Guía del usuario de

AWS App Mesh

AWS App Mesh: Guía del usuario de

Table of Contents

Qué es AWS App Mesh? 1
Adición de App Mesh a una aplicación de ejemplo 1
Componentes de App Mesh 3
Cómo comenzar 4
Acceso a App Mesh 4
Introducción 6
App Mesh y Amazon ECS 6
Escenario 7
Requisitos previos 7
Paso 1: Crear una malla y un servicio virtual 8
Paso 2: Crear un nodo virtual 9
Paso 3: Crear un enrutador virtual y una ruta 11
Paso 4: Revisar y crear 13
Paso 5: Crear recursos adicionales 14
Paso 6: Actualizar los servicios 19
Temas avanzados 35
App Mesh y Kubernetes 36
Requisitos previos 37
Paso 1: Instalar los componentes de integración 38
Paso 2: Implementar recursos de App Mesh 44
Paso 3: Crear o actualizar servicios 58
Paso 4: Limpiar 64
App Mesh y Amazon EC2 65
Escenario 7
Requisitos previos 7
Paso 1: Crear una malla y un servicio virtual 67
Paso 2: Crear un nodo virtual 68
Paso 3: Crear un enrutador virtual y una ruta 11
Paso 4: Revisar y crear 13
Paso 5: Crear recursos adicionales 14
Paso 6: Actualizar los servicios 19
Ejemplos de App Mesh 89
Conceptos 90
Mallas 90

Creación de una malla de servicios 91
Eliminación de una malla 94
Servicios virtuales 95
Creación de un servicio virtual 95
Eliminación de un servicio virtual 98
Puertas de enlace virtuales 100
Creación de una puerta de enlace virtual 101
Implementación de una puerta de enlace virtual 106
Eliminación de una puerta de enlace virtual 107
Rutas de puertas de enlace 108
Nodos virtuales 115
Creación de un nodo virtual 116
Eliminación de un nodo virtual 127
Enrutadores virtuales 129
Creación de un enrutador virtual 130
Eliminación de un enrutador virtual 132
Rutas 134
Envoy 146
Variantes de imagen de Envoy 146
Variables de configuración de Envoy 150
Variables obligatorias 151
Variables opcionales 151
Valores predeterminados de Envoy establecidos por App Mesh 159
Política de reintentos de ruta predeterminada 159
Interruptor predeterminado 161
Updating/migrating a Envoy 1.17 161
Secret Discovery Service con SPIRE 162
Cambios de las expresiones regulares 162
Referencias retrospectivas 165
Agente para Envoy 165
Observabilidad 168
Registro 168
Firelens y Cloudwatch 171
Métricas de Envoy 171
Métricas de la aplicación de ejemplo 174
Exportación de métricas 178

Rastreo 188
X-Ray 189
Jaeger 191
Datadog para el rastreo 155
Herramientas 193
CloudFormation 193
AWS CDK 193
Controlador de App Mesh para Kubernetes 194
Terraform 194
Trabajo con mallas compartidas 195
Otorgar permisos para compartir mallas 195
Otorgar permiso para compartir una malla 195
Otorgar permisos para una malla 196
Requisitos previos para compartir mallas 198
Servicios relacionados 198
Uso compartido de una malla 198
Dejar de compartir una malla compartida 199
Identificación de una malla compartida 200
Facturación y medición 200
Cuotas de instancias 201
Trabajo con otros servicios 202
Creación de recursos de App Mesh con AWS CloudFormation 202
App Mesh y CloudFormation plantillas 203
Obtenga más información sobre CloudFormation 203
App Mesh activado AWS Outposts 203
Requisitos previos 204
Limitaciones 204
Consideraciones sobre la conectividad de red 204
Creación de un proxy de App Mesh Envoy en un Outpost 204
Prácticas recomendadas 206
Instrumentación de todas las rutas con reintentos 206
Ajuste de la velocidad de implementación 207
Escalado horizontal antes de la reducción horizontal 208
Implementación de comprobaciones de estado de contenedores 208
Optimice la resolución de DNS 209
Protección de aplicaciones 210

seguridad de la capa de transporte (TLS) 211
Requisitos del certificado 212
Certificados de autenticación TLS 213
Cómo App Mesh configura Envoys para negociar TLS 215
Verificación del cifrado 217
Renovación de certificados 218
Configure las cargas de trabajo de Amazon ECS para usar la autenticación TLS con AWS App Mesh 218
Configure las cargas de trabajo de Kubernetes para usar la autenticación TLS con AWS App Mesh 219
Autenticación TLS mutua 220
Certificados de la autenticación TLS mutua 220
Configuración de puntos de conexión de malla 221
Migración de los servicios a la autenticación TLS mutua 222
Verificación de la autenticación TLS mutua 222
Tutoriales de la autenticación TLS mutua de App Mesh 223
Identity and Access Management 223
Público 224
Autenticación con identidades 225
Administración del acceso con políticas 226
¿Cómo AWS App Mesh funciona con IAM 228
Ejemplos de políticas basadas en identidades 232
AWS políticas gestionadas 237
Cómo utilizar roles vinculados a servicios 240
Autorización de proxy de Envoy 244
Resolución de problemas 250
CloudTrail registros 252
Eventos de gestión de App Mesh en CloudTrail 254
Ejemplos de eventos de App Mesh 254
Protección de datos 255
Cifrado de datos 256
Validación de conformidad 257
Seguridad de la infraestructura 257
Puntos de conexión de VPC de tipo interfaz (AWS PrivateLink) 258
Resiliencia 260
Recuperación ante desastres en AWS App Mesh 261

Configuración y análisis de vulnerabilidades 261
Resolución de problemas 262
Prácticas recomendadas 262
Habilitar la interfaz de administración del proxy de Envoy 263
Habilite la integración de Envoy DogStats D para reducir las métricas 263
Habilitación de registros de acceso 264
Habilitación del registro de depuración de Envoy en los entornos de preproducción 264
Monitorización de la conectividad de Envoy Proxy con el plano de control de App Mesh 265
Configuración 265
No se puede extraer la imagen del contenedor de Envoy 265
No se puede conectar con el servicio de administración de Envoy de App Mesh 266
Envoy se desconectó del servicio de administración de Envoy de App Mesh mostrando un texto de error 267
La comprobación de estado del contenedor de Envoy, la sonda de disponibilidad o la sonda de vivacidad producen errores 270
La comprobación de estado desde el equilibrador de carga hasta el punto de conexión de malla está fallando 270
La puerta de enlace virtual no acepta tráfico en los puertos 1024 o inferiores 271
Conectividad 272
No se puede resolver el nombre DNS de un servicio virtual 272
No se puede conectar a un backend de servicio virtual 273
No se puede conectar a un servicio externo 275
No se puede conectar a un servidor MySQL o SMTP 276
No se puede conectar a un servicio modelado como un nodo virtual TCP o enrutador virtual en App Mesh 277
La conectividad es correcta para un servicio que no figura como backend de servicio virtual de un nodo virtual 278
Algunas solicitudes producen un error con el código de estado HTTP 503 cuando un servicio virtual tiene un proveedor de nodos virtuales 278
No se puede conectar a un sistema de archivos de Amazon EFS 279
La conectividad funciona correctamente, pero la solicitud entrante no aparece en los registros, rastreos o métricas de acceso de Envoy 280
Establecer las variables de entorno HTTP_PROXY/HTTPS_PROXY a nivel de contenedor no funciona como se esperaba. 280
Se agotan los tiempos de espera de las solicitudes ascendentes incluso después de configurar el tiempo de espera de las rutas. 281

Envoy responde con una solicitud HTTP incorrecta. 282
No se ha podido configurar correctamente el tiempo de espera. 283
Escalado 283
La conectividad falla y las comprobaciones de estado de los contenedores fallan al escalar más de 50 réplicas para una puerta de enlace virtual node/virtual 283
Las solicitudes producen un error 503 cuando el backend de un servicio virtual se escala vertical u horizontalmente 284
El contenedor de Envoy se bloquea debido a un error de segmentación al aumentar la carga 284
El aumento de los recursos predeterminados no se refleja en los límites de servicio 285
La aplicación se bloquea debido a la gran cantidad de llamadas de comprobación de estado. 285
Observabilidad 286
No puedo ver los AWS X-Ray rastros de mis aplicaciones 286
No puedo ver las métricas de Envoy para mis aplicaciones en las CloudWatch métricas de Amazon 287
No se pueden configurar reglas de muestreo personalizadas para las AWS X-Ray trazas ... 288
Seguridad 289
No se puede conectar a un servicio virtual de backend con una política de cliente TLS 290
No se puede conectar a un servicio virtual de backend cuando la aplicación es el origen de TLS 291
No se puede asegurar que la conectividad entre los proxies de Envoy utilice TLS 292
Solución de problemas de TLS con el equilibrador de carga elástico 294
Kubernetes 295
Los recursos de App Mesh creados en Kubernetes no se encuentran en App Mesh 295
Las comprobaciones de disponibilidad y vivacidad de los pods tienen errores después de inyectar el sidecar de Envoy 296
Los pods no se registran o se dan de baja como instancias AWS Cloud Map 296
No es posible determinar dónde se ejecuta un pod para un recurso de App Mesh 297
No se puede determinar cómo qué recurso de App Mesh se está ejecutando un pod 298
Los enviados del cliente no pueden comunicarse con App Mesh Envoy Management Service si está deshabilitado IMDSv1 298
IRSA no funciona en el contenedor de aplicaciones cuando App Mesh está habilitada y Envoy está inyectado 299
Service Quotas 301
Historial de revisión 303

..... CCCX

Qué es AWS App Mesh?

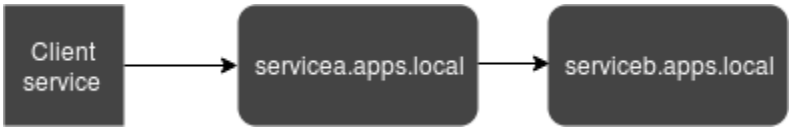
> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

AWS App Mesh es una malla de servicios que facilita la supervisión y el control de los servicios. Una malla de servicios es una capa de infraestructura dedicada a gestionar la comunicación entre servicios, normalmente a través de una serie de proxies de red ligeros implementados junto con el código de la aplicación. App Mesh estandariza cómo se comunican sus servicios, ofrece visibilidad integral y lo ayuda a garantizar la alta disponibilidad de las aplicaciones. App Mesh ofrece visibilidad y controles de tráfico de red coherentes para cada microservicio en una aplicación.

Adición de App Mesh a una aplicación de ejemplo

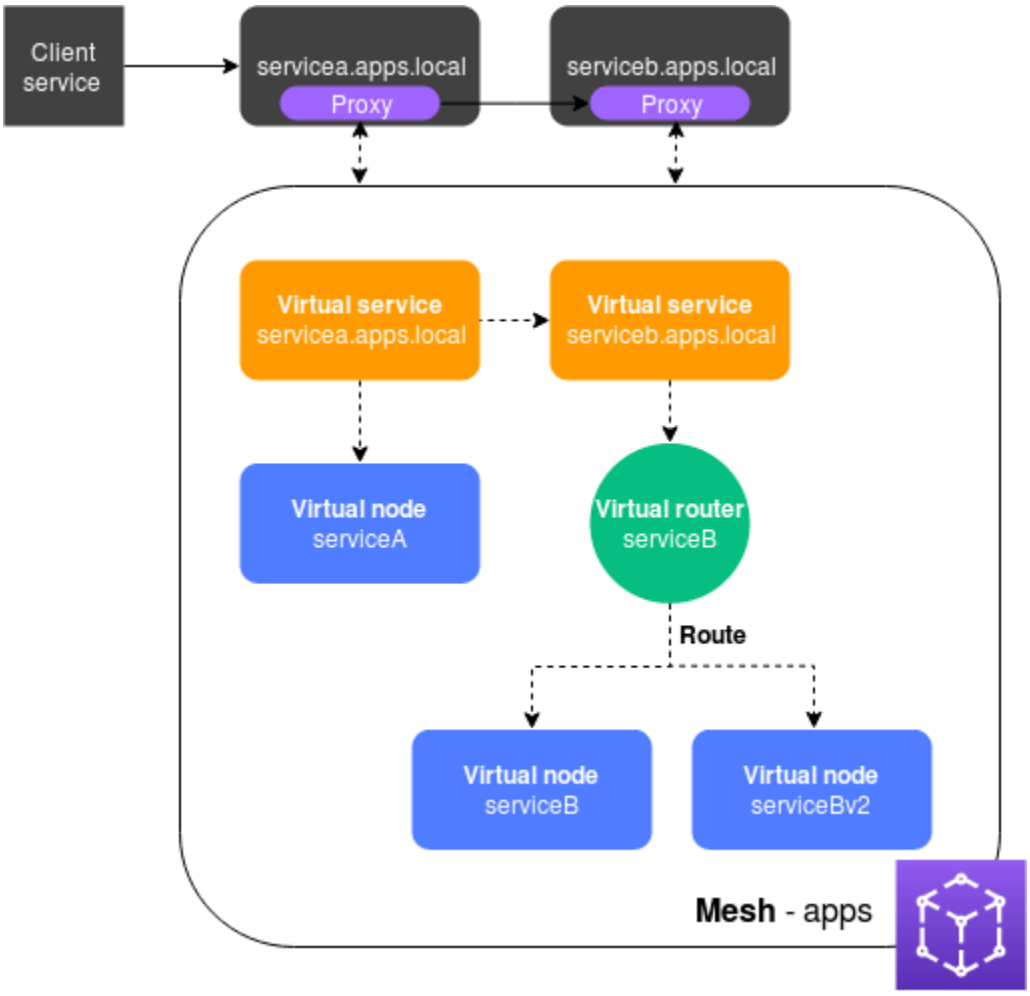
> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS se suspenderá el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Considere la siguiente aplicación de ejemplo simple que no usa App Mesh. Los dos servicios se pueden ejecutar en AWS Fargate instancias de Amazon Elastic Container Service (Amazon ECS), Amazon Elastic Kubernetes Service (Amazon EKS), Kubernetes en instancias de Amazon Elastic Compute Cloud (Amazon EC2) o en instancias de Amazon EC2 con Docker.



En esta ilustración, tanto `serviceA` y `serviceB` se pueden detectar a través del espacio de nombres `apps.local`. Supongamos, por ejemplo, que decide implementar una nueva versión de `serviceb.apps.local` denominada `servicebv2.apps.local`. A continuación, quiere dirigir un porcentaje del tráfico de `servicea.apps.local` a `serviceb.apps.local` y un porcentaje a `servicebv2.apps.local`. Cuando esté seguro de que `servicebv2` funciona bien, querrá enviarle el 100 % del tráfico.

App Mesh puede ayudarlo a hacerlo sin cambiar el código de la aplicación ni los nombres de servicio registrados. Si utiliza App Mesh con esta aplicación de ejemplo, la malla podría tener el aspecto que se muestra en la siguiente ilustración.



En esta configuración, los servicios ya no se comunican entre sí directamente. En cambio, se comunican entre sí a través de un proxy. El proxy implementado con el servicio `servicea.apps.local` lee la configuración de App Mesh y envía el tráfico a `serviceb.apps.local` o `servicebv2.apps.local` en función de la configuración.

Componentes de App Mesh

App Mesh consta de los siguientes componentes, como se mostró en el ejemplo anterior:

- Malla de servicios: es un límite lógico para el tráfico de red entre los servicios que residen dentro de ella. En el ejemplo, la malla se denomina `apps` y contiene todos los demás recursos de la malla. Para obtener más información, consulte Mallas de servicios.
- Servicios virtuales: un servicio virtual es una abstracción de un servicio real que se proporciona mediante un nodo virtual directa o indirectamente a través de un enrutador virtual. En la ilustración, dos servicios virtuales representan los dos servicios reales. Los nombres de los servicios virtuales son los nombres detectables de los servicios reales. Cuando un servicio virtual y un servicio real tienen el mismo nombre, varios servicios pueden comunicar entre sí con los mismos nombres que utilizaban antes de implementar App Mesh. Para obtener más información, consulte Servicios virtuales.
- Nodos virtuales: un nodo virtual actúa como un puntero lógico a un servicio detectable, por ejemplo, un servicio de Amazon ECS o de Kubernetes. Para cada servicio virtual, tendrá al menos un nodo virtual. En la ilustración, el servicio virtual `servicea.apps.local` obtiene la información de configuración del nodo virtual denominado `serviceA`. El nodo virtual `serviceA` está configurado con el nombre `servicea.apps.local` para la detección de servicios. El servicio virtual `serviceb.apps.local` está configurado para enrutar el tráfico a los nodos virtuales `serviceB` y `serviceBv2` a través de un enrutador virtual denominado `serviceB`. Para obtener más información, consulte Nodos virtuales.
- Enrutadores virtuales y rutas: los enrutadores virtuales gestionan el tráfico de uno o más servicios virtuales dentro de la malla. Una ruta está asociada a un enrutador virtual. La ruta se usa para hacer coincidir las solicitudes del enrutador virtual y distribuir el tráfico a sus nodos virtuales asociados. En la ilustración anterior, el enrutador virtual `serviceB` tiene una ruta que dirige un porcentaje del tráfico al nodo virtual `serviceB` y un porcentaje del tráfico al nodo virtual `serviceBv2`. Puede establecer el porcentaje de tráfico enrutado a un nodo virtual concreto y cambiarlo con el tiempo. Puede enrutar el tráfico en función de criterios como los encabezados HTTP, las rutas URL o los nombres de los métodos y servicios gRPC. Puede configurar políticas de reintentos para reintentar una conexión si se produce un error en la respuesta. Por ejemplo, en la ilustración, la política de reintentos de la ruta puede especificar que una conexión a `serviceb.apps.local` se reintente cinco veces, con diez segundos entre reintentos, si `serviceb.apps.local` devuelve tipos de errores específicos. Para obtener más información, consulte Enrutadores virtuales y Rutas.

- Proxy: los servicios se configuran para que usen el proxy después de crear la malla y sus recursos. El proxy lee la configuración de App Mesh y dirige el tráfico de forma adecuada. En la ilustración, todas las comunicaciones de `servicea.apps.local` a `serviceb.apps.local` pasan por el proxy implementado con cada servicio. Los servicios se comunican entre sí mediante los mismos nombres de detección de servicios que utilizaban antes de introducir App Mesh. Como el proxy lee la configuración de App Mesh, puede controlar cómo se comunican los dos servicios entre sí. Si desea cambiar la configuración de App Mesh, no necesita cambiar ni volver a implementar los propios servicios ni los proxies. Para obtener más información, consulte Imagen de Envoy.

Cómo comenzar

Para usar App Mesh, debe tener un servicio existente en ejecución en Amazon ECS AWS Fargate, Amazon EKS, Kubernetes en Amazon EC2 o Amazon EC2 con Docker.

Para empezar a usar App Mesh, consulte las siguientes guías:

- Introducción a App Mesh y Amazon ECS
- Introducción a App Mesh y Kubernetes
- Introducción a App Mesh y Amazon EC2

Acceso a App Mesh

Puede trabajar con App Mesh de las siguientes formas:

Consola de administración de AWS

La consola es una interfaz basada en navegador que puede utilizar para administrar sus recursos de App Mesh. Puede abrir la consola App Mesh en https://console.aws.amazon.com/appmesh/.

AWS CLI

Proporciona comandos para un amplio conjunto de AWS productos y es compatible con Windows, Mac y Linux. Para empezar, consulte la Guía del usuario de AWS Command Line Interface. Para obtener más información acerca de los comandos de App Mesh, consulte appmesh en la Referencia de los comandos de la AWS CLI.

AWS Tools for Windows PowerShell

Proporciona comandos para un amplio conjunto de AWS productos para quienes escriben en el PowerShell entorno. Para empezar, consulte la Herramientas de AWS para PowerShell Guía del usuario de . Para obtener más información sobre los cmdlets de App Mesh, consulte App Mesh en la Referencia de PowerShellcmdlets AWS Tools for.

AWS CloudFormation

Le permite crear una plantilla que describa todos los AWS recursos que desee. Con la plantilla, CloudFormation aprovisiona y configura los recursos por usted. Para empezar, consulte la Guía del usuario de AWS CloudFormation. Para obtener más información sobre los tipos de recursos de App Mesh, consulte la Referencia de tipos de recursos de App Mesh en la Referencia de plantillas de AWS CloudFormation.

AWS SDK

También proporcionamos SDK que le permiten tener acceso a App Mesh mediante una serie de lenguajes de programación. Los SDK se encargan automáticamente de tareas como las siguientes:

- Firmar criptográficamente sus solicitudes de servicio
- Reintentar solicitudes
- Tratar las respuestas a errores

Para obtener más información sobre los SDK disponibles, consulte Herramientas para Amazon Web Services.

Para obtener más información sobre las API de App Mesh, consulte la Referencia de la API de AWS App Mesh.

Introducción a App Mesh

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Puede usar App Mesh con aplicaciones que implemente en Amazon ECS, Kubernetes (que implemente en sus propias instancias de Amazon EC2 o que se ejecuten en Amazon EKS) y Amazon EC2. Para empezar a usar App Mesh, seleccione uno de los servicios en los que tenga implementadas las aplicaciones que desee usar con App Mesh. Siempre puede habilitar las aplicaciones en los demás servicios para que funcionen también con App Mesh después de leer una de las Guías de introducción.

Temas

- Introducción al AWS App Mesh y Amazon ECS
- Introducción al AWS App Mesh y Kubernetes
- Introducción al AWS App Mesh y Amazon EC2
- Ejemplos de App Mesh

Introducción al AWS App Mesh y Amazon ECS

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Este tema le ayuda a utilizarlo AWS App Mesh con un servicio real que se ejecuta en Amazon ECS. Este tutorial abarca las características básicas de varios tipos de recursos de App Mesh.

Escenario

Para ilustrar cómo usar App Mesh, suponga que tiene una aplicación con las siguientes características:

- Consta de dos servicios denominados `serviceA` y `serviceB`.
- Ambos servicios están registrados en un espacio de nombres denominado `apps.local`.
- `ServiceA`se comunica con `serviceB` más HTTP/2 del puerto 80.
- Ya ha implementado la versión 2 de `serviceB` y la ha registrado con el nombre `serviceBv2` en el espacio de nombres `apps.local`.

Tiene los siguientes requisitos:

- Desea enviar el 75 por ciento del tráfico desde `serviceA` `serviceB` y el 25 por ciento del tráfico hacia `serviceBv2` First. Si solo envías el 25 por ciento a`serviceBv2`, puedes validar que está libre de errores antes de enviar el 100 por ciento del tráfico desde allí`serviceA`.
- Desea poder ajustar fácilmente la proporción del tráfico para que el 100 % del tráfico vaya hacia `serviceBv2` una vez que se demuestre que es de confianza. Una vez que se envía todo el tráfico a `serviceBv2`, desea suspender `serviceB`.
- No quiere tener que cambiar ningún código de aplicación existente o registro de detección de servicios para que sus servicios reales cumplan los requisitos anteriores.

Para satisfacer sus requisitos, ha decidido crear una malla de servicios de App Mesh con servicios virtuales, nodos virtuales, un enrutador virtual y una ruta. Después de implementar la malla, actualiza los servicios para utilizar el proxy de Envoy. Una vez actualizados, sus servicios se comunican entre sí a través del proxy de Envoy en lugar de directamente.

Requisitos previos

Important

Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información,

visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

- Comprensión previa de los conceptos de App Mesh. Para obtener más información, consulte Qué es AWS App Mesh?.
- Comprensión previa de los conceptos de Amazon ECSs. Para obtener más información, consulte Qué es Amazon ECS en la Guía para desarrolladores de Amazon Elastic Container Service.
- App Mesh es compatible con los servicios de Linux que están registrados con DNS o con ambos. AWS Cloud Map Para utilizar esta guía de introducción, le recomendamos que tenga tres servicios existentes que estén registrados con DNS. En los procedimientos de este tema se presupone que los servicios existentes se llaman `serviceA`, `serviceB` y `serviceBv2`, y que todos los servicios son detectables a través de un espacio de nombres denominado `apps.local`.

  Puede crear una malla de servicios y sus recursos incluso aunque los servicios no existan, pero no puede usar la malla hasta que haya implementado servicios reales. Para obtener más información acerca de la detección de servicios en Amazon ECS, consulte Detección de servicios. Para crear un servicio de Amazon ECS con la detección de servicios, consulte Tutorial: Creación de un servicio mediante la detección de servicios. Si aún no tiene los servicios en ejecución, puede Crear un servicio de Amazon ECS mediante la detección de servicios.

Paso 1: Crear una malla y un servicio virtual

Una malla de servicios es un límite lógico para el tráfico de red entre los servicios que residen dentro de ella. Para obtener más información, consulte Mallas de servicios. Un servicio virtual es una abstracción de un servicio real. Para obtener más información, consulte Servicios virtuales.

Cree los siguientes recursos :

- Una malla denominada `apps`, ya que todos los servicios del escenario están registrados en el espacio de nombres `apps.local`.
- Un servicio virtual llamado `serviceb.apps.local`, ya que el servicio virtual representa un servicio que se puede detectar con ese nombre y no desea cambiar el código para hacer referencia a otro nombre. Un servicio virtual llamado `servicea.apps.local` se agrega en un paso posterior.

Puedes usar la AWS CLI versión 1.18.116 Consola de administración de AWS o superior o la 2.0.38 o superior para completar los siguientes pasos. Si utiliza el AWS CLI, utilice el `aws --version` comando para comprobar la versión instalada. AWS CLI Si no tiene instalada 1.18.116 o posterior o bien la 2.0.38 o posterior, debe instalar o actualizar la AWS CLI. Seleccione la pestaña de la herramienta que desea utilizar.

Consola de administración de AWS

1. Abre el asistente de primera ejecución de la consola App Mesh en https://console.aws.amazon.com/appmesh/get-started.
2. Para Nombre de malla, escriba **apps**.
3. En Nombre del servicio virtual, escriba **serviceb.apps.local**.
4. Para continuar, elija Siguiente.

AWS CLI

1. Cree una malla con el comando `create-mesh`.

```
aws appmesh create-mesh --mesh-name apps
```

2. Cree un servicio virtual con el comando `create-virtual-service`.

```
aws appmesh create-virtual-service --mesh-name apps --virtual-service-name
 serviceb.apps.local --spec {}
```

Paso 2: Crear un nodo virtual

Un nodo virtual actúa como un puntero lógico a un servicio real. Para obtener más información, consulte Nodos virtuales.

Cree un nodo virtual denominado `serviceB`, ya que uno de los nodos virtuales representa el servicio real denominado `serviceB`. El servicio real que representa el nodo virtual es detectable a través de DNS con un nombre de host de `serviceb.apps.local`. Como alternativa, puede descubrir los servicios reales utilizando. AWS Cloud Map El nodo virtual escuchará el tráfico mediante el HTTP/2 protocolo del puerto 80. También se admiten otros protocolos, así como comprobaciones de estado. Creará nodos virtuales para `serviceA` y `serviceBv2` en un paso posterior.

Consola de administración de AWS

1. En Nombre del nodo virtual, escriba **`serviceB`**.
2. En Método de detección de servicios, elija DNS y escriba **`serviceb.apps.local`** en Nombre de host DNS.
3. En Configuración del agente de escucha, elija http2 en Protocolo y escriba **`80`** en Puerto.
4. Para continuar, elija Siguiente.

AWS CLI

1. Cree un archivo denominado `create-virtual-node-serviceb.json` con el siguiente contenido:

```
{
    "meshName": "apps",
    "spec": {
        "listeners": [
            {
                "portMapping": {
                    "port": 80,
                    "protocol": "http2"
                }
            }
        ],
        "serviceDiscovery": {
            "dns": {
                "hostname": "serviceB.apps.local"
            }
        }
    },
    "virtualNodeName": "serviceB"
}
```

2. Cree el nodo virtual con el comando create-virtual-node utilizando el archivo JSON como entrada.

```
aws appmesh create-virtual-node --cli-input-json file://create-virtual-node-
serviceb.json
```

Paso 3: Crear un enrutador virtual y una ruta

Los routers virtuales enrutan el tráfico de uno o más servicios virtuales dentro de la malla. Para obtener más información, consulte Enrutadores virtuales y Rutas.

Cree los siguientes recursos :

- Un enrutador virtual llamado `serviceB`, ya que el servicio virtual `serviceB.apps.local` no inicia la comunicación saliente con ningún otro servicio. Recuerde que el servicio virtual que creó anteriormente es una abstracción de su servicio `serviceb.apps.local` real. El servicio virtual envía tráfico al router virtual. El router virtual escucha el tráfico mediante el HTTP/2 protocolo del puerto 80. También se admiten otros protocolos.
- Una ruta llamada `serviceB`. Enruta el cien por cien de su tráfico al nodo virtual `serviceB`. La ponderación se realiza en un paso posterior una vez que haya añadido el nodo virtual `serviceBv2`. Aunque no se incluye en esta guía, puede agregar criterios de filtro adicionales para la ruta y agregar una política de reintento para que el proxy de Envoy realice varios intentos de enviar tráfico a un nodo virtual cuando experimenta un problema de comunicación.

Consola de administración de AWS

1. En Nombre del enrutador virtual, escriba **`serviceB`**.
2. En Configuración del agente de escucha, elija http2 en Protocolo y especifique **`80`** en Puerto.
3. En Nombre de ruta, escriba **`serviceB`**.
4. En Tipo de ruta, elija http2.
5. En Nombre de nodo virtual, en Configuración de destino, seleccione `serviceB` y escriba **`100`** para Ponderación.
6. En Configuración de coincidencia, elija un Método.
7. Para continuar, elija Siguiente.

AWS CLI

1. Cree un router virtual.

   a. Cree un archivo denominado `create-virtual-router.json` con el siguiente contenido:

```
{
    "meshName": "apps",
    "spec": {
        "listeners": [
            {
                "portMapping": {
                    "port": 80,
                    "protocol": "http2"
                }
            }
        ]
    },
    "virtualRouterName": "serviceB"
}
```

b. Cree el router virtual con el comando create-virtual-router utilizando el archivo JSON como entrada.

```
aws appmesh create-virtual-router --cli-input-json file://create-virtual-router.json
```

2. Cree una ruta.

a. Cree un archivo denominado `create-route.json` con el siguiente contenido:

```
{
    "meshName" : "apps",
    "routeName" : "serviceB",
    "spec" : {
        "httpRoute" : {
            "action" : {
                "weightedTargets" : [
                    {
                        "virtualNode" : "serviceB",
                        "weight" : 100
                    }
                ]
            },
            "match" : {
                "prefix" : "/"
            }
        }
```

```
    },
    "virtualRouterName" : "serviceB"
}
```

b. Cree la ruta con el comando create-route utilizando el archivo JSON como entrada.

```
aws appmesh create-route --cli-input-json file://create-route.json
```

Paso 4: Revisar y crear

Revise la configuración comparándola con las instrucciones anteriores.

Consola de administración de AWS

Elija Editar si necesita realizar cambios en cualquier sección. Una vez esté satisfecho con la configuración, elija Crear malla.

La pantalla Estado muestra todos los recursos de malla que se han creado. Puede ver los recursos creados en la consola seleccionando Ver malla.

AWS CLI

Revise la configuración de la malla que creó con el comando describe-mesh.

```
aws appmesh describe-mesh --mesh-name apps
```

Revise la configuración del servicio virtual que creó con el comando describe-virtual-service.

```
aws appmesh describe-virtual-service --mesh-name apps --virtual-service-name
 serviceb.apps.local
```

Revise la configuración del nodo virtual que creó con el comando describe-virtual-node.

```
aws appmesh describe-virtual-node --mesh-name apps --virtual-node-name serviceB
```

Revise la configuración del router virtual que creó con el comando describe-virtual-router.

```
aws appmesh describe-virtual-router --mesh-name apps --virtual-router-name serviceB
```

Revise la configuración de la ruta que creó con el comando describe-route.

```
aws appmesh describe-route --mesh-name apps \
    --virtual-router-name serviceB  --route-name serviceB
```

Paso 5: Crear recursos adicionales

Para completar el escenario, debe:

- Crear un nodo virtual denominado `serviceBv2` y otro denominado `serviceA`. Ambos nodos virtuales escuchan las solicitudes a través del HTTP/2 puerto 80. Para el nodo virtual `serviceA`, configure un backend de `serviceb.apps.local`. Todo el tráfico saliente del nodo virtual `serviceA` se envía al servicio virtual denominado `serviceb.apps.local`. Aunque no se trata en esta guía, también puede especificar una ruta de acceso de archivo en la que escribir registros de acceso para un nodo virtual.
- Cree un servicio virtual adicional llamado `servicea.apps.local`, que envíe todo el tráfico directamente al nodo virtual `serviceA`.
- Actualizar la ruta de `serviceB` que creó en un paso anterior para enviar el 75 % de su tráfico al nodo virtual `serviceB` y el 25 % de su tráfico al nodo virtual `serviceBv2`. Con el tiempo, puede continuar modificando las proporciones hasta que `serviceBv2` reciba el 100 % del tráfico. Una vez que se envíe todo el tráfico hacia `serviceBv2`, puede cerrar y suspender el nodo virtual `serviceB` y el servicio real. Cuando cambia las ponderaciones, el código no requiere ninguna modificación, ya que los nombres del servicio real y virtual `serviceb.apps.local` no cambian. Recuerde que el servicio virtual `serviceb.apps.local` envía tráfico al router virtual, que enruta el tráfico a los nodos virtuales. Los nombres de detección de servicios para los nodos virtuales se pueden cambiar en cualquier momento.

Consola de administración de AWS

1. En el panel de navegación izquierdo, seleccione Mallas.
2. Seleccione la malla `apps` que creó en un paso anterior.
3. En el panel de navegación izquierdo, seleccione Nodos virtuales.
4. Elija Crear nodo virtual.
5. En Nombre del nodo virtual, escriba **`serviceBv2`**, en Método de detección de servicios, elija DNS, y en Nombre de host DNS escriba **`servicebv2.apps.local`**.

6. En Configuración del agente de escucha, seleccione http2 para Protocolo y escriba **`80`** para Puerto.
7. Elija Crear nodo virtual.
8. Elija de nuevo Crear nodo virtual. Escriba **`serviceA`** en el Nombre del nodo virtual. En Método de detección de servicios, elija DNS y en Nombre de host DNS, escriba **`servicea.apps.local`**.
9. En Escriba un nombre de servicio virtual, en Nuevo backend, escriba **`serviceb.apps.local`**.
10. En Configuración del agente de escucha, elija http2 para Protocolo, escriba **`80`** para Puerto y, a continuación, elija Crear nodo virtual.
11. En el panel de navegación izquierdo, seleccione Routers virtuales y, a continuación, seleccione el router virtual `serviceB` de la lista.
12. En Rutas, seleccione la ruta llamada `ServiceB` que creó en un paso anterior y elija Editar.
13. En Destinos, Nombre del nodo virtual, cambie el valor de Ponderación de `serviceB` a **`75`**.
14. Elija Agregar objetivo, elija `serviceBv2` en la lista desplegable y establezca el valor de Ponderación en **`25`**.
15. Seleccione Save.
16. En el panel de navegación izquierdo, seleccione Servicios virtuales y, a continuación, elija Crear servicio virtual.
17. Escriba **`servicea.apps.local`** en Nombre del servicio virtual, seleccione Nodo virtual en Proveedor, seleccione `serviceA` en Nodo virtual y, a continuación, elija Crear servicio virtual.

AWS CLI

1. Cree el nodo virtual `serviceBv2`.

   a. Cree un archivo denominado `create-virtual-node-servicebv2.json` con el siguiente contenido:

```
{
    "meshName": "apps",
    "spec": {
        "listeners": [
            {
```

```
                "portMapping": {
                    "port": 80,
                    "protocol": "http2"
                }
            }
        ],
        "serviceDiscovery": {
            "dns": {
                "hostname": "serviceBv2.apps.local"
            }
        }
    },
    "virtualNodeName": "serviceBv2"
}
```

b. Cree el nodo virtual.

```
aws appmesh create-virtual-node --cli-input-json file://create-virtual-node-servicebv2.json
```

2. Cree el nodo virtual `serviceA`.

a. Cree un archivo denominado `create-virtual-node-servicea.json` con el siguiente contenido:

```
{
   "meshName" : "apps",
   "spec" : {
      "backends" : [
         {
            "virtualService" : {
               "virtualServiceName" : "serviceb.apps.local"
            }
         }
      ],
      "listeners" : [
         {
            "portMapping" : {
               "port" : 80,
               "protocol" : "http2"
            }
         }
      ],
```

```
      "serviceDiscovery" : {
         "dns" : {
            "hostname" : "servicea.apps.local"
         }
      }
   },
   "virtualNodeName" : "serviceA"
}
```

b. Cree el nodo virtual.

```
aws appmesh create-virtual-node --cli-input-json file://create-virtual-node-servicea.json
```

3. Actualice el servicio virtual `serviceb.apps.local` que creó en un paso anterior para enviar su tráfico al router virtual `serviceB`. Cuando el servicio virtual se creó originalmente, no envió tráfico a ninguna parte, ya que el router virtual `serviceB` aún no se había creado.

   a. Cree un archivo denominado `update-virtual-service.json` con el siguiente contenido:

```
{
   "meshName" : "apps",
   "spec" : {
      "provider" : {
         "virtualRouter" : {
            "virtualRouterName" : "serviceB"
         }
      }
   },
   "virtualServiceName" : "serviceb.apps.local"
}
```

   b. Actualice el servicio virtual con el comando update-virtual-service.

```
aws appmesh update-virtual-service --cli-input-json file://update-virtual-service.json
```

4. Actualice la ruta `serviceB` que creó en un paso anterior.

   a. Cree un archivo denominado `update-route.json` con el siguiente contenido:

```
{
   "meshName" : "apps",
   "routeName" : "serviceB",
   "spec" : {
      "http2Route" : {
         "action" : {
            "weightedTargets" : [
               {
                  "virtualNode" : "serviceB",
                  "weight" : 75
               },
               {
                  "virtualNode" : "serviceBv2",
                  "weight" : 25
               }
            ]
         },
         "match" : {
            "prefix" : "/"
         }
      }
   },
   "virtualRouterName" : "serviceB"
}
```

b. Actualice la ruta con el comando update-route.

```
aws appmesh update-route --cli-input-json file://update-route.json
```

5. Cree el servicio virtual `serviceA`.

   a. Cree un archivo denominado `create-virtual-servicea.json` con el siguiente contenido:

```
{
   "meshName" : "apps",
   "spec" : {
      "provider" : {
         "virtualNode" : {
            "virtualNodeName" : "serviceA"
         }
      }
```

```
    },
    "virtualServiceName" : "servicea.apps.local"
}
```

b. Cree el servicio virtual.

```
aws appmesh create-virtual-service --cli-input-json file://create-virtual-servicea.json
```

Resumen de malla

Antes de crear la malla de servicio, tenía tres servicios reales denominados `servicea.apps.local`, `serviceb.apps.local` y `servicebv2.apps.local`. Además de los servicios reales, ahora tiene una malla de servicio que contiene los siguientes recursos que representan los servicios reales:

- Dos servicios virtuales. El proxy envía todo el tráfico desde el servicio virtual `servicea.apps.local` al servicio virtual `serviceb.apps.local` a través de un router virtual.
- Tres nodos virtuales denominados `serviceA`, `serviceB` y `serviceBv2`. El proxy de Envoy utiliza la información de detección de servicios configurada para los nodos virtuales para buscar las direcciones IP de los servicios reales.
- Un router virtual con una ruta que indica al proxy de Envoy que enrute el 75 % del tráfico entrante al nodo virtual `serviceB` y el 25 % del tráfico al nodo virtual `serviceBv2`.

Paso 6: Actualizar los servicios

Después de crear su malla, debe realizar las siguientes tareas:

- Autorice al proxy de Envoy que implementa con cada tarea de Amazon ECS a que lea la configuración de uno o más nodos virtuales. Para obtener más información acerca de cómo autorizar el proxy, consulte Autorización del proxy.
- Actualice cada una de sus definiciones de tareas de Amazon ECS para que utilicen el proxy de Envoy.

Credenciales

El contenedor Envoy requiere AWS Identity and Access Management credenciales para firmar las solicitudes que se envían al servicio App Mesh. Para las tareas de Amazon ECS implementadas con el tipo de lanzamiento de Amazon EC2, las credenciales pueden provenir del rol de instancia o de un rol de IAM de la tarea. Las tareas de Amazon ECS implementadas con Fargate en contenedores de Linux no tienen acceso al servidor de metadatos de Amazon EC2 que proporciona las credenciales del perfil de IAM de la instancia. Para proporcionar las credenciales, debe asociar un rol de tarea de IAM a cualquier tarea implementada con el tipo de contenedor Fargate en Linux.

Si una tarea se implementa con el tipo de lanzamiento de Amazon EC2 y se bloquea el acceso al servidor de metadatos de Amazon EC2, tal y como se describe en la anotación Importante en Rol de IAM para tareas, también debe asociarse a la tarea un rol de IAM de tarea. El rol que asigne a la instancia o tarea debe tener una política de IAM asociada como se describe en Autorización de proxy.

Para actualizar la definición de la tarea mediante el AWS CLI

Utiliza el AWS CLI comando Amazon ECS `register-task-definition`. El ejemplo de definición de tarea que aparece a continuación muestra cómo configurar App Mesh para su servicio.

> **Note**
>
> La configuración de App Mesh para Amazon ECS a través de la consola no está disponible.

json de definición de tarea

Configuración del proxy

Para configurar su servicio de Amazon ECS para que utilice App Mesh, la definición de tarea del servicio debe tener la siguiente sección de configuración de proxy. Establezca la configuración de proxy `type` en `APPMESH` y `containerName` en `envoy`. Establezca los siguientes valores de propiedad según corresponda.

`IgnoredUID`

El proxy de Envoy no enruta el tráfico de los procesos que utilizan este ID de usuario. Puede elegir cualquier identificador de usuario que desee para este valor de propiedad, pero este identificador debe ser el mismo que el ID de `user` del contenedor de Envoy en la definición de tarea. Este emparejamiento permite a Envoy pasar por alto su propio tráfico sin utilizar el proxy. Nuestros ejemplos utilizan *1337* con fines históricos.

`ProxyIngressPort`

Este es el puerto de entrada del contenedor de proxy de Envoy. Ajuste este valor en `15000`.

`ProxyEgressPort`

Este es el puerto de salida del contenedor de proxy de Envoy. Ajuste este valor en `15001`.

`AppPorts`

Especifique los puertos de entrada de escucha de los contenedores de aplicación. En este ejemplo, el contenedor de aplicación escucha en el puerto *`9080`*. El puerto que especifique debe coincidir con el puerto configurado en el agente de escucha de nodo virtual.

`EgressIgnoredIPs`

Envoy no utiliza un proxy para el tráfico en estas direcciones IP. Establezca este valor en `169.254.170.2,169.254.169.254`, que ignora el servidor de metadatos de Amazon EC2 y el punto de conexión de metadatos de la tarea de Amazon ECS. El punto de conexión de metadatos proporciona roles de IAM para las credenciales de tareas. Puede añadir direcciones adicionales.

`EgressIgnoredPorts`

Puede añadir una lista de puertos separados por comas. Envoy no utiliza un proxy para el tráfico en estas direcciones IP. El puerto 22 no se tiene en cuenta aunque no aparezca ningún puerto.

> **Note**
>
> El número máximo de puertos de salida que se pueden ignorar es 15.

```
"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"
   },
   {
    "name": "EgressIgnoredPorts",
    "value": "22"
   }
  ]
}
```

Dependencia de Envoy de contenedor de aplicación

Los contenedores de aplicación en las definiciones de tarea deben esperar a que arranque y comience el proxy de Envoy antes de poder comenzar. Para asegurarse de que esto ocurra, establezca una sección `dependsOn` en cada definición de contenedor de la aplicación para que espere a que el contenedor de Envoy notifique el estado `HEALTHY`. El siguiente código muestra un ejemplo de definición de contenedor de aplicación con esta dependencia. Todas las propiedades del siguiente ejemplo son necesarias. Algunos de los valores de las propiedades también son obligatorios, pero otros sí*replaceable*.

```
{
 "name": "appName",
 "image": "appImage",
 "portMappings": [{
  "containerPort": 9080,
  "hostPort": 9080,
  "protocol": "tcp"
 }],
 "essential": true,
 "dependsOn": [{
  "containerName": "envoy",
  "condition": "HEALTHY"
 }]
}
```

Definición de contenedor de Envoy

Las definiciones de tareas de Amazon ECS deben incluir una imagen del contenedor de App Mesh Envoy.

Todas las regiones compatibles se pueden *Region-code* sustituir por cualquier región que no sea `me-south-1` `ap-east-1``ap-southeast-3`,`eu-south-1`,`il-central-1`, `yaf-south-1`.

Standard

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

FIPS-compliant

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod-
fips
```

`me-south-1`

Standard

```
772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

`ap-east-1`

Standard

```
856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

`ap-southeast-3`

Standard

```
909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

`eu-south-1`

Standard

```
422531588944.dkr.ecr.eu-south-1.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

`il-central-1`

Standard

```
564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

`af-south-1`

Standard

```
924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

`Public repository`

Standard

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.34.13.1-prod
```

FIPS-compliant

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.34.13.1-prod-fips
```

Important

Solo se admite el uso de la versión v1.9.0.0-prod o posterior con App Mesh.

Debe usar la imagen del contenedor de App Mesh Envoy hasta que el equipo del proyecto de Envoy fusione los cambios compatibles con App Mesh. Para obtener más información, consulta el tema de la GitHub hoja de ruta.

Todas las propiedades del siguiente ejemplo son necesarias. Algunos de los valores de las propiedades también son obligatorios, pero otros sí. *`replaceable`*

Note

- La definición de contenedor de Envoy se debe marcar como `essential`.

- Recomendamos asignar unidades de CPU de `512` y al menos `64` MiB de memoria al contenedor de Envoy. En Fargate, el valor más bajo que podrá establecer son `1024` MiB de memoria.
- El nombre del nodo virtual del servicio de Amazon ECS debe establecerse en el valor de la propiedad `APPMESH_RESOURCE_ARN`. Esta propiedad requiere la versión `1.15.0` o posterior de la imagen de Envoy. Para obtener más información, consulte Envoy.
- El valor de la configuración `user` debe coincidir con el valor `IgnoredUID` de la configuración del proxy de definición de tarea. En este ejemplo, usaremos *`1337`*.
- La comprobación de estado que se muestra aquí espera a que el contenedor de Envoy arranque correctamente antes de notificar a Amazon ECS que el contenedor de Envoy está en buen estado y listo para que se inicien los contenedores de la aplicación.
- De forma predeterminada, App Mesh utiliza el nombre del recurso que se especificó en `APPMESH_RESOURCE_ARN` cuando Envoy hace referencia a sí mismo en métricas y registros de seguimiento. Puede anular este comportamiento estableciendo la variable de entorno `APPMESH_RESOURCE_CLUSTER` con su propio nombre. Esta propiedad requiere la versión `1.15.0` o posterior de la imagen de Envoy. Para obtener más información, consulte Envoy.

En el siguiente bloque se muestra un ejemplo de definición de contenedor de Envoy.

