinda métricas de uso de CloudWatc h que proporcionan visibilidad del uso de recursos de Fargate On-Demand y Fargate Spot de su cuenta. Para obtener más información, consulte Métricas de uso.	3 de agosto de 2020

Cambio	Descripción	Fecha
AWS Copilot versión 0.1.0	La nueva CLI de AWS Copilot que se lanzó proporcio na comandos de alto nivel para simplificar el modelado, la creación, el lanzamiento y la administr ación de aplicaciones en contenedores en Amazon ECS desde un entorno de desarrollo local. Para obtener más información, consulte Creación de recursos de Amazon ECS mediante la interfaz de la línea de comandos de AWS Copilot.	9 de julio de 2020
Programación de obsolescencia de versiones de la plataform a de AWS Fargate	Se agregó el programa de obsolescencia de versiones de la plataforma de Fargate. Para obtener más información, consulte Obsolecencia de la versión de la plataforma AWS Fargate Linux.	8 de julio de 2020
Ampliación de las regiones de AWS Fargate	Amazon ECS en AWS Fargate se ha expandido a la región Europa (Milán).	25 de junio de 2020
Lanzamiento de la AMI de Amazon Linux 2 (Neuron) optimizada por Amazon ECS	Amazon ECS lanzó una AMI de Amazon Linux 2 (Neuron) optimizada por Amazon ECS para realizar cargas de trabajo inferenciales. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.	24 de junio de 2020
Compatibilidad agregada para eliminar proveedor es de capacidad	Amazon ECS ahora admite la eliminación de proveedores capacidad de grupo de Auto Scaling.	11 de junio de 2020
Actualización a la versión 1.4.0 de la plataforma de AWS Fargate	A partir del 28 de mayo de 2020, cualquier nueva tarea de Fargate que se lance a través de la versión 1.4.0 de la plataforma tendrá su almacenamiento efímero de 20 GB cifrado con un algoritmo de cifrado AES-256 mediante una clave de cifrado administrada de AWS Fargate. Para obtener más información, consulte Almacenamiento efímero de tareas de Fargate para Amazon ECS.	28 de mayo de 2020

Cambio	Descripción	Fecha
Compatibilidad con archivos de variables de entorno	Se ha añadido compatibilidad para especificar archivos de variables de entorno en una definición de tarea, lo que permite agregar variables de entorno en bloque a los contenedores. Para obtener más información, consulte Transferencia de una variable de entorno individual a un contenedor de Amazon ECS.	18 de mayo de 2020
Ampliación de las regiones de AWS Fargate	AWS Fargate with Amazon ECS se ha ampliado a la región África (Ciudad del Cabo).	11 de mayo de 2020
Cuota de servicio actualizada	Se ha actualizado la cuota de servicio siguiente: • Los clústeres por cuenta se han incrementado de `2,000` a `10,000`. Para obtener más información, consulte Puntos de conexión y cuotas de Amazon ECS.	17 de abril de 2020

Cambio	Descripción	Fecha
Plataforma de AWS Fargate versión 1.4.0	Se ha lanzado la versión 1.4.0 de la plataforma de AWS Fargate, que contiene las siguientes caracterí sticas: • Se ha agregado compatibilidad con el uso de volúmenes del sistema de archivos de Amazon EFS para el almacenamiento de tareas persisten tes. Para obtener más información, consulte Uso de volúmenes de Amazon EFS con Amazon ECS. • El almacenamiento de tareas efímeras se ha incrementado en 20 GB. Para obtener más informaci ón, consulte Almacenamiento efímero de tareas de Fargate para Amazon ECS. • Se ha actualizado el comportamiento del tráfico de red de entrada y salida de las tareas. A partir de la versión 1.4 de la plataforma, todas las tareas de Fargate reciben una única interfaz de red elástica (que se conoce como ENI de la tarea) y todo el tráfico de red fluye a través de esa ENI dentro de la VPC y podrá verse a través de los registros de flujo de la VPC. Para obtener más información, consulte Opciones de redes de tareas de Amazon ECS para el tipo de inicialización de Fargate. • Las ENI de tarea permiten utilizar tramas gigantes. Las interfaces de red se configuran con una unidad de transmisión máxima (MTU), que es el tamaño de la carga útil más grande que cabe en una sola trama. Cuanto mayor sea la MTU, mayor será la carga de la aplicación que cabe en una sola trama, lo que reduce la sobrecarga por trama y aumenta la eficiencia. Esta compatibilidad con las tramas	8 de abril de 2020

Cambio	Descripción	Fecha
	gigantes reducirá la sobrecarga cuando la ruta de red entre la tarea y el destino admita el uso de estas tramas, como el tráfico que permanece dentro de la VPC. • CloudWatch Container Insights incluirá métricas de rendimiento de red para las tareas de Fargate. Para obtener más información, consulte Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada. • Se ha agregado compatibilidad con el punto de enlace de metadatos de tareas v4, que proporciona información adicional sobre las tareas de Fargate, incluidas las estadísticas de red de la tarea y la zona de disponibilidad en la que se ejecuta la tarea. Para obtener más información, consulte Versión 4 del punto de conexión de metadatos de tareas de Amazon ECS. • Se ha agregado compatibilidad con el parámetro `SYS_PTRACE` de Linux en las definiciones de contenedores. Para obtener más información, consulte Parámetros de Linux. • El agente de contenedor de Fargate sustituye al de Amazon ECS para todas las tareas de Fargate. Este cambio no debería tener ningún efecto en el modo en el que se ejecutan las tareas. • El entorno de ejecución del contenedor ahora utiliza Containerd en lugar de Docker. Este cambio no debería tener ningún efecto en el modo en el que se ejecutan las tareas. Como podrá ver, algunos	

Cambio	Descripción	Fecha
	mensajes de error que se originan con el entorno de ejecución del contenedor cambiarán y ya no mencionarán a Docker sino que devolverán errores más generales. Para obtener más información, consulte Versiones de la plataforma Fargate para Amazon ECS.	
Compatibilidad con el sistema de archivos de Amazon EFS para volúmenes de tareas	Los sistemas de archivos de Amazon EFS pueden utilizarse como volúmenes de datos tanto en las tareas de Amazon ECS como en las de Fargate. Para obtener más información, consulte Uso de volúmenes de Amazon EFS con Amazon ECS.	8 de abril de 2020
Versión 4 del punto de conexión de metadatos de tareas de Amazon ECS	A partir de la versión 1.39.0 del agente de contenedor de Amazon ECS y la versión 1.4.0 de la plataforma de Fargate, se ha incluido una variable de entorno con el nombre `ECS_CONTAINER_METADATA_URI_V4` en cada contenedor de las tareas. Cuando consulta la versión 4 del punto de enlace de metadatos de tareas, hay diversos metadatos y estadísticas de Docker disponibles para las tareas. Para obtener más información, consulte Versión 4 del punto de conexión de metadatos de tareas de Amazon ECS.	8 de abril de 2020
Compatibilidad para la introducción de versiones específicas de secretos de Secrets Manager como variables de entorno	Ahora se admite la especificación de información confidencial utilizando versiones específicas de secretos de Secrets Manager. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.	24 de febrero de 2020

Cambio	Descripción	Fecha
Se han agregado opciones de configuración de implementación de CodeDeploy adicionales para implementaciones “blue/green”	El servicio de CodeDeploy ha agregado nuevas configuraciones de implementación de valor controlado y lineal para el tipo de implementación de Amazon ECS. También está disponible la posibilidad de definir configuraciones de implementación personalizadas. Para obtener más información, consulte Implementaciones azul/verde de CodeDeploy para Amazon ECS.	6 de febrero de 2020
Incorporación del parámetro de definición de tarea efsVolumeConfiguration	El parámetro de definición de tareas `efsVolumeConfiguration` se encuentra en la vista previa pública, lo que facilita la utilización de los sistemas de archivos de Amazon EFS con las tareas de Amazon ECS. Para obtener más información, consulte Uso de volúmenes de Amazon EFS con Amazon ECS.	17 de enero de 2020
Actualización del comportamiento de registro del agente de contenedor de Amazon ECS	Se han actualizado las ubicaciones de registro de agente de contenedor y el comportamiento de rotación de Amazon ECS. Para obtener más información, consulte Parámetros de configuración del registro del agente de contenedor de Amazon ECS.	13 de enero de 2020
Fargate Spot	En Amazon ECS, se agregó compatibilidad con la ejecución de tareas mediante Fargate Spot. Para obtener más información, consulte Clústeres de Amazon ECS para Fargate.	3 de diciembre de 2019
Escalado automático de clústeres	El Auto Scaling de clústeres de Amazon ECS permite tener más control sobre la forma de escalado de las tareas dentro de un clúster. Para obtener más información, consulte Administración automática de la capacidad de Amazon ECS con el escalado automático de clústeres.	3 de diciembre de 2019

Cambio	Descripción	Fecha
Proveedores de capacidad de clúster	Los proveedores de capacidad de clúster de Amazon ECS determinan la infraestructura que se va a utilizar para las tareas. Para obtener más información, consulte Clústeres de Amazon ECS.	3 de diciembre de 2019
Creación de un clúster en AWS Outposts	Amazon ECS ahora admite la creación de clústeres en AWS Outposts. Para obtener más información, consulte the section called “Amazon Elastic Container Service en AWS Outposts”.	3 de diciembre de 2019
Eventos de acciones de servicio	Amazon ECS ahora envía eventos a Amazon EventBridge cuando se producen algunas acciones de servicio. Para obtener más información, consulte Eventos de acciones de servicio de Amazon ECS.	25 de noviembre de 2019
La AMI optimizada para GPU de Amazon ECS admite instancias G4	Amazon ECS ha agregado compatibilidad con la familia de tipos de instancias G4 cuando se utiliza la AMI optimizada para GPU de Amazon ECS. Para obtener más información, consulte Definiciones de tareas de Amazon ECS para cargas de trabajo de GPU.	8 de octubre de 2019
FireLens para Amazon ECS	FireLens para Amazon ECS es un producto de disponibilidad general. FireLens para Amazon ECS permite utilizar parámetros de definición de tareas para dirigir los registros a un destino de socios o servicio de AWS para el almacenamiento y el análisis de registros. Para obtener más información, consulte Envío de registros de Amazon ECS a un servicio de AWS o AWS Partner.	30 de septiembre de 2019
Ampliación de las regiones de AWS Fargate	AWS Fargate en Amazon ECS se ha ampliado a las regiones de Europa (París), Europa (Estocolmo) y Medio Oriente (Baréin).	30 de septiembre de 2019