```
{
 "name": "envoy",
 "image": "840364872350.dkr.ecr.us-west-2.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-
prod",
 "essential": true,
 "environment": [{
  "name": "APPMESH_RESOURCE_ARN",
  "value": "arn:aws:appmesh:us-west-2:111122223333:mesh/apps/virtualNode/serviceB"
 }],
 "healthCheck": {
  "command": [
   "CMD-SHELL",
   "curl -s http://localhost:9901/server_info | grep state | grep -q LIVE"
  ],
  "startPeriod": 10,
  "interval": 5,
  "timeout": 2,
  "retries": 3
```

```
 },
 "user": "1337"
}
```

Ejemplos de definiciones de tarea

Las definiciones de tareas de Amazon ECS de ejemplo siguientes muestran cómo combinar los ejemplos de arriba en una definición de tarea para `taskB`. Se proporcionan ejemplos para crear tareas para ambos tipos de lanzamiento de Amazon ECS con o sin uso de AWS X-Ray. Cambie los *`replaceable`* valores, según proceda, para crear definiciones de tareas para las tareas nombradas `taskBv2` y `taskA` del escenario. Sustituya el nombre de malla y el nombre del nodo virtual por el valor `APPMESH_RESOURCE_ARN` y una lista de los puertos donde la aplicación escucha el valor `AppPorts` de la configuración de proxy. De forma predeterminada, App Mesh utiliza el nombre del recurso que se especificó en `APPMESH_RESOURCE_ARN` cuando Envoy hace referencia a sí mismo en métricas y registros de seguimiento. Puede anular este comportamiento estableciendo la variable de entorno `APPMESH_RESOURCE_CLUSTER` con su propio nombre. Todas las propiedades de los siguientes ejemplos son necesarias. Algunos de los valores de las propiedades también son obligatorios, pero otros sí*`replaceable`*.

Si está ejecutando una tarea de Amazon ECS como se describe en la sección Credenciales, debe agregar un rol de IAM de tarea existente a los ejemplos.

> **Important**
>
> Fargate debe usar un valor de puerto superior a 1024.

Example Definición de tareas de JSON para Amazon ECS: Fargate en contenedores de Linux

```
{

   "family" : "taskB",
   "memory" : "1024",
   "cpu" : "0.5 vCPU",
   "proxyConfiguration" : {
      "containerName" : "envoy",
      "properties" : [
         {
            "name" : "ProxyIngressPort",
            "value" : "15000"
         },
```

```
            {
               "name" : "AppPorts",
               "value" : "9080"
            },
            {
               "name" : "EgressIgnoredIPs",
               "value" : "169.254.170.2,169.254.169.254"
            },
            {
               "name": "EgressIgnoredPorts",
               "value": "22"
            },
            {
               "name" : "IgnoredUID",
               "value" : "1337"
            },
            {
               "name" : "ProxyEgressPort",
               "value" : "15001"
            }
         ],
         "type" : "APPMESH"
      },
      "containerDefinitions" : [
         {
            "name" : "appName",
            "image" : "appImage",
            "portMappings" : [
               {
                  "containerPort" : 9080,
                  "protocol" : "tcp"
               }
            ],
            "essential" : true,
            "dependsOn" : [
               {
                  "containerName" : "envoy",
                  "condition" : "HEALTHY"
               }
            ]
         },
         {
            "name" : "envoy",
```

```
            "image" : "840364872350.dkr.ecr.us-west-2.amazonaws.com/aws-appmesh-
envoy:v1.34.13.1-prod",
            "essential" : true,
            "environment" : [
               {
                  "name" : "APPMESH_VIRTUAL_NODE_NAME",
                  "value" : "mesh/apps/virtualNode/serviceB"
               }
            ],
            "healthCheck" : {
               "command" : [
                  "CMD-SHELL",
                  "curl -s http://localhost:9901/server_info | grep state | grep -q LIVE"
               ],
               "interval" : 5,
               "retries" : 3,
               "startPeriod" : 10,
               "timeout" : 2
            },
            "memory" : 500,
            "user" : "1337"
         }
      ],
      "requiresCompatibilities" : [ "FARGATE" ],
      "taskRoleArn" : "arn:aws:iam::123456789012:role/ecsTaskRole",
      "executionRoleArn" : "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
      "networkMode" : "awsvpc"
}
```

Example Definición de tareas de JSON para Amazon ECS con AWS X-Ray - Fargate en contenedores Linux

X-Ray permite recopilar datos sobre las solicitudes que atiende una aplicación y proporciona herramientas que puede utilizar para visualizar el flujo de tráfico. El uso del X-Ray controlador para Envoy permite a Envoy enviar información de rastreo a X-Ray. Puede habilitar el X-Ray rastreo mediante la configuración de Envoy. Según la configuración, Envoy envía los datos de rastreo al X-Ray daemon que funciona como contenedor lateral y el daemon reenvía los rastreos al servicio. X-Ray Una vez publicados los rastreos X-Ray, puede usar la X-Ray consola para visualizar el gráfico de llamadas de servicio y solicitar los detalles del rastreo. El siguiente JSON representa una definición de tarea para habilitar la integración de X-Ray.

```
{
```

```
"family" : "taskB",
"memory" : "1024",
"cpu" : "512",
"proxyConfiguration" : {
   "containerName" : "envoy",
   "properties" : [
      {
         "name" : "ProxyIngressPort",
         "value" : "15000"
      },
      {
         "name" : "AppPorts",
         "value" : "9080"
      },
      {
         "name" : "EgressIgnoredIPs",
         "value" : "169.254.170.2,169.254.169.254"
      },
      {
         "name": "EgressIgnoredPorts",
         "value": "22"
      },
      {
         "name" : "IgnoredUID",
         "value" : "1337"
      },
      {
         "name" : "ProxyEgressPort",
         "value" : "15001"
      }
   ],
   "type" : "APPMESH"
},
"containerDefinitions" : [
   {
      "name" : "appName",
      "image" : "appImage",
      "portMappings" : [
         {
            "containerPort" : 9080,
            "protocol" : "tcp"
         }
```

```
         ],
         "essential" : true,
         "dependsOn" : [
            {
               "containerName" : "envoy",
               "condition" : "HEALTHY"
            }
         ]
      },
      {

         "name" : "envoy",
         "image" : "840364872350.dkr.ecr.us-west-2.amazonaws.com/aws-appmesh-
envoy:v1.34.13.1-prod",
         "essential" : true,
         "environment" : [
            {
               "name" : "APPMESH_VIRTUAL_NODE_NAME",
               "value" : "mesh/apps/virtualNode/serviceB"
            },
            {
               "name": "ENABLE_ENVOY_XRAY_TRACING",
               "value": "1"
            }
         ],
         "healthCheck" : {
            "command" : [
               "CMD-SHELL",
               "curl -s http://localhost:9901/server_info | grep state | grep -q LIVE"
            ],
            "interval" : 5,
            "retries" : 3,
            "startPeriod" : 10,
            "timeout" : 2
         },
         "memory" : 500,
         "user" : "1337"
      },
      {
         "name" : "xray-daemon",
         "image" : "amazon/aws-xray-daemon",
         "user" : "1337",
         "essential" : true,
         "cpu" : "32",
```

```
         "memoryReservation" : "256",
         "portMappings" : [
            {
               "containerPort" : 2000,
               "protocol" : "udp"
            }
         ]
      }
   ],
   "requiresCompatibilities" : [ "FARGATE" ],
   "taskRoleArn" : "arn:aws:iam::123456789012:role/ecsTaskRole",
   "executionRoleArn" : "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
   "networkMode" : "awsvpc"
}
```

Example JSON para la definición de tarea de Amazon ECS: tipo de lanzamiento de EC2

```
{
  "family": "taskB",
  "memory": "256",
  "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"
```

```
      },
      {
        "name": "EgressIgnoredPorts",
        "value": "22"
      }
    ]
  },
  "containerDefinitions": [
    {
      "name": "appName",
      "image": "appImage",
      "portMappings": [
        {
          "containerPort": 9080,
          "hostPort": 9080,
          "protocol": "tcp"
        }
      ],
      "essential": true,
      "dependsOn": [
        {
          "containerName": "envoy",
          "condition": "HEALTHY"
        }
      ]
    },
    {
      "name": "envoy",
      "image": "840364872350.dkr.ecr.us-west-2.amazonaws.com/aws-appmesh-
envoy:v1.34.13.1-prod",
      "essential": true,
      "environment": [
        {
          "name": "APPMESH_VIRTUAL_NODE_NAME",
          "value": "mesh/apps/virtualNode/serviceB"
        }
      ],
      "healthCheck": {
        "command": [
          "CMD-SHELL",
          "curl -s http://localhost:9901/server_info | grep state | grep -q LIVE"
        ],
        "startPeriod": 10,
        "interval": 5,
```

```
        "timeout": 2,
        "retries": 3
      },
      "user": "1337"
    }
  ],
  "requiresCompatibilities" : [ "EC2" ],
  "taskRoleArn" : "arn:aws:iam::123456789012:role/ecsTaskRole",
  "executionRoleArn" : "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
  "networkMode": "awsvpc"
}
```

Example Definición de tareas de JSON para Amazon ECS con AWS X-Ray - Tipo de lanzamiento de EC2

```
{
  "family": "taskB",
  "memory": "256",
   "cpu" : "1024",
  "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"
      },
```

```
      {
        "name": "EgressIgnoredPorts",
        "value": "22"
      }
    ]
  },
  "containerDefinitions": [
    {
      "name": "appName",
      "image": "appImage",
      "portMappings": [
        {
          "containerPort": 9080,
          "hostPort": 9080,
          "protocol": "tcp"
        }
      ],
      "essential": true,
      "dependsOn": [
        {
          "containerName": "envoy",
          "condition": "HEALTHY"
        }
      ]
    },
    {
      "name": "envoy",
      "image": "840364872350.dkr.ecr.us-west-2.amazonaws.com/aws-appmesh-
envoy:v1.34.13.1-prod",
      "essential": true,
      "environment": [
        {
          "name": "APPMESH_VIRTUAL_NODE_NAME",
          "value": "mesh/apps/virtualNode/serviceB"
        },
        {
         "name": "ENABLE_ENVOY_XRAY_TRACING",
         "value": "1"
        }
      ],
      "healthCheck": {
        "command": [
          "CMD-SHELL",
          "curl -s http://localhost:9901/server_info | grep state | grep -q LIVE"
```

```
        ],
        "startPeriod": 10,
        "interval": 5,
        "timeout": 2,
        "retries": 3
      },
      "user": "1337"
    },
    {
      "name": "xray-daemon",
      "image": "amazon/aws-xray-daemon",
      "user": "1337",
      "essential": true,
      "cpu": 32,
      "memoryReservation": 256,
      "portMappings": [
        {
          "containerPort": 2000,
          "protocol": "udp"
        }
      ]
    }
  ],
  "requiresCompatibilities" : [ "EC2" ],
  "taskRoleArn" : "arn:aws:iam::123456789012:role/ecsTaskRole",
  "executionRoleArn" : "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
  "networkMode": "awsvpc"
}
```

Temas avanzados

Implementaciones de valores controlados mediante App Mesh

Las versiones e implementaciones de valores controlados ayudan a cambiar el tráfico entre una versión antigua de una aplicación y una versión recién implementada. También monitoriza el estado de la versión recién implementada. Si hay algún problema con la nueva versión, la implementación de valores controlados puede devolver automáticamente el tráfico a la versión anterior. Las implementaciones de valores controlados permiten cambiar el tráfico entre versiones de la aplicación con más control.

Para obtener más información sobre cómo aplicar las implementaciones de valores controlados para Amazon ECS mediante App Mesh, consulte Crear una canalización con las implementaciones de valores controlados para Amazon ECS mediante App Mesh

> **Note**
>
> Para ver más ejemplos y tutoriales de App Mesh, consulte el repositorio de ejemplos de App Mesh.

Introducción al AWS App Mesh y Kubernetes

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte a. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Cuando te integras AWS App Mesh con Kubernetes mediante el controlador App Mesh para Kubernetes, administras los recursos de App Mesh, como mallas, servicios virtuales, nodos virtuales, enrutadores virtuales y rutas a través de Kubernetes. También puede agregar automáticamente las imágenes del contenedor sidecar de App Mesh a las especificaciones del pod de Kubernetes. Este tutorial le guiará a través de la instalación del controlador de App Mesh para Kubernetes para que esa integración sea posible.

El controlador va acompañado de la implementación de las siguientes definiciones de recursos personalizados de Kubernetes: `meshes`, `virtual services`, `virtual nodes` y `virtual routers`. El controlador supervisa la creación, modificación y eliminación de los recursos personalizados y realiza cambios en los recursos de App Mesh correspondientes: the section called “Mallas” the section called “Servicios virtuales”, the section called “Nodos virtuales”, the section called “Puertas de enlace virtuales”, the section called “Rutas de puertas de enlace” y the section called “Enrutadores virtuales” (incluidas the section called “Rutas”) mediante la API de App Mesh. Para obtener más información o contribuir al controlador, consulta el proyecto. GitHub

El controlador también instala un webhook que inyecta los siguientes contenedores en los pods de Kubernetes etiquetados con el nombre que especifique.

- Proxy de App Mesh Envoy: Envoy utiliza la configuración definida en el plano de control de App Mesh para determinar dónde enviar el tráfico de la aplicación.
- Administrador de rutas del proxy de App Mesh: actualiza las reglas `iptables` del espacio de nombres de red de un pod que enrutan el tráfico entrante y saliente a través de Envoy. Este contenedor funciona como un contenedor init de Kubernetes dentro del pod.

Requisitos previos

- Comprensión previa de los conceptos de App Mesh. Para obtener más información, consulte Qué es AWS App Mesh?.
- Comprensión previa de los conceptos de Kubernetes. Para obtener más información, consulte Qué es Kubernetes en la documentación de Kubernetes.
- Un clúster de Kubernetes de existente. Si no dispone de un clúster existente, consulte Introducción a Amazon EKS en la Guía del usuario de Amazon EKS. Si ejecuta su propio clúster de Kubernetes en Amazon EC2, asegúrese de que Docker esté autenticado en el repositorio de Amazon ECR en el que se encuentra la imagen de Envoy. Para obtener más información, consulte la Imagen de Envoy, Autenticación del registro en la Guía del usuario de Amazon Elastic Container Registry y Extracción de una imagen de un registro privado en la documentación de Kubernetes.
- App Mesh es compatible con los servicios de Linux que están registrados con DNS o con ambos. AWS Cloud Map Para utilizar esta guía de introducción, le recomendamos que tenga tres servicios existentes que estén registrados con DNS. En los procedimientos de este tema se presupone que los servicios existentes se llaman `serviceA`, `serviceB` y `serviceBv2`, y que todos los servicios son detectables a través de un espacio de nombres denominado `apps.local`.

  Puede crear una malla de servicios y sus recursos incluso aunque los servicios no existan, pero no puede usar la malla hasta que haya implementado servicios reales.
- La AWS CLI versión 1.18.116 o posterior o la 2.0.38 o posterior instalada. Para instalar o actualizar el AWS CLI, consulte Instalación del. AWS CLI
- Un cliente `kubectl` configurado para comunicarse con el clúster de Kubernetes. Si utiliza Amazon Elastic Kubernetes Service, puede seguir las instrucciones para instalar `kubectl` y configurar un archivo `kubeconfig`.
- Tiene instalada la versión 3.0 o posterior. Si no tiene instalado Helm, consulte Uso de Helm con Amazon EKS en la Guía del usuario de Amazon EKS.

- Actualmente, Amazon EKS solo admite preferencias de IP solo `IPv4_ONLY` y solo `IPv6_ONLY`, ya que Amazon EKS actualmente solo admite pods capaces de atender solo el tráfico `IPv4` o solo el tráfico `IPv6`.

En los pasos restantes se presupone que los servicios reales se llaman `serviceA`, `serviceB` y `serviceBv2`, y que todos los servicios son detectables a través de un espacio de nombres denominado `apps.local`.

Paso 1: Instalar los componentes de integración

Instale los componentes de integración una vez en cada clúster que aloja los pods que desee utilizar con App Mesh.

Para instalar los componentes de integración

1. Para los pasos restantes de este procedimiento es necesario un clúster sin una versión preliminar del controlador instalada. Si ha instalado una versión preliminar o no está seguro de si dispone de ella, puede descargar y ejecutar un script que compruebe si hay una versión preliminar instalada en el clúster.

   ```
   curl -o pre_upgrade_check.sh https://raw.githubusercontent.com/aws/eks-charts/
   master/stable/appmesh-controller/upgrade/pre_upgrade_check.sh
   sh ./pre_upgrade_check.sh
   ```

   Si el script devuelve `Your cluster is ready for upgrade. Please proceed to the installation instructions`, puede continuar con el siguiente paso. Si se devuelve un mensaje diferente, deberá completar los pasos de actualización para continuar. Para obtener más información sobre la actualización de una versión preliminar, consulte Actualizar en GitHub.

2. Agregue el repositorio `eks-charts` a Helm.

   ```
   helm repo add eks https://aws.github.io/eks-charts
   ```

3. Instale las definiciones de recursos personalizados (CRD) de App Mesh Kubernetes.

   ```
   kubectl apply -k "https://github.com/aws/eks-charts/stable/appmesh-controller/crds?
   ref=master"
   ```

4. Cree un espacio de nombres de Kubernetes para el controlador.

```
kubectl create ns appmesh-system
```

5. Establezca las siguientes variables para usarlas en los pasos posteriores. Reemplace *cluster-name* y *Region-code* por los valores del clúster existente.

```
export CLUSTER_NAME=cluster-name
export AWS_REGION=Region-code
```

6. (Opcional) Si desea ejecutar el controlador en Fargate, tiene que crear un perfil de Fargate. Si no tiene instalado `eksctl`, consulte Instalación o actualización de `eksctl` en la Guía del usuario de Amazon EKS. Si prefiere crear el perfil mediante la consola, consulte Creación de un perfil de Fargate en la Guía del usuario de Amazon EKS.

```
eksctl create fargateprofile --cluster $CLUSTER_NAME --name appmesh-system --
namespace appmesh-system
```

7. Cree un proveedor de identidad de OpenID Connect (OIDC) para su clúster. Si no tiene instalado `eksctl`, puede instalarlo siguiendo las instrucciones de Instalación o actualización de `eksctl` en la Guía del usuario de Amazon EKS. Si prefiere crear el proveedor mediante la consola, consulte Habilitación de roles de IAM para cuentas de servicio en su clúster en la Guía del usuario de Amazon EKS.

```
eksctl utils associate-iam-oidc-provider \
    --region=$AWS_REGION \
    --cluster $CLUSTER_NAME \
    --approve
```

8. Cree un rol de IAM, asígnele las políticas AWSCloudMapFullAccess AWS gestionadas AWSAppMeshFullAccessy vincúlelo a la cuenta de servicio de `appmesh-controller` Kubernetes. El rol permite al controlador añadir, quitar y cambiar recursos de App Mesh.

> **Note**
>
> El comando crea un rol de AWS IAM con un nombre generado automáticamente. No puede especificar el nombre de rol de IAM que se crea.

```
eksctl create iamserviceaccount \
```

```
    --cluster $CLUSTER_NAME \
    --namespace appmesh-system \
    --name appmesh-controller \
    --attach-policy-arn  arn:aws:iam::aws:policy/
AWSCloudMapFullAccess,arn:aws:iam::aws:policy/AWSAppMeshFullAccess \
    --override-existing-serviceaccounts \
    --approve
```

Si prefiere crear la cuenta de servicio mediante Consola de administración de AWS o AWS CLI, consulte Creación de un rol y una política de IAM para su cuenta de servicio en la Guía del usuario de Amazon EKS. Si utiliza Consola de administración de AWS o AWS CLI para crear la cuenta, también debe asignar la función a una cuenta de servicio de Kubernetes. Para obtener más información, consulte Especificación de un rol de IAM para su cuenta de servicio en la Guía del usuario de Amazon EKS.

9. Implemente el controlador de App Mesh. Para obtener una lista de todas las opciones de configuración, consulte Configuración activado GitHub.

   1. Para implementar el controlador de App Mesh en un clúster privado, primero debe habilitar App Mesh y los puntos de conexión de Amazon VPC de detección de servicios en la subred privada enlazada. También debe establecer el `accountId`.

      ```
      --set accountId=$AWS_ACCOUNT_ID
      ```

      Para habilitar el X-Ray rastreo en un clúster privado, habilite los puntos de enlace Amazon VPC X-Ray y Amazon ECR. El controlador usa `public.ecr.aws/xray/aws-xray-daemon:latest` de forma predeterminada, así que coloque esta imagen en una ubicación local y colóquela en el repositorio de ECR personal.

      > **Note**
      >
      > Los puntos de conexión de Amazon VPC actualmente no admiten los repositorios públicos de Amazon ECR.

      En el siguiente ejemplo, se muestra la implementación del controlador con las configuraciones para. X-Ray

      ```
      helm upgrade -i appmesh-controller eks/appmesh-controller \
      ```

```
    --namespace appmesh-system \
    --set region=$AWS_REGION \
    --set serviceAccount.create=false \
    --set serviceAccount.name=appmesh-controller \
    --set accountId=$AWS_ACCOUNT_ID \
    --set log.level=debug \
    --set tracing.enabled=true \
    --set tracing.provider=x-ray \
    --set xray.image.repository=your-account-id.dkr.ecr.your-
region.amazonaws.com/your-repository \
    --set xray.image.tag=your-xray-daemon-image-tag
```

Compruebe si el X-Ray daemon se ha inyectado correctamente al vincular la implementación de la aplicación con el nodo virtual o la puerta de enlace.

Para obtener más información, consulte Clústeres privados en la Guía del usuario de Amazon EKS.

2. Implemente el controlador de App Mesh para otros clústeres. Para obtener una lista de todas las opciones de configuración, consulte Configuración en GitHub.

```
helm upgrade -i appmesh-controller eks/appmesh-controller \
    --namespace appmesh-system \
    --set region=$AWS_REGION \
    --set serviceAccount.create=false \
    --set serviceAccount.name=appmesh-controller
```

> **Note**
>
> Si su familia de clústeres de Amazon EKS es `IPv6`, defina el nombre del clúster al implementar el controlador App Mesh añadiendo la siguiente opción al comando anterior `--set clusterName=$CLUSTER_NAME`.

> **Important**
>
> Si el clúster está en las regiones `me-south-1`, `ap-east-1`, `ap-southeast-3`, `eu-south-1`, `il-central-1` o `af-south-1`, debe añadir las siguientes opciones al comando anterior:

Sustituya *account-id* y *Region-code* por uno de los conjuntos de valores adecuados.

- Para la imagen del sidecar:
  - ```
    --set image.repository=account-id.dkr.ecr.Region-code.amazonaws.com/amazon/appmesh-controller
    ```
  - 772975370895.dkr.ecr.me-south-1.amazonaws. com/aws-appmesh-envoy: v1.34.13.1-prod
  - 856666278305.dkr.ecr.ap-east-1.amazonaws. com/aws-appmesh-envoy: v1.34.13.1-prod
  - 909464085924.dkr.ecr.ap-southeast-3.amazonaws. com/aws-appmesh-envoy: v1.34.13.1-prod
  - 422531588944.dkr.ecr.eu-south-1.amazonaws. com/aws- appmesh-envoy: v1.34.13.1-prod
  - 564877687649.dkr.ecr.il-central-1.amazonaws. com/aws-appmesh-envoy: v1.34.13.1-prod
  - 924023996002.dkr.ecr.af-south-1.amazonaws. com/aws-appmesh-envoy: v1.34.13.1-prod
- Los URI de las imágenes más antiguas se encuentran en el registro de cambios. GitHub Las AWS cuentas en las que están presentes las imágenes han cambiado de versión`v1.5.0`. Las versiones anteriores de las imágenes se alojan en cuentas de AWS que se encuentran en los registros de imágenes de contenedores de Amazon de Amazon Elastic Kubernetes Service.
- Para la imagen del controlador:
  - ```
    --set sidecar.image.repository=account-id.dkr.ecr.Region-code.amazonaws.com/aws-appmesh-envoy
    ```
  - 772975370895.dkr.ecr.me-south-1.amazonaws. com/amazon/appmesh-controller: v1.13.1
  - 856666278305.dkr.ecr.ap-east-1.amazonaws. com/amazon/appmesh-controller: v1.13.1
  - 909464085924.dkr.ecr.ap-southeast-3.amazonaws. com/amazon/appmesh-controller: v1.13.1

- 422531588944.dkr.ecr.eu-south-1.amazonaws. com/amazon/appmesh-controller: v1.13.1
- 564877687649.dkr.ecr.il-central-1.amazonaws. com/amazon/appmesh-controller: v1.13.1
- 924023996002.dkr.ecr.af-south-1.amazonaws. com/amazon/appmesh-controller: v1.13.1

- Para la imagen de inicio del sidecar:
  - `--set sidecar.image.repository=account-id.dkr.ecr.Region-code.amazonaws.com/aws-appmesh-envoy`
  - 772975370895.dkr.ecr.me-south-1.amazonaws. com/aws- appmesh-proxy-route-manager: v7-prod
  - 856666278305.dkr.ecr.ap-east-1.amazonaws. com/aws- appmesh-proxy-route-manager: v7-prod
  - 909464085924.dkr.ecr.ap-southeast-3.amazonaws. com/aws- appmesh-proxy-route-manager: v7-prod
  - 422531588944.dkr.ecr.eu-south-1.amazonaws. com/aws- appmesh-proxy-route-manager: v7-prod
  - 564877687649.dkr.ecr.il-central-1.amazonaws. com/aws- appmesh-proxy-route-manager: v7-prod
  - 924023996002.dkr.ecr.af-south-1.amazonaws. com/aws- appmesh-proxy-route-manager: v7-prod

> **Important**
>
> Solo se admite el uso de la versión v1.9.0.0-prod o posterior con App Mesh.

10. Confirme que la versión del controlador es `v1.4.0` o posterior. Puede revisar el registro de cambios en el inicio de sesión. GitHub

```
kubectl get deployment appmesh-controller \
    -n appmesh-system \
    -o json  | jq -r ".spec.template.spec.containers[].image" | cut -f2 -d ':'
```

> **Note**
>
> Si ve el registro del contenedor en ejecución, puede ver una línea que incluye el siguiente texto, que puede omitirse de forma segura.
>
> ```
> Neither -kubeconfig nor -master was specified. Using the inClusterConfig.
>  This might not work.
> ```

Paso 2: Implementar recursos de App Mesh

Cuando implementa una aplicación en Kubernetes, también tiene que crear los recursos personalizados de Kubernetes para que el controlador pueda crear los recursos de App Mesh correspondientes. El siguiente procedimiento lo ayuda a implementar recursos de App Mesh con algunas de sus características. Puedes encontrar ejemplos de manifiestos para implementar otras funciones de recursos de App Mesh en las `v1beta2` subcarpetas de muchas de las carpetas de funciones que aparecen en los tutoriales de App Mesh en. GitHub

> **Important**
>
> Una vez que el controlador cree un recurso de App Mesh, recomendamos que solo realice cambios en el recurso de App Mesh (o que lo elimine) mediante el controlador. Si realiza cambios o elimina el recurso mediante App Mesh, el controlador no cambiará ni volverá a crear el recurso de App Mesh modificado o eliminado durante diez horas, de forma predeterminada. Puede configurar esta duración para que sea inferior. Para obtener más información, consulte Configuración en. GitHub

Para implementar recursos de App Mesh

1. Cree un espacio de nombres de Kubernetes en el que implementar recursos de App Mesh.

   a. Guarde el siguiente contenido en un archivo llamado `namespace.yaml` en el equipo.

   ```
   apiVersion: v1
   kind: Namespace
   metadata:
     name: my-apps
   ```

```
  labels:
    mesh: my-mesh
    appmesh.k8s.aws/sidecarInjectorWebhook: enabled
```

b. Cree el espacio de nombres.

```
kubectl apply -f namespace.yaml
```

2. Cree una malla de servicios de App Mesh.

a. Guarde el siguiente contenido en un archivo llamado `mesh.yaml` en el equipo. El archivo se utiliza para crear un recurso de malla denominado *my-mesh*. Una malla de servicios es un límite lógico para el tráfico de red entre los servicios que residen dentro de ella.

```
apiVersion: appmesh.k8s.aws/v1beta2
kind: Mesh
metadata:
  name: my-mesh
spec:
  namespaceSelector:
    matchLabels:
      mesh: my-mesh
```

b. Cree la malla.

```
kubectl apply -f mesh.yaml
```

c. Consulte los detalles del recurso de malla Kubernetes que se ha creado.

```
kubectl describe mesh my-mesh
```

Output

```
Name:         my-mesh
Namespace:
Labels:       <none>
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"appmesh.k8s.aws/
v1beta2","kind":"Mesh","metadata":{"annotations":{},"name":"my-mesh"},"spec":
{"namespaceSelector":{"matchLa...
API Version:  appmesh.k8s.aws/v1beta2
Kind:         Mesh
```

```
Metadata:
  Creation Timestamp:  2020-06-17T14:51:37Z
  Finalizers:
    finalizers.appmesh.k8s.aws/mesh-members
    finalizers.appmesh.k8s.aws/aws-appmesh-resources
  Generation:        1
  Resource Version:  6295
  Self Link:         /apis/appmesh.k8s.aws/v1beta2/meshes/my-mesh
  UID:               111a11b1-c11d-1e1f-gh1i-j11k1l111m711
Spec:
  Aws Name:  my-mesh
  Namespace Selector:
    Match Labels:
      Mesh:  my-mesh
Status:
  Conditions:
    Last Transition Time:  2020-06-17T14:51:37Z
    Status:                True
    Type:                  MeshActive
  Mesh ARN:                arn:aws:appmesh:us-west-2:111122223333:mesh/my-mesh
  Observed Generation:     1
Events:                    <none>
```

d. Consulte los detalles sobre la malla de servicios de App Mesh que creó el controlador.

```
aws appmesh describe-mesh --mesh-name my-mesh
```

Output

```
{
    "mesh": {
        "meshName": "my-mesh",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:111122223333:mesh/my-mesh",
            "createdAt": "2020-06-17T09:51:37.920000-05:00",
            "lastUpdatedAt": "2020-06-17T09:51:37.920000-05:00",
            "meshOwner": "111122223333",
            "resourceOwner": "111122223333",
            "uid": "111a11b1-c11d-1e1f-gh1i-j11k1l111m711",
            "version": 1
        },
        "spec": {},
        "status": {
```

```
            "status": "ACTIVE"
        }
    }
}
```

3. Cree un nodo virtual de App Mesh. Un nodo virtual actúa como un puntero lógico a una implementación de Kubernetes.

   a. Guarde el siguiente contenido en un archivo llamado `virtual-node.yaml` en el equipo. El archivo se utilizará para crear un nodo virtual de App Mesh denominado *`my-service-a`* en el espacio de nombres *`my-apps`*. El nodo virtual representa un servicio de Kubernetes que se crea en un paso posterior. El valor de `hostname` es el nombre de host de DNS completo del servicio real que representa este nodo virtual.

```
apiVersion: appmesh.k8s.aws/v1beta2
kind: VirtualNode
metadata:
  name: my-service-a
  namespace: my-apps
spec:
  podSelector:
    matchLabels:
      app: my-app-1
  listeners:
    - portMapping:
        port: 80
        protocol: http
  serviceDiscovery:
    dns:
      hostname: my-service-a.my-apps.svc.cluster.local
```

   Los nodos virtuales tienen capacidades, como el cifrado de extremo a extremo y las comprobaciones de estado, que no se explican en este tutorial. Para obtener más información, consulte the section called “Nodos virtuales”. Para ver todas las configuraciones disponibles para un nodo virtual que puede establecer en la especificación anterior, ejecute el siguiente comando.

```
aws appmesh create-virtual-node --generate-cli-skeleton yaml-input
```

   b. Implemente el nodo virtual.

```
kubectl apply -f virtual-node.yaml
```

c. Consulte los detalles del recurso del nodo virtual Kubernetes que se ha creado.

```
kubectl describe virtualnode my-service-a -n my-apps
```

Output

```
Name:         my-service-a
Namespace:    my-apps
Labels:       <none>
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"appmesh.k8s.aws/
v1beta2","kind":"VirtualNode","metadata":{"annotations":{},"name":"my-service-
a","namespace":"my-apps"},"s...
API Version:  appmesh.k8s.aws/v1beta2
Kind:         VirtualNode
Metadata:
  Creation Timestamp:  2020-06-17T14:57:29Z
  Finalizers:
    finalizers.appmesh.k8s.aws/aws-appmesh-resources
  Generation:        2
  Resource Version:  22545
  Self Link:         /apis/appmesh.k8s.aws/v1beta2/namespaces/my-apps/
virtualnodes/my-service-a
  UID:               111a11b1-c11d-1e1f-gh1i-j11k1l111m711
Spec:
  Aws Name:  my-service-a_my-apps
  Listeners:
    Port Mapping:
      Port:      80
      Protocol:  http
  Mesh Ref:
    Name:  my-mesh
    UID:   111a11b1-c11d-1e1f-gh1i-j11k1l111m711
  Pod Selector:
    Match Labels:
      App:  nginx
  Service Discovery:
    Dns:
      Hostname:  my-service-a.my-apps.svc.cluster.local
Status:
```

```
  Conditions:
    Last Transition Time:  2020-06-17T14:57:29Z
    Status:                True
    Type:                  VirtualNodeActive
  Observed Generation:     2
  Virtual Node ARN:        arn:aws:appmesh:us-west-2:111122223333:mesh/my-mesh/
virtualNode/my-service-a_my-apps
Events:                    <none>
```

d. Consulte los detalles del nodo virtual que el controlador creó en App Mesh.

> **Note**
>
> Aunque el nombre del nodo virtual creado en Kubernetes es *my-service-a*, el nombre del nodo virtual creado en App Mesh es *my-service-a_my-apps*. El controlador anexa el nombre del espacio de nombres de Kubernetes al nombre del nodo virtual de App Mesh cuando crea el recurso de App Mesh. El nombre del espacio de nombres se agrega porque en Kubernetes se pueden crear nodos virtuales con el mismo nombre en diferentes espacios de nombres, pero en App Mesh un nombre de nodo virtual debe ser único dentro de una malla.

```
aws appmesh describe-virtual-node --mesh-name my-mesh --virtual-node-name my-
service-a_my-apps
```

Output

```
{
    "virtualNode": {
        "meshName": "my-mesh",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:111122223333:mesh/my-mesh/
virtualNode/my-service-a_my-apps",
            "createdAt": "2020-06-17T09:57:29.840000-05:00",
            "lastUpdatedAt": "2020-06-17T09:57:29.840000-05:00",
            "meshOwner": "111122223333",
            "resourceOwner": "111122223333",
            "uid": "111a11b1-c11d-1e1f-gh1i-j11k1l111m711",
            "version": 1
        },
        "spec": {
```

```
            "backends": [],
            "listeners": [
                {
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
                    }
                }
            ],
            "serviceDiscovery": {
                "dns": {
                    "hostname": "my-service-a.my-apps.svc.cluster.local"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualNodeName": "my-service-a_my-apps"
    }
}
```

4. Cree un enrutador virtual de App Mesh. Los routers virtuales controlan el tráfico de uno o más servicios virtuales dentro de la malla.

   a. Guarde el siguiente contenido en un archivo llamado `virtual-router.yaml` en el equipo. El archivo se utiliza para crear un enrutador virtual para dirigir el tráfico al nodo virtual denominado `my-service-a` que se creó en el paso anterior. El controlador crea el enrutador virtual de App Mesh y los recursos de la ruta. Puede especificar muchas más capacidades para sus rutas y utilizar protocolos distintos de `http`. Para obtener más información, consulte the section called “Enrutadores virtuales” y the section called “Rutas”. Tenga en cuenta que el nombre del nodo virtual al que se hace referencia es el nombre del nodo virtual de Kubernetes, no el nombre del nodo virtual de App Mesh que ha creado el controlador en App Mesh.

```
apiVersion: appmesh.k8s.aws/v1beta2
kind: VirtualRouter
metadata:
  namespace: my-apps
  name: my-service-a-virtual-router
spec:
  listeners:
```

```
    - portMapping:
        port: 80
        protocol: http
  routes:
    - name: my-service-a-route
      httpRoute:
        match:
          prefix: /
        action:
          weightedTargets:
            - virtualNodeRef:
                name: my-service-a
              weight: 1
```

(Opcional) Para ver todas las configuraciones disponibles para un enrutador virtual que puede establecer en la especificación anterior, ejecute el siguiente comando.

```
aws appmesh create-virtual-router --generate-cli-skeleton yaml-input
```

Para ver todas las configuraciones disponibles para una ruta que puede establecer en la especificación anterior, ejecute el siguiente comando.

```
aws appmesh create-route --generate-cli-skeleton yaml-input
```

b. Implemente el enrutador virtual.

```
kubectl apply -f virtual-router.yaml
```

c. Consulte el recurso del enrutador virtual Kubernetes que se ha creado.

```
kubectl describe virtualrouter my-service-a-virtual-router -n my-apps
```

Salida abreviada

```
Name:         my-service-a-virtual-router
Namespace:    my-apps
Labels:       <none>
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"appmesh.k8s.aws/
v1beta2","kind":"VirtualRouter","metadata":{"annotations":{},"name":"my-
service-a-virtual-router","namespac...
```

```
API Version:  appmesh.k8s.aws/v1beta2
Kind:         VirtualRouter
...
Spec:
  Aws Name:  my-service-a-virtual-router_my-apps
  Listeners:
    Port Mapping:
      Port:      80
      Protocol:  http
  Mesh Ref:
    Name:  my-mesh
    UID:   111a11b1-c11d-1e1f-gh1i-j11k1l111m711
  Routes:
    Http Route:
      Action:
        Weighted Targets:
          Virtual Node Ref:
            Name:  my-service-a
          Weight:  1
      Match:
        Prefix:  /
    Name:        my-service-a-route
Status:
  Conditions:
    Last Transition Time:  2020-06-17T15:14:01Z
    Status:                True
    Type:                  VirtualRouterActive
  Observed Generation:     1
  Route AR Ns:
    My - Service - A - Route:  arn:aws:appmesh:us-west-2:111122223333:mesh/my-
mesh/virtualRouter/my-service-a-virtual-router_my-apps/route/my-service-a-route
  Virtual Router ARN:          arn:aws:appmesh:us-west-2:111122223333:mesh/my-
mesh/virtualRouter/my-service-a-virtual-router_my-apps
Events:                        <none>
```

d. Consulte el recurso del enrutador virtual que el controlador creó en App Mesh. Especifique `my-service-a-virtual-router_my-apps` para `name`, porque cuando el controlador creó el enrutador virtual en App Mesh, anexó el nombre del espacio de nombres Kubernetes al nombre del enrutador virtual.

```
aws appmesh describe-virtual-router --virtual-router-name my-service-a-virtual-
router_my-apps --mesh-name my-mesh
```

Output

```
{
    "virtualRouter": {
        "meshName": "my-mesh",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:111122223333:mesh/my-mesh/
virtualRouter/my-service-a-virtual-router_my-apps",
            "createdAt": "2020-06-17T10:14:01.547000-05:00",
            "lastUpdatedAt": "2020-06-17T10:14:01.547000-05:00",
            "meshOwner": "111122223333",
            "resourceOwner": "111122223333",
            "uid": "111a11b1-c11d-1e1f-gh1i-j11k1l111m711",
            "version": 1
        },
        "spec": {
            "listeners": [
                {
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
                    }
                }
            ]
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualRouterName": "my-service-a-virtual-router_my-apps"
    }
}
```

e. Consulte el recurso de la ruta que el controlador creó en App Mesh. No se creó un recurso de ruta en Kubernetes porque la ruta forma parte de la configuración del enrutador virtual en Kubernetes. La información de la ruta se muestra en los detalles del recurso Kubernetes en el subpaso c. El controlador no anexó el nombre del espacio de nombres de Kubernetes al nombre de la ruta de App Mesh al crear la ruta en App Mesh porque los nombres de ruta son exclusivos para un enrutador virtual.

```
aws appmesh describe-route \
    --route-name my-service-a-route \
```

```
    --virtual-router-name my-service-a-virtual-router_my-apps \
    --mesh-name my-mesh
```

Output

```
{
    "route": {
        "meshName": "my-mesh",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:111122223333:mesh/my-mesh/
virtualRouter/my-service-a-virtual-router_my-apps/route/my-service-a-route",
            "createdAt": "2020-06-17T10:14:01.577000-05:00",
            "lastUpdatedAt": "2020-06-17T10:14:01.577000-05:00",
            "meshOwner": "111122223333",
            "resourceOwner": "111122223333",
            "uid": "111a11b1-c11d-1e1f-gh1i-j11k1l111m711",
            "version": 1
        },
        "routeName": "my-service-a-route",
        "spec": {
            "httpRoute": {
                "action": {
                    "weightedTargets": [
                        {
                            "virtualNode": "my-service-a_my-apps",
                            "weight": 1
                        }
                    ]
                },
                "match": {
                    "prefix": "/"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualRouterName": "my-service-a-virtual-router_my-apps"
    }
}
```

5. Cree un servicio virtual de App Mesh. Un servicio virtual es una abstracción de un servicio real que se proporcionan mediante un nodo virtual directa o indirectamente a través de un router

virtual. Los servicios dependientes llaman al servicio virtual por su nombre. Aunque el nombre no es pertinente para App Mesh, se recomienda asignar al servicio virtual el nombre de dominio completo del servicio real al que hace referencia. Al nombrar sus servicios virtuales de esa manera, no tendrá que cambiar el código de aplicación para que haga referencia a un nombre distinto. Las solicitudes se dirigen al nodo virtual o enrutador virtual que se especifica como proveedor del servicio virtual.

a. Guarde el siguiente contenido en un archivo llamado `virtual-service.yaml` en el equipo. El archivo se utiliza para crear un servicio virtual que utilice un proveedor de enrutador virtual para dirigir el tráfico al nodo virtual denominado `my-service-a` que se creó en el paso anterior. El valor de `awsName` en `spec` es el nombre de dominio completo (FQDN) del servicio de Kubernetes real que abstrae este servicio virtual. El servicio Kubernetes se crea en the section called “Paso 3: Crear o actualizar servicios”. Para obtener más información, consulte the section called “Servicios virtuales”.

```
apiVersion: appmesh.k8s.aws/v1beta2
kind: VirtualService
metadata:
  name: my-service-a
  namespace: my-apps
spec:
  awsName: my-service-a.my-apps.svc.cluster.local
  provider:
    virtualRouter:
      virtualRouterRef:
        name: my-service-a-virtual-router
```

Para ver todas las configuraciones disponibles para un servicio virtual que puede establecer en la especificación anterior, ejecute el siguiente comando.

```
aws appmesh create-virtual-service --generate-cli-skeleton yaml-input
```

b. Cree el servicio virtual.

```
kubectl apply -f virtual-service.yaml
```

c. Consulte los detalles del recurso del servicio virtual Kubernetes que se ha creado.

```
kubectl describe virtualservice my-service-a -n my-apps
```

Output

```
Name:         my-service-a
Namespace:    my-apps
Labels:       <none>
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"appmesh.k8s.aws/
v1beta2","kind":"VirtualService","metadata":{"annotations":{},"name":"my-
service-a","namespace":"my-apps"}...
API Version:  appmesh.k8s.aws/v1beta2
Kind:         VirtualService
Metadata:
  Creation Timestamp:  2020-06-17T15:48:40Z
  Finalizers:
    finalizers.appmesh.k8s.aws/aws-appmesh-resources
  Generation:        1
  Resource Version:  13598
  Self Link:         /apis/appmesh.k8s.aws/v1beta2/namespaces/my-apps/
virtualservices/my-service-a
  UID:               111a11b1-c11d-1e1f-gh1i-j11k1l111m711
Spec:
  Aws Name:  my-service-a.my-apps.svc.cluster.local
  Mesh Ref:
    Name:  my-mesh
    UID:   111a11b1-c11d-1e1f-gh1i-j11k1l111m711
  Provider:
    Virtual Router:
      Virtual Router Ref:
        Name:  my-service-a-virtual-router
Status:
  Conditions:
    Last Transition Time:  2020-06-17T15:48:40Z
    Status:                True
    Type:                  VirtualServiceActive
  Observed Generation:     1
  Virtual Service ARN:     arn:aws:appmesh:us-west-2:111122223333:mesh/my-mesh/
virtualService/my-service-a.my-apps.svc.cluster.local
Events:                    <none>
```

d. Consulte los detalles del recurso del servicio virtual que el controlador creó en App Mesh. El controlador de Kubernetes no anexó el nombre del espacio de nombres de Kubernetes al

nombre del servicio virtual de App Mesh al crear el servicio virtual en App Mesh, porque el nombre del servicio virtual es un FQDN único.