Cambio	Descripción	Fecha
Deep Learning Containers con Elastic Inference en Amazon ECS	Amazon ECS permite asociar aceleradores de Amazon Elastic Inference a los contenedores para que la ejecución de cargas de trabajo de inferencia de aprendizaje profundo sea más eficaz.	3 de septiembre de 2019
FireLens para Amazon ECS	FireLens para Amazon ECS se encuentra en vista previa pública. FireLens para Amazon ECS permite utilizar parámetros de definición de tareas para dirigir los registros a un destino de socios o servicio de AWS para el almacenamiento y el análisis de registros. Para obtener más información, consulte Envío de registros de Amazon ECS a un servicio de AWS o AWS Partner.	30 de agosto de 2019
Información de contenedores de CloudWatch	CloudWatch Container Insights ya está disponible con carácter general. Le permite recopilar, agregar y resumir métricas y registros de microservicios y aplicaciones en contenedores. Para obtener más información, consulte Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada.	30 de agosto de 2019
Configuración del intercambio de nivel de contenedor	Amazon ECS ahora admite el control de la utilización del espacio de memoria de intercambio en las instancias de contenedor de Linux en el nivel de contenedor. Con una configuración de intercambio por contenedor, cada contenedor incluido en una definición de tarea puede tener el intercambio habilitado o deshabilitado, y para aquellos que lo tienen habilitado, se puede limitar la cantidad máxima de espacio de intercambio utilizado. Para obtener más información, consulte Administración del espacio de memoria de intercambio de contenedores en Amazon ECS.	16 de agosto de 2019

Cambio	Descripción	Fecha
Ampliación de las regiones de AWS Fargate	AWS Fargate con Amazon ECS se ha ampliado a la región Asia-Pacífico (Hong Kong).	6 de agosto de 2019
Enlace troncal de interfaz de red elástica	Se han agregado otros tipos de instancias de Amazon EC2 compatibles con la característica de redes troncales ENI. Para obtener más información, consulte Instancias admitidas para un aumento de las interfaces de red de contenedores de Amazon ECS.	1 de agosto de 2019
Registro de varios grupos de destino con un servicio	Se ha agregado compatibilidad para especificar varios grupos de destino en una definición de servicio. Para obtener más información, consulte Registro de varios grupos de destino en un servicio de Amazon ECS.	30 de julio de 2019
Especificación de información confidencial mediante secretos de Secrets Manager	Se ha agregado un tutorial sobre la especificación de información confidencial mediante secretos de Secrets Manager. Para obtener más información, consulte Especificación de información confidencial mediante secretos de Secrets Manager en Amazon ECS.	20 de julio de 2019
Información de contenedores de CloudWatch	Amazon ECS ha agregado compatibilidad con CloudWatch Container Insights. Para obtener más información, consulte Supervisión de contenedores de Amazon ECS mediante Información de contenedores con capacidad de observabilidad mejorada.	9 de julio de 2019
Permisos de nivel de recursos para servicios y conjuntos de tareas de servicios de Amazon ECS	Amazon ECS ha ampliado la compatibilidad de los permisos de nivel de recursos para los servicios y conjuntos de tareas de los servicios de Amazon ECS. Para obtener más información, consulte Cómo funciona Amazon Elastic Container Service con IAM.	27 de junio de 2019

Cambio	Descripción	Fecha
Nueva AMI optimizada para Amazon ECS con parches para AWS-2019-005	Amazon ECS ha actualizado la AMI optimizada para Amazon ECS a fin de resolver las vulnerabilidades descritas en AWS-2019-005.	17 de junio de 2019
Enlace troncal de interfaz de red elástica	Amazon ECS presenta compatibilidad con el lanzamiento de instancias de contenedor con tipos de instancia de Amazon EC2 admitidos que tienen una mayor densidad de interfaz de red elástica (ENI). El uso de estos tipos de instancias y la elección de la configuración de cuenta `awsvpcTrunking` proporcio nan una mayor densidad de ENI en las instancias de contenedor recién lanzadas, lo cual permite colocar más tareas en cada instancia de contenedor. Para obtener más información, consulte Aumento de las interfaces de red de instancias de contenedor de Linux de Amazon ECS.	6 de junio de 2019
Actualización a la versión 1.3.0 de la plataforma de AWS Fargate	A partir del 1 de mayo de 2019, cualquier nueva tarea de Fargate que se lance es compatible con el controlador de registros `splunk` además del controlad or de registros `awslogs`. Para obtener más informaci ón, consulte Almacenamiento y registro.	1 de mayo de 2019
Actualización a la versión 1.3.0 de la plataforma de AWS Fargate	A partir del 1 de mayo de 2019, cualquier nueva tarea de Fargate que se lanza admite hacer referenci a a información confidencial en la configuración de registros de un contenedor mediante el parámetro de definición de contenedor `secretOptions` . Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.	1 de mayo de 2019

Cambio	Descripción	Fecha
Actualización a la versión 1.3.0 de la plataforma de AWS Fargate	A partir del 2 de abril de 2019, cualquier nueva tarea de Fargate que se lance admite introducir informaci ón confidencial en sus contenedores mediante su almacenamiento en secretos de AWS Secrets Manager o en parámetros del Parameter Store de AWS Systems Manager y la posterior referencia a ellos en la definición de contenedor. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.	2 de abril de 2019
Actualización a la versión 1.3.0 de la plataforma de AWS Fargate	A partir del 27 de marzo de 2019, cualquier nueva tarea de Fargate lanzada puede utilizar parámetros de definición de tareas adicionales que le permiten definir una configuración del proxy, dependencias para el inicio y apagado del contenedor, y un valor de tiempo de espera de inicio y detención por contenedor. Para obtener más información, consulte Configuración del proxy, Dependencia de contenedor y Tiempos de espera de contenedor.	27 de marzo de 2019
Amazon ECS presenta el tipo de implementación externa	El tipo de implementación externa permite que utilice cualquier controlador de implementación de terceros para tener un control completo del proceso de implementación de un servicio de Amazon ECS. Para obtener más información, consulte Implementación de los servicios de Amazon ECS mediante un controlador de terceros.	27 de marzo de 2019

Cambio	Descripción	Fecha
AWS Deep Learning Containers en Amazon ECS	Los Deep Learning Containers de AWS son un conjunto de imágenes de Docker que se utilizan para entrenar y trabajar con modelos en TensorFlow en Amazon Elastic Container Service (Amazon ECS). Los Deep Learning Containers proporcionan entornos optimizados con bibliotecas de TensorFlow, CUDA de Nvidia (para instancias de GPU) y MKL de Intel (para instancias de CPU) y están disponibles en Amazon ECR. Para obtener más información, consulte Uso de contenedores de aprendizaje profundo de AWS en Amazon ECS.	27 de marzo de 2019
Amazon ECS presenta administración de dependencias de contenedores mejorada	Amazon ECS presenta parámetros de definición de tareas adicionales que le permiten definir dependenc ias para el encendido y apagado de contenedores, como también un valor de tiempo de espera de inicio y detención por contenedor. Para obtener más informaci ón, consulte Dependencia de contenedor.	7 de marzo de 2019
Amazon ECS presenta la API `PutAccoun tSettingDefault`	Amazon ECS presenta la API `PutAccoun tSettingDefault`, la cual permite a un usuario definir el estado de formato de ARN/ID predeterm inado incluido para todos los usuarios y roles de la cuenta. Anteriormente, para configurar el estado de suscripción predeterminado de la cuenta era necesario utilizar la cuenta de usuario del propietario. Para obtener más información, consulte Nombres de recursos de Amazon (ARN) e ID.	8 de febrero de 2019

Cambio	Descripción	Fecha
Amazon ECS admite cargas de trabajo de GPU	Amazon ECS introduce compatibilidad con cargas de trabajo de GPU para que pueda crear clústeres con instancias de contenedor habilitadas para GPU. En una definición de tarea puede especificar el número de GPU necesarias y el agente de ECS adjuntará las GPU físicas al contenedor. Para obtener más información, consulte Definiciones de tareas de Amazon ECS para cargas de trabajo de GPU.	4 de febrero de 2019
Compatibilidad ampliada con secretos en Amazon ECS	Amazon ECS ha ampliado la compatibilidad con la utilización de secretos de AWS Secrets Manager directamente en las definiciones de tareas a fin de incluir información confidencial en los contenedores. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.	21 de enero de 2019
Puntos de conexión de VPC de interfaz (AWS PrivateLink)	Se ha agregado compatibilidad para configurar puntos de enlace de la VPC de interfaz basados en AWS PrivateLink. Esto le permite crear una conexión privada entre su VPC y Amazon ECS sin que se requiera el acceso a través de Internet, a través de una instancia NAT, una conexión de VPN o AWS Direct Connect. Para obtener más información, consulte Puntos de enlace de la VPC de interfaz de (AWS PrivateLink).	26 de diciembre de 2018

Cambio	Descripción	Fecha
Plataforma de AWS Fargate versión 1.3.0	Se ha publicado una nueva versión de la plataforma de AWS Fargate que contiene lo siguiente: • Se ha agregado compatibilidad para el uso de parámetros AWS Systems Manager Parameter Store para inyectar información confidencial en los contenedores. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS. • Se ha agregado el reciclaje de tareas para las tareas de Fargate, un proceso que consiste en actualizar las tareas que forman parte de un servicio de Amazon ECS. Para obtener más información, consulte Retirada y mantenimiento de tareas para AWS Fargate en Amazon ECS. Para obtener más información, consulte Versiones de la plataforma Fargate para Amazon ECS.	17 de diciembre de 2018

Cambio	Descripción	Fecha
Límites de servicios actualizados	Se han actualizado los siguientes límites de servicio: • El número de clústeres por región y por cuenta se incrementado de `1000` a `2000`. • El número de instancias de contenedor por clúster se incrementado de `1000` a `2000`. • El número de servicios por clúster se incrementado de `500` a `1000`. Para obtener más información, consulte Cuotas de servicio de Amazon ECS.	14 de diciembre de 2018
Ampliación de las regiones de AWS Fargate	AWS Fargate con Amazon ECS se ha ampliado a las regiones Asia-Pacífico (Mumbai) y Canadá (Central). Para obtener más información, consulte Regiones compatibles con Amazon ECS en AWS Fargate.	7 de diciembre de 2018
Implementaciones “blue/green” de Amazon ECS	Amazon ECS ha agregado compatibilidad con las implementaciones “blue/green” a través de CodeDeploy. Este tipo de implementación permite verificar una nueva implementación de un servicio antes de enviarle el tráfico de producción. Para obtener más información, consulte Implementaciones azul/verde de CodeDeploy para Amazon ECS.	27 de noviembre de 2018
Lanzamiento de la AMI de Amazon Linux 2 (arm64) optimizada para Amazon ECS	Amazon ECS ha lanzado una AMI de Amazon Linux 2 optimizada para Amazon ECS para la arquitectura arm64. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.	26 de noviembre de 2018

Cambio	Descripción	Fecha
Incorporación de compatibilidad para opciones adicionales de Docker en las definicio nes de tareas	Amazon ECS ha agregado compatibilidad con los siguientes indicadores de Docker en las definiciones de tarea: • Modo IPC • Modo PID	16 de noviembre de 2018
Compatibilidad con los secretos de Amazon ECS	Amazon ECS ha agregado compatibilidad con la utilización de parámetros de Parameter Store de AWS Systems Manager para inyectar información confidenc ial en los contenedores. Para obtener más información, consulte Transferencia de datos confidenciales a un contenedor de Amazon ECS.	15 de noviembre de 2018
Etiquetado de recursos	Amazon ECS ha agregado compatibilidad para incorporar etiquetas de metadatos a sus servicios, definiciones de tareas, tareas, clústeres e instancias de contenedor. Para obtener más información, consulte Etiquetado de los recursos de Amazon ECS.	15 de noviembre de 2018
Ampliación de las regiones de AWS Fargate	AWS Fargate con Amazon ECS se ha ampliado a las regiones EE. UU. Oeste (Norte de California) y Asia-Pacífico (Seúl). Para obtener más información, consulte AWS Fargate para Amazon ECS.	7 de noviembre de 2018