```
aws appmesh describe-virtual-service --virtual-service-name my-service-a.my-apps.svc.cluster.local --mesh-name my-mesh
```

Output

```
{
    "virtualService": {
        "meshName": "my-mesh",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:111122223333:mesh/my-mesh/
virtualService/my-service-a.my-apps.svc.cluster.local",
            "createdAt": "2020-06-17T10:48:40.182000-05:00",
            "lastUpdatedAt": "2020-06-17T10:48:40.182000-05:00",
            "meshOwner": "111122223333",
            "resourceOwner": "111122223333",
            "uid": "111a11b1-c11d-1e1f-gh1i-j11k1l111m711",
            "version": 1
        },
        "spec": {
            "provider": {
                "virtualRouter": {
                    "virtualRouterName": "my-service-a-virtual-router_my-apps"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualServiceName": "my-service-a.my-apps.svc.cluster.local"
    }
}
```

Aunque no se trata en este tutorial, el controlador también puede implementar the section called “Puertas de enlace virtuales” y the section called “Rutas de puertas de enlace” de App Mesh. Para obtener información detallada sobre la implementación de estos recursos con el controlador, consulte Configuración de la puerta de enlace entrante o un ejemplo de manifiesto que incluya los recursos. GitHub

Paso 3: Crear o actualizar servicios

Todos los pods que desee usar con App Mesh deben tener los contenedores sidecar de App Mesh asociados a ellos. El inyector añade automáticamente los contenedores sidecar a cualquier pod implementado con una etiqueta que especifique.

1. Habilite la autorización de proxy. Recomendamos que habilite cada implementación de Kubernetes para transmitir solo la configuración de su propio nodo virtual de App Mesh.

   a. Guarde el siguiente contenido en un archivo llamado `proxy-auth.json` en el equipo. Asegúrese de reemplazarla por la *alternate-colored values* suya propia.

      JSON

      ```
      {
          "Version":"2012-10-17",
          "Statement": [
              {
                  "Effect": "Allow",
                  "Action": "appmesh:StreamAggregatedResources",
                  "Resource": [
                      "arn:aws:appmesh:us-east-1:111122223333:mesh/my-mesh/
      virtualNode/my-service-a_my-apps"
                  ]
              }
          ]
      }
      ```

   b. Creación de la política.

      ```
      aws iam create-policy --policy-name my-policy --policy-document file://proxy-
      auth.json
      ```

   c. Cree un rol de IAM, asocie la política que creó en el paso anterior, cree una cuenta de servicio de Kubernetes y enlace la política a la cuenta de servicio de Kubernetes. El rol permite al controlador añadir, quitar y cambiar recursos de App Mesh.

      ```
      eksctl create iamserviceaccount \
          --cluster $CLUSTER_NAME \
          --namespace my-apps \
          --name my-service-a \
      ```

```
    --attach-policy-arn  arn:aws:iam::111122223333:policy/my-policy \
    --override-existing-serviceaccounts \
    --approve
```

Si prefiere crear la cuenta de servicio mediante Consola de administración de AWS o AWS CLI, consulte Creación de un rol y una política de IAM para su cuenta de servicio en la Guía del usuario de Amazon EKS. Si utiliza Consola de administración de AWS o AWS CLI para crear la cuenta, también debe asignar la función a una cuenta de servicio de Kubernetes. Para obtener más información, consulte Especificación de un rol de IAM para su cuenta de servicio en la Guía del usuario de Amazon EKS.

2. (Opcional) Si desea realizar la implementación en pods de Fargate, debe crear un perfil de Fargate. Si no tiene instalado `eksctl`, puede instalarlo siguiendo las instrucciones de Instalación o actualización de `eksctl` en la Guía del usuario de Amazon EKS. Si prefiere crear el perfil mediante la consola, consulte Creación de un perfil de Fargate en la Guía del usuario de Amazon EKS.

```
eksctl create fargateprofile --cluster my-cluster --region Region-code --name my-service-a --namespace my-apps
```

3. Cree un servicio y una implementación de Kubernetes. Si tiene una implementación existente que desea usar con App Mesh, debe implementar un nodo virtual, como hizo en el paso secundario 3 del the section called “Paso 2: Implementar recursos de App Mesh”. Actualice la implementación para asegurarse de que su etiqueta coincida con la etiqueta que configuró en el nodo virtual, de modo que los contenedores sidecar se añadan automáticamente a los pods y estos se vuelvan a implementar.

   a. Guarde el siguiente contenido en un archivo llamado `example-service.yaml` en el equipo. Si cambia el nombre del espacio de nombres y utiliza pods de Fargate, asegúrese de que el nombre del espacio de nombres coincida con el que definió en su perfil de Fargate.

```
apiVersion: v1
kind: Service
metadata:
  name: my-service-a
  namespace: my-apps
  labels:
    app: my-app-1
spec:
```

```
  selector:
    app: my-app-1
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-service-a
  namespace: my-apps
  labels:
    app: my-app-1
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-app-1
  template:
    metadata:
      labels:
        app: my-app-1
    spec:
      serviceAccountName: my-service-a
      containers:
      - name: nginx
        image: nginx:1.19.0
        ports:
        - containerPort: 80
```

Important

El valor de app `matchLabels` `selector` en la especificación debe coincidir con el valor que especificó al crear el nodo virtual en el subpaso 3 de the section called "Paso 2: Implementar recursos de App Mesh". De lo contrario, los contenedores sidecar no se inyectarán en el pod. En el ejemplo anterior, el valor de la etiqueta es `my-app-1`. Si implementa una puerta de enlace virtual en lugar de un nodo virtual, el manifiesto `Deployment` solo debe incluir el contenedor de Envoy. Para obtener más información acerca de la imagen que debe utilizar, consulte Envoy. Para ver un ejemplo de manifiesto, consulta el ejemplo de implementación en. GitHub

b. Implemente el servicio.

```
kubectl apply -f example-service.yaml
```

c. Consulte el servicio y la implementación.

```
kubectl -n my-apps get pods
```

Output

```
NAME                                READY   STATUS    RESTARTS   AGE
my-service-a-54776556f6-2cxd9   2/2     Running   0          10s
my-service-a-54776556f6-w26kf   2/2     Running   0          18s
my-service-a-54776556f6-zw5kt   2/2     Running   0          26s
```

d. Consulte los detalles de uno de los pods que se ha implementado.

```
kubectl -n my-apps describe pod my-service-a-54776556f6-2cxd9
```

Salida abreviada

```
Name:         my-service-a-54776556f6-2cxd9
Namespace:    my-app-1
Priority:     0
Node:         ip-192-168-44-157.us-west-2.compute.internal/192.168.44.157
Start Time:   Wed, 17 Jun 2020 11:08:59 -0500
Labels:       app=nginx
              pod-template-hash=54776556f6
Annotations:  kubernetes.io/psp: eks.privileged
Status:       Running
IP:           192.168.57.134
IPs:
  IP:           192.168.57.134
Controlled By:  ReplicaSet/my-service-a-54776556f6
Init Containers:
  proxyinit:
    Container ID:   docker://
e0c4810d584c21ae0cb6e40f6119d2508f029094d0e01c9411c6cf2a32d77a59
    Image:          111345817488.dkr.ecr.us-west-2.amazonaws.com/aws-appmesh-
proxy-route-manager:v2
```

```
    Image ID:       docker-pullable://111345817488.dkr.ecr.us-
west-2.amazonaws.com/aws-appmesh-proxy-route-manager
    Port:           <none>
    Host Port:      <none>
    State:          Terminated
      Reason:       Completed
      Exit Code:    0
      Started:      Fri, 26 Jun 2020 08:36:22 -0500
      Finished:     Fri, 26 Jun 2020 08:36:22 -0500
    Ready:          True
    Restart Count:  0
    Requests:
      cpu:     10m
      memory:  32Mi
    Environment:
      APPMESH_START_ENABLED:         1
      APPMESH_IGNORE_UID:            1337
      APPMESH_ENVOY_INGRESS_PORT:    15000
      APPMESH_ENVOY_EGRESS_PORT:     15001
      APPMESH_APP_PORTS:             80
      APPMESH_EGRESS_IGNORED_IP:     169.254.169.254
      APPMESH_EGRESS_IGNORED_PORTS:  22
      AWS_ROLE_ARN:                  arn:aws:iam::111122223333:role/eksctl-app-
mesh-addon-iamserviceaccount-my-a-Role1-NMNCVWB6PL0N
      AWS_WEB_IDENTITY_TOKEN_FILE:   /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
    ...
Containers:
  nginx:
    Container ID:   docker://
be6359dc6ecd3f18a1c87df7b57c2093e1f9db17d5b3a77f22585ce3bcab137a
    Image:          nginx:1.19.0
    Image ID:       docker-pullable://nginx
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Fri, 26 Jun 2020 08:36:28 -0500
    Ready:          True
    Restart Count:  0
    Environment:
      AWS_ROLE_ARN:                 arn:aws:iam::111122223333:role/eksctl-app-
mesh-addon-iamserviceaccount-my-a-Role1-NMNCVWB6PL0N
      AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
```

```
    ...
  envoy:
    Container ID:
 docker://905b55cbf33ef3b3debc51cb448401d24e2e7c2dbfc6a9754a2c49dd55a216b6
    Image:          840364872350.dkr.ecr.us-west-2.amazonaws.com/aws-appmesh-
envoy:v1.12.4.0-prod
    Image ID:       docker-pullable://840364872350.dkr.ecr.us-
west-2.amazonaws.com/aws-appmesh-envoy
    Port:           9901/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Fri, 26 Jun 2020 08:36:36 -0500
    Ready:          True
    Restart Count:  0
    Requests:
      cpu:     10m
      memory:  32Mi
    Environment:
      APPMESH_RESOURCE_ARN:         arn:aws:iam::111122223333:mesh/my-mesh/
virtualNode/my-service-a_my-apps
      APPMESH_PREVIEW:              0
      ENVOY_LOG_LEVEL:              info
      AWS_REGION:                   us-west-2
      AWS_ROLE_ARN:                 arn:aws:iam::111122223333:role/eksctl-app-
mesh-addon-iamserviceaccount-my-a-Role1-NMNCVWB6PL0N
      AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
...
Events:
  Type    Reason     Age   From
   Message
  ----    ------     ----  ----
   -------
  Normal  Pulling    30s   kubelet, ip-192-168-44-157.us-
west-2.compute.internal  Pulling image "111345817488.dkr.ecr.us-
west-2.amazonaws.com/aws-appmesh-proxy-route-manager:v2"
  Normal  Pulled     23s   kubelet, ip-192-168-44-157.us-
west-2.compute.internal  Successfully pulled image "111345817488.dkr.ecr.us-
west-2.amazonaws.com/aws-appmesh-proxy-route-manager:v2"
  Normal  Created    21s   kubelet, ip-192-168-44-157.us-
west-2.compute.internal  Created container proxyinit
  Normal  Started    21s   kubelet, ip-192-168-44-157.us-
west-2.compute.internal  Started container proxyinit
```

```
  Normal  Pulling    20s   kubelet, ip-192-168-44-157.us-
west-2.compute.internal  Pulling image "nginx:1.19.0"
  Normal  Pulled     16s   kubelet, ip-192-168-44-157.us-
west-2.compute.internal  Successfully pulled image "nginx:1.19.0"
  Normal  Created    15s   kubelet, ip-192-168-44-157.us-
west-2.compute.internal  Created container nginx
  Normal  Started    15s   kubelet, ip-192-168-44-157.us-
west-2.compute.internal  Started container nginx
  Normal  Pulling    15s   kubelet, ip-192-168-44-157.us-
west-2.compute.internal  Pulling image "840364872350.dkr.ecr.us-
west-2.amazonaws.com/aws-appmesh-envoy:v1.12.4.0-prod"
  Normal  Pulled     8s    kubelet, ip-192-168-44-157.us-
west-2.compute.internal  Successfully pulled image "840364872350.dkr.ecr.us-
west-2.amazonaws.com/aws-appmesh-envoy:v1.12.4.0-prod"
  Normal  Created    7s    kubelet, ip-192-168-44-157.us-
west-2.compute.internal  Created container envoy
  Normal  Started    7s    kubelet, ip-192-168-44-157.us-
west-2.compute.internal  Started container envoy
```

En el resultado anterior, puede ver que el controlador ha añadido los contenedores `proxyinit` y `envoy` al pod. Si implementó el servicio de ejemplo en Fargate, el controlador agregó el contenedor `envoy` al pod, pero no el contenedor `proxyinit`.

4. (Opcional) Instale complementos como Prometheus, Grafana, Jaeger y AWS X-Ray Datadog. Para obtener más información, consulta los complementos de App Mesh GitHub y la sección Observabilidad de la Guía del usuario de App Mesh.

> **Note**
>
> Para ver más ejemplos y tutoriales de App Mesh, consulte el repositorio de ejemplos de App Mesh.

Paso 4: Limpiar

Elimine todos los recursos de ejemplo creados en este tutorial. El controlador también elimina los recursos creados en la malla de servicios de App Mesh `my-mesh`.

```
kubectl delete namespace my-apps
```

Si ha creado un perfil de Fargate para el servicio de ejemplo, elimínelo.

```
eksctl delete fargateprofile --name my-service-a --cluster my-cluster --region Region-code
```

Elimine la malla.

```
kubectl delete mesh my-mesh
```

(Opcional) Puede eliminar los componentes de integración de Kubernetes.

```
helm delete appmesh-controller -n appmesh-system
```

(Opcional) Si implementó los componentes de integración de Kubernetes en Fargate, elimine el perfil de Fargate.

```
eksctl delete fargateprofile --name appmesh-system --cluster my-cluster --
region Region-code
```

Introducción al AWS App Mesh y Amazon EC2

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Este tema le ayuda a utilizarlo AWS App Mesh con un servicio real que se ejecuta en Amazon EC2. Este tutorial abarca las características básicas de varios tipos de recursos de App Mesh.

Escenario

Para ilustrar cómo usar App Mesh, suponga que tiene una aplicación con las siguientes características:

- Consta de dos servicios denominados `serviceA` y `serviceB`.
- Ambos servicios están registrados en un espacio de nombres denominado `apps.local`.
- `ServiceA`se comunica con `serviceB` más HTTP/2 del puerto 80.
- Ya ha implementado la versión 2 de `serviceB` y la ha registrado con el nombre `serviceBv2` en el espacio de nombres `apps.local`.

Tiene los siguientes requisitos:

- Desea enviar el 75 por ciento del tráfico desde `serviceA` `serviceB` y el 25 por ciento del tráfico hacia `serviceBv2` First. Si solo envías el 25 por ciento a`serviceBv2`, puedes validar que está libre de errores antes de enviar el 100 por ciento del tráfico desde allí`serviceA`.
- Desea poder ajustar fácilmente la proporción del tráfico para que el 100 % del tráfico vaya hacia `serviceBv2` una vez que se demuestre que es de confianza. Una vez que se envía todo el tráfico a `serviceBv2`, desea suspender `serviceB`.
- No quiere tener que cambiar ningún código de aplicación existente o registro de detección de servicios para que sus servicios reales cumplan los requisitos anteriores.

Para satisfacer sus requisitos, ha decidido crear una malla de servicios de App Mesh con servicios virtuales, nodos virtuales, un enrutador virtual y una ruta. Después de implementar la malla, actualiza los servicios para utilizar el proxy de Envoy. Una vez actualizados, sus servicios se comunican entre sí a través del proxy de Envoy en lugar de directamente.

Requisitos previos

App Mesh es compatible con los servicios de Linux que están registrados con DNS o con ambos. AWS Cloud Map Para utilizar esta guía de introducción, le recomendamos que tenga tres servicios existentes que estén registrados con DNS. Puede crear una malla de servicios y sus recursos incluso aunque los servicios no existan, pero no puede usar la malla hasta que haya implementado servicios reales.

Si aún no tiene los servicios en ejecución, puede lanzar instancias de Amazon EC2 e implementar aplicaciones en ellas. Para obtener más información, consulte el tutorial: Introducción a las instancias Linux de Amazon EC2 en la Guía del usuario de Amazon EC2. En los pasos restantes se presupone que los servicios reales se llaman `serviceA`, `serviceB` y `serviceBv2`, y que todos los servicios son detectables a través de un espacio de nombres denominado `apps.local`.

Paso 1: Crear una malla y un servicio virtual

Una malla de servicios es un límite lógico para el tráfico de red entre los servicios que residen dentro de ella. Para obtener más información, consulte Mallas de servicios. Un servicio virtual es una abstracción de un servicio real. Para obtener más información, consulte Servicios virtuales.

Cree los siguientes recursos :

- Una malla denominada `apps`, ya que todos los servicios del escenario están registrados en el espacio de nombres `apps.local`.
- Un servicio virtual llamado `serviceb.apps.local`, ya que el servicio virtual representa un servicio que se puede detectar con ese nombre y no desea cambiar el código para hacer referencia a otro nombre. Un servicio virtual llamado `servicea.apps.local` se agrega en un paso posterior.

Puede utilizar la AWS CLI versión 1.18.116 Consola de administración de AWS o superior o la 2.0.38 o superior para completar los siguientes pasos. Si utiliza el AWS CLI, utilice el `aws --version` comando para comprobar la versión instalada. AWS CLI Si no tiene instalada 1.18.116 o posterior o bien la 2.0.38 o posterior, debe instalar o actualizar la AWS CLI. Seleccione la pestaña de la herramienta que desea utilizar.

Consola de administración de AWS

1. Abre el asistente de primera ejecución de la consola App Mesh en https://console.aws.amazon.com/appmesh/get-started.
2. Para Nombre de malla, escriba **`apps`**.
3. En Nombre del servicio virtual, escriba **`serviceb.apps.local`**.
4. Para continuar, elija Siguiente.

AWS CLI

1. Cree una malla con el comando `create-mesh`.

   ```
   aws appmesh create-mesh --mesh-name apps
   ```

2. Cree un servicio virtual con el comando `create-virtual-service`.

```
aws appmesh create-virtual-service --mesh-name apps --virtual-service-name
 serviceb.apps.local --spec {}
```

Paso 2: Crear un nodo virtual

Un nodo virtual actúa como un puntero lógico a un servicio real. Para obtener más información, consulte Nodos virtuales.

Cree un nodo virtual denominado `serviceB`, ya que uno de los nodos virtuales representa el servicio real denominado `serviceB`. El servicio real que representa el nodo virtual es detectable a través de DNS con un nombre de host de `serviceb.apps.local`. También puede detectar servicios reales mediante AWS Cloud Map. El nodo virtual escucha el tráfico mediante el HTTP/2 protocolo del puerto 80. También se admiten otros protocolos, así como comprobaciones de estado. Creará nodos virtuales para `serviceA` y `serviceBv2` en un paso posterior.

Consola de administración de AWS

1. En Nombre del nodo virtual, escriba **`serviceB`**.
2. En Método de detección de servicios, elija DNS y escriba **`serviceb.apps.local`** en Nombre de host DNS.
3. En Configuración del agente de escucha, elija http2 en Protocolo y escriba **`80`** en Puerto.
4. Para continuar, elija Siguiente.

AWS CLI

1. Cree un archivo denominado `create-virtual-node-serviceb.json` con el siguiente contenido:

```
{
    "meshName": "apps",
    "spec": {
        "listeners": [
            {
                "portMapping": {
                    "port": 80,
                    "protocol": "http2"
                }
```

```
            }
        ],
        "serviceDiscovery": {
            "dns": {
                "hostname": "serviceB.apps.local"
            }
        }
    },
    "virtualNodeName": "serviceB"
}
```

2. Cree el nodo virtual con el comando create-virtual-node utilizando el archivo JSON como entrada.

```
aws appmesh create-virtual-node --cli-input-json file://create-virtual-node-serviceb.json
```

Paso 3: Crear un enrutador virtual y una ruta

Los routers virtuales enrutan el tráfico de uno o más servicios virtuales dentro de la malla. Para obtener más información, consulte Enrutadores virtuales y Rutas.

Cree los siguientes recursos :

- Un enrutador virtual llamado `serviceB`, ya que el servicio virtual `serviceB.apps.local` no inicia la comunicación saliente con ningún otro servicio. Recuerde que el servicio virtual que creó anteriormente es una abstracción de su servicio `serviceb.apps.local` real. El servicio virtual envía tráfico al router virtual. El router virtual escucha el tráfico mediante el HTTP/2 protocolo del puerto 80. También se admiten otros protocolos.
- Una ruta llamada `serviceB`. Enruta el cien por cien de su tráfico al nodo virtual `serviceB`. La ponderación se realiza en un paso posterior una vez que haya añadido el nodo virtual `serviceBv2`. Aunque no se incluye en esta guía, puede agregar criterios de filtro adicionales para la ruta y agregar una política de reintento para que el proxy de Envoy realice varios intentos de enviar tráfico a un nodo virtual cuando experimenta un problema de comunicación.

Consola de administración de AWS

1. En Nombre del enrutador virtual, escriba **`serviceB`**.

2. En Configuración del agente de escucha, elija http2 en Protocolo y especifique **`80`** en Puerto.
3. En Nombre de ruta, escriba **`serviceB`**.
4. En Tipo de ruta, elija http2.
5. En Nombre de nodo virtual, en Configuración de destino, seleccione `serviceB` y escriba **`100`** para Ponderación.
6. En Configuración de coincidencia, elija un Método.
7. Para continuar, elija Siguiente.

AWS CLI

1. Cree un router virtual.

   a. Cree un archivo denominado `create-virtual-router.json` con el siguiente contenido:

   ```
   {
       "meshName": "apps",
       "spec": {
           "listeners": [
               {
                   "portMapping": {
                       "port": 80,
                       "protocol": "http2"
                   }
               }
           ]
       },
       "virtualRouterName": "serviceB"
   }
   ```

   b. Cree el router virtual con el comando create-virtual-router utilizando el archivo JSON como entrada.

   ```
   aws appmesh create-virtual-router --cli-input-json file://create-virtual-router.json
   ```

2. Cree una ruta.

   a. Cree un archivo denominado `create-route.json` con el siguiente contenido:

```
{
    "meshName" : "apps",
    "routeName" : "serviceB",
    "spec" : {
        "httpRoute" : {
            "action" : {
                "weightedTargets" : [
                    {
                        "virtualNode" : "serviceB",
                        "weight" : 100
                    }
                ]
            },
            "match" : {
                "prefix" : "/"
            }
        }
    },
    "virtualRouterName" : "serviceB"
}
```

b. Cree la ruta con el comando create-route utilizando el archivo JSON como entrada.

```
aws appmesh create-route --cli-input-json file://create-route.json
```

Paso 4: Revisar y crear

Revise la configuración comparándola con las instrucciones anteriores.

Consola de administración de AWS

Elija Editar si necesita realizar cambios en cualquier sección. Una vez esté satisfecho con la configuración, elija Crear malla.

La pantalla Estado muestra todos los recursos de malla que se han creado. Puede ver los recursos creados en la consola seleccionando Ver malla.

AWS CLI

Revise la configuración de la malla que creó con el comando describe-mesh.

```
aws appmesh describe-mesh --mesh-name apps
```

Revise la configuración del servicio virtual que creó con el comando describe-virtual-service.

```
aws appmesh describe-virtual-service --mesh-name apps --virtual-service-name
 serviceb.apps.local
```

Revise la configuración del nodo virtual que creó con el comando describe-virtual-node.

```
aws appmesh describe-virtual-node --mesh-name apps --virtual-node-name serviceB
```

Revise la configuración del router virtual que creó con el comando describe-virtual-router.

```
aws appmesh describe-virtual-router --mesh-name apps --virtual-router-name serviceB
```

Revise la configuración de la ruta que creó con el comando describe-route.

```
aws appmesh describe-route --mesh-name apps \
    --virtual-router-name serviceB  --route-name serviceB
```

Paso 5: Crear recursos adicionales

Para completar el escenario, debe:

- Crear un nodo virtual denominado `serviceBv2` y otro denominado `serviceA`. Ambos nodos virtuales escuchan las solicitudes a través del HTTP/2 puerto 80. Para el nodo virtual `serviceA`, configure un backend de `serviceb.apps.local`. Todo el tráfico saliente del nodo virtual `serviceA` se envía al servicio virtual denominado `serviceb.apps.local`. Aunque no se trata en esta guía, también puede especificar una ruta de acceso de archivo en la que escribir registros de acceso para un nodo virtual.
- Cree un servicio virtual adicional llamado `servicea.apps.local`, que envíe todo el tráfico directamente al nodo virtual `serviceA`.
- Actualizar la ruta de `serviceB` que creó en un paso anterior para enviar el 75 % de su tráfico al nodo virtual `serviceB` y el 25 % de su tráfico al nodo virtual `serviceBv2`. Con el tiempo, puede continuar modificando las proporciones hasta que `serviceBv2` reciba el 100 % del tráfico. Una vez que se envíe todo el tráfico hacia `serviceBv2`, puede cerrar y suspender el nodo virtual

`serviceB` y el servicio real. Cuando cambia las ponderaciones, el código no requiere ninguna modificación, ya que los nombres del servicio real y virtual `serviceb.apps.local` no cambian. Recuerde que el servicio virtual `serviceb.apps.local` envía tráfico al router virtual, que enruta el tráfico a los nodos virtuales. Los nombres de detección de servicios para los nodos virtuales se pueden cambiar en cualquier momento.

Consola de administración de AWS

1. En el panel de navegación izquierdo, seleccione Mallas.
2. Seleccione la malla `apps` que creó en un paso anterior.
3. En el panel de navegación izquierdo, seleccione Nodos virtuales.
4. Elija Crear nodo virtual.
5. En Nombre del nodo virtual, escriba **`serviceBv2`**, en Método de detección de servicios, elija DNS, y en Nombre de host DNS escriba **`servicebv2.apps.local`**.
6. En Configuración del agente de escucha, seleccione http2 para Protocolo y escriba **`80`** para Puerto.
7. Elija Crear nodo virtual.
8. Elija de nuevo Crear nodo virtual. Escriba **`serviceA`** en el Nombre del nodo virtual. En Método de detección de servicios, elija DNS y en Nombre de host DNS, escriba **`servicea.apps.local`**.
9. En Escriba un nombre de servicio virtual, en Nuevo backend, escriba **`serviceb.apps.local`**.
10. En Configuración del agente de escucha, elija http2 para Protocolo, escriba **`80`** para Puerto y, a continuación, elija Crear nodo virtual.
11. En el panel de navegación izquierdo, seleccione Routers virtuales y, a continuación, seleccione el router virtual `serviceB` de la lista.
12. En Rutas, seleccione la ruta llamada `ServiceB` que creó en un paso anterior y elija Editar.
13. En Destinos, Nombre del nodo virtual, cambie el valor de Ponderación de `serviceB` a **`75`**.
14. Elija Agregar objetivo, elija `serviceBv2` en la lista desplegable y establezca el valor de Ponderación en **`25`**.
15. Seleccione Save.
16. En el panel de navegación izquierdo, seleccione Servicios virtuales y, a continuación, elija Crear servicio virtual.

17. Escriba **`servicea.apps.local`** en Nombre del servicio virtual, seleccione Nodo virtual en Proveedor, seleccione `serviceA` en Nodo virtual y, a continuación, elija Crear servicio virtual.

AWS CLI

1. Cree el nodo virtual `serviceBv2`.

   a. Cree un archivo denominado `create-virtual-node-servicebv2.json` con el siguiente contenido:

```
{
    "meshName": "apps",
    "spec": {
        "listeners": [
            {
                "portMapping": {
                    "port": 80,
                    "protocol": "http2"
                }
            }
        ],
        "serviceDiscovery": {
            "dns": {
                "hostname": "serviceBv2.apps.local"
            }
        }
    },
    "virtualNodeName": "serviceBv2"
}
```

   b. Cree el nodo virtual.

```
aws appmesh create-virtual-node --cli-input-json file://create-virtual-node-servicebv2.json
```

2. Cree el nodo virtual `serviceA`.

   a. Cree un archivo denominado `create-virtual-node-servicea.json` con el siguiente contenido:

```
{
   "meshName" : "apps",
   "spec" : {
      "backends" : [
         {
            "virtualService" : {
               "virtualServiceName" : "serviceb.apps.local"
            }
         }
      ],
      "listeners" : [
         {
            "portMapping" : {
               "port" : 80,
               "protocol" : "http2"
            }
         }
      ],
      "serviceDiscovery" : {
         "dns" : {
            "hostname" : "servicea.apps.local"
         }
      }
   },
   "virtualNodeName" : "serviceA"
}
```

b. Cree el nodo virtual.

```
aws appmesh create-virtual-node --cli-input-json file://create-virtual-node-servicea.json
```

3. Actualice el servicio virtual `serviceb.apps.local` que creó en un paso anterior para enviar su tráfico al router virtual `serviceB`. Cuando el servicio virtual se creó originalmente, no envió tráfico a ninguna parte, ya que el router virtual `serviceB` aún no se había creado.

   a. Cree un archivo denominado `update-virtual-service.json` con el siguiente contenido:

```
{
   "meshName" : "apps",
   "spec" : {
```

```
      "provider" : {
         "virtualRouter" : {
            "virtualRouterName" : "serviceB"
         }
      }
   },
   "virtualServiceName" : "serviceb.apps.local"
}
```

b. Actualice el servicio virtual con el comando update-virtual-service.

```
aws appmesh update-virtual-service --cli-input-json file://update-virtual-service.json
```

4. Actualice la ruta `serviceB` que creó en un paso anterior.

a. Cree un archivo denominado `update-route.json` con el siguiente contenido:

```
{
   "meshName" : "apps",
   "routeName" : "serviceB",
   "spec" : {
      "http2Route" : {
         "action" : {
            "weightedTargets" : [
               {
                  "virtualNode" : "serviceB",
                  "weight" : 75
               },
               {
                  "virtualNode" : "serviceBv2",
                  "weight" : 25
               }
            ]
         },
         "match" : {
            "prefix" : "/"
         }
      }
   },
   "virtualRouterName" : "serviceB"
}
```

b. Actualice la ruta con el comando update-route.

```
aws appmesh update-route --cli-input-json file://update-route.json
```

5. Cree el servicio virtual `serviceA`.

a. Cree un archivo denominado `create-virtual-servicea.json` con el siguiente contenido:

```
{
   "meshName" : "apps",
   "spec" : {
      "provider" : {
         "virtualNode" : {
            "virtualNodeName" : "serviceA"
         }
      }
   },
   "virtualServiceName" : "servicea.apps.local"
}
```

b. Cree el servicio virtual.

```
aws appmesh create-virtual-service --cli-input-json file://create-virtual-
servicea.json
```

Resumen de malla

Antes de crear la malla de servicio, tenía tres servicios reales denominados `servicea.apps.local`, `serviceb.apps.local` y `servicebv2.apps.local`. Además de los servicios reales, ahora tiene una malla de servicio que contiene los siguientes recursos que representan los servicios reales:

- Dos servicios virtuales. El proxy envía todo el tráfico desde el servicio virtual `servicea.apps.local` al servicio virtual `serviceb.apps.local` a través de un router virtual.
- Tres nodos virtuales denominados `serviceA`, `serviceB` y `serviceBv2`. El proxy de Envoy utiliza la información de detección de servicios configurada para los nodos virtuales para buscar las direcciones IP de los servicios reales.

- Un router virtual con una ruta que indica al proxy de Envoy que enrute el 75 % del tráfico entrante al nodo virtual `serviceB` y el 25 % del tráfico al nodo virtual `serviceBv2`.

Paso 6: Actualizar los servicios

Después de crear su malla, debe realizar las siguientes tareas:

- Autorice al proxy de Envoy a que implemente con cada servicio para leer la configuración de uno o más nodos virtuales. Para obtener más información acerca de cómo autorizar el proxy, consulte Autorización de proxy de Envoy.
- Para actualizar el servicio existente, siga los pasos que se indican a continuación.

Para configurar una instancia de Amazon EC2 como miembro del nodo virtual

1. Crear un rol de IAM.

   a. Cree un archivo denominado `ec2-trust-relationship.json` con el siguiente contenido.

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

   b. Cree un rol de IAM con el siguiente comando.

```
aws iam create-role --role-name mesh-virtual-node-service-b --assume-role-
policy-document file://ec2-trust-relationship.json
```

2. Asocie políticas de IAM al rol que le permite leer de Amazon ECR y solo la configuración de un nodo virtual de App Mesh específico.

   a. Cree un archivo llamado `virtual-node-policy.json` con el siguiente contenido. `apps` es el nombre de la malla que creó en the section called “Paso 1: Crear una malla y un servicio virtual” y `serviceB` es el nombre del nodo virtual que creó en the section called “Paso 2: Crear un nodo virtual”. *111122223333*Sustitúyalo por tu ID *us-west-2* de cuenta y por la región en la que creaste la conexión.

      JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "appmesh:StreamAggregatedResources",
            "Resource": [
                "arn:aws:appmesh:us-west-2:111122223333:mesh/apps/
virtualNode/serviceB"
            ]
        }
    ]
}
```

   b. Cree la política mediante el siguiente comando.

```
aws iam create-policy --policy-name virtual-node-policy --policy-document
 file://virtual-node-policy.json
```

   c. Asocie la política que ha creado en el anterior paso al rol, para que este solo pueda leer la configuración del nodo virtual `serviceB` desde App Mesh.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::111122223333:policy/
virtual-node-policy --role-name mesh-virtual-node-service-b
```

   d. Asocie la política administrada `AmazonEC2ContainerRegistryReadOnly` al rol para que pueda extraer la imagen del contenedor de Envoy de Amazon ECR.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly --role-name mesh-virtual-node-service-b
```

3. Lance una instancia de Amazon EC2 con el rol de IAM que ha creado.
4. Conéctese a la instancia mediante SSH.
5. Instala Docker y el AWS CLI en tu instancia según la documentación del sistema operativo.
6. Autentíquese en el repositorio de Amazon ECR de Envoy en la región de la que desee que el cliente de Docker extraiga la imagen:

   - Todas las regiones excepto `me-south-1`, `ap-east-1`, `ap-southeast-3`, `eu-south-1`, `il-central-1` y `af-south-1`. Puedes *us-west-2* sustituirlo por cualquier región compatible`me-south-1`, exceptoap-east-1,ap-southeast-3, eu-south-1il-central-1, yaf-south-1.

     ```
     $aws ecr get-login-password \
         --region us-west-2 \
     | docker login \
         --username AWS \
         --password-stdin 840364872350.dkr.ecr.us-west-2.amazonaws.com
     ```

   - Región de `me-south-1`

     ```
     $aws ecr get-login-password \
         --region me-south-1 \
     | docker login \
         --username AWS \
         --password-stdin 772975370895.dkr.ecr.me-south-1.amazonaws.com
     ```

   - Región de `ap-east-1`

     ```
     $aws ecr get-login-password \
         --region ap-east-1 \
     | docker login \
         --username AWS \
         --password-stdin 856666278305.dkr.ecr.ap-east-1.amazonaws.com
     ```

7. Ejecute uno de los siguientes comandos para iniciar el contenedor de App Mesh Envoy en la instancia, en función de la región de la que desee extraer la imagen. *serviceB*Los valores *apps* y son los nombres de malla y nodo virtual definidos en el escenario. Esta información

indica al proxy qué configuración del nodo virtual debe leer en App Mesh. Para completar el escenario, también debe completar estos pasos para las instancias de Amazon EC2 que alojan los servicios representados por los nodos virtuales `serviceBv2` y `serviceA`. Para su propia aplicación, reemplace estos valores con los suyos propios.

- Todas las regiones excepto `me-south-1`, `ap-east-1`, `ap-southeast-3`, `eu-south-1`, `il-central-1` y `af-south-1`. Puede *`Region-code`* sustituirlo por cualquier región compatible`me-south-1`, excepto las `af-south-1` regiones `ap-east-1` `ap-southeast-3``eu-south-1`,`il-central-1`,, y. Puede reemplazar *`1337`* con cualquier valor entre `0` y `2147483647`.

```
sudo docker run --detach --env APPMESH_RESOURCE_ARN=mesh/apps/
virtualNode/serviceB  \
-u 1337 --network host 840364872350.dkr.ecr.region-code.amazonaws.com/aws-
appmesh-envoy:v1.34.13.1-prod
```

- Región `me-south-1`. Puede reemplazar *`1337`* con cualquier valor entre `0` y `2147483647`.

```
sudo docker run --detach --env APPMESH_RESOURCE_ARN=mesh/apps/
virtualNode/serviceB  \
-u 1337 --network host 772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-
envoy:v1.34.13.1-prod
```

- Región `ap-east-1`. Puede reemplazar *`1337`* con cualquier valor entre `0` y `2147483647`.

```
sudo docker run --detach --env APPMESH_RESOURCE_ARN=mesh/apps/
virtualNode/serviceB  \
-u 1337 --network host 856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-
envoy:v1.34.13.1-prod
```

> **Note**
>
> La propiedad `APPMESH_RESOURCE_ARN` requiere una versión `1.15.0` o posterior de la imagen de Envoy. Para obtener más información, consulte Envoy.

> **Important**
>
> Solo se admite el uso de la versión v1.9.0.0-prod o posterior con App Mesh.

8. Seleccione Show more a continuación. Cree un archivo llamado `envoy-networking.sh` en la instancia con el siguiente contenido. *8000*Sustitúyalo por el puerto que utiliza el código de la aplicación para el tráfico entrante. Puede cambiar el valor de `APPMESH_IGNORE_UID`, pero el valor debe ser el mismo que el que ha especificado en el paso anterior, por ejemplo `1337`. Puede añadir direcciones adicionales a `APPMESH_EGRESS_IGNORED_IP` si es necesario. No modifique ninguna otra línea.

```
#!/bin/bash -e

#
# Start of configurable options
#


#APPMESH_START_ENABLED="0"
APPMESH_IGNORE_UID="1337"
APPMESH_APP_PORTS="8000"
APPMESH_ENVOY_EGRESS_PORT="15001"
APPMESH_ENVOY_INGRESS_PORT="15000"
APPMESH_EGRESS_IGNORED_IP="169.254.169.254,169.254.170.2"

# Enable routing on the application start.
[ -z "$APPMESH_START_ENABLED" ] && APPMESH_START_ENABLED="0"

# Enable IPv6.
[ -z "$APPMESH_ENABLE_IPV6" ] && APPMESH_ENABLE_IPV6="0"

# Egress traffic from the processess owned by the following UID/GID will be
 ignored.
if [ -z "$APPMESH_IGNORE_UID" ] && [ -z "$APPMESH_IGNORE_GID" ]; then
    echo "Variables APPMESH_IGNORE_UID and/or APPMESH_IGNORE_GID must be set."
    echo "Envoy must run under those IDs to be able to properly route it's egress
 traffic."
    exit 1
fi
```

```
# Port numbers Application and Envoy are listening on.
if [ -z "$APPMESH_ENVOY_EGRESS_PORT" ]; then
    echo "APPMESH_ENVOY_EGRESS_PORT must be defined to forward traffic from the
 application to the proxy."
    exit 1
fi

# If an app port was specified, then we also need to enforce the proxies ingress
 port so we know where to forward traffic.
if [ ! -z "$APPMESH_APP_PORTS" ] && [ -z "$APPMESH_ENVOY_INGRESS_PORT" ]; then
    echo "APPMESH_ENVOY_INGRESS_PORT must be defined to forward traffic from the
 APPMESH_APP_PORTS to the proxy."
    exit 1
fi

# Comma separated list of ports for which egress traffic will be ignored, we always
 refuse to route SSH traffic.
if [ -z "$APPMESH_EGRESS_IGNORED_PORTS" ]; then
    APPMESH_EGRESS_IGNORED_PORTS="22"
else
    APPMESH_EGRESS_IGNORED_PORTS="$APPMESH_EGRESS_IGNORED_PORTS,22"
fi

#
# End of configurable options
#

function initialize() {
    echo "=== Initializing ==="
    if [ ! -z "$APPMESH_APP_PORTS" ]; then
        iptables -t nat -N APPMESH_INGRESS
        if [ "$APPMESH_ENABLE_IPV6" == "1" ]; then
            ip6tables -t nat -N APPMESH_INGRESS
        fi
    fi
    iptables -t nat -N APPMESH_EGRESS
    if [ "$APPMESH_ENABLE_IPV6" == "1" ]; then
        ip6tables -t nat -N APPMESH_EGRESS
    fi
}

function enable_egress_routing() {
    # Stuff to ignore
    [ ! -z "$APPMESH_IGNORE_UID" ] && \
```

```
        iptables -t nat -A APPMESH_EGRESS \
        -m owner --uid-owner $APPMESH_IGNORE_UID \
        -j RETURN

    [ ! -z "$APPMESH_IGNORE_GID" ] && \
        iptables -t nat -A APPMESH_EGRESS \
        -m owner --gid-owner $APPMESH_IGNORE_GID \
        -j RETURN

    [ ! -z "$APPMESH_EGRESS_IGNORED_PORTS" ] && \
        for IGNORED_PORT in $(echo "$APPMESH_EGRESS_IGNORED_PORTS" | tr "," "\n");
do
          iptables -t nat -A APPMESH_EGRESS \
          -p tcp \
          -m multiport --dports "$IGNORED_PORT" \
          -j RETURN
        done

    if [ "$APPMESH_ENABLE_IPV6" == "1" ]; then
      # Stuff to ignore ipv6
      [ ! -z "$APPMESH_IGNORE_UID" ] && \
          ip6tables -t nat -A APPMESH_EGRESS \
          -m owner --uid-owner $APPMESH_IGNORE_UID \
          -j RETURN

      [ ! -z "$APPMESH_IGNORE_GID" ] && \
          ip6tables -t nat -A APPMESH_EGRESS \
          -m owner --gid-owner $APPMESH_IGNORE_GID \
          -j RETURN

      [ ! -z "$APPMESH_EGRESS_IGNORED_PORTS" ] && \
        for IGNORED_PORT in $(echo "$APPMESH_EGRESS_IGNORED_PORTS" | tr "," "\n");
do
          ip6tables -t nat -A APPMESH_EGRESS \
          -p tcp \
          -m multiport --dports "$IGNORED_PORT" \
          -j RETURN
        done
    fi

    # The list can contain both IPv4 and IPv6 addresses. We will loop over this
list
    # to add every IPv4 address into `iptables` and every IPv6 address into
`ip6tables`.
```

```
    [ ! -z "$APPMESH_EGRESS_IGNORED_IP" ] && \
        for IP_ADDR in $(echo "$APPMESH_EGRESS_IGNORED_IP" | tr "," "\n"); do
            if [[ $IP_ADDR =~ .*:.* ]]
            then
                [ "$APPMESH_ENABLE_IPV6" == "1" ] && \
                    ip6tables -t nat -A APPMESH_EGRESS \
                        -p tcp \
                        -d "$IP_ADDR" \
                        -j RETURN
            else
                iptables -t nat -A APPMESH_EGRESS \
                    -p tcp \
                    -d "$IP_ADDR" \
                    -j RETURN
            fi
        done

    # Redirect everything that is not ignored
    iptables -t nat -A APPMESH_EGRESS \
        -p tcp \
        -j REDIRECT --to $APPMESH_ENVOY_EGRESS_PORT

    # Apply APPMESH_EGRESS chain to non local traffic
    iptables -t nat -A OUTPUT \
        -p tcp \
        -m addrtype ! --dst-type LOCAL \
        -j APPMESH_EGRESS

    if [ "$APPMESH_ENABLE_IPV6" == "1" ]; then
        # Redirect everything that is not ignored ipv6
        ip6tables -t nat -A APPMESH_EGRESS \
            -p tcp \
            -j REDIRECT --to $APPMESH_ENVOY_EGRESS_PORT
        # Apply APPMESH_EGRESS chain to non local traffic ipv6
        ip6tables -t nat -A OUTPUT \
            -p tcp \
            -m addrtype ! --dst-type LOCAL \
            -j APPMESH_EGRESS
    fi

}

function enable_ingress_redirect_routing() {
    # Route everything arriving at the application port to Envoy
```

```
    iptables -t nat -A APPMESH_INGRESS \
        -p tcp \
        -m multiport --dports "$APPMESH_APP_PORTS" \
        -j REDIRECT --to-port "$APPMESH_ENVOY_INGRESS_PORT"

    # Apply AppMesh ingress chain to everything non-local
    iptables -t nat -A PREROUTING \
        -p tcp \
        -m addrtype ! --src-type LOCAL \
        -j APPMESH_INGRESS

    if [ "$APPMESH_ENABLE_IPV6" == "1" ]; then
        # Route everything arriving at the application port to Envoy ipv6
        ip6tables -t nat -A APPMESH_INGRESS \
            -p tcp \
            -m multiport --dports "$APPMESH_APP_PORTS" \
            -j REDIRECT --to-port "$APPMESH_ENVOY_INGRESS_PORT"

        # Apply AppMesh ingress chain to everything non-local ipv6
        ip6tables -t nat -A PREROUTING \
            -p tcp \
            -m addrtype ! --src-type LOCAL \
            -j APPMESH_INGRESS
    fi
}

function enable_routing() {
    echo "=== Enabling routing ==="
    enable_egress_routing
    if [ ! -z "$APPMESH_APP_PORTS" ]; then
        enable_ingress_redirect_routing
    fi
}

function disable_routing() {
    echo "=== Disabling routing ==="
    iptables -t nat -F APPMESH_INGRESS
    iptables -t nat -F APPMESH_EGRESS

    if [ "$APPMESH_ENABLE_IPV6" == "1" ]; then
        ip6tables -t nat -F APPMESH_INGRESS
        ip6tables -t nat -F APPMESH_EGRESS
    fi
}
```

```
function dump_status() {
    echo "=== iptables FORWARD table ==="
    iptables -L -v -n
    echo "=== iptables NAT table ==="
    iptables -t nat -L -v -n

    if [ "$APPMESH_ENABLE_IPV6" == "1" ]; then
        echo "=== ip6tables FORWARD table ==="
        ip6tables -L -v -n
        echo "=== ip6tables NAT table ==="
        ip6tables -t nat -L -v -n
    fi
}

function clean_up() {
    disable_routing
    ruleNum=$(iptables -L PREROUTING -t nat --line-numbers | grep APPMESH_INGRESS |
 cut -d " " -f 1)
    iptables -t nat -D PREROUTING $ruleNum

    ruleNum=$(iptables -L OUTPUT -t nat --line-numbers | grep APPMESH_EGRESS | cut
 -d " " -f 1)
    iptables -t nat -D OUTPUT $ruleNum

    iptables -t nat -X APPMESH_INGRESS
    iptables -t nat -X APPMESH_EGRESS

    if [ "$APPMESH_ENABLE_IPV6" == "1" ]; then
        ruleNum=$(ip6tables -L PREROUTING -t nat --line-numbers | grep
 APPMESH_INGRESS | cut -d " " -f 1)
        ip6tables -t nat -D PREROUTING $ruleNum

        ruleNum=$(ip6tables -L OUTPUT -t nat --line-numbers | grep APPMESH_EGRESS |
 cut -d " " -f 1)
        ip6tables -t nat -D OUTPUT $ruleNum

        ip6tables -t nat -X APPMESH_INGRESS
        ip6tables -t nat -X APPMESH_EGRESS
    fi
}

function main_loop() {
    echo "=== Entering main loop ==="
```

```
    while read -p '> ' cmd; do
        case "$cmd" in
            "quit")
                clean_up
                break
                ;;
            "status")
                dump_status
                ;;
            "enable")
                enable_routing
                ;;
            "disable")
                disable_routing
                ;;
            *)
                echo "Available commands: quit, status, enable, disable"
                ;;
        esac
    done
}

function print_config() {
    echo "=== Input configuration ==="
    env | grep APPMESH_ || true
}

print_config

initialize

if [ "$APPMESH_START_ENABLED" == "1" ]; then
    enable_routing
fi

main_loop
```

9. Para configurar reglas de `iptables` para enrutar el tráfico de la aplicación al proxy Envoy, ejecute el script que ha creado en el paso anterior.

```
sudo ./envoy-networking.sh
```

10. Inicie el código de aplicación del nodo virtual.

> **Note**
>
> Para ver más ejemplos y tutoriales de App Mesh, consulte el repositorio de ejemplos de App Mesh.

Ejemplos de App Mesh

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

AWS App Mesh En el siguiente end-to-end repositorio encontrará tutoriales que muestran cómo actuar y ejemplos de código para la integración con varios AWS servicios:

Ejemplos de App Mesh

Conceptos de App Mesh

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

App Mesh se compone de los siguientes conceptos:

- Mallas de servicios
- Servicios virtuales
- Puertas de enlace virtuales
- Nodos virtuales
- Enrutadores virtuales

Mallas de servicios

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Una malla de servicios es un límite lógico para el tráfico de red entre los servicios que residen dentro de ella. Una vez que haya creado la malla de servicios, puede crear servicios virtuales, nodos virtuales, routers virtuales y rutas para distribuir el tráfico entre las aplicaciones en la malla.

Creación de una malla de servicios

> **Note**
>
> Al crear una malla, debe agregar un selector de espacios de nombres. Si el selector de espacios de nombres está vacío, selecciona todos los espacios de nombres. Para restringir los espacios de nombres, use una etiqueta para asociar los recursos de App Mesh a la malla creada.

Consola de administración de AWS

Para crear una malla de servicios mediante el Consola de administración de AWS

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija Crear malla.
3. En Nombre de la malla, especifique un nombre para la malla de servicios.
4. (Opcional) Seleccione Permitir el tráfico externo. De forma predeterminada, los proxies de la malla solo reenvían el tráfico entre sí. Si permite el tráfico externo, los proxies de la malla también reenvían el tráfico TCP directamente a los servicios que no se implementan con un proxy definido en la malla.

   > **Note**
   >
   > Si especifica algún backend en un nodo virtual al usar `ALLOW_ALL`, debe especificar todas las salidas de ese nodo virtual como backends. De lo contrario, `ALLOW_ALL` ya no funcionará para ese nodo virtual.

5. Preferencia de la versión de IP

   Controle qué versión de IP debe usarse para el tráfico dentro de la malla activando la opción Anulación del comportamiento predeterminado de la versión de IP. De forma predeterminada, App Mesh usa varias versiones de IP.

   > **Note**
   >
   > La malla aplica la preferencia de IP a todos los nodos virtuales y puertas de enlace virtuales dentro de una malla. Este comportamiento se puede anular en un

nodo virtual individual configurando la preferencia de IP al crear o editar el nodo. La preferencia IP no se puede anular en una puerta de enlace virtual porque la configuración de las puertas de enlace virtuales, que les permite escuchar tanto IPv4 el tráfico como el IPv6 tráfico, es la misma independientemente de la preferencia que se establezca en la malla.

- Predeterminado/a
  - El solucionador de DNS de Envoy prefiere `IPv6` y recurre a `IPv4`.
  - Usamos la dirección `IPv4` devuelta por AWS Cloud Map , si está disponible, y recurrimos a la dirección `IPv6`.
  - El punto de conexión creado para la aplicación local usa una dirección `IPv4`.
  - Los oyentes de Envoy se enlazan a todas las direcciones `IPv4`.
- IPv6 preferido
  - El solucionador de DNS de Envoy prefiere `IPv6` y recurre a `IPv4`.
  - Se utiliza la dirección `IPv6` devuelta por AWS Cloud Map , si está disponible, y se recurre a `IPv4`.
  - El punto de conexión creado para la aplicación local usa una dirección `IPv6`.
  - Los oyentes de Envoy se enlazan a todas las direcciones `IPv4` y `IPv6`.
- IPv4 preferido
  - El solucionador de DNS de Envoy prefiere `IPv4` y recurre a `IPv6`.
  - Usamos la dirección `IPv4` devuelta por AWS Cloud Map , si está disponible, y recurrimos a la dirección `IPv6`.
  - El punto de conexión creado para la aplicación local usa una dirección `IPv4`.
  - Los oyentes de Envoy se enlazan a todas las direcciones `IPv4` y `IPv6`.
- IPv6 únicamente
  - El solucionador de DNS de Envoy solo usa `IPv6`.
  - Solo se utiliza la dirección `IPv6` devuelta por AWS Cloud Map . Si AWS Cloud Map devuelve una `IPv4` dirección, no se utiliza ninguna dirección IP y los resultados vacíos se devuelven al Envoy.
  - El punto de conexión creado para la aplicación local usa una dirección `IPv6`.
  - Los oyentes de Envoy se enlazan a todas las direcciones `IPv4` y `IPv6`.

- IPv4 únicamente
  - El solucionador de DNS de Envoy solo usa `IPv4`.
  - Solo se utiliza la dirección `IPv4` devuelta por AWS Cloud Map . Si AWS Cloud Map devuelve una `IPv6` dirección, no se utiliza ninguna dirección IP y los resultados vacíos se devuelven al Envoy.
  - El punto de conexión creado para la aplicación local usa una dirección `IPv4`.
  - Los oyentes de Envoy se enlazan a todas las direcciones `IPv4` y `IPv6`.

6. Elija Crear malla para finalizar.
7. (Opcional) Comparta la malla con otras cuentas. Una malla compartida permite que los recursos creados por cuentas diferentes se comuniquen entre sí en la misma malla. Para obtener más información, consulte Trabajo con mallas compartidas.

AWS CLI

Para crear una malla mediante la AWS CLI.

Cree una malla de servicios con el siguiente comando (sustituya los *red* valores por los suyos propios):

1. ```
   aws appmesh create-mesh --mesh-name meshName
   ```
2. Ejemplo de salida:

```
{
    "mesh":{
        "meshName":"meshName",
        "metadata":{
            "arn":"arn:aws:appmesh:us-west-2:123456789012:mesh/meshName",
            "createdAt":"2022-04-06T08:45:50.072000-05:00",
            "lastUpdatedAt":"2022-04-06T08:45:50.072000-05:00",
            "meshOwner": "123456789012",
            "resourceOwner": "123456789012",
            "uid":"a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version":1
        },
        "spec":{},
        "status":{
            "status":"ACTIVE"
        }
```
```

```
    }
}
```

Para obtener más información sobre cómo crear una malla con App Mesh, consulta el comando create-mesh en la AWS CLI referencia. AWS CLI

Eliminación de una malla

Consola de administración de AWS

Para eliminar una puerta de enlace virtual mediante el Consola de administración de AWS

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija la malla que desea eliminar. Se muestran todas las mallas de su propiedad y que se han compartido con usted.
3. En el cuadro de confirmación, escriba **`delete`** y, a continuación, haga clic en Eliminar.

AWS CLI

Para eliminar una malla mediante el AWS CLI

1. Usa el siguiente comando para eliminar la malla (reemplaza los *`red`* valores por los tuyos):

```
aws appmesh delete-mesh \
     --mesh-name meshName
```

2. Ejemplo de salida:

```
{
    "mesh": {
        "meshName": "meshName",
        "metadata": {
            "arn":"arn:aws:appmesh:us-west-2:123456789012:mesh/meshName",
            "createdAt": "2022-04-06T08:45:50.072000-05:00",
            "lastUpdatedAt": "2022-04-07T11:06:32.795000-05:00",
            "meshOwner": "123456789012",
            "resourceOwner": "123456789012",
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
```

```
        },
        "spec": {},
        "status": {
            "status": "DELETED"
        }
    }
}
```

Para obtener más información sobre cómo eliminar una malla con App Mesh, consulta el comando delete-mesh en la AWS CLI referencia. AWS CLI

Servicios virtuales

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Un servicio virtual es una abstracción de un servicio real que se proporcionan mediante un nodo virtual directa o indirectamente a través de un router virtual. Los servicios dependientes llaman al servicio virtual por su `virtualServiceName` y dichas solicitudes se dirigen al nodo virtual o router virtual que se especifica como el proveedor del servicio virtual.

Creación de un servicio virtual

Consola de administración de AWS

Para crear un servicio virtual mediante Consola de administración de AWS

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija la malla en la que desea crear el servicio virtual. Se muestran todas las mallas que son de su propiedad y que se han compartido con usted.
3. Elija Virtual services (Servicios virtuales) en el panel de navegación izquierdo.
4. Elija Create Virtual service (Crear servicio virtual).

5. En Virtual service name (Nombre del servicio virtual), seleccione un nombre para el servicio virtual. Puede elegir cualquier nombre, pero se recomienda usar el nombre de detección de servicios del servicio real de destino, por ejemplo `my-service.default.svc.cluster.local`, para que sea más fácil correlacionar sus servicios virtuales con los servicios reales. De esta manera, no es necesario cambiar el código para hacer referencia a un nombre diferente al que se refiere actualmente. El nombre que especifique debe ser una dirección IP que no sea de bucle invertido, porque el contenedor de aplicaciones debe poder resolver correctamente el nombre antes de enviar la solicitud al proxy de Envoy. Puede usar cualquier dirección IP que no sea de bucle invertido, porque ni la aplicación ni los contenedores del proxy se comunican con esta dirección IP. El proxy se comunica con otros servicios virtuales a través de los nombres configurados para ellos en App Mesh, no a través de las direcciones IP que resuelven los nombres.
6. En Provider (Proveedor), elija el tipo de proveedor de su servicio virtual:
   - Si desea que el servicio virtual propague tráfico por varios nodos virtuales, seleccione Virtual router (Router virtual) y elija el router virtual que usará en el menú desplegable.
   - Si desea que el servicio virtual alcance un nodo virtual directamente sin un enrutador virtual, seleccione Nodo virtual y elija el nodo virtual que desea usar en el menú desplegable.

     > **Note**
     >
     > App Mesh puede crear automáticamente una política predeterminada de reintentos de ruta de Envoy para cada proveedor de nodos virtuales que defina a partir del 29 de julio de 2020, aunque no pueda definir dicha política a través de la API de App Mesh. Para obtener más información, consulte Política de reintentos de ruta predeterminada.

   - Si no desea que el servicio virtual dirija el tráfico en este momento (por ejemplo, si los nodos virtuales o routers virtuales todavía no existen), seleccione None (Ninguno). Puede actualizar el proveedor para este servicio virtual más adelante.
7. Elija Create virtual service (Crear servicio virtual) para finalizar.

AWS CLI

Para crear un servicio virtual mediante la AWS CLI.

Cree un servicio virtual con un proveedor de nodos virtuales mediante el siguiente comando y un archivo JSON de entrada (sustituya los *red* valores por los suyos):

1. ```
   aws appmesh create-virtual-service \
   --cli-input-json file://create-virtual-service-virtual-node.json
   ```

2. Contenido del ejemplo create-virtual-service-virtual -node.json:

   ```
   {
       "meshName": "meshName",
       "spec": {
           "provider": {
               "virtualNode": {
                   "virtualNodeName": "nodeName"
               }
           }
       },
       "virtualServiceName": "serviceA.svc.cluster.local"
   }
   ```

3. Ejemplo de salida:

   ```
   {
       "virtualService": {
           "meshName": "meshName",
           "metadata": {
               "arn": "arn:aws:appmesh:us-west-2:210987654321:mesh/meshName/
   virtualService/serviceA.svc.cluster.local",
               "createdAt": "2022-04-06T09:45:35.890000-05:00",
               "lastUpdatedAt": "2022-04-06T09:45:35.890000-05:00",
               "meshOwner": "123456789012",
               "resourceOwner": "210987654321",
               "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
               "version": 1
           },
           "spec": {
               "provider": {
                   "virtualNode": {
                       "virtualNodeName": "nodeName"
                   }
               }
           },
           "status": {
   ```
   ```

```
            "status": "ACTIVE"
        },
        "virtualServiceName": "serviceA.svc.cluster.local"
    }
}
```

Para obtener más información sobre la AWS CLI creación de un servicio virtual con App Mesh, consulte el create-virtual-servicecomando en la AWS CLI referencia.

Eliminación de un servicio virtual

> **Note**
>
> No se puede eliminar un servicio virtual al que hace referencia una ruta de puerta de enlace. Primero debe eliminar la ruta de la puerta de enlace.

Consola de administración de AWS

Para eliminar un servicio virtual mediante el Consola de administración de AWS

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija la malla en la que desee eliminar un servicio virtual. Se muestran todas las mallas que son de su propiedad y que se han compartido con usted.
3. Elija Virtual services (Servicios virtuales) en el panel de navegación izquierdo.
4. Elija el servicio virtual que desee eliminar y haga clic en Eliminar en la esquina superior derecha. Solo puede eliminar una puerta de enlace virtual en la que su cuenta aparezca como propietaria del recurso.
5. En el cuadro de confirmación, escriba **`delete`** y, a continuación, haga clic en Eliminar.

AWS CLI

Para eliminar un servicio virtual mediante el AWS CLI

1. Utilice el siguiente comando para eliminar el servicio virtual (sustituya los *`red`* valores por los suyos propios):

```
aws appmesh delete-virtual-service \
     --mesh-name meshName \
     --virtual-service-name serviceA.svc.cluster.local
```

2. Ejemplo de salida:

```
{
    "virtualService": {
        "meshName": "meshName",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:210987654321:mesh/meshName/
virtualService/serviceA.svc.cluster.local",
            "createdAt": "2022-04-06T09:45:35.890000-05:00",
            "lastUpdatedAt": "2022-04-07T10:39:42.772000-05:00",
            "meshOwner": "123456789012",
            "resourceOwner": "210987654321",
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 2
        },
        "spec": {
            "provider": {
                "virtualNode": {
                    "virtualNodeName": "nodeName"
                }
            }
        },
        "status": {
            "status": "DELETED"
        },
        "virtualServiceName": "serviceA.svc.cluster.local"
    }
}
```

Para obtener más información sobre cómo eliminar un servicio virtual con App Mesh, consulta el delete-virtual-servicecomando en la AWS CLI referencia. AWS CLI

Puertas de enlace virtuales

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Una puerta de enlace virtual permite que los recursos que están fuera de la malla se comuniquen con los recursos que están dentro de la malla. La puerta de enlace virtual representa un proxy de Envoy que se ejecuta en un servicio de Amazon ECS, en un servicio de Kubernetes o en una instancia de Amazon. EC2 A diferencia de un nodo virtual, que representa un Envoy que se ejecuta con una aplicación, una puerta de enlace virtual representa un Envoy implementado por sí mismo.

Los recursos externos deben poder resolver un nombre DNS en una dirección IP asignada al servicio o la instancia que ejecuta Envoy. Luego, Envoy puede obtener acceso a toda la configuración de App Mesh para los recursos que se encuentran dentro de la malla. La configuración para gestionar las solicitudes entrantes en la puerta de enlace virtual se especifica mediante Rutas de puerta de enlace.

> **Important**
>
> Una puerta de enlace virtual con HTTP o un agente de HTTP2 escucha reescribe el nombre de host de la solicitud entrante con el nombre del servicio virtual de destino de Gateway Route y, de forma predeterminada, se reescribe el prefijo coincidente de Gateway Route. `/` Por ejemplo, si ha configurado la ruta de la puerta de enlace, haga coincidir el prefijo con `/chapter` y, si la solicitud entrante es `/chapter/1`, la solicitud se reescribiría como `/1`. Para configurar las reescrituras, consulte la sección Creación de una ruta de puerta de enlace de Rutas de puerta de enlace.
>
> Al crear una puerta de enlace virtual, `proxyConfiguration` y `user` no deben configurarse.

Para completar un end-to-end tutorial, consulte Configuración de la puerta de enlace entrante.

Creación de una puerta de enlace virtual

> **Note**
>
> Al crear una puerta de enlace virtual, debe agregar un selector de espacios de nombres con una etiqueta para identificar la lista de espacios de nombres con los que asociar las rutas de puerta de enlace a la puerta de enlace virtual creada.

Consola de administración de AWS

Para crear una puerta de enlace virtual mediante el Consola de administración de AWS

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija la malla en la que desee crear la puerta de enlace virtual. Se muestran todas las mallas que son de su propiedad y que se han compartido con usted.
3. Elija Puertas de enlace virtuales en el panel de navegación izquierdo.
4. Seleccione Crear una puerta de enlace virtual.
5. En Nombre de puerta de enlace virtual, escriba un nombre para la puerta de enlace.
6. (Opcional, pero recomendado) Configure los Valores predeterminados de la política del cliente.
    a. (Opcional) Seleccione Aplicar TLS si desea que la puerta de enlace solo se comunique con los servicios virtuales que utilizan la seguridad de la capa de transporte (TLS).
    b. (Opcional) Para Puertos, especifique uno o más puertos en los que desee aplicar la comunicación TLS con los servicios virtuales.
    c. Para Método de validación, seleccione una de las siguientes opciones. El certificado que especifique ya debe existir y cumplir unos requisitos específicos. Para obtener más información, consulte Requisitos del certificado.
        - Alojamiento de AWS Private Certificate Authority: seleccione uno o más certificados existentes.
        - Alojamiento de Envoy Secret Discovery Service (SDS): escriba el nombre del secreto que Envoy obtiene mediante Secret Discovery Service.
        - Alojamiento de archivos local: especifique la ruta al archivo de la cadena de certificados en el sistema de archivos en el que esté implementado Envoy.

d. (Opcional) Escriba un Nombre alternativo del asunto. Para agregar más SANs, seleccione Agregar SAN. SANs debe tener formato FQDN o URI.

e. (Opcional) Seleccione Proporcionar un certificado de cliente y una de las siguientes opciones para proporcionar un certificado de cliente cuando un servidor lo solicite y habilitar la autenticación TLS mutua. Para obtener más información sobre la TLS mutua, consulte los documentos de Autenticación TLS mutua de App Mesh.

- Alojamiento de Envoy Secret Discovery Service (SDS): escriba el nombre del secreto que Envoy obtiene mediante Secret Discovery Service.
- Alojamiento de archivos local: especifique la ruta al archivo en cadena del certificado, así como la clave privada, en el sistema de archivos en el que esté implementado Envoy. Para obtener información completa sobre end-to-end cómo implementar una malla con una aplicación de ejemplo que utiliza el cifrado con archivos locales, consulte Configuración de TLS con certificados TLS proporcionados por archivos activados. GitHub

7. (Opcional) Para configurar el registro, seleccione Registro. Escriba la ruta de los registros de acceso HTTP que desea que utilice Envoy. Te recomendamos la `/dev/stdout` ruta para que puedas usar los controladores de registro de Docker para exportar tus registros de Envoy a un servicio como Amazon CloudWatch Logs.

> **Note**
>
> Los registros deben ser recibidos por un agente en su aplicación y enviados a un destino. Esta ruta de archivo solo indica a Envoy donde enviar los registros.

8. Configure el oyente.

a. Seleccione un protocolo y especifique el puerto en el que Envoy escucha el tráfico. El oyente http permite la transición de la conexión a websockets. Puede hacer clic en Agregar agente de escucha para añadir varios oyentes. El botón Eliminar eliminará ese oyente.

b. (Opcional) Habilitar grupo de conexiones

La agrupación de conexiones limita el número de conexiones que el Envoy de puerta de enlace virtual puede establecer simultáneamente. Su objetivo es evitar que su instancia de Envoy se sobrecargue de conexiones y le permite ajustar la configuración del tráfico a las necesidades de sus aplicaciones.

Puede configurar los ajustes del grupo de conexiones del lado de destino para un oyente de puerta de enlace virtual. App Mesh establece la configuración del conjunto de conexiones del lado del cliente en infinita de forma predeterminada, lo que simplifica la configuración de la malla.

> **Note**
>
> Los protocolos portMapping `connectionPool` y `connectionPool` deben ser iguales. Si su protocolo de oyente es `grpc` o `http2`, especifique `maxRequests` únicamente. Si su protocolo de oyente es `http`, puede especificar tanto `maxConnections` como `maxPendingRequests`.

- En Número máximo de conexiones, especifique el número máximo de conexiones salientes.
- En Número máximo de solicitudes, especifique el número máximo de solicitudes paralelas que se pueden establecer con el Envoy de puerta de enlace virtual.
- (Opcional) En Número máximo de solicitudes pendientes, especifique el número de solicitudes de desbordamiento después del Número máximo de conexiones que un Envoy pone en espera. El valor predeterminado es `2147483647`.

c. (Opcional) Si desea configurar una comprobación de estado para su oyente, seleccione Habilitar la comprobación de estado.

Una política de comprobación de estado es opcional, pero si especifica algún valor para una política de comprobación de estado, debe especificar valores para el Umbral de buen estado, Intervalo de comprobación de estado, Protocolo de comprobación de estado, Tiempo de espera de comprobación de estado y Umbral de mal estado.

- En Protocolo de comprobación de estado, elija un protocolo. Si especifica grpc, el servicio debe cumplir el Protocolo de comprobación de estado GRPC.
- En Puerto de comprobación de estado, especifique el puerto en el que se debe ejecutar la comprobación de estado.
- En Umbral de buen estado, especifique el número de comprobaciones de estado correctas consecutivas que deben producirse antes de declarar que el oyente está en buen estado.

- En Intervalo de comprobación de estado, especifique el período de tiempo en milisegundos entre cada ejecución de comprobación de estado.
- En Ruta, especifique la ruta de destino de cada solicitud de comprobación de estado. Este valor solo se usa si el Protocolo de comprobación de estado es `http` o `http2`. El valor se ignora en el caso de los demás protocolos.
- En Periodo de espera, especifique, en milisegundos, el plazo de tiempo que se va a esperar cuando se reciba una respuesta de comprobación de estado.
- En Umbral de estado incorrecto, especifique el número de comprobaciones de estado incorrectas consecutivas que deben producirse antes de declarar que el agente de escucha está en mal estado.

d. (Opcional) Si desea especificar si los clientes se comunican con esta puerta de enlace virtual mediante TLS, seleccione Habilitar la terminación de TLS.

- En Modo, seleccione el modo para el que desea que se configure TLS en el oyente.
- En Método del certificado, seleccione una de las siguientes opciones. El certificado debe cumplir unos requisitos específicos. Para obtener más información, consulte Requisitos del certificado.
  - AWS Certificate Manager alojamiento: selecciona un certificado existente.
  - Alojamiento de Envoy Secret Discovery Service (SDS): escriba el nombre del secreto que Envoy obtiene mediante el Secret Discovery Service.
  - Alojamiento local de archivos: especifique la ruta a la Cadena de certificados y a los archivos de Clave privada del sistema de archivos en el que está implementado Envoy.
- (Opcional) Seleccione Exigir certificado de cliente y una de las siguientes opciones para habilitar la autenticación TLS mutua si el cliente proporciona un certificado. Para obtener más información sobre la TLS mutua, consulte los documentos de Autenticación TLS mutua de App Mesh.
  - Alojamiento de Envoy Secret Discovery Service (SDS): escriba el nombre del secreto que Envoy obtiene mediante el Secret Discovery Service.
  - Alojamiento de archivos local: especifique la ruta al archivo de la cadena de certificados en el sistema de archivos en el que esté implementado Envoy.
- (Opcional) Escriba un Nombre alternativo del asunto. Para agregar más SANs, seleccione Agregar SAN. SANs debe tener formato FQDN o URI.

9. Elija Crear puerta de enlace virtual para finalizar.

AWS CLI

Para crear una puerta de enlace virtual mediante la AWS CLI.

Cree una puerta de enlace virtual con el siguiente comando e introduzca JSON (sustituya los *red* valores por los suyos):

1.
```
aws appmesh create-virtual-gateway \
--mesh-name meshName \
--virtual-gateway-name virtualGatewayName \
--cli-input-json file://create-virtual-gateway.json
```

2. Contenido del ejemplo create-virtual-gateway .json:

```
{
    "spec": {
      "listeners": [
        {
          "portMapping": {
            "port": 9080,
            "protocol": "http"
          }
        }
      ]
    }
}
```

3. Ejemplo de salida:

```
{
    "virtualGateway": {
        "meshName": "meshName",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:123456789012:mesh/meshName/
virtualGateway/virtualGatewayName",
            "createdAt": "2022-04-06T10:42:42.015000-05:00",
            "lastUpdatedAt": "2022-04-06T10:42:42.015000-05:00",
            "meshOwner": "123456789012",
            "resourceOwner": "123456789012",
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "spec": {
```

```
            "listeners": [
                {
                    "portMapping": {
                        "port": 9080,
                        "protocol": "http"
                    }
                }
            ]
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualGatewayName": "virtualGatewayName"
    }
}
```

Para obtener más información sobre cómo crear una puerta de AWS CLI enlace virtual con App Mesh, consulte el create-virtual-gatewaycomando en la AWS CLI referencia.

Implementación de una puerta de enlace virtual

Implemente un servicio de Amazon ECS o Kubernetes que contenga solo el contenedor de Envoy. También puedes implementar el contenedor Envoy en una EC2 instancia de Amazon. Para obtener más información, consulta Cómo empezar a usar App Mesh y Amazon EC2. Para obtener más información sobre cómo implementar en Amazon ECS, consulte Introducción a App Mesh y Amazon ECS o Introducción a AWS App Mesh y Kubernetes para implementar en Kubernetes. Debe establecer la variable de entorno `APPMESH_RESOURCE_ARN` en `mesh/`*`mesh-name`*`/virtualGateway/`*`virtual-gateway-name`* y no debe especificar la configuración del proxy para que el tráfico del proxy no se redirija hacia sí mismo. De forma predeterminada, App Mesh utiliza el nombre del recurso que se especificó en `APPMESH_RESOURCE_ARN` cuando Envoy hace referencia a sí mismo en métricas y registros de seguimiento. Puede anular este comportamiento estableciendo la variable de entorno `APPMESH_RESOURCE_CLUSTER` con su propio nombre.

Recomendamos implementar varias instancias del contenedor y configurar un Equilibrador de carga de red para equilibrar la carga del tráfico a las instancias. El nombre de detección de servicios del balanceador de cargas es el nombre que deseas que usen los servicios externos para acceder a los recursos que están en la malla, por ejemplo. *`myapp.example.com`* Para obtener más información, consulte Creación de un balanceador de carga de red (Amazon ECS), Creación de un balanceador

de carga externo (Kubernetes) o Tutorial: Aumente la disponibilidad de su aplicación en Amazon. EC2 También puede encontrar más ejemplos y tutoriales en nuestros ejemplos de App Mesh.

Habilitación de la autorización de proxy para Envoy. Para obtener más información, consulte Autorización de proxy de Envoy.

Eliminación de una puerta de enlace virtual

Consola de administración de AWS

Para eliminar una puerta de enlace virtual mediante el Consola de administración de AWS

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija la malla desde la que desea eliminar una puerta de enlace virtual. Se muestran todas las mallas que son de su propiedad y que se han compartido con usted.
3. Elija Puertas de enlace virtuales en el panel de navegación izquierdo.
4. Seleccione la puerta de enlace virtual que desee eliminar y elija Eliminar. No puede eliminar una puerta de enlace virtual si tiene alguna ruta de puerta de enlace asociada. Primero debe eliminar todas las rutas de puerta de enlace asociadas. Solo puede eliminar una puerta de enlace virtual en la que su cuenta aparezca como propietaria del recurso.
5. En el cuadro de confirmación, escriba **`delete`** y, a continuación, elija Eliminar.

AWS CLI

Para eliminar una puerta de enlace virtual mediante el AWS CLI

1. Utilice el siguiente comando para eliminar la puerta de enlace virtual (sustituya los *red* valores por los suyos propios):

```
aws appmesh delete-virtual-gateway \
     --mesh-name meshName \
     --virtual-gateway-name virtualGatewayName
```

2. Ejemplo de salida:

```
{
    "virtualGateway": {
        "meshName": "meshName",
        "metadata": {
```

```
            "arn": "arn:aws:appmesh:us-west-2:123456789012:mesh/meshName/
virtualGateway/virtualGatewayName",
            "createdAt": "2022-04-06T10:42:42.015000-05:00",
            "lastUpdatedAt": "2022-04-07T10:57:22.638000-05:00",
            "meshOwner": "123456789012",
            "resourceOwner": "123456789012",
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 2
        },
        "spec": {
            "listeners": [
                {
                    "portMapping": {
                        "port": 9080,
                        "protocol": "http"
                    }
                }
            ]
        },
        "status": {
            "status": "DELETED"
        },
        "virtualGatewayName": "virtualGatewayName"
    }
}
```

Para obtener más información sobre cómo eliminar una puerta de AWS CLI enlace virtual con App Mesh, consulta el delete-virtual-gatewaycomando en la AWS CLI referencia.

Rutas de puertas de enlace

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Una ruta de gateway se asocia a una gateway virtual y dirige el tráfico a un servicio virtual existente. Si una ruta coincide con una solicitud, puede distribuir tráfico a un servicio virtual de destino. Este tema lo ayuda a trabajar con rutas de puertas de enlace en una malla de servicios.

Creación de una ruta de puerta de enlace

Consola de administración de AWS

Para crear una ruta de puerta de enlace mediante el Consola de administración de AWS

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija la malla en la que desea crear la ruta de puerta de enlace. Se muestran todas las mallas que son de su propiedad y que se han compartido con usted.
3. Elija Puertas de enlace virtuales en el panel de navegación izquierdo.
4. Elija la puerta de enlace virtual a la que desee asociar una nueva ruta de puerta de enlace. Si no hay ninguna en la lista, deberá crear una puerta de enlace virtual primero. Solo puede crear una ruta de puerta de enlace para una puerta de enlace virtual cuya cuenta figure como propietaria del recurso.
5. En la tabla Rutas de puerta de enlace, elija Crear una ruta de puerta de enlace.
6. En Nombre de ruta de puerta de enlace, especifique el nombre que se va a utilizar para la ruta de puerta de enlace.
7. En Tipo de ruta de puerta de enlace, elija http, http2 o grpc.
8. Seleccione un nombre de servicio virtual existente. Si no hay ninguno en la lista, debe crear un servicio virtual primero.
9. Elija el puerto que corresponda al destino del Puerto del proveedor de servicios virtuales. El puerto del proveedor de servicios virtuales es obligatorio cuando el proveedor (enrutador o nodo) del servicio virtual seleccionado tiene varios oyentes.
10. (Opcional) En Prioridad, especifique la prioridad de esta ruta de puerta de enlace.
11. En Configuración de coincidencia, especifique:

    - Si el tipo seleccionado es http/http2:
      - (Opcional) Método: especifica el encabezado del método que debe coincidir en las solicitudes http/http2 entrantes.
      - (Opcional) Coincidencia de puerto: hace coincidir el puerto del tráfico entrante. La coincidencia de puerto es necesaria si esta puerta de enlace virtual tiene varios oyentes.

- (Opcional) Nombre de host exacto/de sufijo: especifica el nombre de host que debe coincidir en la solicitud entrante para enrutar al servicio virtual de destino.
- Prefix/Exact/RegexRuta (opcional) - El método para hacer coincidir la ruta de la URL.
  - Coincidencia de prefijo: una solicitud coincidente de una ruta de puerta de enlace se reescribe con el nombre del servicio virtual de destino y, de forma predeterminada, el prefijo coincidente se reescribe como `/`. Según cómo configure el servicio virtual, podría utilizar un enrutador virtual para enrutar la solicitud a nodos virtuales diferentes, en función de prefijos o encabezados específicos.

    Important

    - No puede especificar `/aws-appmesh*` ni `/aws-app-mesh*` para la Coincidencia de prefijo. Estos prefijos están reservados para el uso interno de App Mesh en el futuro.
    - Si se definen varias rutas de puerta de enlace, una solicitud coincide con la ruta con el prefijo más largo. Por ejemplo, si existieran dos rutas de puerta de enlace, una con el prefijo de `/chapter` y la otra con el prefijo de `/`, la solicitud de `www.example.com/chapter/` coincidiría con la ruta de la puerta de enlace con el prefijo `/chapter`.

    Note

    Si habilita la coincidencia basada en ruta/prefijo, App Mesh habilita la normalización de rutas (normalize_path y merge_slashes) para reducir la probabilidad de que se produzcan vulnerabilidades de confusión de rutas. Se producen vulnerabilidades de confusión de ruta cuando las partes que participan en la solicitud utilizan representaciones de ruta diferentes.

  - Coincidencia exacta: el parámetro exacto desactiva la coincidencia parcial de una ruta y garantiza que solo devuelva la ruta si la ruta coincide EXACTAMENTE con la URL actual.
  - Concordancia de expresiones regulares: se utiliza para describir patrones en los URLs que varias páginas pueden identificar una sola página del sitio web.
- (Opcional) Parámetros de consulta: este campo le permite hacer coincidir los parámetros de la consulta.

- (Opcional) Encabezados: especifica los encabezados de http y http2. Debe coincidir con la solicitud entrante para enrutarse al servicio virtual de destino.

- Si grpc es el tipo seleccionado:
  - El Tipo de coincidencia de nombre de alojamiento y la Coincidencia exacta/de sufijo(opcional): especifica el nombre de host que debe coincidir en la solicitud entrante para enrutarla al servicio virtual de destino.
  - nombre del servicio grpc - El servicio grpc actúa como una API para su aplicación y se define con. ProtoBuf

    > **Important**
    >
    > No puede especificar `/aws.app-mesh*` ni `aws.appmesh` para el Nombre del servicio. Estos nombres de servicio están reservados para el uso interno de App Mesh en el futuro.

  - (Opcional) Metadatos: especifica los metadatos de grpc. Debe coincidir con la solicitud entrante para enrutarse al servicio virtual de destino.

12. (Opcional) Para la configuración de Reescribir:

- Si el tipo seleccionado es http/http2:
  - Si Prefijo es el tipo de coincidencia seleccionado:
    - Anulación de la reescritura automática del nombre de alojamiento: de forma predeterminada, el nombre de host se reescribe con el nombre del servicio virtual de destino.
    - Anulación de la reescritura automática del prefijo: cuando está activada, Reescritura de prefijos especifica el valor del prefijo reescrito.
  - Si Coincidencia exacta es el tipo de coincidencia seleccionado:
    - Anulación de la reescritura automática del nombre de alojamiento: de forma predeterminada, el nombre de host se reescribe con el nombre del servicio virtual de destino.
    - Reescritura de la ruta: especifica el valor de la ruta reescrita. No hay una ruta predeterminada.
  - Si Coincidencia de ruta regex es el tipo de coincidencia seleccionado:

- Anulación de la reescritura automática del nombre de alojamiento: de forma predeterminada, el nombre de host se reescribe con el nombre del servicio virtual de destino.
- Reescritura de la ruta: especifica el valor de la ruta reescrita. No hay una ruta predeterminada.

- Si grpc es el tipo seleccionado:
  - Anulación de la reescritura automática del nombre de alojamiento: de forma predeterminada, el nombre de host se reescribe con el nombre del servicio virtual de destino.

13. Elija Crear una ruta de puerta de enlace para finalizar.

AWS CLI

Para crear una ruta de puerta de enlace mediante la AWS CLI.

Cree una ruta de puerta de enlace con el siguiente comando e introduzca JSON (sustituya los *red* valores por los suyos):

1.
```
aws appmesh create-virtual-gateway \
--mesh-name meshName \
--virtual-gateway-name virtualGatewayName \
--gateway-route-name gatewayRouteName \
--cli-input-json file://create-gateway-route.json
```

2. Contenido del ejemplo create-gateway-route .json:

```
{
    "spec": {
        "httpRoute" : {
            "match" : {
                "prefix" : "/"
            },
            "action" : {
                "target" : {
                    "virtualService": {
                        "virtualServiceName": "serviceA.svc.cluster.local"
                    }
                }
            }
```

```
        }
    }
}
```

3. Ejemplo de salida:

```
{
    "gatewayRoute": {
        "gatewayRouteName": "gatewayRouteName",
        "meshName": "meshName",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:210987654321:mesh/meshName/
virtualGateway/virtualGatewayName/gatewayRoute/gatewayRouteName",
            "createdAt": "2022-04-06T11:05:32.100000-05:00",
            "lastUpdatedAt": "2022-04-06T11:05:32.100000-05:00",
            "meshOwner": "123456789012",
            "resourceOwner": "210987654321",
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "spec": {
            "httpRoute": {
                "action": {
                    "target": {
                        "virtualService": {
                            "virtualServiceName": "serviceA.svc.cluster.local"
                        }
                    }
                },
                "match": {
                    "prefix": "/"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualGatewayName": "gatewayName"
    }
}
```

Para obtener más información sobre cómo crear una ruta de puerta de AWS CLI enlace con App Mesh, consulta el create-gateway-routecomando en la AWS CLI referencia.

Eliminación de una ruta de puerta de enlace

Consola de administración de AWS

Para eliminar una ruta de puerta de enlace mediante el Consola de administración de AWS

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija la malla desde la que desea eliminar una ruta de puerta de enlace. Se muestran todas las mallas que son de su propiedad y que se han compartido con usted.
3. Elija Puertas de enlace virtuales en el panel de navegación izquierdo.
4. Elija la puerta de enlace virtual de la que desea eliminar una ruta de puerta de enlace.
5. En la tabla Rutas de puerta de enlace, elija la ruta de puerta de enlace que desee eliminar y seleccione Eliminar. Solo puede eliminar una ruta de puerta de enlace si su cuenta aparece como Propietaria del recurso.
6. En el cuadro de confirmación, escriba **`delete`** y, a continuación, haga clic en Eliminar.

AWS CLI

Para eliminar una ruta de puerta de enlace mediante el AWS CLI

1. Utilice el siguiente comando para eliminar la ruta de la puerta de enlace (sustituya los *`red`* valores por los suyos propios):

```
aws appmesh delete-gateway-route \
     --mesh-name meshName \
     --virtual-gateway-name virtualGatewayName \
     --gateway-route-name gatewayRouteName
```

2. Ejemplo de salida:

```
{
    "gatewayRoute": {
        "gatewayRouteName": "gatewayRouteName",
        "meshName": "meshName",
        "metadata": {
```

```
            "arn": "arn:aws:appmesh:us-west-2:210987654321:mesh/meshName/
virtualGateway/virtualGatewayName/gatewayRoute/gatewayRouteName",
            "createdAt": "2022-04-06T11:05:32.100000-05:00",
            "lastUpdatedAt": "2022-04-07T10:36:33.191000-05:00",
            "meshOwner": "123456789012",
            "resourceOwner": "210987654321",
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 2
        },
        "spec": {
            "httpRoute": {
                "action": {
                    "target": {
                        "virtualService": {
                            "virtualServiceName": "serviceA.svc.cluster.local"
                        }
                    }
                },
                "match": {
                    "prefix": "/"
                }
            }
        },
        "status": {
            "status": "DELETED"
        },
        "virtualGatewayName": "virtualGatewayName"
    }
}
```

Para obtener más información sobre cómo eliminar una ruta de puerta de AWS CLI enlace con App Mesh, consulta el delete-gateway-routecomando en la AWS CLI referencia.

Nodos virtuales

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS

App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Un nodo virtual actúa como un puntero lógico a un determinado grupo de tareas, como, por ejemplo, un servicio ECS de Amazon o una implementación de Kubernetes. Al crear un nodo virtual, debe especificar un método de detección de servicios para su grupo de tareas. Todo el tráfico de entrada que el nodo virtual espera debe especificarse como un oyente. Cualquier servicio virtual al que un nodo virtual envía tráfico saliente se especifica como backend.

Los metadatos de respuesta del nuevo nodo virtual contienen el Nombre de recurso de Amazon (ARN) que se asocia al nodo virtual. Establezca este valor como la variable de entorno `APPMESH_RESOURCE_ARN` del contenedor del proxy de Envoy del grupo de tareas en la definición de tarea de Amazon ECS o la especificación del pod de Kubernetes. Por ejemplo, el valor podría ser `arn:aws:appmesh:us-west-2:111122223333:mesh/myMesh/virtualNode/myVirtualNode`. Después, esto se asigna a los parámetros `node.id` y `node.cluster` de Envoy. Debe utilizar la versión `1.15.0` o posterior de la imagen de Envoy al establecer esta variable. Para obtener más información sobre las variables de App Mesh Envoy, consulte Envoy.

> **Note**
>
> De forma predeterminada, App Mesh utiliza el nombre del recurso que se especificó en `APPMESH_RESOURCE_ARN` cuando Envoy hace referencia a sí mismo en métricas y registros de seguimiento. Puede anular este comportamiento estableciendo la variable de entorno `APPMESH_RESOURCE_CLUSTER` con su propio nombre.

Creación de un nodo virtual

Consola de administración de AWS

Para crear un nodo virtual mediante Consola de administración de AWS

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija la malla en la que desea crear el nodo virtual. Se muestran todas las mallas que son de su propiedad y que se han compartido con usted.
3. Elija Nodos virtuales en el panel de navegación izquierdo.

4. Seleccione Crear nodo virtual y, a continuación, especifique la configuración del nodo virtual.
5. En Nombre del nodo virtual, escriba un nombre para el nodo virtual.
6. En Método de detección de servicios, elija una de las siguientes opciones:

   - DNS: especifique el nombre de host DNS del servicio real que representa el nodo virtual. El proxy de Envoy se implementa en una VPC de Amazon. El proxy envía solicitudes de resolución de nombres al servidor DNS que está configurado para la VPC. Si se resuelve el nombre de host, el servidor DNS devuelve una o más direcciones IP. Para obtener más información acerca de la configuración DNS de VPC, consulte Uso de DNS con su VPC. Para Tipo de respuesta DNS (opcional), especifique los tipos de puntos de conexión devueltos por el solucionador de DNS. Equilibrador de carga significa que el solucionador de DNS devuelve un conjunto de puntos de conexión con equilibrio de carga. Puntos de enlace significa que la resolución de DNS devuelve todos los puntos de conexión. De forma predeterminada, se supone que el tipo de respuesta es Equilibrador de carga.

     > **Note**
     >
     > Si usa Route53, deberá utilizar el Equilibrador de carga.

   - AWS Cloud Map: especifique un Nombre de servicio y un Espacio de nombres HTTP existentes. Si lo desea, también puede especificar los atributos que App Mesh puede consultar AWS Cloud Map seleccionando Añadir fila y especificando una clave y un valor. Solo se devolverán las instancias que coincidan con todos los key/value pares especificados. Para poder AWS Cloud Map utilizarla, tu cuenta debe tener el rol `AWSServiceRoleForAppMesh` vinculado al servicio. Para obtener más información al respecto AWS Cloud Map, consulta la Guía para AWS Cloud Map desarrolladores.
   - Ninguno: seleccione si su nodo virtual no espera tráfico de entrada.

7. Preferencia de versión de IP

   Controle qué versión de IP debe usarse para el tráfico dentro de la malla activando la opción Anulación del comportamiento predeterminado de la versión de IP. De forma predeterminada, App Mesh usa varias versiones de IP.

> **Note**
>
> La configuración de la preferencia IP en el nodo virtual solo anula la preferencia IP establecida para la malla en este nodo específico.

- Predeterminado
  - El solucionador de DNS de Envoy prefiere `IPv6` y recurre a `IPv4`.
  - Usamos la `IPv4` dirección devuelta AWS Cloud Map si está disponible y recurrimos a usar la `IPv6` dirección.
  - El punto de conexión creado para la aplicación local usa una dirección `IPv4`.
  - Los oyentes de Envoy se enlazan a todas las direcciones `IPv4`.
- IPv6 preferido
  - El solucionador de DNS de Envoy prefiere `IPv6` y recurre a `IPv4`.
  - La `IPv6` dirección devuelta por AWS Cloud Map se utiliza si está disponible y se vuelve a utilizar la `IPv4` dirección
  - El punto de conexión creado para la aplicación local usa una dirección `IPv6`.
  - Los oyentes de Envoy se enlazan a todas las direcciones `IPv4` y `IPv6`.
- IPv4 preferido
  - El solucionador de DNS de Envoy prefiere `IPv4` y recurre a `IPv6`.
  - Usamos la `IPv4` dirección devuelta AWS Cloud Map si está disponible y recurrimos a usar la `IPv6` dirección.
  - El punto de conexión creado para la aplicación local usa una dirección `IPv4`.
  - Los oyentes de Envoy se enlazan a todas las direcciones `IPv4` y `IPv6`.
- IPv6 únicamente
  - El solucionador de DNS de Envoy solo usa `IPv6`.
  - Solo se utiliza la `IPv6` dirección devuelta por AWS Cloud Map . Si AWS Cloud Map devuelve una dirección`IPv4`, no se utiliza ninguna dirección IP y se devuelven resultados vacíos a Envoy.
  - El punto de conexión creado para la aplicación local usa una dirección `IPv6`.
  - Los oyentes de Envoy se enlazan a todas las direcciones `IPv4` y `IPv6`.
- IPv4 únicamente

- El solucionador de DNS de Envoy solo usa `IPv4`.
- Solo se utiliza la `IPv4` dirección devuelta por AWS Cloud Map . Si AWS Cloud Map devuelve una dirección`IPv6`, no se utiliza ninguna dirección IP y se devuelven resultados vacíos a Envoy.
- El punto de conexión creado para la aplicación local usa una dirección `IPv4`.
- Los oyentes de Envoy se enlazan a todas las direcciones `IPv4` y `IPv6`.