Cambio	Descripción	Fecha
Límites de servicios actualizados	Se han actualizado los siguientes límites de servicio: • Se ha aumentado el número de tareas que utilizan el tipo de lanzamiento de Fargate por cada cuenta y región de `20` a `50`. • Se ha aumentado el número de direcciones IP públicas de tareas que utilizan el tipo lanzamiento de Fargate de `20` a `50`. Para obtener más información, consulte Cuotas de servicio de Amazon ECS.	31 de octubre de 2018
Ampliación de las regiones de AWS Fargate	AWS Fargate con Amazon ECS se ha ampliado a la región Europa (Londres). Para obtener más información, consulte AWS Fargate para Amazon ECS.	26 de octubre de 2018
Lanzamiento de la AMI de Amazon Linux 2 optimizada para Amazon ECS	Amazon ECS provee AMI de Linux optimizadas para el servicio en dos variantes. La versión más reciente recomendada se basa en x;. Amazon ECS también provee AMI, pero recomendamos migrar las cargas de trabajo a la variante Amazon Linux 2, dado que la compatibilidad con la AMI de Amazon Linux finalizará a más tardar el 30 de junio de 2020. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.	18 de octubre de 2018

Cambio	Descripción	Fecha
Versión 3 del punto de enlace de metadatos de tareas de Amazon ECS	A partir de la versión 1.21.0 del agente de contenedor de Amazon ECS, el agente introduce una variable de entorno con el nombre `ECS_CONTAINER_METADATA_URI` en cada contenedor de las tareas. Al consultar la versión 3 del punto de enlace de metadatos de tareas, están disponibles diversos metadatos de tareas y estadísticas de Docker para las tareas que utilizan el modo de red `awsvpc` en un punto de enlace HTTP proporcionado por el agente de contenedor de Amazon ECS. Para obtener más información, consulte Supervisión de las cargas de trabajo mediante metadatos de Amazon ECS.	18 de octubre de 2018
Ampliación de las regiones de detección de servicios de Amazon ECS	La detección de servicios de Amazon ECS se ha ampliado y ahora admite las regiones Canadá (Central), América del Sur (São Paulo), Asia-Pacífico (Seúl), Asia-Pacífico (Mumbai) y Europa (París). Para obtener más información, consulte Uso de la detección de servicios para conectar los servicios de Amazon ECS con nombres de DNS.	27 de septiembre de 2018
Incorporación de compatibilidad para opciones adicionales de Docker en las definiciones de contenedor	Amazon ECS ahora admite la utilización de los siguientes indicadores de Docker en las definiciones de contenedor: • Controles del sistema • Interactivo • Pseudo terminal	17 de septiembre de 2018

Cambio	Descripción	Fecha
Compatibilidad con la autenticación de registros privados para Amazon ECS utilizando tareas de AWS Fargate	Amazon ECS ahora permite la utilización de tareas de Fargate mediante la autenticación de registros privados a través de AWS Secrets Manager. Esta característica le permite almacenar sus credencia les de forma segura y hacer referencia a ellas en la definición de contenedor, lo que permite que las tareas utilicen imágenes privadas. Para obtener más información, consulte Uso de imágenes de contenedor que no sean de AWS en Amazon ECS.	10 de septiembre de 2018
Ampliación de las regiones de detección de servicios de Amazon ECS	La detección de servicios de Amazon ECS se ha ampliado y ahora admite las regiones Asia-Pacífico (Singapur), Asia-Pacífico (Sídney), Asia-Pacífico (Tokio), EU (Fráncfort) y EU (Londres). Para obtener más información, consulte Uso de la detección de servicios para conectar los servicios de Amazon ECS con nombres de DNS.	30 de agosto de 2018
Tareas programadas que admiten tareas de Fargate	Amazon ECS ahora admite el uso de tareas programadas para el tipo de lanzamiento de Fargate.	28 de agosto de 2018
Autenticación de registros privados mediante AWS Secrets Manager	Amazon ECS ahora admite autenticación de registros privados mediante AWS Secrets Manager. Esta característica le permite almacenar sus credencia les de forma segura y hacer referencia a ellas en la definición de contenedor, lo que permite que las tareas utilicen imágenes privadas. Para obtener más información, consulte Uso de imágenes de contenedor que no sean de AWS en Amazon ECS.	16 de agosto de 2018

Cambio	Descripción	Fecha
Incorporación de compatibilidad con volúmenes de Docker	Amazon ECS ahora admite la utilización de de volúmenes de Docker. Para obtener más información, consulte Opciones de almacenamiento para las tareas de Amazon ECS.	9 de agosto de 2018
Ampliación de las regiones de AWS Fargate	AWS Fargate con Amazon ECS se ha ampliado a las regiones Europa (Fráncfort), Asia-Pacífico (Singapur) y Asia-Pacífico (Sídney). Para obtener más información, consulte AWS Fargate para Amazon ECS.	19 de julio de 2018

Cambio	Descripción	Fecha
Incorporación de estrategias de programación del servicio de Amazon ECS	Amazon ECS presentó el concepto de estrategias de programador de servicio. Existen dos estrategias del programador de servicio: • `REPLICA`: la estrategia de programación de réplicas sitúa y mantiene en el clúster el número de tareas deseado. De forma predeterminada, el programador de servicio distribuye las tareas en zonas de disponibilidad. Puede utilizar estrategias y restricciones de ubicación de tareas para personalizar las decisiones de ubicación de las tareas. Para obtener más información, consulte Estrategia de programación de réplicas. • `DAEMON`: la estrategia de programación del daemon implementa exactamente una tarea en cada instancia de contenedor activa que cumpla todas las restricciones de ubicación de tareas que se especifiquen para el clúster. Cuando se utiliza esta estrategia, no es necesario especificar un número deseado de tareas, ni una estrategia de ubicación de tareas ni utilizar políticas de Auto Scaling de servicios. Para obtener más información, consulte Estrategia de programación de daemon. Note Las tareas de Fargate no admiten la estrategia de programación de `DAEMON`. .	12 de junio de 2018