8. (Opcional) Valores predeterminados de la política del cliente: configure los requisitos predeterminados al comunicar con los servicios virtuales de backend.

> **Note**
>
> - Si desea habilitar la seguridad de la capa de transporte (TLS) para un nodo virtual existente, le recomendamos que cree un nodo virtual nuevo, que represente el mismo servicio que el nodo virtual existente, en el que habilitar TLS. A continuación, transfiera gradualmente el tráfico al nuevo nodo virtual mediante un enrutador y una ruta virtuales. Para obtener más información acerca de la creación de una ruta y el ajuste de las ponderaciones para la transición, consulte Rutas. Si actualiza un nodo virtual existente que presta servicio al tráfico con TLS, existe la posibilidad de que los proxies de Envoy del cliente descendente reciban el contexto de validación de TLS antes de que el proxy de Envoy del nodo virtual que ha actualizado reciba el certificado. Esto puede provocar errores de negociación de TLS en los proxies de Envoy descendentes.
> - La autorización de proxy debe estar habilitada para el proxy de Envoy implementado con la aplicación representada por los nodos virtuales del servicio de backend. Le recomendamos que, cuando habilite la autorización de proxy, restrinja el acceso únicamente a los nodos virtuales con los que se comunica este nodo virtual.

- (Opcional) Seleccione Aplicar TLS si quiere que el nodo virtual se comunique con todos los backends mediante la seguridad de la capa de transporte (TLS).
- (Opcional) Si solo quiere exigir el uso de TLS para uno o más puertos específicos, indique un número en Puertos. Para añadir puertos adicionales, seleccione Agregar puerto. Si no especifica ningún puerto, se aplicará la seguridad TLS a todos los puertos.

- En Método de validación, seleccione una de las siguientes opciones. El certificado que especifique ya debe existir y cumplir unos requisitos específicos. Para obtener más información, consulte Requisitos del certificado.
  - Alojamiento de AWS Private Certificate Authority: seleccione uno o más certificados existentes. Para ver un recorrido completo end-to-end sobre la implementación de una malla con una aplicación de ejemplo que utiliza el cifrado con un certificado ACM, consulte Configuración de TLS con AWS Certificate Manager activado. GitHub
  - Alojamiento de Envoy Secret Discovery Service (SDS): escriba el nombre del secreto que Envoy obtendrá mediante Secret Discovery Service.
  - Alojamiento de archivos local: especifique la ruta al archivo de la cadena de certificados en el sistema de archivos en el que está implementado Envoy. Para obtener información completa sobre end-to-end la implementación de una malla con una aplicación de ejemplo que utiliza el cifrado con archivos locales, consulte Configuración de TLS con certificados TLS proporcionados por archivos activados. GitHub
- (Opcional) Escriba un Nombre alternativo del asunto. Para añadir más SANs, selecciona Añadir SAN. SANs debe tener formato FQDN o URI.
- (Opcional) Seleccione Proporcionar un certificado de cliente y una de las siguientes opciones para proporcionar un certificado de cliente cuando un servidor lo solicite y habilitar la autenticación TLS mutua. Para obtener más información sobre la TLS mutua, consulte los documentos de Autenticación TLS mutua de App Mesh.
  - Alojamiento de Envoy Secret Discovery Service (SDS): escriba el nombre del secreto que Envoy obtendrá mediante Secret Discovery Service.
  - Alojamiento de archivos local: especifique la ruta al archivo de la cadena de certificados, así como la clave privada, en el sistema de archivos en el que está implementado Envoy.

9. (Opcional) Backends de servicios: especifique el servicio virtual de App Mesh con el que se comunicará el nodo virtual.

   - Escriba un nombre de servicio virtual de App Mesh o nombre de recurso de Amazon (ARN) completo para el servicio virtual con el que se comunica el nodo virtual.
   - (Opcional) Si desea establecer una configuración de TLS única para un backend, seleccione Configuración de TLS y, a continuación, Anular los valores predeterminados.
     - (Opcional) Seleccione Aplicar TLS si quiere que el nodo virtual se comunique con todos los backends mediante TLS.

- (Opcional) Si solo quiere exigir el uso de TLS para uno o más puertos específicos, escriba un número en Puertos. Para añadir puertos adicionales, seleccione Agregar puerto. Si no especifica ningún puerto, se aplicará la seguridad TLS a todos los puertos.
- En Método de validación, seleccione una de las siguientes opciones. El certificado que especifique ya debe existir y cumplir unos requisitos específicos. Para obtener más información, consulte Requisitos del certificado.
  - Alojamiento de AWS Private Certificate Authority: seleccione uno o más certificados existentes.
  - Alojamiento de Envoy Secret Discovery Service (SDS): escriba el nombre del secreto que Envoy obtendrá mediante Secret Discovery Service.
  - Alojamiento de archivos local: especifique la ruta al archivo de la cadena de certificados en el sistema de archivos en el que está implementado Envoy.
- (Opcional) Escriba un Nombre alternativo del asunto. Para agregar más SANs, seleccione Agregar SAN. SANsdebe tener formato FQDN o URI.
- (Opcional) Seleccione Proporcionar un certificado de cliente y una de las siguientes opciones para proporcionar un certificado de cliente cuando un servidor lo solicite y habilitar la autenticación TLS mutua. Para obtener más información sobre la TLS mutua, consulte los documentos de Autenticación de TLS mutua de App Mesh.
  - Alojamiento de Envoy Secret Discovery Service (SDS): escriba el nombre del secreto que Envoy obtendrá mediante Secret Discovery Service.
  - Alojamiento de archivos local: especifique la ruta al archivo de la cadena de certificados, así como la clave privada, en el sistema de archivos en el que está implementado Envoy.
- Para añadir backends adicionales, seleccione Agregar backend.

10. (Opcional) Registro

    Para configurar el registro, escriba la ruta de registro de acceso HTTP que desea que utilice Envoy. Te recomendamos la `/dev/stdout` ruta para que puedas usar los controladores de registro de Docker para exportar tus registros de Envoy a un servicio como Amazon CloudWatch Logs.

> **Note**
>
> Los registros deben ser recibidos por un agente en su aplicación y enviados a un destino. Esta ruta de archivo solo indica a Envoy donde enviar los registros.

11. Configuración del oyente

    Los oyentes admiten los protocolos `HTTP`, `HTTP/2`, `GRPC` y `TCP`. No admiten `HTTPS`

    a. Si el nodo virtual espera tráfico de entrada, especifique un puerto y protocolo para ese oyente. El oyente http permite la transición de la conexión a websockets. Puede hacer clic en Agregar agente de escucha para añadir varios oyentes. El botón Eliminar eliminará ese oyente.

    b. (Opcional) Habilitar grupo de conexiones

       La agrupación de conexiones limita el número de conexiones que un Envoy puede establecer simultáneamente con el clúster de aplicaciones local. Su objetivo es proteger su aplicación local frente a la sobrecarga de conexiones y le permite ajustar la configuración del tráfico a las necesidades de sus aplicaciones.

       Puede configurar los ajustes del grupo de conexiones del lado de destino para un oyente de nodos virtuales. App Mesh establece la configuración del conjunto de conexiones del lado del cliente en infinita de forma predeterminada, lo que simplifica la configuración de la malla.

       > **Note**
       >
       > Los protocolos connectionPool y portMapping deben ser los mismos. Si el protocolo del oyente es tcp, especifique solo maxConnections. Si e protocolo del oyente es grpc o http2, especifique solo maxRequests. Si su protocolo de escucha es http, puede especificar MaxConnections y. maxPendingRequests

       - En Número máximo de conexiones, especifique el número máximo de conexiones salientes.

- (Opcional) En Número máximo de solicitudes pendientes, especifique el número de solicitudes de desbordamiento después del Número máximo de conexiones que un Envoy pondrá en espera. El valor predeterminado es `2147483647`.

c. (Opcional) Habilite la detección de valores atípicos

La detección de valores atípicos aplicada al cliente de Envoy permite a los clientes tomar medidas casi inmediatas en las conexiones en las que se hayan observado errores graves conocidos. Es una forma de implementación de un interruptor que rastrea el estado de los hosts individuales del servicio inicial.

La detección de valores atípicos determina de forma dinámica si los puntos de conexión de un clúster ascendente funcionan de forma diferente a los demás y los elimina del conjunto de equilibrio de carga en buen estado.

> **Note**
>
> Para configurar de forma eficaz la detección de valores atípicos para un nodo virtual de servidor, el método de detección de servicios de ese nodo virtual puede ser un AWS Cloud Map DNS con el campo de tipo de respuesta establecido en. `ENDPOINTS` Si utiliza el método de detección de servicios DNS con el tipo de respuesta `LOADBALANCER`, el proxy de Envoy solo elegirá una dirección IP única para el enrutamiento al servicio ascendente. Esto anula el comportamiento de detección de valores atípicos de expulsar un host en mal estado de un conjunto de hosts. Consulte la sección sobre el método de detección de servicios para obtener más información sobre el comportamiento del proxy de Envoy en relación con el tipo de detección de servicios.

- En Errores del servidor, especifique el número de 5xx errores consecutivos necesarios para la expulsión.
- En Intervalo de detección de valores atípicos, especifique la unidad y el intervalo de tiempo entre el análisis de barrido de expulsión.
- En Duración base de la expulsión, especifique la unidad y el periodo de tiempo base durante los que se expulsa un host.
- En Porcentaje de expulsión, especifique el porcentaje máximo de hosts del grupo de equilibrio de carga que se puede expulsar.

d. (Opcional) Habilitar la comprobación de estado: configure los ajustes de una política de comprobación de estado.

   Una política de comprobación de estado es opcional, pero si especifica algún valor para una política de comprobación de estado, debe especificar valores para el Umbral de buen estado, Intervalo de comprobación de estado, Protocolo de comprobación de estado, Tiempo de espera de comprobación de estado y Umbral de mal estado.

   - En Protocolo de comprobación de estado, elija un protocolo. Si especifica grpc, el servicio debe cumplir el Protocolo de comprobación de estado GRPC.
   - En Puerto de comprobación de estado, especifique el puerto en el que se debe ejecutar la comprobación de estado.
   - En Umbral de buen estado, especifique el número de comprobaciones de estado correctas consecutivas que deben producirse antes de declarar que el oyente está en buen estado.
   - En Intervalo de comprobación de estado, especifique el período de tiempo en milisegundos entre cada ejecución de comprobación de estado.
   - En Ruta, especifique la ruta de destino de cada solicitud de comprobación de estado. Este valor solo se usa si el Protocolo de comprobación de estado es `http` o `http2`. El valor se ignora para los demás protocolos.
   - En Período de tiempo, especifique, en milisegundos, el plazo de tiempo que se va a esperar cuando se recibe una respuesta de comprobación de estado.
   - En Umbral de estado incorrecto, especifique el número de comprobaciones de estado incorrectas consecutivas que deben producirse antes de declarar que el oyente está en mal estado.

e. (Opcional) Habilitar la terminación de TLS: configure la forma en que otros nodos virtuales se comunican con este nodo virtual mediante TLS.

   - En Modo, seleccione el modo para el que desea que se configure TLS en el oyente.
   - En Método del certificado, seleccione una de las siguientes opciones: El certificado debe cumplir unos requisitos específicos. Para obtener más información, consulte Requisitos del certificado.
     - AWS Certificate Manager alojamiento: seleccione un certificado existente.
     - Alojamiento de Envoy Secret Discovery Service (SDS): escriba el nombre del secreto que Envoy obtendrá mediante Secret Discovery Service.

- Alojamiento de archivos local: especifique la ruta al archivo de la cadena de certificados, así como la clave privada, en el sistema de archivos en el que está implementado el proxy de Envoy.

- (Opcional) Seleccione Exigir certificado de cliente y una de las siguientes opciones para habilitar la autenticación TLS mutua cuando un cliente proporcione un certificado. Para obtener más información sobre la TLS mutua, consulte los documentos de Autenticación TLS mutua de App Mesh.
  - Alojamiento de Envoy Secret Discovery Service (SDS): escriba el nombre del secreto que Envoy obtendrá mediante Secret Discovery Service.
  - Alojamiento de archivos local: especifique la ruta al archivo de la cadena de certificados en el sistema de archivos en el que está implementado Envoy.
- (Opcional) Escriba un Nombre alternativo del asunto. Para agregar más SANs, seleccione Agregar SAN. SANsdebe tener formato FQDN o URI.

f. (Opcional) Tiempos de espera

> **Note**
>
> Si especifica un tiempo de espera superior al predeterminado, asegúrese de configurar un enrutador virtual y una ruta con un tiempo de espera superior al predeterminado. Sin embargo, si reduce el tiempo de espera a un valor inferior al predeterminado; opcionalmente, puede actualizar los tiempos de espera en la ruta. Para obtener más información, consulte Rutas.

- Tiempo de espera de la solicitud: puede especificar un tiempo de espera de solicitud si ha seleccionado grpc, http o http2 para el protocolo del oyente. El valor predeterminado es de 15 segundos. Un valor de `0` deshabilita el tiempo de espera.
- Duración de inactividad: puede especificar una duración de inactividad para cualquier protocolo del oyente. El valor predeterminado es de 300 segundos.

12. Elija Crear nodo virtual para finalizar.

AWS CLI

Para crear un nodo virtual mediante la AWS CLI.

Cree un nodo virtual que utilice DNS para la detección de servicios mediante el siguiente comando y un archivo JSON de entrada (sustituya los *red* valores por los suyos):

1. ```
   aws appmesh create-virtual-node \
   --cli-input-json file://create-virtual-node-dns.json
   ```

2. Contenido del ejemplo create-virtual-node-dns .json:

   ```
   {
       "meshName": "meshName",
       "spec": {
           "listeners": [
               {
                   "portMapping": {
                       "port": 80,
                       "protocol": "http"
                   }
               }
           ],
           "serviceDiscovery": {
               "dns": {
                   "hostname": "serviceBv1.svc.cluster.local"
               }
           }
       },
       "virtualNodeName": "nodeName"
   }
   ```

3. Ejemplo de código de salida:

   ```
   {
       "virtualNode": {
           "meshName": "meshName",
           "metadata": {
               "arn": "arn:aws:appmesh:us-west-2:210987654321:mesh/meshName/
   virtualNode/nodeName",
               "createdAt": "2022-04-06T09:12:24.348000-05:00",
               "lastUpdatedAt": "2022-04-06T09:12:24.348000-05:00",
               "meshOwner": "123456789012",
               "resourceOwner": "210987654321",
               "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
               "version": 1
           },
   ```
   ```

```
        "spec": {
            "listeners": [
                {
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
                    }
                }
            ],
            "serviceDiscovery": {
                "dns": {
                    "hostname": "serviceBv1.svc.cluster.local"
                }
            }
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualNodeName": "nodeName"
    }
}
```

Para obtener más información sobre la AWS CLI creación de un nodo virtual con App Mesh, consulte el create-virtual-nodecomando en la AWS CLI referencia.

Eliminación de un nodo virtual

> **Note**
>
> No puede eliminar un nodo virtual si está especificado como destino en cualquier ruta o como proveedor en cualquier servicio virtual.

Consola de administración de AWS

Para eliminar un nodo virtual mediante el Consola de administración de AWS

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija la malla en la que desee eliminar un nodo virtual. Se muestran todas las mallas que son de su propiedad y que se han compartido con usted.

3. Elija Nodos virtuales en el panel de navegación izquierdo.
4. En la tabla Nodos virtuales, elija el nodo virtual que desee eliminar y seleccione Eliminar. Para eliminar un nodo virtual, el ID de su cuenta debe figurar en las columnas Propietario de la malla o Propietario del recurso del nodo virtual.
5. En el cuadro de confirmación, escriba **`delete`** y elija Eliminar.

AWS CLI

Para eliminar un nodo virtual mediante el AWS CLI

1. Utilice el siguiente comando para eliminar el nodo virtual (sustituya los *`red`* valores por los suyos propios):

```
aws appmesh delete-virtual-node \
     --mesh-name meshName \
     --virtual-node-name nodeName
```

2. Ejemplo de código de salida:

```
{
    "virtualNode": {
        "meshName": "meshName",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:210987654321:mesh/meshName/
virtualNode/nodeName",
            "createdAt": "2022-04-06T09:12:24.348000-05:00",
            "lastUpdatedAt": "2022-04-07T11:03:48.120000-05:00",
            "meshOwner": "123456789012",
            "resourceOwner": "210987654321",
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 2
        },
        "spec": {
            "backends": [],
            "listeners": [
                {
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
                    }
                }
```

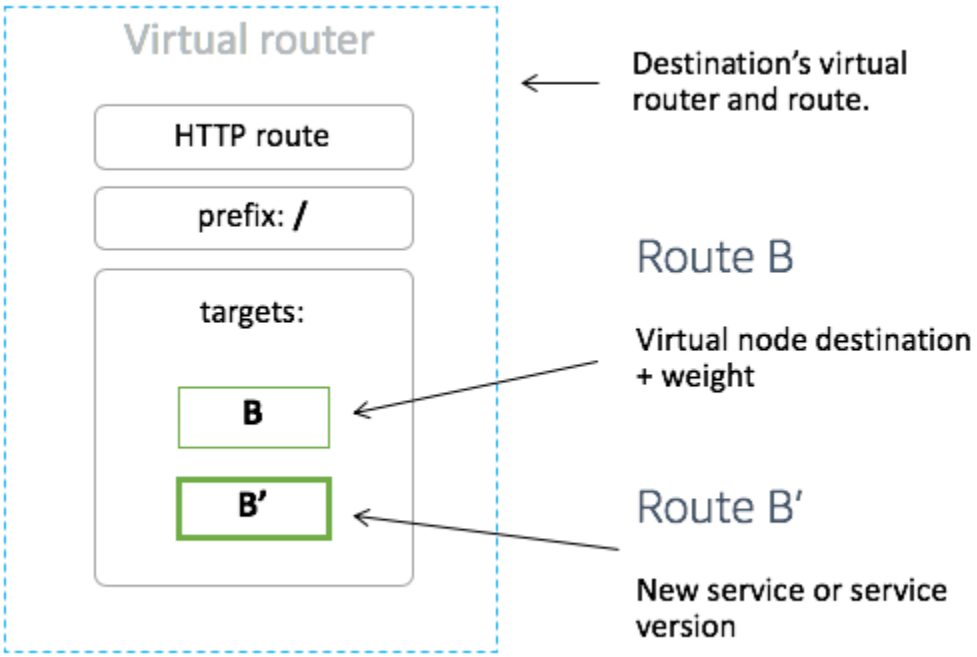
```
            ],
            "serviceDiscovery": {
                "dns": {
                    "hostname": "serviceBv1.svc.cluster.local"
                }
            }
        },
        "status": {
            "status": "DELETED"
        },
        "virtualNodeName": "nodeName"
    }
}
```

Para obtener más información sobre cómo eliminar un nodo virtual con App Mesh, consulta el delete-virtual-nodecomando en la AWS CLI referencia. AWS CLI

Enrutadores virtuales

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Los routers virtuales controlan el tráfico de uno o más servicios virtuales dentro de la malla. Después de crear un router virtual, puede crear y asociar las rutas para el router virtual que dirigen las solicitudes entrantes a diferentes nodos virtuales.



Todo el tráfico de entrada que el enrutador virtual espera se debe especificar como un oyente.

Creación de un enrutador virtual

Consola de administración de AWS

Para crear un enrutador virtual mediante Consola de administración de AWS

> **Note**
>
> Al crear un enrutador virtual, debe agregar un selector de espacios de nombres con una etiqueta para identificar la lista de espacios de nombres para asociar las rutas al enrutador virtual creado.

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija la malla en la que desea crear el enrutador virtual. Se muestran todas las mallas que son de su propiedad y que se han compartido con usted.
3. Elija Routers virtuales en el panel de navegación izquierdo.
4. Elija Crear router virtual.

5. En Nombre del router virtual, especifique un nombre para el router virtual. Se admiten hasta 255 letras, números, guiones y guiones bajos.
6. (Opcional) En Configuración del agente de escucha, especifique un puerto y protocolo para el enrutador virtual. El oyente de `http` permite la transición de la conexión a websockets. Puede hacer clic en Agregar agente de escucha para añadir varios oyentes. El botón Eliminar eliminará ese oyente.
7. Elija Crear router virtual para finalizar.

AWS CLI

Para crear un enrutador virtual mediante la AWS CLI.

Cree un router virtual con el siguiente comando e introduzca JSON (sustituya los *red* valores por los suyos):

1.
```
aws appmesh create-virtual-router \
     --cli-input-json file://create-virtual-router.json
```
2. Contenido del ejemplo create-virtual-router.json
3.
```
{
    "meshName": "meshName",
    "spec": {
        "listeners": [
            {
                "portMapping": {
                    "port": 80,
                    "protocol": "http"
                }
            }
        ]
    },
    "virtualRouterName": "routerName"
}
```
4. Ejemplo de código de salida:

```
{
    "virtualRouter": {
        "meshName": "meshName",
        "metadata": {
```

```
            "arn": "arn:aws:appmesh:us-west-2:210987654321:mesh/meshName/
virtualRouter/routerName",
            "createdAt": "2022-04-06T11:49:47.216000-05:00",
            "lastUpdatedAt": "2022-04-06T11:49:47.216000-05:00",
            "meshOwner": "123456789012",
            "resourceOwner": "210987654321",
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "spec": {
            "listeners": [
                {
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
                    }
                }
            ]
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualRouterName": "routerName"
    }
}
```


Para obtener más información sobre cómo crear un router virtual con App Mesh, consulta el comando create-virtual-router en la referencia. AWS CLI AWS CLI

Eliminación de un enrutador virtual

> **Note**
>
> No puede eliminar un enrutador virtual si tiene alguna ruta o si está especificado como proveedor de algún servicio virtual.

Consola de administración de AWS

Para eliminar un enrutador virtual mediante el Consola de administración de AWS

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija la malla en la que desee eliminar un enrutador virtual. Se muestran todas las mallas de su propiedad y que se han compartido con usted.
3. Elija Routers virtuales en el panel de navegación izquierdo.
4. En la tabla Enrutadores virtuales, elija el enrutador virtual que desee eliminar y seleccione Eliminar en la esquina superior derecha. Para eliminar un enrutador virtual, el ID de su cuenta debe figurar en las columnas Propietario de la malla o Propietario del recurso del enrutador virtual.
5. En el cuadro de confirmación, escriba **`delete`** y, a continuación, haga clic en Eliminar.

AWS CLI

Para eliminar un router virtual mediante AWS CLI

1. Utilice el siguiente comando para eliminar el router virtual (sustituya los *`red`* valores por los suyos propios):

```
aws appmesh delete-virtual-router \
     --mesh-name meshName \
     --virtual-router-name routerName
```

2. Ejemplo de código de salida:

```
{
    "virtualRouter": {
        "meshName": "meshName",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:210987654321:mesh/meshName/
virtualRouter/routerName",
            "createdAt": "2022-04-06T11:49:47.216000-05:00",
            "lastUpdatedAt": "2022-04-07T10:49:53.402000-05:00",
            "meshOwner": "123456789012",
            "resourceOwner": "210987654321",
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 2
        },
```

```
        "spec": {
            "listeners": [
                {
                    "portMapping": {
                        "port": 80,
                        "protocol": "http"
                    }
                }
            ]
        },
        "status": {
            "status": "DELETED"
        },
        "virtualRouterName": "routerName"
    }
}
```

Para obtener más información sobre cómo eliminar un router virtual con App Mesh, consulta el comando delete-virtual-router en la referencia. AWS CLI AWS CLI

Rutas

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Se asocia una ruta a un enrutador virtual. La ruta se usa para hacer coincidir las solicitudes del enrutador virtual y distribuir el tráfico a sus nodos virtuales asociados. Si una ruta coincide con una solicitud, puede distribuir el tráfico a uno o varios nodos virtuales de destino. Puede especificar la ponderación relativa para cada nodo virtual. Este tema lo ayuda a trabajar con rutas en una malla de servicio.

Creación de una ruta

Consola de administración de AWS

Para crear una ruta mediante el Consola de administración de AWS

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija la malla en la que desea crear la ruta. Se muestran todas las mallas de su propiedad y que se han compartido con usted.
3. Elija Routers virtuales en el panel de navegación izquierdo.
4. Elija el enrutador virtual que desea asociar a una nueva ruta. Si no hay ninguno en la lista, debe crear un enrutador virtual primero.
5. En la tabla Rutas, elija Crear ruta. Para crear una ruta, el ID de su cuenta debe figurar como Propietario del recurso de la ruta.
6. En Route name (Nombre de la ruta), especifique el nombre que se va a utilizar para la ruta.
7. En Tipo de ruta, elija el protocolo que desea para la ruta. El protocolo que seleccione debe coincidir con el protocolo del oyente que seleccionó para su enrutador virtual y el nodo virtual al que está enrutando el tráfico.
8. (Opcional) En Prioridad de la ruta, especifique una prioridad de 0 a 1000 para usarla en la ruta. Las rutas se asignan en función del valor especificado, siendo 0 la máxima prioridad.
9. (Opcional) Elija Configuración adicional. De los protocolos que aparecen a continuación, elija el protocolo que haya seleccionado para Tipo de ruta y especifique la configuración que desee en la consola.
10. En Configuración de destino, seleccione el nodo virtual de App Mesh existente al que dirigir el tráfico y especifique una Ponderación. Puede elegir Agregar objetivo para añadir destinos adicionales. El porcentaje de todos los destinos debe sumar 100. Si no aparece ningún nodo virtual, primero debe crear uno. Si el nodo virtual seleccionado tiene varios oyentes, se requiere el Puerto de destino.
11. Para la configuración de Coincidencia, especifique:

    La configuración de Coincidencia no está disponible para *`tcp`*

    - Si http/http2es el tipo seleccionado:
      - (Opcional) Método: especifica el encabezado del método que debe coincidir en las solicitudes http/http2 entrantes.

- (Opcional) Coincidencia de puerto: hace coincidir el puerto del tráfico entrante. La coincidencia de puerto es necesaria si este enrutador virtual tiene varios oyentes.
- Prefix/Exact/RegexRuta (opcional): método para hacer coincidir la ruta de la URL.
  - Coincidencia de prefijo: una solicitud coincidente de una ruta de puerta de enlace se reescribe con el nombre del servicio virtual de destino y, de forma predeterminada, el prefijo coincidente se reescribe con `/`. Según cómo configure el servicio virtual, podría utilizar un enrutador virtual para enrutar la solicitud a diferentes nodos virtuales, en función de prefijos o encabezados específicos.

    > **Note**
    >
    > Si habilita la coincidencia basada en Ruta/prefijo, App Mesh habilita la normalización de rutas (normalize_path y merge_slashes) para reducir la probabilidad de que se produzcan vulnerabilidades de confusión de rutas. Las vulnerabilidades de confusión de rutas se producen cuando las partes que participan en la solicitud utilizan diferentes representaciones de rutas.

  - Coincidencia exacta: el parámetro exacto desactiva la coincidencia parcial de una ruta y garantiza que solo devuelva la ruta si la ruta coincide EXACTAMENTE con la URL actual.
  - Coincidencia de regex: se utiliza para describir patrones en los que varias URL pueden identificar realmente una sola página del sitio web.
- (Opcional) Parámetros de consulta: este campo permite hacer coincidir los parámetros de la consulta.
- (Opcional) Encabezados: especifica los encabezados de http y http2. Debe coincidir con la solicitud entrante para enrutarse al servicio virtual de destino.

- Si grpc es el tipo seleccionado:
  - Nombre del servicio: el servicio de destino para el que se debe hacer coincidir la solicitud.
  - Nombre del método: el método de destino para el que se debe hacer coincidir la solicitud.
  - (Opcional) Metadatos: especifica `Match` en función de la presencia de metadatos. Todos deben coincidir para que se procese la solicitud.

12. Seleccione Crear ruta.

AWS CLI

Para crear una ruta mediante la AWS CLI.

Cree una ruta gRPC con el siguiente comando e introduzca JSON (sustituya los *red* valores por los suyos):

1. ```
   aws appmesh create-route \
        --cli-input-json file://create-route-grpc.json
   ```

2. Contenido del ejemplo create-route-grpc.json

```
{
    "meshName" : "meshName",
    "routeName" : "routeName",
    "spec" : {
       "grpcRoute" : {
          "action" : {
             "weightedTargets" : [
                {
                   "virtualNode" : "nodeName",
                   "weight" : 100
                }
             ]
          },
          "match" : {
             "metadata" : [
                {
                   "invert" : false,
                   "match" : {
                      "prefix" : "123"
                   },
                   "name" : "myMetadata"
                }
             ],
             "methodName" : "nameOfmethod",
             "serviceName" : "serviceA.svc.cluster.local"
          },
          "retryPolicy" : {
             "grpcRetryEvents" : [ "deadline-exceeded" ],
             "httpRetryEvents" : [ "server-error", "gateway-error" ],
             "maxRetries" : 3,
             "perRetryTimeout" : {
```
```

```
                "unit" : "s",
                "value" : 15
             },
             "tcpRetryEvents" : [ "connection-error" ]
          }
       },
       "priority" : 100
    },
    "virtualRouterName" : "routerName"
}
```

3. Ejemplo de código de salida:

```
{
    "route": {
        "meshName": "meshName",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:210987654321:mesh/meshName/
virtualRouter/routerName/route/routeName",
            "createdAt": "2022-04-06T13:48:20.749000-05:00",
            "lastUpdatedAt": "2022-04-06T13:48:20.749000-05:00",
            "meshOwner": "123456789012",
            "resourceOwner": "210987654321",
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 1
        },
        "routeName": "routeName",
        "spec": {
            "grpcRoute": {
                "action": {
                    "weightedTargets": [
                        {
                            "virtualNode": "nodeName",
                            "weight": 100
                        }
                    ]
                },
                "match": {
                    "metadata": [
                        {
                            "invert": false,
                            "match": {
                                "prefix": "123"
                            },
```

```
                            "name": "myMetadata"
                        }
                    ],
                    "methodName": "nameOfMehod",
                    "serviceName": "serviceA.svc.cluster.local"
                },
                "retryPolicy": {
"grpcRetryEvents": [
                        "deadline-exceeded"
                    ],
                    "httpRetryEvents": [
                        "server-error",
                        "gateway-error"
                    ],
                    "maxRetries": 3,
                    "perRetryTimeout": {
                        "unit": "s",
                        "value": 15
                    },
                    "tcpRetryEvents": [
                        "connection-error"
                    ]
                }
            },
            "priority": 100
        },
        "status": {
            "status": "ACTIVE"
        },
        "virtualRouterName": "routerName"
    }
}
```

Para obtener más información sobre cómo crear una ruta con App Mesh, consulta el comando create-route en la AWS CLI referencia. AWS CLI

gRPC

(Opcional) Match

- (Opcional) Escriba el Nombre del servicio de destino para que coincida con la solicitud. Si no especifica un nombre, las solicitudes coincidirán con cualquier servicio.
- (Opcional) Escriba el Nombre del método de destino para que coincida con la solicitud. Si no especifica un nombre, las solicitudes coincidirán con cualquier método. Si especifica un nombre de método, debe especificar un nombre de servicio.

(Opcional) Metadatos

Elija Agregar metadatos.

- (Opcional) Escriba el Nombre de los metadatos en el que desee basar la ruta, seleccione un Tipo de coincidencia e introduzca un Valor de coincidencia. Si selecciona Invertir, coincidirá con lo contrario. Por ejemplo, si especifica un Nombre de los metadatos `myMetadata`, un Tipo de coincidencia exacto, un Valor de coincidencia de `123` y selecciona Invertir, la ruta coincidirá con cualquier solicitud que tenga un nombre de metadatos que comience por un nombre distinto de `123`.
- (Opcional) Seleccione Agregar metadatos para añadir hasta diez elementos de metadatos.

(Opcional) Política de reintentos

Una política de reintentos permite a los clientes protegerse de errores intermitentes de red o errores intermitentes del lado del servidor. Una política de reintentos es opcional, pero recomendable. Los valores del tiempo de espera de los reintentos definen el tiempo de espera por reintento (incluido el intento inicial). Si no define una política de reintentos, App Mesh puede crear automáticamente una política predeterminada para cada una de sus rutas. Para obtener más información, consulte Política de reintentos de ruta predeterminada.

- En Tiempo de espera de reintento, especifique el número de unidades de la duración del tiempo de espera. Se requiere un valor si selecciona cualquier evento de reintento de protocolo.
- En Unidad de tiempo de espera de reintento, seleccione una unidad. Se requiere un valor si selecciona cualquier evento de reintento de protocolo.

- En Número máximo de reintentos especifique el número máximo de reintentos en caso de que se produzca un error en la solicitud. Se requiere un valor si selecciona cualquier evento de reintento de protocolo. Recomendamos un valor de al menos dos.
- Seleccione uno o más Eventos de reintento HTTP. Recomendamos seleccionar al menos stream-error y gateway-error.
- Seleccione un Evento de reintento HTTP.
- Seleccione uno o varios Eventos de reintento de gRPC. Recomendamos seleccionar al menos los eventos cancelados y no disponibles.

(Opcional) Tiempos de espera

- El valor predeterminado es de 15 segundos. Si especificó una Política de reintentos, la duración que indique aquí debe ser siempre mayor o igual a la duración de los reintentos multiplicada por el Número máximo de reintentos que haya definido en Política de reintentos para que la política de reintentos se complete. Si especifica una duración superior a 15 segundos, asegúrese de que el tiempo de espera definido para el oyente de cualquier Destino de nodo virtual también sea superior a 15 segundos. Para obtener más información, consulte Nodos virtuales.
- Un valor de `0` deshabilita el tiempo de espera.
- El periodo tiempo máximo que la ruta puede permanecer inactiva.

HTTP y HTTP/2

(Opcional) Coincidencia

- Especifique el Prefijo con el que debe coincidir la ruta. Por ejemplo, si el nombre de servicio virtual es `service-b.local` y desea que la ruta se empareje con solicitudes para `service-b.local/metrics`, el prefijo debe ser `/metrics`. Especificación de las rutas `/` de todo el tráfico.
- (Opcional) Seleccione un Método.
- (Opcional) Seleccione un Esquema. Aplicable sólo para rutas HTTP2.

(Opcional) Encabezados

- (Opcional) Seleccione Agregar encabezado. Escriba el Nombre de encabezado en función del cual desee realizar la ruta, seleccione un Tipo de coincidencia e introduzca un Valor de coincidencia. Si selecciona Invertir, coincidirá con lo contrario. Por ejemplo, si especifica un encabezado

denominado `clientRequestId` con el Prefijo de `123` y selecciona Invertir, la ruta coincidirá con cualquier solicitud que tenga un encabezado que comience con un nombre distinto de `123`.

- (Opcional) Seleccione Agregar encabezado. Puede agregar hasta diez encabezados.

(Opcional) Política de reintentos

Una política de reintentos permite a los clientes protegerse de errores intermitentes de red o errores intermitentes del lado del servidor. Una política de reintentos es opcional, pero recomendable. Los valores del tiempo de espera de los reintentos definen el tiempo de espera por reintento (incluido el intento inicial). Si no define una política de reintentos, App Mesh puede crear automáticamente una política predeterminada para cada una de sus rutas. Para obtener más información, consulte Política de reintentos de ruta predeterminada.

- En Tiempo de espera de reintento, especifique el número de unidades de la duración del tiempo de espera. Se requiere un valor si selecciona cualquier evento de reintento de protocolo.
- En Unidad de tiempo de espera de reintento, seleccione una unidad. Se requiere un valor si selecciona cualquier evento de reintento de protocolo.
- En Número máximo de reintentos especifique el número máximo de reintentos en caso de que se produzca un error en la solicitud. Se requiere un valor si selecciona cualquier evento de reintento de protocolo. Recomendamos un valor de al menos dos.
- Seleccione uno o más Eventos de reintento HTTP. Recomendamos seleccionar al menos stream-error y gateway-error.
- Seleccione un Evento de reintento de TCP.

(Opcional) Tiempos de espera

- Tiempo de espera de la solicitud: el valor predeterminado es de 15 segundos. Si especificó una Política de reintentos, la duración que indique aquí debe ser siempre mayor o igual a la duración de los reintentos multiplicada por el Número máximo de reintentos que haya definido en Política de reintentos para que la política de reintentos se complete.
- Duración de inactividad: el valor predeterminado es de 300 segundos.
- Un valor de `0` deshabilita el tiempo de espera.

> **Note**
>
> Si especifica un tiempo de espera superior al predeterminado, asegúrese de que el tiempo de espera especificado para el oyente para todos los participantes del nodo virtual también sea superior al predeterminado. Sin embargo, si reduce el tiempo de espera a un valor inferior al predeterminado; opcionalmente, puede actualizar los tiempos de espera en los nodos virtuales. Para obtener más información, consulte Nodos virtuales.

TCP

(Opcional) Tiempos de espera

- Duración de inactividad: el valor predeterminado es de 300 segundos.
- Un valor de `0` deshabilita el tiempo de espera.

Eliminación de una ruta

Consola de administración de AWS

Para eliminar una ruta mediante Consola de administración de AWS

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. Elija la malla en la que desea eliminar una ruta. Se muestran todas las mallas que son de su propiedad y que se han compartido con usted.
3. Elija Routers virtuales en el panel de navegación izquierdo.
4. Elija el enrutador en el que desea eliminar una ruta.
5. En la tabla Rutas, elija la ruta que desee eliminar y seleccione Eliminar en la esquina superior derecha.
6. En el cuadro de confirmación, escriba **`delete`** y, a continuación, haga clic en Eliminar.

AWS CLI

Para eliminar una ruta mediante el AWS CLI

1. Utilice el siguiente comando para eliminar la ruta (sustituya los *`red`* valores por los suyos propios):

```
aws appmesh delete-route \
     --mesh-name meshName \
     --virtual-router-name routerName \
     --route-name routeName
```

2. Ejemplo de código de salida:

```
{
    "route": {
        "meshName": "meshName",
        "metadata": {
            "arn": "arn:aws:appmesh:us-west-2:210987654321:mesh/meshName/
virtualRouter/routerName/route/routeName",
            "createdAt": "2022-04-06T13:46:54.750000-05:00",
            "lastUpdatedAt": "2022-04-07T10:43:57.152000-05:00",
            "meshOwner": "123456789012",
            "resourceOwner": "210987654321",
            "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "version": 2
        },
        "routeName": "routeName",
        "spec": {
            "grpcRoute": {
                "action": {
                    "weightedTargets": [
                        {
                            "virtualNode": "nodeName",
                            "weight": 100
                        }
                    ]
                },
                "match": {
                    "metadata": [
                        {
                            "invert": false,
                            "match": {
                                "prefix": "123"
                            },
                            "name": "myMetadata"
                        }
                    ],
                    "methodName": "methodName",
```

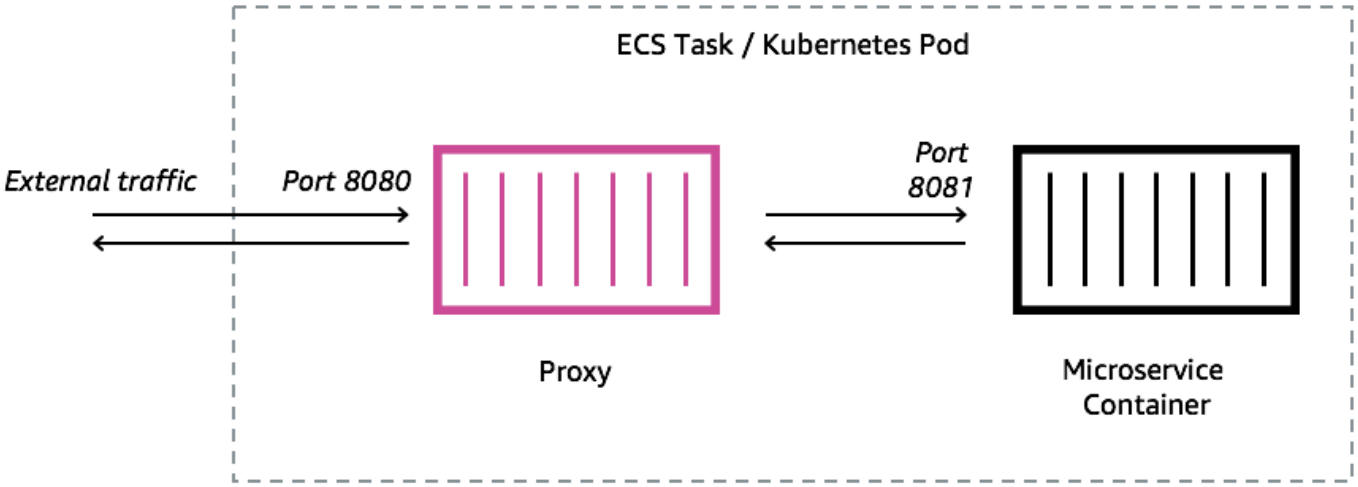
```
                    "serviceName": "serviceA.svc.cluster.local"
                },
                "retryPolicy": {
                    "grpcRetryEvents": [
                        "deadline-exceeded"
                    ],
                    "httpRetryEvents": [
                        "server-error",
                        "gateway-error"
                    ],
                    "maxRetries": 3,
                    "perRetryTimeout": {
                        "unit": "s",
                        "value": 15
                    },
                    "tcpRetryEvents": [
                        "connection-error"
                    ]
                }
            },
            "priority": 100
        },
        "status": {
            "status": "DELETED"
        },
        "virtualRouterName": "routerName"
    }
}
```

Para obtener más información sobre cómo eliminar una ruta con App Mesh, consulta el comando delete-route en la AWS CLI referencia. AWS CLI

Imagen de Envoy

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

AWS App Mesh es una red de servicios basada en el proxy de Envoy.



Debe añadir un proxy de Envoy a la tarea de Amazon ECS, al pod de Kubernetes o a la instancia de Amazon EC2 representada por su punto de conexión de App Mesh, como un nodo virtual o una puerta de enlace virtual. App Mesh vende una imagen de contenedor proxy de Envoy parcheada con las últimas actualizaciones de vulnerabilidad y rendimiento. App Mesh prueba cada nueva versión de proxy de Envoy comparándola con el conjunto de funciones App Mesh antes de poner a tu disposición una nueva imagen.

Variantes de imagen de Envoy

App Mesh ofrece dos variantes de la imagen del contenedor proxy de Envoy. La diferencia entre ambos es la forma en que el proxy de Envoy se comunica con el plano de datos de App Mesh y la forma en que los proxies de Envoy se comunican entre sí. Una es una imagen estándar, que se

comunica con los puntos finales del servicio App Mesh estándar. La otra variante es FIPS-compliant la que se comunica con los puntos finales del servicio FIPS de App Mesh y aplica la criptografía FIPS en la comunicación TLS entre los servicios de App Mesh.

Puede elegir una imagen regional de la lista siguiente o una imagen de nuestro repositorio público denominada `aws-appmesh-envoy`.

Important

- A partir del 30 de junio de 2023, solo la imagen de Envoy `v1.17.2.0-prod` o una versión posterior será compatible con App Mesh. En el caso de los clientes actuales que hayan utilizado anteriormente una imagen de Envoy`v1.17.2.0`, aunque las imágenes de Envoys existentes seguirán siendo compatibles, recomendamos encarecidamente migrar a la versión más reciente.
- Como práctica recomendada, se recomienda encarecidamente actualizar la versión de Envoy a la última versión de forma periódica. Solo se valida la versión más reciente de Envoy con los parches de seguridad, las versiones de funciones y las mejoras de rendimiento más recientes.
- La versión `1.17` fue una actualización importante de Envoy. Consulte Updating/migrating la versión 1.17 de Envoy para obtener más información.
- La versión `1.20.0.1` o posterior es compatible con `ARM64`.
- Para obtener asistencia para `IPv6`, se requiere la versión `1.20` o posterior de Envoy.

Note

El FIPS solo está disponible en las regiones de EE. UU. y Canadá.

Todas las regiones compatibles se pueden *`Region-code`* sustituir por cualquier región que no sea `me-south-1`ap-east-1,`ap-southeast-3`, `eu-south-1`il-central-1, `yaf-south-1`.

Standard

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

FIPS-compliant

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod-
fips
```

me-south-1

Standard

```
772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

ap-east-1

Standard

```
856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

ap-southeast-3

Standard

```
909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

eu-south-1

Standard

```
422531588944.dkr.ecr.eu-south-1.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

il-central-1

Standard

```
564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

af-south-1

Standard

```
924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

`Public repository`

Standard

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.34.13.1-prod
```

FIPS-compliant

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.34.13.1-prod-fips
```

> **Note**
>
> Recomendamos asignar 512 unidades de CPU y al menos 64 MiB de memoria al contenedor de Envoy. En Fargate, la cantidad mínima de memoria que puede establecer es de 1024 MiB. La asignación de recursos al contenedor de Envoy se puede aumentar si los datos del contenedor u otras métricas indican que los recursos son insuficientes debido a una carga mayor.

> **Note**
>
> Todas las versiones de lanzamiento de la imagen `aws-appmesh-envoy` a partir de la `v1.22.0.0` están diseñadas como una imagen de Docker sin distribución. Hicimos este cambio para poder reducir el tamaño de la imagen y reducir nuestra exposición a las vulnerabilidades en los paquetes no utilizados presentes en la imagen. Si está creando a partir de la imagen aws-appmesh-envoy y cuenta con algunas de las funcionalidades y paquetes base de AL2 (por ejemplo, yum), le sugerimos que copie los archivos binarios del interior de una imagen `aws-appmesh-envoy` para crear una nueva imagen de Docker con base en AL2.
>
> Ejecute este script para generar una imagen de Docker personalizada con la etiqueta `aws-appmesh-envoy:v1.22.0.0-prod-al2:`
>
> ```
> cat << EOF > Dockerfile
> FROM public.ecr.aws/appmesh/aws-appmesh-envoy:v1.22.0.0-prod as envoy
>
> FROM public.ecr.aws/amazonlinux/amazonlinux:2
> RUN yum -y update && \
>     yum clean all && \
> ```

```
    rm -rf /var/cache/yum

COPY --from=envoy /usr/bin/envoy /usr/bin/envoy
COPY --from=envoy /usr/bin/agent /usr/bin/agent
COPY --from=envoy /aws_appmesh_aggregate_stats.wasm /
aws_appmesh_aggregate_stats.wasm

CMD [ "/usr/bin/agent" ]
EOF

docker build -f Dockerfile -t aws-appmesh-envoy:v1.22.0.0-prod-al2 .
```

El acceso a esta imagen de contenedor en Amazon ECR lo controla AWS Identity and Access Management (IAM). Por lo tanto, debe usar IAM para comprobar que tiene acceso de lectura a Amazon ECR. Por ejemplo, al utilizar Amazon ECS, puede asignar un rol de ejecución de tareas adecuado a una tarea de Amazon ECS. Si utiliza políticas de IAM que limitan el acceso a recursos de Amazon ECR específicos, asegúrese de que permite el acceso al nombre de recurso de Amazon (ARN) específico de la región que identifica al repositorio `aws-appmesh-envoy`. Por ejemplo, en la región `us-west-2`, permite el acceso al siguiente recurso: `arn:aws:ecr:us-west-2:840364872350:repository/aws-appmesh-envoy`. Para obtener más información, consulte Políticas administradas de Amazon ECR. Si utiliza Docker en una instancia de Amazon EC2, autentique Docker en el repositorio. Para obtener más información, consulte Autenticación del registro.

De vez en cuando lanzamos nuevas características de App Mesh que dependen de los cambios de Envoy que aún no se han fusionado con las imágenes originales de Envoy. Para usar estas nuevas funciones de App Mesh antes de que los cambios de Envoy se fusionen previamente, debes usar la imagen del contenedor de Mesh-vended App Envoy. Para ver una lista de cambios, consulta la hoja de GitHubruta de App Mesh sobre los problemas relacionados con la `Envoy Upstream` etiqueta. Recomendamos que utilice la imagen del contenedor de App Mesh Envoy como la mejor opción compatible.

Variables de configuración de Envoy

Important

Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte a. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App

Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Use las siguientes variables de entorno para configurar los contenedores de Envoy para sus grupos de tareas de nodos virtuales de App Mesh.

> **Note**
>
> App Mesh Envoy 1.17 no es compatible con la API xDS v2 de Envoy. Si utiliza variables de configuración de Envoy que aceptan archivos de configuración de Envoy, debe actualizarlas a la última versión v3 xDS de la API.

Variables obligatorias

La siguiente variable de entorno es necesaria para todos los contenedores de App Mesh Envoy. Esta variable solo se puede usar con la versión `1.15.0` o posterior de la imagen de Envoy. Si utiliza una versión anterior de la imagen, debe configurar la variable `APPMESH_VIRTUAL_NODE_NAME` en su lugar.

`APPMESH_RESOURCE_ARN`

Al agregar el contenedor de Envoy a un grupo de tareas, establezca esta variable de entorno en el ARN del nodo virtual o la puerta de enlace virtual que representa el grupo de tareas. La lista siguiente contiene ARN de ejemplo:

- Nodo virtual: arn:aws:appmesh: ::mesh/ /VirtualNode/ *Region-code 111122223333 meshName virtualNodeName*
- Puerta de enlace virtual: arn:aws:appmesh: *Region-code* ::mesh/ /VirtualGateway/ *111122223333 meshName virtualGatewayName*

Variables opcionales

La siguiente variable de entorno es opcional para los contenedores de App Mesh Envoy.

`ENVOY_LOG_LEVEL`

Especifica el nivel de registro del contenedor de Envoy.

Valores válidos: `trace`, `debug`, `info`, `warn`, `error`, `critical`, `off`

Valor predeterminado: `info`

`ENVOY_INITIAL_FETCH_TIMEOUT`

Especifica el tiempo que Envoy espera la primera respuesta de configuración del servidor de administración durante el proceso de inicialización.

Para obtener más información, consulte Orígenes de configuración en la documentación de Envoy. Si se establece en `0`, no hay tiempo de espera.

Valor predeterminado: `0`

`ENVOY_CONCURRENCY`

Establece la opción de línea de comandos `--concurrency` al iniciar Envoy. No se establece de forma predeterminada. Esta opción está disponible desde la versión `v1.24.0.0-prod` o superior de Envoy.

Para obtener más información, consulte Opciones de línea de comandos en la documentación de Envoy.

Variables de administración

Utilice estas variables de entorno para configurar la interfaz administrativa de Envoy.

`ENVOY_ADMIN_ACCESS_PORT`

Especifique un puerto de administración personalizado en el que Envoy pueda escuchar. Predeterminado: `9901`.

> **Note**
>
> El puerto de administración de Envoy debe ser diferente de cualquier puerto de escucha de la puerta de enlace virtual o el nodo virtual

`ENVOY_ADMIN_ACCESS_LOG_FILE`

Especifique una ruta personalizada en la que escribir los registros de acceso de Envoy. Predeterminado: `/tmp/envoy_admin_access.log`.

`ENVOY_ADMIN_ACCESS_ENABLE_IPV6`

Activa o desactiva la interfaz de administración de Envoy para que acepte tráfico `IPv6`, lo que permite que esta interfaz acepte tanto tráfico `IPv4` como `IPv6`. De forma predeterminada, este indicador está establecido en falso y Envoy solo escucha tráfico `IPv4`. Esta variable solo se puede usar con la versión 1.22.0 o posterior de la imagen de Envoy.

Variables del agente

Utilice estas variables de entorno para configurar el AWS App Mesh agente para Envoy. Para obtener más información, consulte Agente para Envoy de App Mesh.

`APPNET_ENVOY_RESTART_COUNT`

Especifica el número de veces que el agente reiniciará el proceso del proxy de Envoy dentro de una tarea o pod en ejecución si se cierra. El agente también registra el estado de cierre cada vez que se cierra Envoy para facilitar la solución de problemas. El valor predeterminado de esta variable es `0`. Cuando se establece el valor predeterminado, el agente no intenta reiniciar el proceso.

Valor predeterminado: `0`

Máximo: `10`

`PID_POLL_INTERVAL_MS`

Especifica el intervalo en milisegundos durante el cual el agente comprueba el estado del proceso del proxy de Envoy. El valor predeterminado es `100`.

Valor predeterminado: `100`

Mínimo: `100`

Máximo: `1000`

`LISTENER_DRAIN_WAIT_TIME_S`

Especifica el tiempo en segundos durante el cual el proxy de Envoy espera a que se cierren las conexiones activas antes de finalizar el proceso.

Valor predeterminado: `20`

Mínimo: `5`

Máximo: `110`

`APPNET_AGENT_ADMIN_MODE`

Inicia el servidor de la interfaz de administración del agente y lo enlaza a una dirección tcp o un socket Unix.

Valores válidos: `tcp`, `uds`

`APPNET_AGENT_HTTP_PORT`

Especifique el puerto que se utilizará para enlazar la interfaz de administración del agente en el modo `tcp`. Asegúrese de que el valor del puerto sea > `1024` si `uid` != `0`. Asegúrese de que el puerto sea menor que `65535`.

Valor predeterminado: `9902`

`APPNET_AGENT_ADMIN_UDS_PATH`

Especifique la ruta del socket de dominio de Unix para la interfaz de administración del agente en el modo `uds`.

Valor predeterminado: `/var/run/ecs/appnet_admin.sock`

Variables de rastreo

Puede configurar uno o ninguno de los siguientes controladores de rastreo.

AWS X-Ray variables

Use las siguientes variables de entorno para configurar App Mesh con AWS X-Ray. Para obtener más información, consulte la Guía para desarrolladores de AWS X-Ray.

`ENABLE_ENVOY_XRAY_TRACING`

Permite X-Ray el rastreo `127.0.0.1:2000` como punto final daemon predeterminado. Para habilitarlo, establezca el valor en `1`. El valor predeterminado es `0`.

`XRAY_DAEMON_PORT`

Especifique un valor de puerto para anular el puerto X-Ray daemon predeterminado:. `2000`

`XRAY_SAMPLING_RATE`

Especifique una frecuencia de muestreo para anular la frecuencia de muestreo predeterminada del X-Ray rastreador, que es del `0.05` 5%. Especifique el valor como un decimal entre `0` y `1.00`

(100 %). Este valor se anula si se especifica `XRAY_SAMPLING_RULE_MANIFEST`. Esta variable es compatible con las imágenes de Envoy de la versión `v1.19.1.1-prod` y posterior.

`XRAY_SAMPLING_RULE_MANIFEST`

Especifique una ruta de archivo en el sistema de archivos contenedores de Envoy para configurar las reglas de muestreo personalizadas y localizadas del X-Ray rastreador. Para obtener más información, consulte Reglas de muestreo en la Guía para desarrolladores de AWS X-Ray . Esta variable es compatible con las imágenes de Envoy de la versión `v1.19.1.0-prod` y posterior.

`XRAY_SEGMENT_NAME`

Especifique un nombre de segmento para las trazas para anular el nombre de X-Ray segmento predeterminado. De forma predeterminada, este valor se establecerá en `mesh/resourceName`. Esta variable es compatible con la versión `v1.23.1.0-prod` o posterior de la imagen de Envoy.

Variables de rastreo de Datadog

Las siguientes variables de entorno lo ayudan a configurar App Mesh con el rastreador de agentes de Datadog. Para obtener más información, consulte Configuración del agente en la documentación de Datadog.

`ENABLE_ENVOY_DATADOG_TRACING`

Habilita la recopilación de rastros de Datadog utilizando `127.0.0.1:8126` como punto de conexión predeterminado del agente de Datadog. Para habilitarla, establezca el valor en `1` (el valor predeterminado es `0`).

`DATADOG_TRACER_PORT`

Especifique un valor de puerto para anular el puerto predeterminado del agente de Datadog: `8126`.

`DATADOG_TRACER_ADDRESS`

Especifique una dirección IP para anular la dirección predeterminada del agente de Datadog: `127.0.0.1`.

`DD_SERVICE`

Especifique un nombre de servicio para los rastreos a fin de anular el nombre de servicio predeterminado de Datadog: `envoy-meshName/virtualNodeName`. Esta variable es compatible con las imágenes de Envoy de la versión `v1.18.3.0-prod` y posterior.

Variables de rastreo de Jaeger

Utilice las siguientes variables de entorno para configurar App Mesh con el rastreo de Jaeger. Para obtener más información, consulte Introducción en la documentación de Jaeger. Estas variables son compatibles con las imágenes de Envoy de la versión `1.16.1.0-prod` y posterior.

`ENABLE_ENVOY_JAEGER_TRACING`

Habilita la recopilación de rastros de Jaeger utilizando `127.0.0.1:9411` como punto de conexión predeterminado de Jaeger. Para habilitarla, establezca el valor en `1` (el valor predeterminado es`0`).

`JAEGER_TRACER_PORT`

Especifique un valor de puerto para anular el puerto predeterminado de Jaeger: `9411`.

`JAEGER_TRACER_ADDRESS`

Especifique una dirección IP para anular la dirección predeterminada de Jaeger: `127.0.0.1`.

`JAEGER_TRACER_VERSION`

Especifique si el recopilador necesita rastreos en formato `JSON` o `PROTO` codificado. De forma predeterminada, este valor se establecerá en `PROTO`. Esta variable es compatible con la versión `v1.23.1.0-prod` o posterior de la imagen de Envoy.

Variable de rastreo de Envoy

Defina la siguiente variable de entorno para usar su propia configuración de rastreo.

`ENVOY_TRACING_CFG_FILE`

Especifique una ruta de archivo del sistema de archivos del contenedor de Envoy. Para obtener más información, consulte `config.trace.v3.Tracing` en la documentación de Envoy.

> **Note**
>
> Si la configuración de rastreo requiere especificar un clúster de rastreo, asegúrese de que define la configuración del clúster asociado bajo `static_resources` en el mismo archivo de configuración de rastreo. Por ejemplo, Zipkin tiene un campo `collector_cluster` para el nombre de clúster que aloja los recopiladores de rastreos y dicho clúster debe definirse de forma estática.

DogStatsVariables D

Usa las siguientes variables de entorno para configurar App Mesh con DogStats D. Para obtener más información, consulte la documentación de DogStatsD.

`ENABLE_ENVOY_DOG_STATSD`

Habilita las estadísticas DogStats D `127.0.0.1:8125` como punto final daemon predeterminado. Para habilitarlas, establezca el valor en 1.

`STATSD_PORT`

Especifique un valor de puerto para anular el puerto del daemon DogStats D predeterminado.

`STATSD_ADDRESS`

Especifique un valor de dirección IP para anular la dirección IP predeterminada del DogStats daemon D. Predeterminado: `127.0.0.1`. Esta variable solo puede usarse con la versión `1.15.0` o posterior de la imagen de Envoy.

`STATSD_SOCKET_PATH`

Especifique un socket de dominio Unix para el daemon D. DogStats Si no se especifica esta variable y DogStats D está habilitada, este valor se establece de forma predeterminada en el puerto de direcciones IP del daemon DogStats D. `127.0.0.1:8125` Si se especifica que la `ENVOY_STATS_SINKS_CFG_FILE` variable contiene una configuración de sumideros de estadísticas, anula todas las variables D. DogStats Esta variable es compatible con la versión `v1.19.1.0-prod` o posterior de la imagen de Envoy.

Variables de App Mesh

Las siguientes variables lo ayudan a configurar App Mesh.

`APPMESH_RESOURCE_CLUSTER`

De forma predeterminada, App Mesh utiliza el nombre del recurso especificado en `APPMESH_RESOURCE_ARN` cuando Envoy hace referencia a sí mismo en métricas y rastreos. Puede anular este comportamiento estableciendo la variable de entorno `APPMESH_RESOURCE_CLUSTER` con su propio nombre. Esta variable solo puede usarse con la versión `1.15.0` o posterior de la imagen de Envoy.

`APPMESH_METRIC_EXTENSION_VERSION`

Establezca el valor en 1 para habilitar la extensión de métricas de App Mesh. Para obtener más información acerca del uso de la extensión de métricas de App Mesh, consulte Extensión de métricas de App Mesh.

`APPMESH_DUALSTACK_ENDPOINT`

Establezca el valor en 1 para conectar al punto de conexión de pila doble de App Mesh. Cuando este indicador está establecido, Envoy usa nuestro dominio compatible con pila doble. De forma predeterminada, este indicador está establecido en falso y solo se conecta a nuestro dominio `IPv4`. Esta variable solo se puede usar con la versión 1.22.0 o posterior de la imagen de Envoy.

Variables de Envoy Stats

Utilice las siguientes variables de entorno para configurar App Mesh con Envoy Stats. Para obtener más información, consulte la documentación de Envoy Stats.

`ENABLE_ENVOY_STATS_TAGS`

Permite el uso de las etiquetas definidas por App Mesh `appmesh.mesh` y `appmesh.virtual_node`. Para obtener más información, consulte config.metrics.v3. TagSpecifieren la documentación de Envoy. Para habilitarlas, establezca el valor en 1.

`ENVOY_STATS_CONFIG_FILE`

Especifique una ruta de archivo del sistema de archivos del contenedor de Envoy para reemplazar el archivo de configuración predeterminado de etiquetas de estadísticas por el suyo propio. Para obtener más información, consulte config.metrics.v3. StatsConfig.

> **Note**
>
> Si se establece una configuración de estadísticas personalizada que incluya filtros de estadísticas, Envoy podría entrar en un estado en el que ya no se sincronice correctamente con el estado mundial de App Mesh. Se trata de un error de Envoy. Nuestra recomendación es no filtrar las estadísticas en Envoy. Si el filtrado es absolutamente necesario, hemos incluido un par de soluciones alternativas para este problema en nuestra hoja de ruta.

`ENVOY_STATS_SINKS_CFG_FILE`

Especifique una ruta de archivo del sistema de archivos del contenedor de Envoy para sustituir la configuración predeterminada por la suya propia. Para obtener más información, consulte config.metrics.v3. StatsSinken la documentación de Envoy.

Variables obsoletas

Las variables de entorno `APPMESH_VIRTUAL_NODE_NAME` y `APPMESH_RESOURCE_NAME` ya no son compatibles con la versión `1.15.0` o posterior de Envoy. Sin embargo, siguen siendo compatibles con las mallas existentes. En lugar de usar estas variables con la versión `1.15.0` o posterior de Envoy, use `APPMESH_RESOURCE_ARN` para todos los puntos de conexión de App Mesh.

Valores predeterminados de Envoy establecidos por App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Las siguientes secciones proporcionan información sobre los valores predeterminados de Envoy para la política de reintentos de ruta y el interruptor establecidos por App Mesh.

Política de reintentos de ruta predeterminada

Si no tenías ninguna malla en tu cuenta antes del 29 de julio de 2020, App Mesh crea automáticamente una política de reintentos de ruta de Envoy predeterminada para todas las solicitudes HTTP y gRPC en cualquier malla de tu cuenta a partir del 29 de julio de 2020. HTTP/2 Si tenía alguna malla en su cuenta antes del 29 de julio de 2020, no se creó ninguna política predeterminada para las rutas de Envoy que existieran antes del 29 de julio de 2020 o después. Esto es, a menos que abras un ticket con el servicio de asistencia. AWS Una vez que el equipo de soporte procesa el ticket, se crea la política predeterminada para cualquier ruta futura de Envoy que App Mesh cree en la fecha en que se procesó el ticket o después de dicha fecha. Para obtener

más información sobre las políticas de reintento de rutas de Envoy, consulte config.route.v3.RetryPolicyen la documentación de Envoy.

App Mesh crea una ruta de Envoy cuando se crea una ruta de App Mesh o se define un proveedor de nodos virtuales para un servicio virtual de App Mesh. Aunque puede crear una política de reintentos de ruta de App Mesh, no puede crear una política de reintentos de App Mesh para un proveedor de nodos virtuales.

La política predeterminada no es visible a través de la API de App Mesh. La política predeterminada solo es visible a través de Envoy. Para ver la configuración, debe habilitar la interfaz de administración y enviar una solicitud a Envoy para recibir un `config_dump`. La política predeterminada incluye las siguientes opciones:

- Número máximo de reintentos: 2
- Eventos de reintento de gRPC: `UNAVAILABLE`
- Eventos de reintento HTTP: `503`

> **Note**
>
> No es posible crear una política de reintentos de ruta de App Mesh que busque un código de error HTTP específico. Sin embargo, una política de reintentos de ruta de App Mesh puede buscar `server-error` o`gateway-error`. Ambos incluyen errores `503`. Para obtener más información, consulte Rutas.

- Evento de reintento TCP: `connect-failure` y `refused-stream`

> **Note**
>
> No es posible crear una política de reintentos de ruta de App Mesh que busque cualquiera de estos eventos. Sin embargo, se puede buscar una política de reintentos de ruta de App Mesh para `connection-error`, que equivale a `connect-failure`. Para obtener más información, consulte Rutas.

- Restablecer: Envoy intenta volver a intentarlo si el servidor principal no responde en absoluto (disconnect/reset/se agota el tiempo de espera de lectura).

Interruptor predeterminado

Cuando implementa un Envoy en App Mesh, se establecen los valores predeterminados de Envoy para algunos de los ajustes del interruptor. Para obtener más información, consulte clúster. CircuitBreakers.Thresholdsen la documentación de Envoy. Estos ajustes no se pueden ver a través de la API de App Mesh. Los ajustes solo se pueden ver a través de Envoy. Para ver los ajustes, debe habilitar la interfaz de administración y enviar una solicitud a Envoy para recibir un `config_dump`.

Si no tenía mallas en su cuenta antes del 29 de julio de 2020, por cada Envoy que implemente en una malla creada el 29 de julio de 2020 o después, App Mesh desactiva de forma efectiva los interruptores, cambiando los valores predeterminados de Envoy para los siguientes ajustes. Si tenías alguna malla en tu cuenta antes del 29 de julio de 2020, los valores predeterminados de Envoy se establecen para cualquier Envoy que despliegues en App Mesh a partir del 29 de julio de 2020, a menos que abras un ticket con el servicio de AWS asistencia. Una vez que el equipo de soporte procese el ticket, App Mesh establecerá los valores predeterminados de App Mesh de los siguientes ajustes de Envoy en todos los Envoys que implemente después de la fecha de procesamiento del ticket:

- **`max_requests`** – 2147483647
- **`max_pending_requests`** – 2147483647
- **`max_connections`** – 2147483647
- **`max_retries`** – 2147483647

> **Note**
>
> No importa si sus Envoys tienen los valores de interruptor predeterminados de Envoy o App Mesh, no podrá modificarlos.

Updating/migrating a Envoy 1.17

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte a. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App

Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Secret Discovery Service con SPIRE

Si utiliza SPIRE (Entorno en tiempo de ejecución de SPIFFE) con App Mesh para distribuir certificados de confianza a sus servicios, compruebe que utiliza al menos una versión `0.12.0` del agente de SPIRE (publicado en diciembre de 2020). Esta es la primera versión que es compatible con las versiones de Envoy posteriores a la `1.16`.

Cambios de las expresiones regulares

A partir de Envoy `1.17`, App Mesh configura Envoy para que utilice el motor de expresiones regulares RE2 de forma predeterminada. Este cambio es evidente para la mayoría de los usuarios, pero las coincidencias de rutas o rutas de puerta de enlace ya no permiten las referencias retrospectivas o prospectivas en las expresiones regulares.

Referencia prospectiva positiva y negativa

Positiva: una referencia prospectiva positiva es una expresión entre paréntesis que comienza con `?=`:

```
(?=example)
```

Son las que tienen más utilidad cuando se reemplaza una cadena porque permiten hacer coincidir una cadena sin utilizar los caracteres que forman parte de la coincidencia. Debido a que App Mesh no admite la sustitución de cadenas de expresiones regulares, recomendamos que las sustituya por coincidencias normales.

```
(example)
```

Negativa: una anticipación negativa es una expresión entre paréntesis que comienza con `?!`.

```
ex(?!amp)le
```

Las expresiones entre paréntesis se utilizan para afirmar que parte de la expresión no coincide con una entrada determinada. En la mayoría de los casos, puede sustituirlas por un cuantificador cero.

```
ex(amp){0}le
```

Si la propia expresión es una clase de caracteres, puede anular toda la clase y marcarla como opcional utilizando ?.

```
prefix(?![0-9])suffix => prefix[^0-9]?suffix
```

Dependiendo de su caso de uso, es posible que también pueda cambiar sus rutas para solucionar este problema.

```
{
    "routeSpec": {
        "priority": 0,
        "httpRoute": {
            "match": {
                "headers": [
                    {
                        "name": "x-my-example-header",
                        "match": {
                            "regex": "^prefix(?!suffix)"
                        }
                    }
                ]
            }
        }
    }
}

{
    "routeSpec": {
        "priority": 1,
        "httpRoute": {
            "match": {
                "headers": [
                    {
                        "name": "x-my-example-header",
                        "match": {
                            "regex": "^prefix"
                        }
                    }
                ]
            }
```

```
        }
    }
}
```

La primera coincidencia de rutas busca un encabezado que comience con el “prefijo“ pero no seguido del “sufijo“. La segunda ruta busca todos los demás encabezados que comienzan con el “prefijo“, incluidos los que terminan en el “sufijo“. En su lugar, también se pueden invertir para eliminar la referencia prospectiva negativa.

```
{
    "routeSpec": {
        "priority": 0,
        "httpRoute": {
            "match": {
                "headers": [
                    {
                        "name": "x-my-example-header",
                        "match": {
                            "regex": "^prefix.*?suffix"
                        }
                    }
                ]
            }
        }
    }
}

{
    "routeSpec": {
        "priority": 1,
        "httpRoute": {
            "match": {
                "headers": [
                    {
                        "name": "x-my-example-header",
                        "match": {
                            "regex": "^prefix"
                        }
                    }
                ]
            }
        }
    }
```

```
}
```

En este ejemplo, se invierten las rutas para dar mayor prioridad a los encabezados que terminan con “sufijo“, y todos los demás encabezados que comienzan con “prefijo“ coinciden en la ruta de menor prioridad.

Referencias retrospectivas

Una referencia retrospectiva es una forma de escribir expresiones más cortas, repitiéndolas a un grupo anterior entre paréntesis. Tienen este formato.

```
(group1)(group2)\1
```

Una barra invertida \ seguida de un número actúa como marcador de posición para el enésimo grupo entre paréntesis de la expresión. En este ejemplo, `\1` se utiliza como una forma alternativa de escribir `(group1)` por segunda vez.

```
(group1)(group2)(group1)
```

Se pueden eliminar simplemente sustituyendo la referencia retrospectiva por el grupo al que se hace referencia, como en el ejemplo.

Agente para Envoy

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

El agente es un administrador de procesos dentro de la imagen de Envoy que se vende para App Mesh. El agente garantiza que Envoy siga funcionando, se mantenga en buen estado y reduzca el tiempo de inactividad. Filtra las estadísticas y los datos auxiliares de Envoy para ofrecer una visión depurada del funcionamiento del proxy de Envoy en App Mesh. Esto puede ayudarlo a solucionar los errores relacionados con mayor rapidez.

Puede usar el agente para configurar el número de veces que desea reiniciar el proxy de Envoy en caso de que el proxy deje de funcionar correctamente. Si se produce un error, el agente registra el estado de salida definitivo cuando se cierra Envoy. Puede usarlo para solucionar el error. El agente también facilita el drenaje de conexiones de Envoy, lo que ayuda a que sus aplicaciones sean más resistentes a los errores.

Configure el Agente para Envoy mediante estas variables:

- `APPNET_ENVOY_RESTART_COUNT`: cuando esta variable se establece en un valor distinto de cero, el agente intenta reiniciar el proceso de proxy de Envoy hasta el número que usted haya establecido cuando, al sondear, considera que el estado del proceso del proxy no es correcto. Esto ayuda a reducir el tiempo de inactividad al permitir un reinicio más rápido en comparación con la sustitución de una tarea o un pod por parte del orquestador de contenedores en caso de que se produzca un error en la comprobación de estado del proxy.
- `PID_POLL_INTERVAL_MS`: al configurar esta variable, el valor predeterminado se mantiene en `100`. Si se establece en este valor, el proceso de Envoy se detecta y reinicia más rápidamente cuando se cierra, en comparación con la sustitución de una tarea o un pod mediante las comprobaciones de estado del orquestador de contenedores.
- `LISTENER_DRAIN_WAIT_TIME_S`: al configurar esta variable, tenga en cuenta el tiempo de espera del orquestador de contenedores establecido para detener la tarea o el pod. Por ejemplo, si este valor es superior al tiempo de espera del orquestador, el proxy de Envoy solo podrá drenarse mientras el orquestador detenga forzosamente la tarea o el pod.
- `APPNET_AGENT_ADMIN_MODE`: cuando esta variable se establece en `tcp` o `uds`, el agente proporciona una interfaz de administración local. Esta interfaz de administración sirve como punto final seguro para interactuar con el proxy de Envoy y proporciona lo siguiente APIs para los controles de estado y los datos de telemetría, además de resumir el estado de funcionamiento del proxy.
  - `GET /status`: consulta las estadísticas de Envoy y devuelve la información del servidor.
  - `POST /drain_listeners`: drena todos los oyentes de entrada.
  - `POST /enableLogging?level=<desired_level>`: cambia el nivel de registro de Envoy en todos los registradores.
  - `GET /stats/prometheus`: muestra las estadísticas de Envoy en formato Prometheus.
  - `GET /stats/prometheus?usedonly`: muestra solo las estadísticas que Envoy ha actualizado.

Para obtener más información sobre las variables de configuración del agente, consulte Variables de configuración de Envoy.

El nuevo AWS App Mesh agente se incluye en las imágenes de Envoy optimizadas para App Mesh a partir de la versión `1.21.0.0` y no requiere la asignación de recursos adicionales en las tareas o módulos de los clientes.

Observabilidad de App Mesh

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Una de las ventajas de trabajar con App Mesh es una mayor visibilidad de las aplicaciones de microservicios. App Mesh puede funcionar con muchas soluciones diferentes de registro, métricas y rastreo.

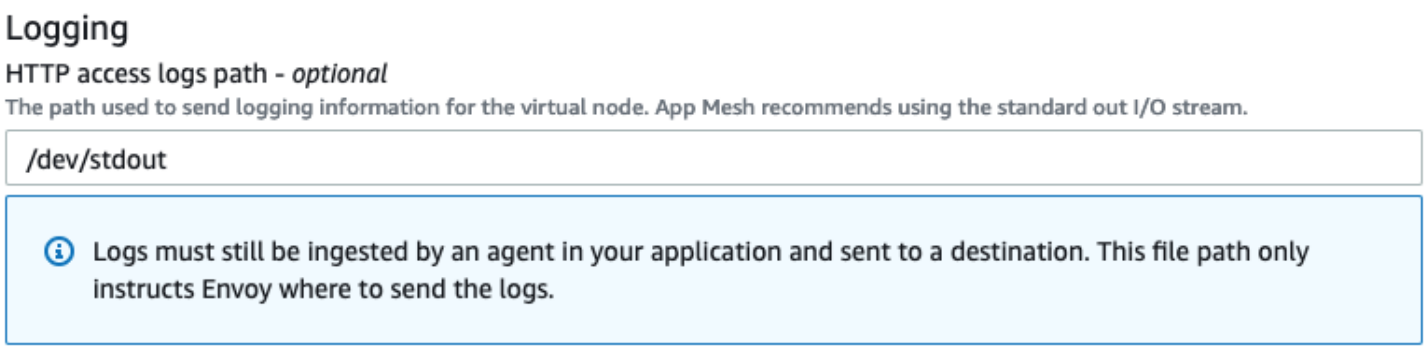
El proxy de Envoy y App Mesh ofrecen los siguientes tipos de herramientas que lo ayudan a obtener una visión más clara de sus aplicaciones y proxies:

- Registro
- Métricas
- Rastreo

Registro

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Al crear sus nodos virtuales y puertas de enlace virtuales, tiene la opción de configurar los registros de acceso de Envoy. En la consola, se encuentra en la sección Registro del nodo virtual y la puerta de enlace virtual para crear o editar flujos de trabajo.



La imagen anterior muestra una ruta de registro de `/dev/stdout` para los registros de acceso de Envoy.

En `format`, especifique uno de los dos formatos posibles, `json` o `text` y el patrón. `json` toma los pares de claves y los transforma en una estructura JSON antes de pasarlos a Envoy.

El siguiente bloque de código muestra la representación en JSON que puede usar en la AWS CLI.

```
      "logging": {
         "accessLog": {
            "file": {
               "path": "/dev/stdout",
                "format" : {
                    // Exactly one of json or text should be specified
                    "json": [ // json will be implemented with key pairs
                        {
                            "key": "string",
                            "value": "string"
                        }
                    ]
                    "text": "string" //e.g. "%LOCAL_REPLY_BODY%:%RESPONSE_CODE%:path=
%REQ(:path)%\n"
                }
            }
         }
      }
```

> **Important**
>
> Asegúrese de comprobar que el patrón de entrada es válido para Envoy, o Envoy rechazará la actualización y guardará los cambios más recientes en `error state`.

Cuando envía los registros de acceso de Envoy a `/dev/stdout`, se mezclan con los registros del contenedor de Envoy. Puede exportarlos a un servicio de almacenamiento y procesamiento de CloudWatch registros, como Logs, utilizando controladores de registro estándar de Docker, como. `awslogs` Para obtener más información, consulte Uso del controlador de registros awslogs en la Guía para desarrolladores de Amazon ECS. Para exportar solo los registros de acceso de Envoy (e ignorar los demás registros del contenedor de Envoy), puede establecer `ENVOY_LOG_LEVEL` en `off`. Puede registrar la solicitud sin una cadena de consulta si incluye la cadena de formato `%REQ_WITHOUT_QUERY(X?Y):Z%`. Para ver ejemplos, consulte Formateador `ReqWithoutQuery`. Para obtener más información, consulte Registro de acceso en la documentación de Envoy.

Habilitar registros de acceso en Kubernetes

Al usar el Controlador de App Mesh para Kubernetes, puede configurar nodos virtuales con registro de acceso añadiendo la configuración de registro a la especificación del nodo virtual, como se muestra en el siguiente ejemplo.

```
---
apiVersion: appmesh.k8s.aws/v1beta2
kind: VirtualNode
metadata:
  name: virtual-node-name
  namespace: namespace
spec:
  listeners:
    - portMapping:
        port: 9080
        protocol: http
  serviceDiscovery:
    dns:
      hostName: hostname
  logging:
    accessLog:
      file:
        path: "/dev/stdout"
```

El clúster debe tener un reenviador de registros para recopilar los registros, por ejemplo, Fluentd. Para obtener más información, consulte Configurar Fluentd como DaemonSet para enviar registros a Logs. CloudWatch

Envoy también escribe varios registros de depuración a partir de sus filtros en `stdout`. Estos registros son útiles para obtener información sobre la comunicación de Envoy con App Mesh y el

tráfico de servicio a servicio. Su nivel de registro específico se puede configurar mediante la variable de entorno `ENVOY_LOG_LEVEL`. Por ejemplo, el texto siguiente proviene de un ejemplo de registro de depuración que muestra el clúster que Envoy identificó para una solicitud HTTP concreta.

```
[debug][router] [source/common/router/router.cc:434] [C4][S17419808847192030829]
 cluster 'cds_ingress_howto-http2-mesh_color_client_http_8080' match for URL '/ping'
```

Firelens y Cloudwatch

Firelens es un enrutador de registros de contenedores que puede utilizar para recopilar registros para Amazon ECS y. AWS Fargate Puede encontrar un ejemplo del uso de Firelens en nuestro Repositorio de muestras de AWS.

Puede utilizarlo CloudWatch para recopilar información de registro y métricas. Puedes encontrar más información CloudWatch en nuestra sección Exportación de métricas de los documentos de App Mesh.

Monitorización de su aplicación mediante métricas de Envoy

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

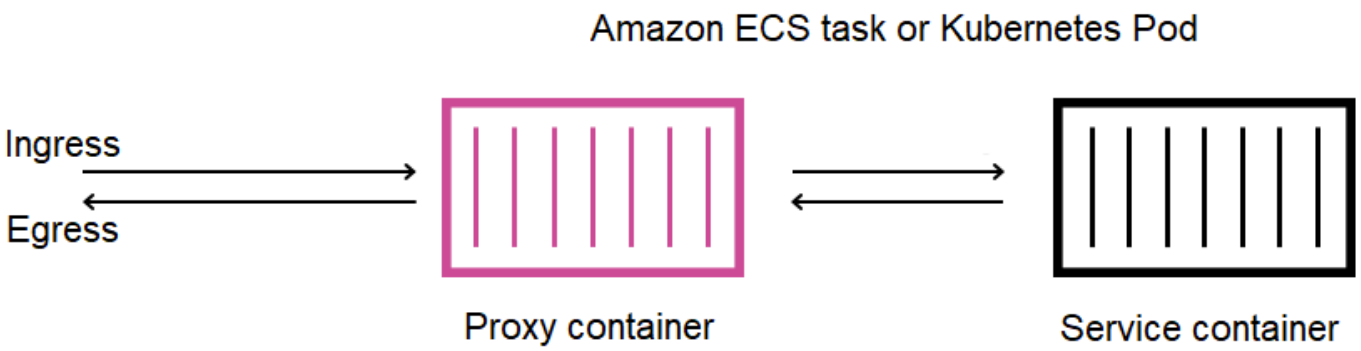
Envoy clasifica sus métricas en las siguientes categorías principales:

- Descendentes: métricas relacionadas con las conexiones y solicitudes que entrar en el proxy.
- Ascendentes: métricas relacionadas con las conexiones salientes y las solicitudes realizadas por el proxy.
- Servidor: métricas que describen el estado interno de Envoy. Incluyen métricas como el tiempo de actividad o la memoria asignada.

En App Mesh, el proxy intercepta el tráfico ascendente y descendente. Por ejemplo, las solicitudes recibidas de sus clientes, así como las solicitudes realizadas por su contenedor de servicios, Envoy las clasifica como tráfico descendente. Para distinguir entre estos diferentes tipos de tráfico ascendente y descendente, App Mesh clasifica aún más las métricas de Envoy en función de la dirección del tráfico relacionadas con su servicio:

- Entrada: métricas y recursos relacionados con las conexiones y solicitudes que fluyen a su contenedor de servicios.
- Salida: métricas y recursos relacionados con las conexiones y solicitudes que fluyen desde su contenedor de servicios y, en última instancia, desde su tarea de Amazon ECS o su pod de Kubernetes.

La siguiente imagen muestra la comunicación entre el proxy y los contenedores de servicios.



Convenciones de nomenclatura para los recursos

Resulta útil entender cómo Envoy ve su malla y cómo sus recursos se corresponden con los recursos que usted define en App Mesh. Estos son los recursos principales de Envoy que App Mesh configura:

- Oyentes: las direcciones y puertos en los que el proxy escucha las conexiones descendentes. En la imagen anterior, App Mesh crea un oyente de entrada para el tráfico que entra en su tarea de Amazon ECS o su pod de Kubernetes y un oyente de salida para el tráfico que sale de su contenedor de servicios.
- Clústeres: un grupo de puntos de conexión ascendentes con nombre a los que el proxy se conecta y dirige el tráfico. En App Mesh, su contenedor de servicios se representa como un clúster, así como todos los demás nodos virtuales a los que se puede conectar su servicio.

- Rutas: corresponden a las rutas que defina en la malla. Contienen las condiciones según las cuales el proxy corresponde a una solicitud, así como el clúster de destino al que se envía la solicitud.
- Asignaciones de carga de clústeres y puntos de conexión: las direcciones IP de los clústeres ascendentes. Cuando se utiliza AWS Cloud Map como mecanismo de detección de servicios para nodos virtuales, App Mesh envía las instancias de servicio detectadas como recursos de punto de conexión a su proxy.
- Secretos: incluyen, entre otros, las claves de cifrado y los certificados TLS. Cuando se usa AWS Certificate Manager como fuente de certificados de cliente y servidor, App Mesh envía certificados públicos y privados a tu proxy como recursos secretos.

App Mesh usa un esquema coherente para nombrar los recursos de Envoy que puede usar para relacionarlos con su malla.

Comprender el esquema de nomenclatura de los oyentes y clústeres es importante para entender las métricas de Envoy en App Mesh.

Nombres de los oyentes

Los oyentes se nombran con el siguiente formato:

```
lds_<traffic direction>_<listener IP address>_<listening port>
```

Normalmente, verá los siguientes oyentes configurados en Envoy:

- `lds_ingress_0.0.0.0_15000`
- `lds_egress_0.0.0.0_15001`

Mediante un complemento CNI de Kubernetes o reglas de tablas IP, el tráfico de la tarea de Amazon ECS o del pod de Kubernetes se dirige a los puertos `15000` y `15001`. App Mesh configura Envoy con estos dos oyentes para aceptar el tráfico de entrada (entrante) y de salida (saliente). Si no tiene configurado un oyente en su nodo virtual, no debería ver ningún oyente de entrada.

Nombres de clúster

La mayoría de los clústeres utilizan el siguiente formato:

```
cds_<traffic direction>_<mesh name>_<virtual node name>_<protocol>_<port>
```

Cada uno de los nodos virtuales con los que se comunican sus servicios tiene su propio clúster. Como se mencionó anteriormente, App Mesh crea un clúster para el servicio que se ejecuta junto a Envoy para que el proxy pueda enviarle el tráfico de entrada.

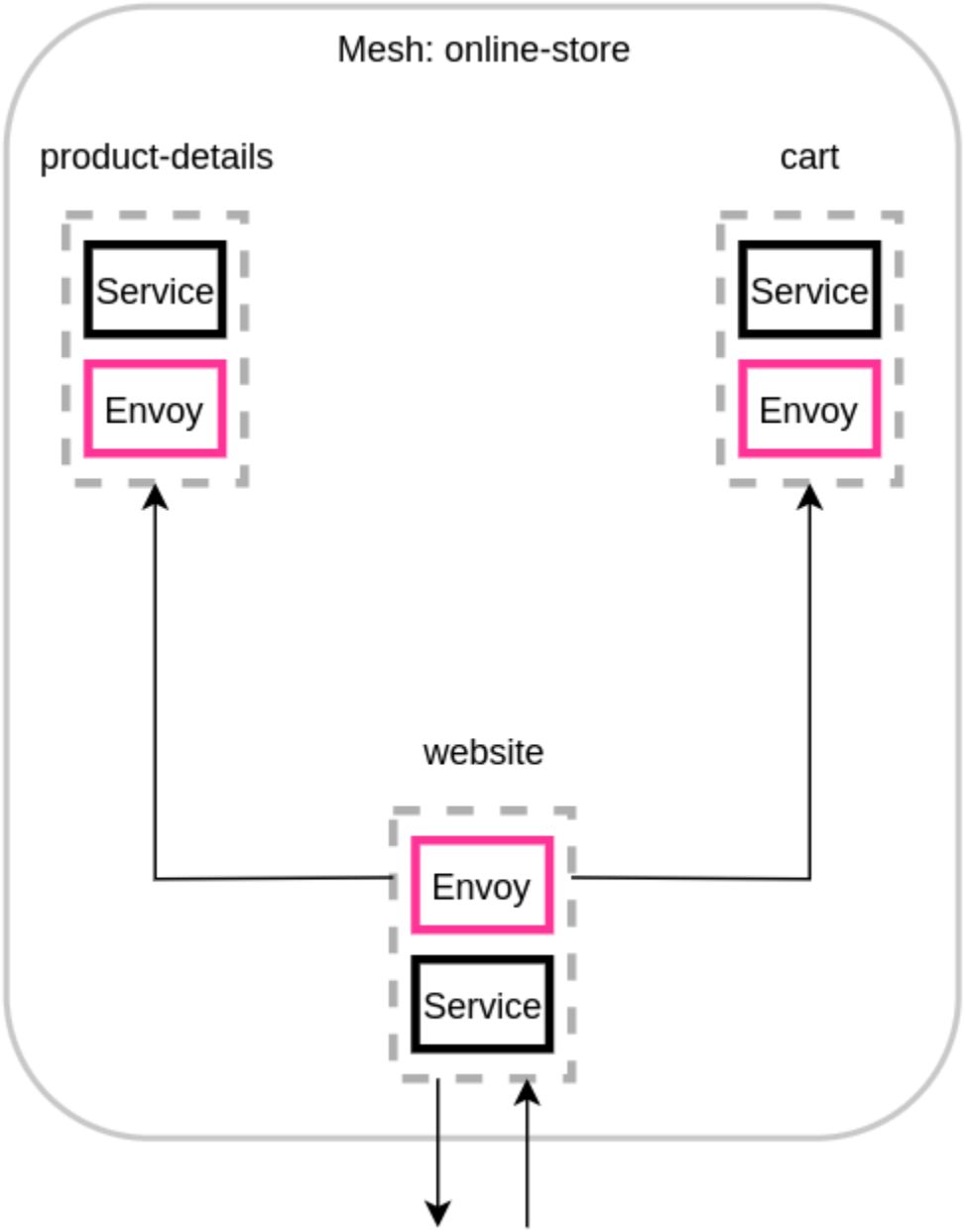
Por ejemplo, si tiene un nodo virtual con el nombre `my-virtual-node` que escucha el tráfico http en el puerto `8080` y dicho nodo virtual está en una malla denominada `my-mesh`, App Mesh crea un clúster con el nombre `cds_ingress_my-mesh_my-virtual-node_http_8080`. Este clúster sirve como destino del tráfico hacia el contenedor de servicios de `my-virtual-node`.

App Mesh también puede crear los siguientes tipos de clústeres especiales adicionales. Estos otros clústeres no se corresponden necesariamente con los recursos que defina de forma explícita en su malla.

- Los clústeres se utilizan para llegar a otros AWS servicios. Este tipo permite que su malla llegue a la mayoría de AWS los servicios de forma predeterminada:`cds_egress_`*`<mesh name>`*`_amazonaws`.
- Clúster utilizado para realizar el enrutamiento de las puertas de enlace virtuales. En general se puede ignorar con toda tranquilidad.
  - Para oyentes individuales: `cds_ingress_`*`<mesh name>_<virtual gateway name>`*`_self_redirect_`*`<protocol>_<port>`*
  - Para oyentes múltiples: `cds_ingress_`*`<mesh name>_<virtual gateway name>`*`_self_redirect_`*`<ingress_listener_port>_<protocol>_<port>`*
- El clúster cuyo punto de conexión puede definir, por ejemplo TLS, cuando recupera secretos mediante Secret Discovery Service de Envoy: `static_cluster_sds_unix_socket`.

Métricas de la aplicación de ejemplo

Para ilustrar las métricas disponibles en Envoy, la siguiente aplicación de ejemplo tiene tres nodos virtuales. Los servicios virtuales, los enrutadores virtuales y las rutas de la malla se pueden ignorar, porque no se reflejan en las métricas de Envoy. En este ejemplo, todos los servicios escuchan el tráfico http en el puerto 8080.



Recomendamos añadir la variable de entorno `ENABLE_ENVOY_STATS_TAGS=1` a los contenedores del proxy de Envoy que se ejecutan en la malla. Esto añade las siguientes dimensiones de métrica a todas las métricas que emite el proxy:

- `appmesh.mesh`
- `appmesh.virtual_node`
- `appmesh.virtual_gateway`

Estas etiquetas se definen para el nombre de la malla, el nodo virtual o la puerta de enlace virtual para permitir filtrar las métricas mediante los nombres de los recursos de la malla.

Nombres de los recursos

El proxy del nodo virtual del sitio web tiene los siguientes recursos:

- Dos oyentes para el tráfico de entrada y salida:
  - `lds_ingress_0.0.0.0_15000`
  - `lds_egress_0.0.0.0_15001`
- Dos clústeres de salida, que representan los dos backends del nodo virtual:
  - `cds_egress_online-store_product-details_http_8080`
  - `cds_egress_online-store_cart_http_8080`
- Un clúster de entrada para el contenedor de servicios del sitio web:
  - `cds_ingress_online-store_website_http_8080`

Métricas del oyente de ejemplo

- `listener.0.0.0.0_15000.downstream_cx_active`: número de conexiones de red de entrada activas a Envoy.
- `listener.0.0.0.0_15001.downstream_cx_active`: número de conexiones de red de salida activas a Envoy. Las conexiones realizadas por su aplicación con servicios externos se incluyen en este recuento.
- `listener.0.0.0.0_15000.downstream_cx_total`: número total de conexiones de red de entrada a Envoy.
- `listener.0.0.0.0_15001.downstream_cx_total`: número total de conexiones de red de salida a Envoy.

Para ver el conjunto completo de métricas del oyente, consulte Estadísticas en la documentación de Envoy.

Métricas del clúster de ejemplo

- `cluster_manager.active_clusters`: número total de clústeres con los que Envoy ha establecido al menos una conexión.

- `cluster_manager.warming_clusters`: número total de clústeres a los que Envoy aún no se ha conectado.

Las siguientes métricas de clúster utilizan el formato `cluster.<cluster name>.<metric name>`. Estos nombres de métricas son exclusivos del ejemplo de aplicación y los emite el contenedor de Envoy del sitio web:

- `cluster.cds_egress_online-store_product-details_http_8080.upstream_cx_total`: número total de conexiones entre el sitio web y los detalles del producto.
- `cluster.cds_egress_online-store_product-details_http_8080.upstream_cx_connect_fail`: número total de conexiones con errores entre el sitio web y los detalles del producto.
- `cluster.cds_egress_online-store_product-details_http_8080.health_check.failure`: número total de comprobaciones de estado con errores entre el sitio web y los detalles del producto.
- `cluster.cds_egress_online-store_product-details_http_8080.upstream_rq_total`: número total de solicitudes realizadas entre el sitio web y los detalles del producto.
- `cluster.cds_egress_online-store_product-details_http_8080.upstream_rq_time`: tiempo que tardan las solicitudes realizadas entre el sitio web y los detalles del producto.
- `cluster.cds_egress_online-store_product-details_http_8080.upstream_rq_2xx`: número de respuestas HTTP 2xx recibidas por el sitio web de los detalles del producto.

Para ver el conjunto completo de métricas HTTP, consulte Estadísticas en la documentación de Envoy.

Métricas del servidor de administración

Envoy también emite métricas relacionadas con su conexión al plano de control de App Mesh, que actúa como servidor de administración de Envoy. Recomendamos monitorizar algunas de estas métricas para avisarlo cuando sus proxies se desincronizan respecto al plano de control durante períodos prolongados. La pérdida de conectividad con el plano de control o las actualizaciones

incorrectas impiden que sus proxies reciban una nueva configuración de App Mesh, incluidos los cambios de malla realizados a través de las API de App Mesh.

- `control_plane.connected_state`: esta métrica se establece en 1 cuando el proxy está conectado a App Mesh; de lo contrario, es 0.
- `*.update_rejected`: número total de actualizaciones de configuración que rechaza Envoy. Por lo general, se deben a una mala configuración del usuario. Por ejemplo, si configura App Mesh para que lea un certificado TLS de un archivo que Envoy no puede leer, se rechazará la actualización que contenga la ruta a dicho certificado.
  - Para un oyente actualizado rechazado, las estadísticas serán `listener_manager.lds.update_rejected`.
  - Para un clúster actualizado rechazado, las estadísticas serán `cluster_manager.cds.update_rejected`.
- `*.update_success`: número de actualizaciones de configuración correctas realizadas por App Mesh en su proxy. Incluyen la carga útil de configuración inicial que se envía cuando se inicia un nuevo contenedor de Envoy.
  - Para un oyente actualizado correctamente, las estadísticas serán `listener_manager.lds.update_success`.
  - Para un clúster actualizado correctamente, las estadísticas serán `cluster_manager.cds.update_success`.

Para ver el conjunto de métricas del servidor de administración, consulte Servidor de administración en la documentación de Envoy.

Exportación de métricas

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS se suspenderá el soporte para. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Envoy emite muchas estadísticas tanto sobre su propio funcionamiento como sobre diversas dimensiones del tráfico entrante y saliente. Para obtener más información sobre las estadísticas

de Envoy, consulte Estadísticas en la documentación de Envoy. Estas métricas están disponibles a través del punto de conexión `/stats` en el puerto de administración del proxy, que suele ser el `9901`.

El prefijo `stat` variará en función de si utiliza uno o varios oyentes. A continuación se muestran algunos ejemplos para ilustrar las diferencias.

> **Warning**
>
> Si actualiza la característica de oyente individual a oyente múltiple, puede enfrentarse a un cambio radical debido al prefijo de estadísticas actualizado que se muestra en la siguiente tabla.
> Sugerimos que utilice la versión `1.22.2.1-prod` o posterior de la imagen de Envoy. Esto le permite ver nombres de métricas similares en su punto de conexión de Prometheus.

Estadísticas de oyente individual (SL)/existente con el prefijo de oyente "ingress"	Estadísticas de oyentes múltiples (ML)/nuevos con el prefijo de oyente "ingress.<protocolo>.<puerto>"	

`http.*ingress*.rds.rds_ingress_http_5555.version_text`	`http.*ingress.http.5555*.rds.rds_ingress_http_5555.version_text`
	
	`http.*ingress.http.6666*.rds.rds_ingress_http_6666.version_text`	

`listener.0.0.0.0_15000.http.*ingress*.downstream_rq_2xx`	`listener.0.0.0.0_15000.http.*ingress.http.5555*.downstream_rq_2xx`
	
	`listener.0.0.0.0_15000.http.*ingress`	

Estadísticas de oyente individual (SL)/existente con el prefijo de oyente "ingress"	Estadísticas de oyentes múltiples (ML)/nuevos con el prefijo de oyente "ingress.<protocolo>.<puerto>"	
	`.http.6666*.downstream_rq_2xx`	

`http.*ingress*.downstream_cx_length_ms`	`http.*ingress.http.5555*.downstream_cx_length_ms`
	
	`http.*ingress.http.6666*.downstream_cx_length_ms`	

Para obtener más información sobre el punto de conexión de estadísticas, consulte Punto de conexión de estadísticas en la documentación de Envoy. Para obtener más información sobre la interfaz de administración, consulte Habilitar la interfaz de administración del proxy de Envoy.

Prometheus para App Mesh con Amazon EKS

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Prometheus es un conjunto de herramientas de alerta y monitorización de código abierto. Una de sus capacidades es especificar un formato para emitir métricas que puedan utilizar otros sistemas. Para obtener información acerca de Prometheus, consulte Información general en la documentación de Prometheus. Envoy puede emitir sus métricas a través de su punto de conexión de estadísticas especificando el parámetro `/stats?format=prometheus`.

Para los clientes que utilizan la versión v1.22.2.1-prod de la imagen de Envoy, hay dos dimensiones adicionales para indicar estadísticas específicas del oyente de entrada:

- `appmesh.listener_protocol`
- `appmesh.listener_port`

A continuación se muestra una comparación entre las estadísticas existentes de Prometheus y las nuevas.

- Estadísticas existentes con el prefijo de oyente "ingress"