Cambio	Descripción	Fecha
Agente de contenedor de Amazon ECS v1.18.0	Se ha publicado una nueva versión del agente de contenedor de Amazon ECS, que agrega la siguiente funcionalidad: • Se ha añadido compatibilidad para personalizar el comportamiento de extracción de imágenes del contenedor mediante el parámetro `ECS_IMAGE_PULL_BEHAVIOR`. Para obtener más información, consulte Configuración del agente de contenedor de Amazon ECS. Para obtener más información, consulte amazon-ecs-agent github.	24 de mayo de 2018
Se agregó compatibilidad con los modos de red `bridge` y `host` al configurar la detección de servicios	Se agregó compatibilidad con la configuración de la detección de servicios para los servicios de Amazon ECS mediante definiciones de tareas que especifican los modos de red `bridge` o `host`. Para obtener más información, consulte Uso de la detección de servicios para conectar los servicios de Amazon ECS con nombres de DNS.	22 de mayo de 2018
Se ha agregado compatibilidad con parámetros de metadatos adicionales de la AMI optimizada para Amazon ECS	Se han agregado subparámetros que permiten recuperar mediante programación el ID de la AMI optimizada para Amazon ECS, el nombre de la imagen, el sistema operativo, la versión del agente de contenedor y la versión de tiempo de ejecución. Consulte los metadatos mediante la API de Parameter Store de Systems Manager. Para obtener más información, consulte Recuperación de metadatos de las AMI de Linux optimizadas para Amazon ECS.	9 de mayo de 2018

Cambio	Descripción	Fecha
Ampliación de las regiones de AWS Fargate	AWS Fargate con Amazon ECS se ha ampliado a las regiones EE. UU. Este (Ohio), EE. UU. Oeste (Oregón) y EU Oeste (Irlanda). Para obtener más información, consulte AWS Fargate para Amazon ECS.	26 de abril de 2018
Recuperación de los metadatos de la AMI optimizada para Amazon ECS	Se ha agregado la capacidad de recuperar los metadatos de la AMI optimizada para Amazon ECS mediante programación a través de la API de Parameter Store de Systems Manager. Para obtener más información, consulte Recuperación de metadatos de las AMI de Linux optimizadas para Amazon ECS.	10 de abril de 2018
Versión de la plataforma de AWS Fargate	Se ha publicado una nueva versión de la plataforma de AWS Fargate que contiene lo siguiente: • Se agregó compatibilidad con Supervisión de las cargas de trabajo mediante metadatos de Amazon ECS. • Se agregó compatibilidad con Comprobación de estado. • Se ha agregado compatibilidad para Uso de la detección de servicios para conectar los servicios de Amazon ECS con nombres de DNS Para obtener más información, consulte Versiones de la plataforma Fargate para Amazon ECS.	26 de marzo de 2018

Cambio	Descripción	Fecha
Detección de servicio de Amazon ECS	Se ha agregado la integración con Route 53 para admitir la detección de servicios de Amazon ECS. Para obtener más información, consulte Uso de la detección de servicios para conectar los servicios de Amazon ECS con nombres de DNS.	22 de marzo de 2018
Compatibilidad con shm-size y tmpfs en Docker	Se ha agregado compatibilidad con los parámetros shm-size y tmpfs de Docker en las definiciones de tareas de Amazon ECS. Para obtener más información sobre la sintaxis actualizada de la CLI de ECS, consulte Parámetros de Linux.	20 de marzo de 2018
Comprobaciones de estado de contenedores	Se ha agregado compatibilidad para las comprobac iones de estado de Docker en las definiciones de contenedor. Para obtener más información, consulte Comprobación de estado.	8 de marzo de 2018
AWS Fargate	Se ha agregado información general para Amazon ECS con AWS Fargate. Para obtener más informaci ón, consulte AWS Fargate para Amazon ECS.	22 de febrero de 2018
Punto de enlace de metadatos de tareas de Amazon ECS	A partir de la versión 1.17.0 del agente de contenedo r de Amazon ECS, están disponibles diversos metadatos de tarea y estadísticas de Docker para las tareas que utilizan el modo de red `awsvpc` en un punto de enlace HTTP proporcionado por el agente de contenedor de Amazon ECS. Para obtener más información, consulte Supervisión de las cargas de trabajo mediante metadatos de Amazon ECS.	8 de febrero de 2018

Cambio	Descripción	Fecha
Service Auto Scaling de Amazon ECS mediante políticas de seguimiento de destino	Se ha agregado compatibilidad con ECS Service Auto Scaling mediante políticas de seguimiento de destino en la consola de Amazon ECS. Para obtener más información, consulte Uso de una métrica objetivo para escalar los servicios de Amazon ECS. Se ha eliminado el tutorial anterior para el escalado por pasos en el asistente de primera ejecución de ECS. Se ha sustituido por el nuevo tutorial de seguimiento de destino.	8 de febrero de 2018
Compatibilidad con Docker 17.09	Se ha agregado compatibilidad con Docker 17.09. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.	18 de enero de 2018
Nuevo comportamiento del programador de servicio	Se ha actualizado la información sobre el comportamiento para las tareas de servicio que no se pueden lanzar. Se ha documentado un nuevo mensaje de evento de servicio que se activa cuando una tarea de servicio tiene errores consecutivos.	11 de enero de 2018
Período de espera para la inicialización de la comprobación de estado de Elastic Load Balancing	Se ha añadido la capacidad de especificar un periodo de espera para las comprobaciones de estado.	27 de diciembre de 2017
CPU y memoria de nivel de tarea	Se ha añadido compatibilidad con la especificación de memoria y CPU en el nivel de tarea en las definiciones de tareas. Para obtener más información, consulte TaskDefinition.	12 de diciembre de 2017