```
envoy_http_downstream_rq_xx{appmesh_mesh="multiple-listeners-
mesh",appmesh_virtual_node="foodteller-
vn",envoy_response_code_class="2",envoy_http_conn_manager_prefix="ingress"} 931433
```

- Nuevas estadísticas con el prefijo de oyente "ingress.<protocolo>.<puerto>" + Imagen de Appmesh Envoy v1.22.2.1-prod o posterior

```
envoy_http_downstream_rq_xx{appmesh_mesh="multiple-listeners-
mesh",appmesh_virtual_node="foodteller-
vn",envoy_response_code_class="2",appmesh_listener_protocol="http",appmesh_listener_port="555
 20
```

- Nuevas estadísticas con el prefijo "ingress.<protocolo>.<puerto>" + Imagebuild de Envoy personalizada

```
envoy_http_http_5555_downstream_rq_xx{appmesh_mesh="multiple-listeners-
mesh",appmesh_virtual_node="foodteller-
vn",envoy_response_code_class="2",envoy_http_conn_manager_prefix="ingress"} 15983
```

Para oyentes múltiples, el clúster especial `cds_ingress_<mesh name>_<virtual gateway name>_self_redirect_<ingress_listener_port>_<protocol>_<port>` será específico de cada oyente.

- Estadísticas existentes con el prefijo de oyente "ingress"

```
envoy_cluster_assignment_stale{appmesh_mesh="multiple-listeners-
mesh",appmesh_virtual_gateway="tellergateway-vg",Mesh="multiple-listeners-
```

```
mesh",VirtualGateway="tellergateway-vg",envoy_cluster_name="cds_ingress_multiple-
listeners-mesh_tellergateway-vg_self_redirect_http_15001"} 0
```

- Nuevas estadísticas con el prefijo "ingress.<protocolo>.<puerto>"

```
envoy_cluster_assignment_stale{appmesh_mesh="multiple-
listeners-mesh",appmesh_virtual_gateway="tellergateway-
vg",envoy_cluster_name="cds_ingress_multiple-listeners-mesh_tellergateway-
vg_self_redirect_1111_http_15001"} 0
envoy_cluster_assignment_stale{appmesh_mesh="multiple-
listeners-mesh",appmesh_virtual_gateway="tellergateway-
vg",envoy_cluster_name="cds_ingress_multiple-listeners-mesh_tellergateway-
vg_self_redirect_2222_http_15001"} 0
```

Instalación de Prometheus

1. Agregue el repositorio EKS a Helm:

```
helm repo add eks https://aws.github.io/eks-charts
```

2. Instale App Mesh Prometheus

```
helm upgrade -i appmesh-prometheus eks/appmesh-prometheus \
--namespace appmesh-system
```

Ejemplo de Prometheus

A continuación se muestra un ejemplo de la creación de `PersistentVolumeClaim` para un almacenamiento persistente de Prometheus.

```
helm upgrade -i appmesh-prometheus eks/appmesh-prometheus \
--namespace appmesh-system \
--set retention=12h \
--set persistentVolumeClaim.claimName=prometheus
```

Tutorial de uso de Prometheus

- App Mesh con EKS. Observabilidad: Prometheus

Para obtener más información acerca de Prometheus y Prometheus con Amazon EKS

- Documentación de Prometheus
- EKS: Métricas del plano de control con Prometheus

CloudWatch para App Mesh

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Emisión de estadísticas de Envoy CloudWatch desde Amazon EKS

Puede instalar el CloudWatch agente en su clúster y configurarlo para que recopile un subconjunto de métricas de sus proxies. Si aún no tiene un clúster de Amazon EKS, puede crear uno siguiendo los pasos de Walkthrough: App Mesh con Amazon EKS activado GitHub. Puede instalar una aplicación de muestra en el clúster siguiendo el mismo tutorial.

Para configurar los permisos de IAM adecuados para su clúster e instalar el agente, siga los pasos de la colección Instalar el CloudWatch agente con Prometheus Metrics. La instalación predeterminada contiene una configuración de extracción de Prometheus que obtiene un útil subconjunto de estadísticas de Envoy. Para obtener más información, consulte Métricas de Prometheus para App Mesh.

Para crear un CloudWatch panel de control personalizado de App Mesh configurado para mostrar las métricas que recopila el agente, sigue los pasos del tutorial Visualización de tus métricas de Prometheus. Sus gráficos comenzarán a llenarse con las métricas correspondientes a medida que el tráfico entre en la aplicación de App Mesh.

Filtrar métricas para CloudWatch

La extensión de métricas de App Mesh dispone de un útil subconjunto de métricas que proporciona información sobre el comportamiento de los recursos que define en su malla. Como el CloudWatch agente admite la extracción de métricas de Prometheus, puedes proporcionar una configuración de extracción para seleccionar las métricas que quieres extraer de Envoy y enviarlas. CloudWatch

Puede encontrar un ejemplo de extracción de métricas con Prometheus en nuestro tutorial Extensión de métricas.

CloudWatch Ejemplo

Puede encontrar un ejemplo de configuración de CloudWatch en nuestro repositorio de AWS muestras.

Tutoriales de uso CloudWatch

- Agregar capacidades de monitorización y registro en nuestro Taller de App Mesh.
- App Mesh con EKS-Observabilidad: CloudWatch
- Uso de la extensión de métricas de App Mesh en ECS

Extensión de métricas de App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Envoy genera cientos de métricas que se dividen en unas pocas dimensiones diferentes. Las métricas no son sencillas en cuanto a la forma en que se relacionan con App Mesh. En el caso de los servicios virtuales, no existe ningún mecanismo para saber con certeza qué servicio virtual se está comunicando con una puerta de enlace virtual o un nodo virtual determinado.

La extensión de métricas App Mesh mejora los proxies de Envoy que se ejecutan en su malla. Esta mejora permite que los proxies emitan métricas adicionales que tienen en cuenta los recursos

que define. Este pequeño subconjunto de métricas adicionales lo ayudará a comprender mejor el comportamiento de los recursos que definió en App Mesh.

Para habilitar la extensión de métricas de App Mesh, establezca la variable de entorno `APPMESH_METRIC_EXTENSION_VERSION` en `1`.

```
APPMESH_METRIC_EXTENSION_VERSION=1
```

Para obtener más información acerca de las variables de configuración de Envoy, consulte Variables de configuración de Envoy.

Métricas relacionadas con el tráfico entrante

- **`ActiveConnectionCount`**
  - `envoy.appmesh.ActiveConnectionCount`: número de conexiones TCP activas.
  - Dimensiones: malla, VirtualNode VirtualGateway
- **`NewConnectionCount`**
  - `envoy.appmesh.NewConnectionCount`: número total de conexiones TCP.
  - Dimensiones: malla VirtualNode, VirtualGateway
- **`ProcessedBytes`**
  - `envoy.appmesh.ProcessedBytes`: bytes TCP totales enviados y recibidos de clientes descendentes.
  - Dimensiones: malla VirtualNode, VirtualGateway
- **`RequestCount`**
  - `envoy.appmesh.RequestCount`: el número de solicitudes HTTP procesadas.
  - Dimensiones: malla VirtualNode, VirtualGateway
- **`GrpcRequestCount`**
  - `envoy.appmesh.GrpcRequestCount`: el número de solicitudes gPRC procesadas.
  - Dimensiones: malla VirtualNode, VirtualGateway

Métricas relacionadas con el tráfico saliente

Verá dimensiones diferentes de sus métricas de salida en función de si provienen de un nodo virtual o de una puerta de enlace virtual.

- **`TargetProcessedBytes`**
  - `envoy.appmesh.TargetProcessedBytes`: bytes TCP totales enviados y recibidos desde destinos situados antes de Envoy.
  - Dimensiones:
    - Dimensiones del nodo virtual: malla VirtualNode, TargetVirtualService, TargetVirtualNode
    - Dimensiones de la puerta de enlace virtual: malla VirtualGateway,, TargetVirtualService, TargetVirtualNode
- **`HTTPCode_Target_2XX_Count`**
  - `envoy.appmesh.HTTPCode_Target_2XX_Count`: el número de solicitudes HTTP a un destino situado antes de Envoy que dieron como resultado una respuesta HTTP dos veces mayor.
  - Dimensiones:
    - Dimensiones del nodo virtual: malla VirtualNode, TargetVirtualService, TargetVirtualNode
    - Dimensiones de la puerta de enlace virtual: malla VirtualGateway,, TargetVirtualService, TargetVirtualNode
- **`HTTPCode_Target_3XX_Count`**
  - `envoy.appmesh.HTTPCode_Target_3XX_Count`: el número de solicitudes HTTP a un destino situado antes de Envoy que dieron como resultado una respuesta HTTP tres veces mayor.
  - Dimensiones:
    - Dimensiones del nodo virtual: malla VirtualNode, TargetVirtualService, TargetVirtualNode
    - Dimensiones de la puerta de enlace virtual: malla VirtualGateway,, TargetVirtualService, TargetVirtualNode
- **`HTTPCode_Target_4XX_Count`**
  - `envoy.appmesh.HTTPCode_Target_4XX_Count`: el número de solicitudes HTTP a un destino situado antes de Envoy que dieron como resultado una respuesta HTTP cuatro veces mayor.
  - Dimensiones:
    - Dimensiones del nodo virtual: malla VirtualNode, TargetVirtualService, TargetVirtualNode
    - Dimensiones de la puerta de enlace virtual: malla VirtualGateway,, TargetVirtualService, TargetVirtualNode
- **`HTTPCode_Target_5XX_Count`**

- `envoy.appmesh.HTTPCode_Target_5XX_Count`: el número de solicitudes HTTP a un destino situado antes de Envoy que dieron como resultado una respuesta HTTP cinco veces mayor.
- Dimensiones:
  - Dimensiones del nodo virtual: malla VirtualNode, TargetVirtualService, TargetVirtualNode
  - Dimensiones de la puerta de enlace virtual: malla VirtualGateway,, TargetVirtualService, TargetVirtualNode

- **`RequestCountPerTarget`**
  - `envoy.appmesh.RequestCountPerTarget`: el número de solicitudes enviadas a un objetivo situado antes de Envoy.
  - Dimensiones:
    - Dimensiones del nodo virtual: malla VirtualNode, TargetVirtualService, TargetVirtualNode
    - Dimensiones de la puerta de enlace virtual: malla VirtualGateway,, TargetVirtualService, TargetVirtualNode
- **`TargetResponseTime`**
  - `envoy.appmesh.TargetResponseTime`: el tiempo transcurrido desde que se realiza una solicitud a un destino situado antes de Envoy hasta que se recibe la respuesta completa.
  - Dimensiones:
    - Dimensiones del nodo virtual: malla VirtualNode, TargetVirtualService, TargetVirtualNode
    - Dimensiones de la puerta de enlace virtual: malla VirtualGateway,, TargetVirtualService, TargetVirtualNode

Datadog para App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Datadog es una aplicación de monitorización y seguridad para el registro, métricas y monitorización integrales de aplicaciones en la nube. Datadog permite que su infraestructura, aplicaciones y aplicaciones de terceros se puedan observar por completo.

Instalación de Datadog

- EKS: para configurar Datadog con EKS, siga estos pasos de la documentación de Datadog.
- ECS EC2: para configurar Datadog con ECS EC2, siga estos pasos de la documentación de Datadog.

Para obtener más información acerca de Datadog

- Documentación de Datadog

Rastreo

Important

Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Important

Para implementar completamente el rastreo, deberá actualizar su aplicación.
Para ver todos los datos disponibles del servicio que haya elegido, tendrá que instrumentar su aplicación con las bibliotecas correspondientes.

Supervisa App Mesh con AWS X-Ray

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS se suspenderá el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

AWS X-Ray es un servicio que proporciona herramientas que le permiten ver, filtrar y obtener información sobre los datos recopilados a partir de las solicitudes que atiende su aplicación. Esta información lo ayuda a identificar problemas y oportunidades para optimizar su aplicación. Puede ver información detallada sobre las solicitudes y respuestas, así como sobre las llamadas descendentes que su aplicación realiza a otros servicios de AWS .

X-Ray se integra con App Mesh para gestionar sus microservicios de Envoy. Los datos de rastreo de Envoy se envían al X-Ray daemon que se ejecuta en su contenedor.

X-Ray Impleméntelo en el código de su aplicación utilizando la guía del SDK específica para su idioma.

Habilita el X-Ray rastreo a través de App Mesh

- Según el tipo de servicio:
  - ECS: en la definición del contenedor proxy de Envoy, establezca la variable de entorno `ENABLE_ENVOY_XRAY_TRACING` en `1` y la variable de entorno `XRAY_DAEMON_PORT` en`2000`.
  - EKS: en la configuración del controlador de App Mesh, incluya `--set tracing.enabled=true` y `--set tracing.provider=x-ray`.
- En su X-Ray contenedor, exponga el puerto `2000` y ejecútelo como usuario`1337`.

X-Ray ejemplos

Una definición de contenedor de Envoy para Amazon ECS

```
{
  "name": "envoy",
```

```
        "image": "840364872350.dkr.ecr.us-west-2.amazonaws.com/aws-appmesh-
envoy:v1.15.1.0-prod",
        "essential": true,
        "environment": [
          {
            "name": "APPMESH_VIRTUAL_NODE_NAME",
            "value": "mesh/myMesh/virtualNode/myNode"
          },
          {
            "name": "ENABLE_ENVOY_XRAY_TRACING",
            "value": "1"
           }
        ],
        "healthCheck": {
          "command": [
            "CMD-SHELL",
            "curl -s http://localhost:9901/server_info | cut -d' ' -f3 | grep -q live"
            ],
           "startPeriod": 10,
           "interval": 5,
           "timeout": 2,
           "retries": 3
      }
```

Actualización del controlador App Mesh para Amazon EKS

```
helm upgrade -i appmesh-controller eks/appmesh-controller \
--namespace appmesh-system \
--set region=${AWS_REGION} \
--set serviceAccount.create=false \
--set serviceAccount.name=appmesh-controller \
--set tracing.enabled=true \
--set tracing.provider=x-ray
```

Tutoriales para usar el X-Ray

- Supervise con AWS X-Ray
- App Mesh con Amazon EKS: Observabilidad: X-Ray
- Rastreo distribuido con X-Ray In the Workshop AWS App Mesh

Para obtener más información AWS X-Ray

- Documentación de AWS X-Ray

Resolución de problemas AWS X-Ray con App Mesh

- No puedo ver los AWS X-Ray rastros de mis aplicaciones.

Jaeger para App Mesh con Amazon EKS

Jaeger es un sistema de rastreo distribuido integral de código abierto. Se puede usar para perfilar redes y para monitorizar. Jaeger también puede ayudarlo a solucionar problemas de aplicaciones nativas en la nube complejas.

Para implementar Jaeger en el código de su aplicación, puede encontrar la guía específica de su idioma en las bibliotecas de rastreo de la documentación de Jaeger.

Instalación de Jaeger mediante Helm

1. Agregue el repositorio de EKS a Helm.

```
helm repo add eks https://aws.github.io/eks-charts
```

2. Instalación de Jaeger para App Mesh

```
helm upgrade -i appmesh-jaeger eks/appmesh-jaeger \
--namespace appmesh-system
```

Ejemplo de Jaeger

A continuación se incluye un ejemplo de la creación de `PersistentVolumeClaim` para un almacenamiento persistente de Jaeger.

```
helm upgrade -i appmesh-controller eks/appmesh-controller \
--namespace appmesh-system \
--set tracing.enabled=true \
```

```
--set tracing.provider=jaeger \
--set tracing.address=appmesh-jaeger.appmesh-system \
--set tracing.port=9411
```

Tutorial de uso de Jaeger

- App Mesh con EKS. Observabilidad: Jaeger

Para obtener más información acerca de Jaeger

- Documentación de Jaeger

Datadog para el rastreo

Datadog se puede utilizar tanto para el rastreo como para las métricas. Para obtener más información e instrucciones de instalación, busque la guía específica del idioma de su aplicación en la Documentación de Datadog.

Herramientas de App Mesh

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

App Mesh ofrece a los clientes la posibilidad de interactuar APIs indirectamente con ella mediante herramientas como:

- CloudFormation
- AWS Cloud Development Kit (AWS CDK)
- Controlador de App Mesh para Kubernetes
- Terraform

App Mesh y CloudFormation

CloudFormation es un servicio que le permite crear una plantilla con todos los recursos que necesita para su aplicación y, a continuación, CloudFormation configurará y aprovisionará los recursos por usted. También configurará todas las dependencias para que pueda centrarse más en la aplicación y menos en la administración de los recursos.

Para obtener más información y ejemplos sobre el uso CloudFormation con App Mesh, consulta la CloudFormation documentación.

App Mesh y AWS CDK

AWS CDK es un marco de desarrollo para usar código para definir su infraestructura de nube y usarlo CloudFormation para aprovisionarla. AWS CDK admite varios lenguajes de programación TypeScript, incluidos Python JavaScript, Java y C#/.Net.

Para obtener más información sobre el uso AWS CDK con App Mesh, consulta la AWS CDK documentación.

Controlador de App Mesh para Kubernetes

El controlador de App Mesh para Kubernetes le ayuda a administrar sus recursos de App Mesh para un clúster de Kubernetes e inyectar sidecars en los pods. Este controlador se usa específicamente con Amazon EKS y le permite administrar sus recursos de un modo nativo de Kubernetes.

Para obtener más información sobre el controlador de App Mesh, consulte la documentación del controlador de App Mesh.

App Mesh y Terraform

Terraform es una infraestructura de código abierto como herramienta de software de código. Terraform puede administrar los servicios en la nube mediante su CLI e interactúa con ellos APIs mediante archivos de configuración declarativos.

Para obtener más información sobre el uso de App Mesh con Terraform, consulte la documentación de Terraform.

Trabajo con mallas compartidas

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Puedes compartir tus mallas de App Mesh entre AWS las cuentas que utilizan el AWS Resource Access Manager servicio. Una malla compartida permite que los recursos creados por diferentes AWS cuentas se comuniquen entre sí en la misma malla.

Una AWS cuenta puede ser el propietario de un recurso en malla, un consumidor en malla o ambas cosas. Los consumidores pueden crear recursos en una malla que se comparte con su cuenta. Los propietarios pueden crear recursos en cualquier malla que posea la cuenta. El propietario de una malla puede compartir una malla con los siguientes tipos de consumidores de malla.

- AWS Cuentas específicas dentro o fuera de su organización en AWS Organizations
- Una unidad organizativa dentro de su organización en AWS Organizations
- Toda su organización en AWS Organizations

Para ver end-to-end cómo compartir una malla, consulta el tutorial sobre GitHub redes multicuentas.

Otorgar permisos para compartir mallas

Al compartir mallas entre cuentas, se requieren permisos para el director de IAM que comparte la malla y permisos a nivel de recursos necesarios para la propia malla.

Otorgar permiso para compartir una malla

Se requiere un conjunto mínimo de permisos para que un director de IAM comparta una malla. Recomendamos utilizar las políticas de IAM `AWSResourceAccessManagerFullAccess` gestionadas `AWSAppMeshFullAccess` y las políticas de IAM para garantizar que sus directores de IAM dispongan de los permisos necesarios para compartir y utilizar mallas compartidas.

Si utilizas una política de IAM personalizada, las acciones y las acciones son `appmesh:PutMeshPolicy` obligatorias`appmesh:GetMeshPolicy`. `appmesh:DeleteMeshPolicy` Estas son acciones de IAM solo de permiso. Si a un director de IAM no se le conceden estos permisos, se producirá un error al intentar compartir la malla mediante el AWS RAM servicio.

Para obtener más información sobre la forma en que el AWS Resource Access Manager servicio usa IAM, consulte Cómo se AWS RAM usa IAM en la Guía del AWS Resource Access Manager usuario.

Otorgar permisos para una malla

Una malla compartida tiene los siguientes permisos.

- Los consumidores pueden enumerar y describir todos los recursos de una malla que se comparte con la cuenta.
- Los propietarios pueden enumerar y describir todos los recursos de cualquier malla que posea la cuenta.
- Los propietarios y los consumidores pueden modificar los recursos de una malla creada por la cuenta, pero no pueden modificar los recursos que haya creado otra cuenta.
- Los consumidores pueden eliminar cualquier recurso de una malla que haya creado la cuenta.
- Los propietarios pueden eliminar cualquier recurso de una malla que haya creado cualquier cuenta.
- Los recursos del propietario solo pueden hacer referencia a otros recursos de la misma cuenta. Por ejemplo, un nodo virtual solo puede hacer referencia AWS Cloud Map a un AWS Certificate Manager certificado que esté en la misma cuenta que el propietario del nodo virtual.
- Los propietarios y los consumidores pueden conectar un proxy de Envoy a App Mesh como un nodo virtual propiedad de la cuenta.
- Los propietarios pueden crear puertas de enlace virtuales y rutas de puerta de enlace virtuales.
- Los propietarios y los consumidores pueden enumerar las etiquetas y tag/untag los recursos en una malla creada por la cuenta. No pueden incluir etiquetas y tag/untag recursos en una malla que no haya creado la cuenta.

Las mallas compartidas utilizan una autorización basada en políticas. Se comparte una malla con un conjunto fijo de permisos. Estos permisos se seleccionan para añadirlos a una política de recursos y también se puede seleccionar una política de IAM opcional en función del rol o usuario de IAM. La

intersección de los permisos permitidos en estas políticas, menos los permisos explícitos denegados, determina el acceso de la entidad principal a la malla.

Al compartir una malla, el AWS Resource Access Manager servicio crea una política administrada denominada `AWSRAMDefaultPermissionAppMesh` y la asocia a tu App Mesh que proporciona los siguientes permisos.

- `appmesh:CreateVirtualNode`
- `appmesh:CreateVirtualRouter`
- `appmesh:CreateRoute`
- `appmesh:CreateVirtualService`
- `appmesh:UpdateVirtualNode`
- `appmesh:UpdateVirtualRouter`
- `appmesh:UpdateRoute`
- `appmesh:UpdateVirtualService`
- `appmesh:ListVirtualNodes`
- `appmesh:ListVirtualRouters`
- `appmesh:ListRoutes`
- `appmesh:ListVirtualServices`
- `appmesh:DescribeMesh`
- `appmesh:DescribeVirtualNode`
- `appmesh:DescribeVirtualRouter`
- `appmesh:DescribeRoute`
- `appmesh:DescribeVirtualService`
- `appmesh:DeleteVirtualNode`
- `appmesh:DeleteVirtualRouter`
- `appmesh:DeleteRoute`
- `appmesh:DeleteVirtualService`
- `appmesh:TagResource`
- `appmesh:UntagResource`

Requisitos previos para compartir mallas

Para utilizar una malla, debe cumplir los siguientes requisitos previos.

- Debes ser el propietario de la malla de tu AWS cuenta. No puede compartir una malla que se haya compartido con usted.
- Para compartir una malla con su organización o con una unidad organizativa de AWS Organizations, debe habilitar el uso compartido con AWS Organizations. Para obtener más información, consulte Habilitar el uso compartido con AWS Organizations en la Guía del usuario de AWS RAM .
- Sus servicios se deben implementar en una VPC de Amazon que tenga conectividad compartida entre las cuentas que incluyen los recursos de malla que desea poner en comunicación entre sí. Una forma de compartir la conectividad de red consiste en implementar todos los servicios que desee utilizar en la malla en una subred compartida. Para obtener más información y conocer las limitaciones, consulte Uso compartido de una subred.
- Los servicios deben poder detectarse a través del DNS o AWS Cloud Map. Para obtener más información sobre la detección de servicios, consulte Nodos virtuales.

Servicios relacionados

El uso compartido de mallas se integra con AWS Resource Access Manager (AWS RAM). AWS RAM es un servicio que le permite compartir sus AWS recursos con cualquier AWS cuenta o a través de AWS Organizations. Con AWS RAM, compartes los recursos de tu propiedad mediante la creación de un recurso compartido. Un uso compartido de recursos especifica los recursos que compartir y los consumidores con quienes compartirlos. Los consumidores pueden ser AWS cuentas individuales, unidades organizativas o toda una organización AWS Organizations.

Para obtener más información al respecto AWS RAM, consulte la Guía AWS RAM del usuario.

Uso compartido de una malla

Compartir una malla permite que los recursos de malla creados por diferentes cuentas se comuniquen entre sí en la misma malla. Solo puede compartir una malla de su propiedad. Para compartir una malla, debe añadirla a un recurso compartido. Un recurso compartido es un AWS RAM recurso que te permite compartir tus recursos entre AWS cuentas. Un uso compartido de recursos especifica los recursos que se deben compartir y los consumidores con quienes se

comparten. Cuando se comparte una malla mediante la consola de Amazon Linux, se añade a un uso compartido de recurso existente. Para añadir la malla a un nuevo uso compartido de recurso, debe crear el uso compartido de recurso mediante la consola de AWS RAM.

Si formas parte de una organización AWS Organizations y el uso compartido dentro de tu organización está activado, los consumidores de tu organización pueden acceder automáticamente a la malla compartida. De lo contrario, los consumidores reciben una invitación para unirse al recurso compartido y se les concede acceso a la malla compartida después de aceptar la invitación.

Puedes compartir una malla de tu propiedad mediante la AWS RAM consola o el AWS CLI.

Para compartir una malla de tu propiedad mediante la AWS RAM consola

Para obtener instrucciones, consulte Creación de un recurso compartido en la Guía del usuario de AWS RAM . Cuando seleccione un tipo de recurso, elija Mallas y, a continuación, la malla que desee compartir. Si no aparece ninguna malla, cree una primero. Para obtener más información, consulte Creación de una malla de servicios.

Para compartir una malla de tu propiedad mediante el AWS CLI

Utilice el comando create-resource-share. Para la opción `--resource-arns`, especifique el ARN de la malla que desea compartir.

Dejar de compartir una malla compartida

Cuando deja de compartir una malla, App Mesh desactiva el acceso a la malla para los antiguos consumidores de la malla. Sin embargo, App Mesh no elimina los recursos creados por los consumidores. Una vez que se deja de compartir la malla, solo el propietario de la malla puede obtener acceso a los recursos y eliminarlos. App Mesh impide que la cuenta que posee los recursos de la malla reciba información de configuración después de dejar de compartir la malla. App Mesh también impide que cualquier otra cuenta con recursos en la malla reciba información de configuración de una malla que se ha dejado de compartir. Solo el propietario de la malla puede dejar de compartirla.

Para dejar de compartir una malla compartida que posee, debe eliminarla del recurso compartido. Puede hacerlo mediante la AWS RAM consola o el AWS CLI.

Para dejar de compartir una malla compartida de tu propiedad mediante la consola AWS RAM

Para obtener instrucciones, consulte Actualización de un recurso compartido en la Guía del usuario de AWS RAM .

Para dejar de compartir una malla compartida de tu propiedad mediante el AWS CLI

Utilice el comando disassociate-resource-share.

Identificación de una malla compartida

Los propietarios y los consumidores pueden identificar las mallas y los recursos de malla compartidos mediante la consola Amazon Linux y AWS CLI

Para identificar una malla compartida mediante la consola de Amazon Linux

1. Abre la consola App Mesh en https://console.aws.amazon.com/appmesh/.
2. En el panel de navegación izquierdo, seleccione Mallas. El ID de cuenta del propietario de cada malla aparece en la columna Propietario de la malla.
3. En el panel de navegación izquierdo, seleccione Servicios virtuales, Enrutadores virtuales o Nodos virtuales. Verá el ID de cuenta del propietario de la malla y el propietario del recurso de cada uno de los recursos.

Para identificar una malla compartida mediante el AWS CLI

Use el comando `aws appmesh list` *`resource`*, por ejemplo, `aws appmesh list-meshes`. El comando devuelve las mallas que posee y las mallas que se comparten con usted. La `meshOwner` propiedad muestra el ID de AWS cuenta del `meshOwner` y la `resourceOwner` propiedad muestra el ID de AWS cuenta del propietario del recurso. Cualquier comando que se ejecute en cualquier recurso de malla devuelve estas propiedades.

Las etiquetas definidas por el usuario que se asocian a una malla compartida solo están disponibles para su Cuenta de AWS. No están disponibles para las demás cuentas con las que se comparte la malla. Se deniega el acceso al comando `aws appmesh list-tags-for-resource` de una malla en otra cuenta.

Facturación y medición

No se aplican cargos por compartir una malla.

Cuotas de instancias

Todas las cuotas de una malla también se aplican a las mallas compartidas, independientemente de quién haya creado los recursos de la malla. Solo el propietario de una malla puede solicitar aumentos de cuota. Para obtener más información, consulte Cuotas de servicio de App Mesh. El servicio de AWS Resource Access Manager también tiene cuotas. Para obtener más información, consulte Cuotas de servicio.

AWS servicios integrados con App Mesh

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS se suspenderá el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

App Mesh funciona con otros AWS servicios para ofrecer soluciones adicionales a los desafíos de su empresa. En este tema se identifican los servicios que utiliza App Mesh para agregar funcionalidad o servicios que App Mesh utiliza para realizar tareas.

Contenido

- Creación de recursos de App Mesh con AWS CloudFormation
- App Mesh activado AWS Outposts

Creación de recursos de App Mesh con AWS CloudFormation

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS se suspenderá el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

App Mesh está integrado con AWS CloudFormation un servicio que te ayuda a modelar y configurar tus AWS recursos para que puedas dedicar menos tiempo a crear y gestionar tus recursos e infraestructura. Creas una plantilla que describe todos los AWS recursos que deseas, por ejemplo, una malla de App Mesh, y CloudFormation se encarga de aprovisionar y configurar esos recursos por ti.

Cuando la utilices CloudFormation, podrás reutilizar la plantilla para configurar los recursos de App Mesh de forma coherente y repetida. Simplemente describe tus recursos una vez y luego aprovisiona los mismos recursos una y otra vez en varias AWS cuentas y regiones.

App Mesh y CloudFormation plantillas

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

App Mesh admite la creación de mallas, rutas, nodos virtuales, enrutadores virtuales y servicios virtuales en CloudFormation. Para obtener más información, incluidos ejemplos de plantillas JSON y YAML para los recursos de App Mesh, consulte la Referencia del tipo de recurso de App Mesh en la Guía del usuario de AWS CloudFormation .

Obtenga más información sobre CloudFormation

Para obtener más información CloudFormation, consulte los siguientes recursos:

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

App Mesh activado AWS Outposts

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

AWS Outposts habilita AWS servicios, infraestructura y modelos operativos nativos en instalaciones locales. En AWS los entornos de Outposts, puedes usar las mismas AWS API, herramientas e

infraestructura que usas en la AWS nube. App Mesh on AWS Outposts es ideal para cargas de trabajo de baja latencia que deben ejecutarse muy cerca de datos y aplicaciones locales. Para obtener más información sobre AWS Outposts, consulta la Guía del usuario de AWS Outposts.

Requisitos previos

Los siguientes son los requisitos previos para usar App Mesh 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 .
- La AWS región del puesto de avanzada debe ser compatible. AWS App Mesh Para obtener una lista de las regiones compatibles, consulte Puntos de conexión y cuotas de AWS App Mesh en la Referencia general de AWS.

Limitaciones

Las siguientes son las limitaciones del uso de App Mesh en AWS Outposts:

- AWS Identity and Access Management, Application Load Balancer, Network Load Balancer, Classic Load Balancer y Amazon Route 53 se ejecutan en la AWS región, no en Outposts. Esto aumentará las latencias entre estos servicios y los contenedores.

Consideraciones sobre la conectividad de red

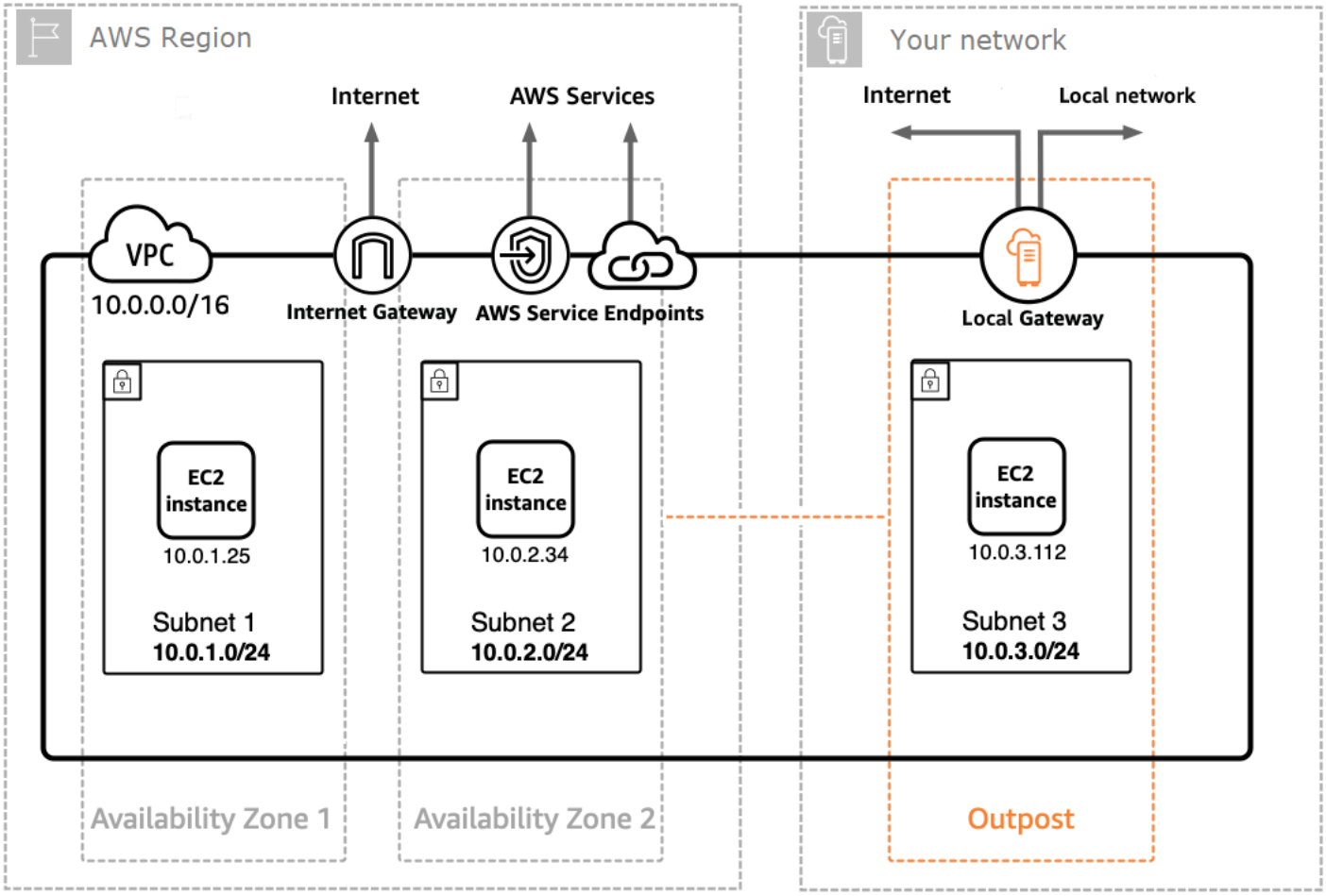
Las siguientes son consideraciones de conectividad de red para Amazon EKS AWS Outposts:

- Si se pierde la conectividad de red entre tu Outpost y su AWS región, los proxies de App Mesh Envoy seguirán funcionando. Sin embargo, no podrá modificar su malla de servicios hasta que se restablezca la conectividad.
- Te recomendamos que proporciones una conectividad fiable, de alta disponibilidad y de baja latencia entre tu Outpost y su región. AWS

Creación de un proxy de App Mesh Envoy en un Outpost

Un Outpost es una extensión de una AWS región y puede ampliar una Amazon VPC en una cuenta para abarcar varias zonas de disponibilidad y cualquier ubicación de Outpost asociada. Al configurar su Outpost, le asocia un grupo de subredes para ampliar su entorno de VPC regional a

sus instalaciones. Las instancias de un Outpost aparecen como parte de su VPC regional, similar a una zona de disponibilidad con subredes asociadas.



Para crear un proxy de App Mesh Envoy en un Outpost, añada la imagen del contenedor de App Mesh Envoy a la tarea de Amazon ECS o al pod de Amazon EKS que se ejecuta en un Outpost. Para obtener más información, consulte Amazon Elastic Container Service on AWS Outposts en la Guía para desarrolladores de Amazon Elastic Container Service y Amazon Elastic Kubernetes Service AWS on Outposts en la Guía del usuario de Amazon EKS.

Prácticas recomendadas para App Mesh

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Para lograr el objetivo de que ninguna solicitud tenga errores durante las implementaciones planificadas y durante la pérdida no planificada de algunos hosts, las prácticas recomendadas de este tema implementan la siguiente estrategia:

- Aumente las probabilidades de que una solicitud se realice correctamente desde el punto de vista de la aplicación mediante una estrategia de reintentos predeterminada segura. Para obtener más información, consulte Instrumentación de todas las rutas con reintentos.
- Aumente la probabilidad de que el reintento de una solicitud tenga éxito maximizando la probabilidad de que el reintento de la solicitud se envíe a un destino real. Para obtener más información, consulte Ajuste de la velocidad de implementación, Escalado horizontal antes de la reducción horizontal y Implementación de comprobaciones de estado de contenedores.

Para reducir o eliminar los errores de forma significativa, recomendamos implementar las recomendaciones en todas las prácticas siguientes.

Instrumentación de todas las rutas con reintentos

Configure todos los servicios virtuales para que usen un enrutador virtual y establezca una política de reintentos predeterminada para todas las rutas. Esto reducirá las solicitudes fallidas al volver a seleccionar un host y enviar una nueva solicitud. Para las políticas de reintento, recomendamos un valor de al menos dos para `maxRetries` y especificar las siguientes opciones para cada tipo de evento de reintento en cada tipo de ruta que admita el tipo de evento de reintento:

- TCP: `connection-error`
- HTTP y HTTP/2: `stream-error` y `gateway-error`
- gRPC: `cancelled` y `unavailable`

Se deben tener en cuenta otros eventos de reintento, ya case-by-case que pueden no ser seguros, por ejemplo, si la solicitud no es idempotente. Deberá tener en cuenta y probar los valores de `maxRetries` y `perRetryTimeout` para encontrar el equilibrio adecuado entre la latencia máxima de una solicitud (`maxRetries` * `perRetryTimeout`) y el aumento de la tasa de éxito de más reintentos. Además, cuando Envoy intente conectarse a un punto de conexión que ya no está presente, es de esperar que la solicitud agote `perRetryTimeout` en su totalidad. Para configurar una política de reintentos, consulte Creación de una ruta y, a continuación, seleccione el protocolo que desee enrutar.

> **Note**
>
> Si implementó una ruta el 29 de julio de 2020 o después y no especificó una política de reintentos, es posible que App Mesh haya creado automáticamente una política de reintentos predeterminada similar a la política anterior para cada ruta que creó el 29 de julio de 2020 o después. Para obtener más información, consulte Política de reintentos de ruta predeterminada.

Ajuste de la velocidad de implementación

Cuando utilice implementaciones continuas, reduzca la velocidad general de la implementación. De forma predeterminada, Amazon ECS configura una estrategia de implementación de un mínimo del 100 % de las tareas en buen estado y del 200 % de las tareas totales. En el momento de la implementación, esto se traduce en dos puntos con gran desviación:

- Los Envoys pueden ver el 100 % del tamaño de la flota de nuevas tareas antes de que estén preparadas para completar las solicitudes (consulte Implementación de comprobaciones de estado de contenedores para las mitigaciones).
- Es posible que los Envoys vean el 100 % del tamaño de la flota de tareas antiguas mientras se completan las tareas.

Cuando se configuran con estas restricciones de implementación, los orquestadores de contenedores puede que pasen a un estado en el que oculten todos los destinos antiguos y muestren todos los nuevos. Como su plano de datos de Envoy finalmente es coherente, podría provocar períodos en los que el conjunto de destinos visibles en su plano de datos difiera desde el punto de vista del orquestador. Para solucionar esta situación, recomendamos mantener un mínimo del 100 % de las tareas en buen estado, pero reducir el total de tareas al 125 %. Esto reducirá la

divergencia y mejorará la fiabilidad de los reintentos. Recomendamos la siguiente configuración para los diferentes tiempos de ejecución de contenedor.

Amazon ECS

Si su servicio tiene un recuento deseado de dos o tres, establezca `maximumPercent` en el 150 %. De lo contrario, establezca `maximumPercent` en el 125 %.

Kubernetes

Configure la `update strategy` de su implementación, estableciendo `maxUnavailable` en el 0 % y `maxSurge` en el 25 %. Para obtener más información sobre las implementaciones, consulte Implementaciones en la documentación de Kubernetes.

Escalado horizontal antes de la reducción horizontal

Al escalar horizontalmente y reducir horizontalmente hay cierta probabilidad de que las solicitudes tengan errores en los reintentos. Aunque existen recomendaciones para las tareas que mitigan los errores al escalar horizontalmente, la única recomendación para reducir horizontalmente es minimizar el porcentaje de tareas reducidas horizontalmente en cualquier momento. Recomendamos utilizar una estrategia de implementación que escale horizontalmente las nuevas tareas de Amazon ECS o las implementaciones de Kubernetes antes de reducir horizontalmente las tareas o implementaciones antiguas. Esta estrategia de escalado reduce el porcentaje de tareas o implementaciones reducidas horizontalmente y, al mismo tiempo, mantiene la misma velocidad. Esta práctica se aplica tanto a las tareas de Amazon ECS como a las implementaciones de Kubernetes.

Implementación de comprobaciones de estado de contenedores

A la hora de escalar verticalmente es posible que los contenedores de una tarea de Amazon ECS estén desordenados y no respondan inicialmente. Recomendamos las siguientes sugerencias para los distintos tiempos de ejecución de los contenedores:

Amazon ECS

Para mitigar esta situación, recomendamos realizar comprobaciones de estado de los contenedores y ordenación de la dependencia de los contenedores para garantizar que Envoy esté funcionando y en buen estado antes de que se inicie cualquier contenedor que requiera conectividad de red

saliente. Para configurar correctamente un contenedor de aplicaciones y un contenedor de Envoy en una definición de tarea, consulte Dependencia de contenedores.

Kubernetes

Ninguna, porque las pruebas de dinamismo y preparación de Kubernetes no se tienen en cuenta a la hora de registrar o anular el registro de instancias en AWS Cloud Map el controlador App Mesh para Kubernetes. Para obtener más GitHub información, consulta el número #132.

Optimice la resolución de DNS

Si utilizas el DNS para la detección de servicios, es fundamental seleccionar el protocolo IP adecuado para optimizar la resolución del DNS al configurar las mallas. App Mesh es compatible con ambas opciones`IPv6`, `IPv4` y su elección puede afectar al rendimiento y la compatibilidad de su servicio. Si su infraestructura no es compatible`IPv6`, le recomendamos que especifique una configuración de IP que se adapte a su infraestructura en lugar de confiar en el `IPv6_PREFERRED` comportamiento predeterminado. El `IPv6_PREFERRED` comportamiento predeterminado puede degradar el rendimiento del servicio.

- IPv6_PREFERRED: esta es la configuración predeterminada. Envoy realiza primero una búsqueda de IPv6 direcciones en el DNS y recurre a ella `IPv4` si no encuentra ninguna `IPv6` dirección. Esto es beneficioso si su infraestructura es compatible principalmente`IPv6`, pero necesita `IPv4` compatibilidad.
- IPv4_PREFERIDO: Envoy busca primero `IPv4` las direcciones y recurre a ellas `IPv6` si no hay ninguna `IPv4` dirección disponible. Use esta configuración si su infraestructura es compatible principalmente`IPv4`, pero tiene alguna `IPv6` compatibilidad.
- IPv6_ONLY: elija esta opción si sus servicios admiten `IPv6` tráfico exclusivamente. Envoy solo realiza búsquedas de `IPv6` direcciones en el DNS, lo que garantiza que todo el tráfico se enrute. `IPv6`
- IPv4_ONLY: elija esta configuración si sus servicios admiten tráfico exclusivamente. `IPv4` Envoy solo realiza búsquedas de `IPv4` direcciones en el DNS, lo que garantiza que todo el tráfico se enrute. `IPv4`

Puede configurar las preferencias de la versión IP tanto a nivel de malla como a nivel de nodo virtual, sustituyendo la configuración de los nodos virtuales a las de nivel de malla.

Para obtener más información, consulte Mallas de servicio y nodos virtuales.

Seguridad en AWS App Mesh

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

La seguridad en la nube AWS es la máxima prioridad. Como AWS cliente, usted se beneficia de una arquitectura de centro de datos y red diseñada para cumplir con los requisitos de las organizaciones más sensibles a la seguridad.

La seguridad es una responsabilidad compartida entre usted AWS y usted. El modelo de responsabilidad compartida la describe como seguridad de la nube y seguridad en la nube:

- Seguridad de la nube: AWS es responsable de proteger la infraestructura que ejecuta AWS los servicios en la AWS nube. AWS también le proporciona servicios que puede utilizar de forma segura. Third-partylos auditores prueban y verifican periódicamente la eficacia de nuestra seguridad como parte de los programas de AWS cumplimiento. Para obtener más información sobre los programas de conformidad que se aplican a AWS App Mesh, consulte Servicios de AWS en el ámbito del programa de conformidad. App Mesh es responsable de proporcionar una configuración segura a los proxies locales, lo que incluye secretos como las claves privadas de los certificados TLS.
- Seguridad en la nube: su responsabilidad viene determinada por el AWS servicio que utilice. Usted es también responsable de otros factores, entre los que se incluyen:
  - La confidencialidad de los datos, los requisitos de la empresa y los reglamentos y la legislación vigentes.
  - La configuración de seguridad del plano de datos de App Mesh, incluida la configuración de los grupos de seguridad que permiten que el tráfico pase de un servicio a otro de la VPC.
  - La configuración de sus recursos informáticos asociados a App Mesh.
  - Las políticas de IAM asociadas a sus recursos informáticos y la configuración que pueden obtener del plano de control de App Mesh.

Esta documentación lo ayuda a comprender cómo aplicar el modelo de responsabilidad compartida cuando se utiliza App Mesh. En los siguientes temas, se explica cómo configurar App Mesh para cumplir sus objetivos de seguridad y conformidad. También aprenderás a usar otros AWS servicios que te ayudan a monitorear y proteger tus recursos de App Mesh.

Principio de seguridad de App Mesh

> Los clientes deberían poder ajustar la seguridad en la medida en que lo necesiten. La plataforma no debería impedir que sean más seguros. Las características de la plataforma son seguras de forma predeterminada.

Temas

- seguridad de la capa de transporte (TLS)
- Autenticación TLS mutua
- Cómo AWS App Mesh funciona con IAM
- Registro de llamadas a la AWS App Mesh API mediante AWS CloudTrail
- Protección de datos en AWS App Mesh
- Validación de conformidad para AWS App Mesh
- Seguridad de infraestructura en AWS App Mesh
- Resiliencia en AWS App Mesh
- Análisis de configuración y vulnerabilidad en AWS App Mesh

seguridad de la capa de transporte (TLS)

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte a AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

En App Mesh, la seguridad de la capa de transporte (TLS) cifra la comunicación entre los proxies de Envoy implementados en los recursos informáticos que están representados en App Mesh por puntos de conexión de malla como Nodos virtuales y Puertas de enlace virtuales. El proxy negocia y

finaliza la TLS. Cuando el proxy se implementa con una aplicación, el código de la aplicación no es responsable de negociar una sesión de TLS. El proxy negocia la TLS en nombre de su aplicación.

App Mesh permite proporcionar el certificado TLS al proxy de las siguientes maneras:

- Un certificado privado de AWS Certificate Manager (ACM) emitido por un AWS Private Certificate Authority (AWS Private CA).
- Un certificado almacenado en el sistema de archivos local de un nodo virtual emitido por su propia autoridad de certificación (CA)
- Un certificado proporcionado por un punto de conexión SDS (Secrets Discovery Service) a través de un socket de dominio Unix local.

La Autorización de proxy de Envoy debe estar habilitada para el proxy de Envoy implementado representado por un punto de conexión de malla. Le recomendamos que, al habilitar la autorización del proxy, restrinja el acceso únicamente al punto de conexión de malla para el que esté habilitando el cifrado.

Requisitos del certificado

Uno de los Nombres alternativos del asunto (SAN) del certificado debe cumplir unos criterios específicos, en función de cómo se detecte el servicio real representado por un punto de conexión de malla.

- DNS: uno de los SAN del certificado debe coincidir con el valor proporcionado en la configuración de detección de servicios del DNS. Para una aplicación con el nombre de detección de servicios de *`mesh-endpoint.apps.local`*, puede crear un certificado que coincida con ese nombre o un certificado con el comodín `*.`*`apps.local`*.
- AWS Cloud Map— Una de las SAN certificadas debe coincidir con el valor proporcionado en la configuración de detección de AWS Cloud Map servicios mediante el formato*`service-name.namespace-name`*. Para una aplicación con la AWS Cloud Map configuración de detección de servicios de ServiceName y *`mesh-endpoint apps.local`* NamespaceName, puede crear un certificado que coincida con el *`mesh-endpoint.apps.local`* nombre o un certificado con el comodín `*.`*`apps.local`*`.`

En ambos mecanismos de detección, si ninguno de los SAN con certificados coincide con la configuración de detección de servicios del DNS, se produce un error en la conexión entre los Envoys y aparece el siguiente mensaje de error, tal y como lo muestra el cliente de Envoy.