Cambio	Descripción	Fecha
Rol de ejecución de tareas	El agente de contenedor de Amazon ECS realiza llamadas a las acciones de la API de ECS en su nombre, de modo que requiere una política y un rol de IAM para que el servicio sepa que el agente le pertenece a usted. El rol de ejecución de tareas abarca las siguientes acciones: • Llamadas a Amazon ECR para extraer la imagen del contenedor • Llamadas a CloudWatch para almacenar los registros de aplicación del contenedor Para obtener más información, consulte Rol de IAM de ejecución de tareas de Amazon ECS.	7 de diciembre de 2017
Los contenedores de Windows admiten GA	Se ha agregado compatibilidad con Windows Server 2016. Para obtener más información, consulte Variantes de AMI optimizadas para Amazon ECS.	5 de diciembre de 2017
AWS Fargate disponible de manera general	Se ha agregado compatibilidad para lanzar servicios de Amazon ECS mediante el tipo de lanzamiento de Fargate. Para obtener más información, consulte Tipos de lanzamiento de Amazon ECS.	29 de noviembre de 2017
Cambio de nombre de Amazon ECS	Se ha cambiado el nombre a Amazon Elastic Container Service (anteriormente era Amazon EC2 Container Service).	21 de noviembre de 2017

Cambio	Descripción	Fecha
Integración en red de las tareas	Las características de redes de tareas que ofrece el modo de red awsvpc proporcionan a las tareas de Amazon ECS las mismas propiedades de redes que poseen las instancias de Amazon EC2. Cuando se utiliza el modo de red awsvpc en una definición de tarea, cada tarea que se lanza desde esa definició n de tarea obtiene su propia interfaz de red elástica, una dirección IP privada principal y un nombre de host DNS interno. La característica de integración en red de tareas simplifica la integración de contenedores en redes y proporciona más control sobre cómo se comunican entre sí las aplicaciones en contenedores y con los demás servicios de las VPC. Para obtener más información, consulte Opciones de redes de tareas de Amazon ECS para el tipo de lanzamiento de EC2.	14 de noviembre de 2017
Metadatos de los contenedores de Amazon ECS	Los contenedores de Amazon ECS ahora pueden obtener acceso a metadatos, como su contenedor de Docker o ID de imagen, la configuración de red o los ARN de Amazon. Para obtener más información, consulte Archivo de metadatos de contenedores de Amazon ECS.	2 de noviembre de 2017
Compatibilidad con Docker 17.06	Se ha agregado compatibilidad con Docker 17.06. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.	2 de noviembre de 2017
Compatibilidad con marcas de Docker: device e init	Se ha añadido compatibilidad con las caracterí sticas device e init de Docker en las definiciones de tareas mediante el parámetro `LinuxParameters` (`devices` e `initProcessEnabled`). Para obtener más información, consulte LinuxParameters.	2 de noviembre de 2017

Cambio	Descripción	Fecha
Compatibilidad con marcas de Docker: cap-add y cap-drop	Se ha añadido compatibilidad con las características cap-add y cap-drop de Docker en las definiciones de tareas mediante el parámetro `LinuxParameters` (`capabilities`). Para obtener más información, consulte LinuxParameters.	22 de septiembre de 2017
Compatibilidad con el Network Load Balancer	Amazon ECS ahora admite Network Load Balancers en la consola de Amazon ECS.	7 de septiembre de 2017
Reemplazos de RunTask	Se ha añadido compatibilidad con los reemplazo s de definición de tarea al ejecutar una tarea. Esto le permite ejecutar una tarea mientras cambia una definición de tarea sin necesidad de crear una nueva revisión de definición de tarea. Para obtener más información, consulte Ejecución de una aplicación como tarea de Amazon ECS.	27 de junio de 2017
Tareas programadas de Amazon ECS	Se ha añadido compatibilidad con la programación de tareas mediante Cron.	7 de junio de 2017
Instancias de spot en la consola de Amazon ECS	Se ha agregado compatibilidad con la creación de instancias de contenedor de flota de spot en la consola de Amazon ECS. Para obtener más información, consulte Lanzamiento de una instancia de contenedor de Linux de Amazon ECS.	6 de junio de 2017
Notificación de Amazon SNS sobre nuevas versiones de AMI optimizadas para Amazon ECS	Se ha agregado la capacidad de suscribirse a notificac iones de SNS sobre las nuevas versiones de AMI optimizadas para Amazon ECS.	23 de marzo de 2017
Microservicios y trabajos por lotes	Se ha agregado documentación para dos casos de uso comunes de Amazon ECS: microservicios y trabajos por lotes.	de febrero de 2017

Cambio	Descripción	Fecha
Vaciado de instancias de contenedor	Se ha añadido compatibilidad con el vaciado de instancias de contenedor, que proporciona un método para eliminar instancias de contenedor de un clúster. Para obtener más información, consulte Drenaje de instancias de contenedor de Amazon ECS.	24 de enero de 2017
Compatibilidad con Docker 1.12	Se ha agregado compatibilidad con Docker 1.12. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.	24 de enero de 2017
Nuevas estrategias de ubicación de tareas	Se ha añadido compatibilidad con estrategias de ubicación de tareas: ubicación en función del atributo, agrupación en bin packing, distribución de zonas de disponibilidad y una por host. Para obtener más información, consulte Uso de estrategias para definir la ubicación de las tareas de Amazon ECS.	29 de diciembre de 2016
Compatibilidad con contenedores de Windows en versión beta	Se ha agregado compatibilidad con los contenedores de Windows Server 2016 (beta). Para obtener más información, consulte Variantes de AMI optimizadas para Amazon ECS.	20 de diciembre de 2016
Compatibilidad con Blox OSS	Se ha añadido compatibilidad con Blox OSS, lo que permite disponer de programadores de tareas personalizados. Para obtener más información, consulte Programación de los contenedores en Amazon ECS.	1 de diciembre de 2016
Secuencia de eventos de Amazon ECS para CloudWatch Events	Amazon ECS ahora envía los cambios de instancia de contenedor y estado de las tareas a CloudWatch Events. Para obtener más información, consulte Automaticización de las respuestas a los errores de Amazon ECS mediante EventBridge.	21 de noviembre de 2016

Cambio	Descripción	Fecha
Registro de contenedores de Amazon ECS en CloudWatch Logs	Se ha agregado compatibilidad con el controlador awslogs para enviar secuencias de registros del contenedor a CloudWatch Logs. Para obtener más información, consulte Envío de registros de Amazon ECS a CloudWatch .	12 de septiembre de 2016
Servicios de Amazon ECS que admiten Elastic Load Balancing para puertos dinámicos	Los balanceadores de carga ahora admiten varias combinaciones de instancia-puerto por agente de escucha, lo que aumenta la flexibilidad de los contenedores. Ahora puede permitir que Docker defina dinámicamente el puerto de host del contenedor y que el programador de ECS registre la instancia-puerto en el balanceador de carga. Para obtener más información, consulte Uso del equilibrador de carga para distribuir el tráfico de servicio de Amazon ECS.	11 de agosto de 2016
Roles de IAM para tareas de Amazon ECS	Se ha agregado la posibilidad de asociar roles de IAM a una tarea. Esto proporciona permisos muy precisos a los contenedores en comparación con los que ofrece un único rol para una instancia de contenedor completa. Para obtener más información, consulte Rol de IAM de tarea de Amazon ECS.	13 de julio de 2016
Compatibilidad con Docker 1.11	Se ha agregado compatibilidad con Docker 1.11. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.	31 de mayo de 2016
Escalado automático de tareas	Amazon ECS agrega compatibilidad con el escalado automático de las tareas que ejecuta un servicio. Para obtener más información, consulte Escalado automático de su servicio de Amazon ECS.	18 de mayo de 2016
Filtrado de definiciones de tareas por familia de tareas	Se ha añadido posibilidad de filtrar las definiciones de tareas basándose en la familia de tareas. Para obtener más información, consulte ListTaskDefinitions.	17 de mayo de 2016

Cambio	Descripción	Fecha
Registro del contenedor de Docker y el agente de Amazon ECS	Amazon ECS agrega la posibilidad de enviar registros del agente de ECS y del contenedor de Docker desde las instancias de contenedor a CloudWatch Logs para simplificar la solución de problemas.	5 de mayo de 2016
La AMI optimizada para ECS ahora admite Amazon Linux 2016.03.	La AMI optimizada para ECS ahora es compatible con Amazon Linux 2016.03. Para obtener más informaci ón, consulte AMI de Linux optimizadas para Amazon ECS.	5 de abril de 2016
Compatibilidad con Docker 1.9	Se ha agregado compatibilidad con Docker 1.9. Para obtener más información, consulte AMI de Linux optimizadas para Amazon ECS.	22 de diciembre de 2015
Métricas de CloudWatch para la reserva de CPU y de memoria del clúster	Amazon ECS incorpora métricas de CloudWatch personalizadas para la reserva de CPU y de memoria.	22 de diciembre de 2015
Nueva experiencia de primer uso de Amazon ECS	La experiencia de primer uso de la consola de Amazon ECS incorpora la creación de roles con cero clics.	23 de noviembre de 2015
Ubicación de tareas en zonas de disponibilidad	El programador de servicio de Amazon ECS permite la ubicación de tareas en zonas de disponibilidad.	8 de octubre de 2015
Métricas de CloudWatch para clústeres y servicios de Amazon ECS	Amazon ECS ha agregado métricas de CloudWatc h personalizadas para el uso de CPU y de memoria para cada instancia de contenedor, servicio y familia de definiciones de tareas de un clúster. Estas métricas nuevas se pueden utilizar para escalar instancias de contenedor en un clúster mediante grupos de Auto Scaling o para crear alarmas de CloudWatch personali zadas.	17 de agosto de 2015

Cambio	Descripción	Fecha
Compatibilidad con puertos UDP	Se ha añadido compatibilidad con los puertos UDP en las definiciones de tareas.	7 de julio de 2015
Anulaciones de variables de entorno	Se ha añadido compatibilidad para reemplazos de deregisterTaskDefinition y de variables de entorno para runTask.	18 de junio de 2015
Actualizaciones automatizadas del agente de Amazon ECS	Se ha añadido la capacidad de ver la versión del agente de ECS que se está ejecutando en una instancia de contenedor. También es posible actualizar el agente de ECS desde la AWS Management Console, la AWS CLI y el SDK.	11 de junio de 2015
Integración del programador de servicios de Amazon ECS y Elastic Load Balancing	Se ha agregado la capacidad de definir un servicio y asociarlo a un balanceador de carga de Elastic Load Balancing.	9 de abril de 2015
GA de Amazon ECS	Disponibilidad general de Amazon ECS en las regiones de EE. UU. Este (Norte de Virginia), EE. UU. Oeste (Oregón), Asia-Pacífico (Tokio) y Europa (Irlanda).	9 de abril de 2015