```
TLS error: 268435581:SSL routines:OPENSSL_internal:CERTIFICATE_VERIFY_FAILED
```

Certificados de autenticación TLS

App Mesh admite varios orígenes para los certificados cuando se utiliza la autenticación TLS.

AWS Private CA

El certificado se debe almacenar en ACM en la misma región y cuenta de AWS que el punto de conexión de malla que utilizará el certificado. No es necesario que el certificado de la CA esté en la misma AWS cuenta, pero sí en la misma región que el punto final de malla. Si no tiene una Autoridad de certificación privada de AWS, debe crear una antes de poder solicitarle un certificado. Para obtener más información sobre cómo solicitar un certificado a una empresa existente que AWS Private CA utiliza ACM, consulte Solicitar un certificado privado. El certificado no puede ser un certificado público.

Las autoridades de certificación privadas que utilice para las políticas de cliente de TLS deben ser autoridades de certificación de usuario raíz.

Para configurar un nodo virtual con certificados y CA de AWS Private CA, el principal (por ejemplo, un usuario o un rol) que utilices para llamar a App Mesh debe tener los siguientes permisos de IAM:

- Para cualquier certificado que añada a la configuración de TLS de un oyente, la entidad principal debe tener el permiso `acm:DescribeCertificate`.
- Para cualquier autoridad de certificación configurada en una política de cliente de TLS, la entidad principal debe tener el permiso `acm-pca:DescribeCertificateAuthority`.

> **Important**
>
> Al compartir las autoridades de certificación con otras cuentas, es posible que esas cuentas obtengan privilegios no deseados para la autoridad de certificación. Recomendamos utilizar políticas basadas en los recursos para restringir el acceso únicamente a `acm-pca:DescribeCertificateAuthority` y `acm-pca:GetCertificateAuthorityCertificate` para las cuentas que no necesiten emitir certificados de la autoridad de certificación.

Puede añadir estos permisos a una política de IAM existente que esté asociada a una entidad principal o crear una entidad principal y una política nuevas y asociar la política a la entidad principal. Para obtener más información, consulte Edición de políticas de IAM, Creación de políticas de IAM y Adición de permisos de identidad de IAM.

> **Note**
>
> Pagas una cuota mensual por el funcionamiento de cada uno de ellos AWS Private CA hasta que los elimines. También paga por los certificados privados que emita cada mes y por los certificados privados que exporte. Para más información, consulte Precios de AWS Certificate Manager.

Al habilitar la autorización de proxy para el proxy de Envoy que representa un punto de conexión de malla, se deben asignar los siguientes permisos de IAM al rol de IAM que utilice:

- Para cualquier certificado configurado en el oyente de un nodo virtual, el rol debe tener el permiso `acm:ExportCertificate`.
- Para cualquier autoridad de certificación configurada en una política de cliente de TLS, el rol debe tener el permiso `acm-pca:GetCertificateAuthorityCertificate`.

Sistema de archivos

Puede distribuir los certificados a Envoy mediante el sistema de archivos. Para ello, haga que la cadena de certificados y la clave privada correspondiente estén disponibles en la ruta del archivo. De esta forma, se puede obtener acceso a estos recursos a través del proxy sidecar de Envoy.

Secret Discovery Service (SDS) de Envoy

Envoy obtiene secretos, por ejemplo, los certificados TLS, de un punto de conexión específico a través del protocolo Secrets Discovery. Para obtener más información sobre este protocolo, consulte la documentación de SDS de Envoy.

App Mesh configura el proxy de Envoy para que utilice un socket de dominio de Unix local en el proxy para que sirva como punto de conexión de Secret Discovery Service (SDS) cuando SDS sirva de origen para sus certificados y cadenas de certificados. Puede configurar la ruta a este punto de conexión mediante la variable de entorno `APPMESH_SDS_SOCKET_PATH`.

> **Important**
>
> El protocolo Secrets Discovery Service local que utiliza el socket de dominio de Unix es compatible con el proxy de App Mesh Envoy versión 1.15.1.0 y versiones posteriores. App Mesh es compatible con el protocolo SDS V2 mediante gRPC.

Integración con el entorno en tiempo de ejecución de SPIFFE (SPIRE)

Puede utilizar cualquier implementación sidecar de la API de SDS, incluidas las cadenas de herramientas existentes, por ejemplo, el entorno en tiempo de ejecución de SPIFFE (SPIRE). SPIRE está diseñado para permitir la implementación de la autenticación TLS mutua entre varias cargas de trabajo en sistemas distribuidos. Acredita la identidad de las cargas de trabajo en tiempo de ejecución. SPIRE también ofrece claves y certificados específicos de la carga de trabajo, de corta duración y que rotan automáticamente directamente a las cargas de trabajo.

Debe configurar el agente de SPIRE como proveedor de SDS para Envoy. Permita que suministre directamente a Envoy el material clave que necesita para proporcionar una autenticación TLS mutua. Ejecute los agentes de SPIRE en sidecars junto a los proxies de Envoy. El agente se encarga de volver a generar las claves y certificados de corta duración según sea necesario. El agente acredita a Envoy y determina qué identidades del servicio y certificados de la autoridad de certificación debe poner a disposición de Envoy cuando este se conecte al servidor de SDS expuesto por el agente de SPIRE.

Durante este proceso, las identidades del servicio y los certificados de la autoridad de certificación se rotan y las actualizaciones se transmiten a Envoy. Envoy los aplica inmediatamente a las nuevas conexiones sin interrupciones ni tiempos de inactividad y sin que las claves privadas entren en contacto con el sistema de archivos.

Cómo App Mesh configura Envoys para negociar TLS

App Mesh utiliza la configuración de punto de conexión de malla tanto del cliente como del servidor para determinar cómo configurar la comunicación entre los Envoys en una malla.

Con políticas de cliente

Cuando una política de cliente exige el uso de TLS y uno de los puertos de la política de cliente coincide con el puerto de la política de servidor, la política de cliente se utiliza para configurar el contexto de validación de TLS del cliente. Por ejemplo, si la política de cliente de una puerta de

enlace virtual coincide con la política de servidor de un nodo virtual, se intentará negociar TLS entre los proxies utilizando la configuración definida en la política de cliente de la puerta de enlace virtual. Si la política de cliente no coincide con el puerto de la política de servidor, la seguridad TLS entre los proxies puede negociarse o no, según la configuración de TLS de la política de servidor.

Sin políticas de cliente

Si el cliente no ha configurado una política de cliente o la política de cliente no coincide con el puerto del servidor, App Mesh utilizará el servidor para determinar si se debe negociar o no la seguridad TLS con el cliente y cómo hacerlo. Por ejemplo, si una puerta de enlace virtual no ha especificado una política de cliente y un nodo virtual no ha configurado la terminación de TLS, la seguridad TLS no se negociará entre los proxies. Si un cliente no ha especificado una política de cliente coincidente y se ha configurado un servidor con los modos `STRICT` o `PERMISSIVE` de TLS, los proxies se configurarán para negociar la seguridad TLS. En función de cómo se hayan proporcionado los certificados para la terminación de TLS, se aplicará el siguiente comportamiento adicional.

- ACM-managed Certificados TLS: cuando un servidor ha configurado la terminación de TLS mediante un ACM-managed certificado, App Mesh configura automáticamente los clientes para que negocien el TLS y validen el certificado con la CA del usuario raíz a la que se encadena el certificado.
- File-based Certificados TLS: cuando un servidor ha configurado la terminación de TLS mediante un certificado del sistema de archivos local del proxy, App Mesh configura automáticamente un cliente para negociar TLS, pero el certificado del servidor no se valida.

Nombres alternativos de asunto

Si lo desea, puede especificar una lista de nombres alternativos de asunto (SAN) en los que pueda confiar. Los SAN deben estar en formato FQDN o URI. Si se proporcionan SAN, Envoy verifica que el nombre alternativo del sujeto del certificado presentado coincida con uno de los nombres de esta lista.

Si no especifica nombres SAN en el punto de conexión de malla de terminación, el proxy de Envoy de ese nodo no verifica el SAN en un certificado de cliente del mismo nivel. Si no especifica nombres SAN en el punto de conexión de malla de origen, el SAN en el certificado proporcionado por el punto de conexión de terminación debe coincidir con la configuración de detección de servicios del punto de conexión de malla.

Para obtener más información, consulte App Mesh TLS: requisitos del certificado.

> **Important**
>
> Solo puede usar nombres SAN comodín si la política de cliente para TLS está establecida en `not enforced`. Si la política de cliente del nodo virtual o la puerta de enlace virtual del cliente está configurada para aplicar TLS, no podrá aceptar un SAN comodín.

Verificación del cifrado

Una vez que haya activado TLS, puede consultar el proxy de Envoy para confirmar que la comunicación está cifrada. El proxy de Envoy emite estadísticas sobre los recursos que pueden ayudarlo a saber si su comunicación TLS funciona correctamente. Por ejemplo, el proxy de Envoy registra estadísticas sobre el número de protocolos de enlace TLS que ha negociado satisfactoriamente para un punto de conexión de malla específico. Determine cuántos protocolos de enlace de manos TLS satisfactorios hubo para un punto de conexión de malla denominado *my-mesh-endpoint* mediante el siguiente comando.

```
curl -s 'http://my-mesh-endpoint.apps.local:9901/stats' | grep ssl.handshake
```

En el resultado que devolvió el siguiente ejemplo, había tres protocolos de enlace para el punto de conexión de malla, por lo que la comunicación está cifrada.

```
listener.0.0.0.0_15000.ssl.handshake: 3
```

El proxy de Envoy también emite estadísticas cuando la negociación de TLS fracasa. Determine si hubo errores de TLS en el punto de conexión de malla.

```
curl -s 'http://my-mesh-endpoint.apps.local:9901/stats' | grep -e "ssl.*\(fail\|error
\)"
```

En el resultado que devolvió el ejemplo, no hubo errores en varias estadísticas, por lo que la negociación de TLS se realizó correctamente.

```
listener.0.0.0.0_15000.ssl.connection_error: 0
listener.0.0.0.0_15000.ssl.fail_verify_cert_hash: 0
listener.0.0.0.0_15000.ssl.fail_verify_error: 0
listener.0.0.0.0_15000.ssl.fail_verify_no_cert: 0
```

```
listener.0.0.0.0_15000.ssl.ssl.fail_verify_san: 0
```

Para obtener más información sobre las estadísticas de TLS de Envoy, consulte Estadísticas del oyente de Envoy.

Renovación de certificados

AWS Private CA

Al renovar un certificado con ACM, el certificado renovado se distribuirá automáticamente a los proxies conectados en un plazo de 35 minutos a partir de la finalización de la renovación. Recomendamos utilizar la renovación administrada para renovar automáticamente los certificados cuando se acerque el final de su período de validez. Para obtener más información, consulte la sección Renovación gestionada de los Amazon-Issued certificados de ACM en la Guía del usuario.
AWS Certificate Manager

Su propio certificado

Al utilizar un certificado del sistema de archivos local, Envoy no volverá a cargar automáticamente el certificado cuando se modifique. Puede reiniciar o volver a implementar el proceso de Envoy para cargar un certificado nuevo. También puede colocar un certificado más reciente en una ruta de archivo diferente y actualizar la configuración del nodo virtual o la puerta de enlace con esa ruta de archivo.

Configure las cargas de trabajo de Amazon ECS para usar la autenticación TLS con AWS App Mesh

Puede configurar su malla para que utilice la autenticación TLS. Asegúrese de que los certificados estén disponibles para los sidecars proxy de Envoy que añada a sus cargas de trabajo. Puede asociar un volumen de EBS o EFS a su sidecar de Envoy, o puede almacenar y recuperar certificados de AWS Secrets Manager.

- Si utiliza la distribución de certificados basada en archivos, asocie un volumen de EBS o EFS a su sidecar de Envoy. Asegúrese de que la ruta al certificado y la clave privada coincidan con la que están configurados. AWS App Mesh
- Si utilizas la SDS-based distribución, añade un sidecar que implemente la API SDS de Envoy con acceso al certificado.

> **Note**
>
> Amazon ECS no admite SPIRE.

Configure las cargas de trabajo de Kubernetes para usar la autenticación TLS con AWS App Mesh

Puede configurar el AWS App Mesh Controller para Kubernetes para habilitar la autenticación TLS para los backends y los oyentes del servicio de nodo virtual y puerta de enlace virtual. Asegúrese de que los certificados estén disponibles para los sidecars del proxy de Envoy que añada a sus cargas de trabajo. Puede ver un ejemplo de cada tipo de distribución en la sección tutorial de Autenticación TLS mutua.

- Si utiliza la distribución de certificados basada en archivos, asocie un volumen de EBS o EFS a su sidecar de Envoy. Asegúrese de que la ruta al certificado y a la clave privada coincidan con la ruta configurada en el controlador. Como alternativa, puede usar un Kubernetes Secret que esté montado en el sistema de archivos.
- Si utilizas la SDS-based distribución, debes configurar un proveedor de SDS local en el nodo que implemente la API de SDS de Envoy. Envoy conectará con él a través de UDS. Para habilitar la compatibilidad con mTLS basadas en SDS en el AppMesh controlador EKS, defina el `enable-sds` indicador en `true` y proporcione la ruta UDS del proveedor de SDS local al controlador mediante el indicador. `sds-uds-path` Si utiliza helm, debe configurarlos como parte de la instalación del controlador:

```
--set sds.enabled=true
```

> **Note**
>
> No podrá utilizar SPIRE para distribuir sus certificados si usa Amazon Elastic Kubernetes Service (Amazon EKS) en modo Fargate.

Autenticación TLS mutua

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

La autenticación TLS (Seguridad de la capa de transporte) mutua es un componente opcional de TLS que ofrece autenticación bidireccional entre pares. La autenticación TLS mutua añade una capa de seguridad a TLS y permite que sus servicios verifiquen al cliente que realiza la conexión.

El cliente de la relación cliente-servidor también proporciona un certificado X.509 durante el proceso de negociación de la sesión. El servidor utiliza este certificado para identificar y autenticar al cliente. Este proceso ayuda a comprobar si el certificado lo ha emitido una entidad de certificación (CA) de confianza y si el certificado es válido. También utiliza el nombre alternativo del asunto (SAN) en el certificado para identificar al cliente.

Puede habilitar la autenticación TLS mutua para todos los protocolos compatibles con. AWS App Mesh Son TCP, HTTP/1.1, HTTP/2, gRPC.

> **Note**
>
> Con App Mesh, puede configurar la autenticación TLS mutua para las comunicaciones entre los proxies de Envoy desde sus servicios. Sin embargo, las comunicaciones entre sus aplicaciones y los proxies de Envoy no están cifradas.

Certificados de la autenticación TLS mutua

AWS App Mesh admite dos posibles fuentes de certificados para la autenticación TLS mutua. Los certificados de cliente de una política de cliente de TLS y la validación del servidor de una configuración de TLS de oyente se pueden obtener de:

- Sistema de archivos: certificados del sistema de archivos local del proxy de Envoy que se está ejecutando. Para distribuir certificados a Envoy, debe indicar las rutas de los archivos de la cadena de certificados y la clave privada de la API de App Mesh.

- El Servicio de Descubrimiento Secreto (SDS) de Envoy: Bring-your-own sidecars que implementan el SDS y permiten enviar los certificados a Envoy. Entre ellas se incluye el entorno en tiempo de ejecución de SPIFFE (SPIRE).

> **Important**
>
> App Mesh no almacena los certificados ni las claves privadas que se utilizan para la autenticación TLS mutua. En su lugar, Envoy los almacena en la memoria.

Configuración de puntos de conexión de malla

Configure la autenticación TLS mutua para sus puntos de conexión de malla, como nodos virtuales o puertas de enlace. Estos puntos de conexión proporcionan certificados y especifican las autoridades de confianza.

Para ello, es necesario aprovisionar certificados X.509 tanto para el cliente como para el servidor y definir de forma explícita los certificados de las autoridades de confianza en el contexto de la validación tanto de la terminación como del origen de TLS.

Confianza dentro de una malla

Los certificados del servidor se configuran en los oyentes de nodos virtuales (terminación de TLS) y los certificados del cliente se configuran en los backends de servicio de nodos virtuales (origen de TLS). Como alternativa a esta configuración, puede definir una política de cliente predeterminada para todos los backends de servicios de un nodo virtual y, a continuación, si hace falta, puede anular esta política para backends específicos según sea necesario. Las puertas de enlace virtuales solo se pueden configurar con una política de cliente predeterminada que se aplique a todos sus backends.

Puede configurar la confianza en diferentes mallas habilitando la autenticación TLS mutua para el tráfico entrante en las puertas de enlace virtuales de ambas mallas.

Confianza fuera de una red

Especifique los certificados del servidor en el oyente de puerta de enlace virtual para la terminación de TLS. Configure el servicio externo que se comunica con su puerta de enlace virtual para presentar los certificados del lado del cliente. Los certificados deben proceder de una

de las mismas autoridades de certificación (CAs) que utilizan los certificados del servidor en el detector de Virtual Gateway para originar el TLS.

Migración de los servicios a la autenticación TLS mutua

Siga estas pautas para mantener la conectividad al migrar sus servicios existentes en App Mesh a la autenticación TLS mutua.

Migración de servicios que se comunican a través de texto sin formato

1. Habilite el modo `PERMISSIVE` para la configuración de TLS en el punto de conexión del servidor. Este modo permite que el tráfico de texto sin formato se conecte al punto de conexión.
2. Configure la autenticación TLS mutua en su servidor, especificando el certificado del servidor, la cadena de confianza y, opcionalmente, el certificado de confianza. SANs
3. Confirme que la comunicación se realiza a través de una conexión TLS.
4. Configure la autenticación TLS mutua en sus clientes, especificando el certificado del cliente, la cadena de confianza y, opcionalmente, el certificado de confianza. SANs
5. Habilite el modo `STRICT` para la configuración de TLS en el servidor.

Migración de servicios que se comunican a través de TLS

1. Configure los ajustes de TLS mutuo en sus clientes, especificando el certificado del cliente y, opcionalmente, el certificado de confianza. SANs El certificado de cliente no se envía a su backend hasta que el servidor de backend lo solicita.
2. Configure los ajustes de TLS mutuo en su servidor, especificando la cadena de confianza y, opcionalmente, la cadena de confianza. SANs Para ello, el servidor solicita un certificado de cliente.

Verificación de la autenticación TLS mutua

Puede consultar la documentación Seguridad de la capa de transporte: verificar el cifrado para saber exactamente cómo Envoy emite estadísticas relacionadas con TLS. Para la autenticación TLS mutua, debe examinar las siguientes estadísticas:

- `ssl.handshake`
- `ssl.no_certificate`

- `ssl.fail_verify_no_cert`
- `ssl.fail_verify_san`

En conjunto, los dos ejemplos de estadísticas siguientes muestran que todas las conexiones TLS correctas que terminan en el nodo virtual tienen su origen en un cliente que proporcionó un certificado.

```
listener.0.0.0.0_15000.ssl.handshake: 3
```

```
listener.0.0.0.0_15000.ssl.no_certificate: 0
```

El siguiente ejemplo de estadística muestra que las conexiones de un nodo de cliente virtual (o puerta de enlace) a un nodo virtual de back-end produjeron un error. El nombre alternativo del sujeto (SAN) que se presenta en el certificado del servidor no coincide con ninguno de los nombres en los que SANs confía el cliente.

```
cluster.cds_egress_my-mesh_my-backend-node_http_9080.ssl.fail_verify_san: 5
```

Tutoriales de la autenticación TLS mutua de App Mesh

- Tutorial de autenticación TLS mutua: en este tutorial se describe cómo puede usar la CLI de App Mesh para crear una aplicación en color con autenticación TLS mutua.
- Tutorial de TLS mutua basada en SDS de Amazon EKS: este tutorial explica cómo puede utilizar la autenticación basada en SDS de TLS mutua con Amazon EKS y el entorno en tiempo de ejecución de SPIFFE (SPIRE).
- Tutorial de TLS mutua basada en archivos de Amazon EKS: este tutorial explica cómo puede utilizar la autenticación basada en archivos de TLS mutua con Amazon EKS y el entorno en tiempo de ejecución de SPIFFE (SPIRE).

Cómo AWS App Mesh funciona con IAM

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte a. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App

Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

AWS Identity and Access Management (IAM) es una herramienta Servicio de AWS que ayuda al administrador a controlar de forma segura el acceso a los AWS recursos. Los administradores de IAM controlan a quién se puede autenticar (puede iniciar sesión) y autorizar (tiene permisos) para utilizar los recursos de App Mesh. La IAM es una Servicio de AWS herramienta que puede utilizar sin coste adicional.

Temas

- Público
- Autenticación con identidades
- Administración del acceso con políticas
- ¿Cómo AWS App Mesh funciona con IAM
- AWS App Mesh ejemplos de políticas basadas en la identidad
- AWS políticas gestionadas para App Mesh
- Uso de roles vinculados a servicios de App Mesh
- Autorización de proxy de Envoy
- Solución de problemas AWS App Mesh de identidad y acceso

Público

La forma de usar AWS Identity and Access Management (IAM) varía según la función que desempeñes:

- Usuario del servicio: solicite permisos al administrador si no puede acceder a las características (consulte Solución de problemas AWS App Mesh de identidad y acceso).
- Administrador del servicio: determine el acceso de los usuarios y envíe las solicitudes de permiso (consulte ¿Cómo AWS App Mesh funciona con IAM).
- Administrador de IAM: escribe las políticas para administrar el acceso (consulte AWS App Mesh ejemplos de políticas basadas en la identidad).

Autenticación con identidades

La autenticación es la forma en que inicias sesión AWS con tus credenciales de identidad. Debe autenticarse como usuario de Usuario raíz de la cuenta de AWS IAM o asumir una función de IAM.

Puede iniciar sesión como una identidad federada con las credenciales de una fuente de identidad, como AWS IAM Identity Center (IAM Identity Center), la autenticación de inicio de sesión único o las credenciales. Google/Facebook Para obtener más información sobre el inicio de sesión, consulte Cómo iniciar sesión en la Cuenta de AWS en la Guía del usuario de AWS Sign-In .

Para el acceso programático, AWS proporciona un SDK y una CLI para firmar criptográficamente las solicitudes. Para obtener más información, consulte AWS Signature Version 4 para solicitudes de API en la Guía del usuario de IAM.

Cuenta de AWS usuario root

Al crear un Cuenta de AWS, se comienza con una identidad de inicio de sesión denominada usuario Cuenta de AWS raíz que tiene acceso completo a todos Servicios de AWS los recursos. Se recomiendaencarecidamente que no utilice el usuario raíz para las tareas diarias. Para ver la lista completa de las tareas que requieren credenciales de usuario raíz, consulte Tareas que requieren credenciales de usuario raíz en la Guía del usuario de IAM.

Usuarios y grupos de IAM

Un usuario de IAM es una identidad con permisos específicos para una sola persona o aplicación. Recomendamos el uso de credenciales temporales en lugar de usuarios de IAM con credenciales de larga duración. Para obtener más información, consulte Exigir a los usuarios humanos que utilicen la federación con un proveedor de identidad para acceder AWS mediante credenciales temporales en la Guía del usuario de IAM.

Un grupo de IAM especifica un conjunto de usuarios de IAM y facilita la administración de los permisos para grupos grandes de usuarios. 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 con permisos específicos que proporciona credenciales temporales. Puede asumir un rol cambiando de un rol de usuario a uno de IAM (consola) o llamando a una AWS CLI operación de AWS API. Para obtener más información, consulte Métodos para asumir un rol en la Guía del usuario de IAM.

Los roles de IAM son útiles para el acceso de usuario federado, los permisos de usuario de IAM temporales, el acceso entre cuentas, el acceso entre servicios y las aplicaciones que se ejecutan en Amazon EC2. Para obtener más información, consulte Acceso a recursos entre cuentas en IAM en la Guía del usuario de IAM.

Administración del acceso con políticas

El acceso se controla AWS creando políticas y adjuntándolas a AWS identidades o recursos. Una política define los permisos cuando están asociados a una identidad o un recurso. AWS evalúa estas políticas cuando un director hace una solicitud. La mayoría de las políticas se almacenan AWS como documentos JSON. Para obtener más información sobre los documentos de políticas de JSON, consulte Información general de políticas de JSON en la Guía del usuario de IAM.

Mediante las políticas, los administradores especifican quién tiene acceso a qué, definiendo qué entidad principal puede realizar acciones sobre qué recursos y en qué condiciones.

De forma predeterminada, los usuarios y los roles no tienen permisos. Un administrador de IAM crea políticas de IAM y las agrega a roles, que los usuarios pueden asumir posteriormente. Las políticas de IAM definen permisos independientemente del método que se utilice para realizar la operación.

Políticas basadas en identidades

Las políticas basadas en identidad son documentos de política de permisos JSON que asocia a una identidad (usuario, grupo o rol). Estas políticas controlan qué acciones pueden realizar las identidades, en qué recursos y en qué condiciones. Para obtener más información sobre cómo crear una política basada en la identidad, consulte Definición de permisos de IAM personalizados con políticas administradas por el cliente en la Guía del usuario de IAM.

Las políticas basadas en identidad pueden ser políticas insertadas (incrustadas directamente en una sola identidad) o políticas administradas (políticas independientes asociadas a varias identidades). Para obtener información sobre cómo elegir entre políticas administradas e insertadas, consulte Selección 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 políticas JSON que se asocian a un recurso. Los ejemplos incluyen 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. Debe especificar una entidad principal en una política basada en recursos.

Las políticas basadas en recursos son políticas insertadas que se encuentran en ese servicio. No puedes usar políticas AWS gestionadas de IAM en una política basada en recursos.

Listas de control de acceso () ACLs

Las listas de control de acceso (ACLs) controlan qué responsables (miembros de la cuenta, usuarios o roles) tienen permisos para acceder a un recurso. ACLs son similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de políticas JSON.

Amazon S3 y Amazon VPC son ejemplos de servicios compatibles. AWS WAF ACLs Para obtener más información ACLs, consulte la descripción general de la lista de control de acceso (ACL) en la Guía para desarrolladores de Amazon Simple Storage Service.

Otros tipos de políticas

AWS admite tipos de políticas adicionales que pueden establecer los permisos máximos otorgados por los tipos de políticas más comunes:

- Límites de permisos: establecen los permisos máximos que una política basada en identidad puede conceder a una entidad de IAM. Para obtener más información, consulte Límites de permisos para las entidades de IAM en la Guía del usuario de IAM.
- Políticas de control de servicios (SCPs): especifican los permisos máximos para una organización o unidad organizativa en AWS Organizations. Para obtener más información, consulte Políticas de control de servicios en la Guía del usuario de AWS Organizations .
- Políticas de control de recursos (RCPs): establece los permisos máximos disponibles para los recursos de tus cuentas. Para obtener más información, consulte Políticas de control de recursos (RCPs) en la Guía del AWS Organizations usuario.
- Políticas de sesión: políticas avanzadas que se pasan como parámetro cuando se crea una sesión temporal para un rol o un usuario federado. Para obtener 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 saber cómo se AWS determina si se debe permitir una solicitud cuando se trata de varios tipos de políticas, consulte la lógica de evaluación de políticas en la Guía del usuario de IAM.

¿Cómo AWS App Mesh funciona con IAM

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte a. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Antes de utilizar IAM para administrar el acceso a App Mesh, debe comprender qué características de IAM están disponibles para su uso con App Mesh. Para obtener una visión general de cómo funcionan App Mesh y otros AWS servicios con IAM, consulte AWS Servicios que funcionan con IAM en la Guía del usuario de IAM.

Temas

- Políticas basadas en identidad de App Mesh
- Políticas basadas en recursos de App Mesh
- Autorización basada en etiquetas de App Mesh
- Roles de IAM de App Mesh

Políticas basadas en identidad de App Mesh

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. App Mesh admite acciones, claves de condición y recursos específicos. Para obtener información sobre todos los elementos que utiliza en una política JSON, consulte Referencia de los elementos de las políticas JSON de IAM en la Guía del usuario de IAM.

Acciones

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Action` de una política JSON describe las acciones que puede utilizar para conceder o denegar el acceso en una política. Incluya acciones en una política para conceder permisos y así llevar a cabo la operación asociada.

Las acciones de políticas de App Mesh utilizan el siguiente prefijo antes de la acción: `appmesh:` Por ejemplo, para conceder a alguien permiso para enumerar las mallas de una cuenta con la operación `appmesh:ListMeshes` de la API, incluya la acción `appmesh:ListMeshes` en su política. Las instrucciones de la política deben incluir un elemento `Action` o un elemento `NotAction`.

Para especificar varias acciones de en una única instrucción, sepárelas con comas del siguiente modo.

```
"Action": [
      "appmesh:ListMeshes",
      "appmesh:ListVirtualNodes"
]
```

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": "appmesh:Describe*"
```

Para ver una lista de las acciones de App Mesh, consulte Acciones definidas por AWS App Mesh en la Guía del usuario de IAM.

Recursos

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Resource` de la política JSON especifica el objeto u objetos a los que se aplica la acción. Como práctica recomendada, especifique un recurso utilizando el Nombre de recurso de Amazon (ARN). En el caso de las acciones que no admiten permisos por recurso, utilice un carácter comodín (*) para indicar que la instrucción se aplica a todos los recursos.

```
"Resource": "*"
```

El recurso `mesh` de App Mesh tiene el siguiente ARN.

```
arn:${Partition}:appmesh:${Region}:${Account}:mesh/${MeshName}
```

Para obtener más información sobre el formato de ARNs, consulte Amazon Resource Names (ARNs) y AWS Service Namespaces.

Por ejemplo, para especificar la malla nombrada *apps* en la *Region-code* región de su declaración, utilice el siguiente ARN.

```
arn:aws:appmesh:Region-code:111122223333:mesh/apps
```

Para especificar todas las instancias que pertenecen a una cuenta específica, utilice el carácter comodín (*).

```
"Resource": "arn:aws:appmesh:Region-code:111122223333:mesh/*"
```

Algunas acciones de App Mesh, como las empleadas para la creación de recursos, no se pueden llevar a cabo en un recurso específico. En dichos casos, debe utilizar el carácter comodín (*).

```
"Resource": "*"
```

En muchas acciones de la API de App Mesh se utilizan varios recursos. Por ejemplo, `CreateRoute` crea una ruta con un destino de nodo virtual, por lo que un usuario de IAM debe tener permisos para usar la ruta y el nodo virtual. Para especificar varios recursos en una sola sentencia, sepárelos ARNs con comas.

```
"Resource": [
      "arn:aws:appmesh:Region-code:111122223333:mesh/apps/virtualRouter/serviceB/route/
*",
      "arn:aws:appmesh:Region-code:111122223333:mesh/apps/virtualNode/serviceB"
]
```

Para ver una lista de los tipos de recursos de App Mesh y sus tipos ARNs, consulte Recursos definidos por AWS App Mesh en la Guía del usuario de IAM. Para obtener información sobre las acciones con las que puede especificar el ARN de cada recurso, consulte Acciones definidas por AWS App Mesh.

Claves de condición

App Mesh admite el uso de algunas claves de condición globales. Para ver todas las claves de condición globales de AWS , consulte Claves de contexto de condición globales de AWSen la Guía del usuario de IAM. Para ver una lista de las claves de condición globales admitidas por App Mesh, consulte Claves de condición de AWS App Mesh en la Guía del usuario de IAM. Para saber qué acciones y recursos puede utilizar con una clave de condición, consulte Acciones definidas por AWS App Mesh.

Ejemplos

Para ver ejemplos de políticas basadas en identidades de App Mesh, consulte AWS App Mesh ejemplos de políticas basadas en la identidad.

Políticas basadas en recursos de App Mesh

App Mesh no admite políticas basadas en recursos. Sin embargo, si utilizas el servicio AWS Resource Access Manager (AWS RAM) para compartir una malla entre AWS los servicios, el servicio aplicará a tu malla una política basada en recursos. AWS RAM Para obtener más información, consulte Otorgar permisos para una malla.

Autorización basada en etiquetas de App Mesh

Puede asociar etiquetas a los recursos de App Mesh o transferirlas en una solicitud a App Mesh. 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 `appmesh:ResourceTag/key-name`, `aws:RequestTag/key-name` o `aws:TagKeys`. Para obtener más información sobre el etiquetado de los recursos de App Mesh, consulte Etiquetado AWS de recursos.

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 Creación de mallas de App Mesh con etiquetas restringidas.

Roles de IAM de App Mesh

Un rol de IAM es una entidad de tu AWS cuenta que tiene permisos específicos.

Uso de credenciales temporales con App Mesh

Puede utilizar credenciales temporales para iniciar sesión con federación, asumir un rol de IAM o asumir un rol de acceso entre cuentas. Para obtener credenciales de seguridad temporales, puede llamar a operaciones de AWS STS API como AssumeRoleo GetFederationToken.

App Mesh admite el uso de credenciales temporales.

Roles vinculados a servicios

Los roles vinculados a un servicio permiten a AWS los servicios acceder a los recursos de otros servicios para completar una acción en tu nombre. Los roles vinculados a servicios aparecen en la

cuenta de IAM y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.

App Mesh admite roles vinculados a servicios. Para obtener más información acerca de cómo crear o administrar roles vinculados a servicios de App Mesh, consulte Uso de roles vinculados a servicios de App Mesh.

Roles de servicio

Esta característica permite que un servicio asuma un rol de servicio en su nombre. Este rol permite que el servicio obtenga acceso a los recursos de otros servicios para completar una acción en su nombre. Los roles de servicio aparecen en su cuenta de IAM y son propiedad de la cuenta. Esto significa que un administrador de IAM puede cambiar los permisos de este rol. Sin embargo, hacerlo podría deteriorar la funcionalidad del servicio.

App Mesh no admite roles de servicio.

AWS App Mesh ejemplos de políticas basadas en la identidad

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte a. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

De forma predeterminada, los roles y usuarios de IAM no tienen permiso para crear o modificar recursos de App Mesh. Tampoco pueden realizar tareas mediante la AWS API Consola de administración de AWS AWS CLI, o. Un administrador de IAM debe crear políticas de IAM que concedan permisos a los usuarios y a los roles para realizar operaciones de la API concretas en los recursos especificados que necesiten. El administrador debe adjuntar esas políticas a los usuarios o grupos de IAM que necesiten esos permisos.

Para obtener información acerca de cómo crear una política basada en identidad de IAM con estos documentos de políticas JSON de ejemplo, consulte Creación de políticas en la pestaña JSON en la Guía del usuario de IAM.

Temas

- Prácticas recomendadas relativas a políticas

- Uso de la consola de App Mesh
- Permitir a los usuarios consultar sus propios permisos
- Creación de una malla
- Enumeración y descripción de todas las mallas
- Creación de mallas de App Mesh con etiquetas restringidas

Prácticas recomendadas relativas a políticas

Las políticas basadas en identidades determinan si alguien puede crear, obtener acceso o eliminar los recursos de App Mesh de la 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:

- Comience con las políticas AWS administradas y avance hacia los permisos con privilegios mínimos: para empezar a conceder permisos a sus usuarios y cargas de trabajo, utilice las políticas AWS administradas que otorgan permisos en muchos casos de uso comunes. Están disponibles en su. Cuenta de AWS Le recomendamos que reduzca aún más los permisos definiendo políticas administradas por el AWS cliente que sean específicas para sus casos de uso. Con el fin de obtener más información, consulte 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, consulte 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, puede escribir una condición de políticas para especificar que todas las solicitudes deben enviarse utilizando SSL. También puedes usar condiciones para conceder el acceso a las acciones del servicio si se utilizan a través de una acción específica Servicio de AWS, por ejemplo CloudFormation. Para obtener más información, consulte 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.

- Requerir autenticación multifactor (MFA): si tiene un escenario que requiere usuarios de IAM o un usuario raíz en Cuenta de AWS su cuenta, active la MFA para 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.

Uso de la consola de App Mesh

Para acceder a la AWS App Mesh consola, debe tener un conjunto mínimo de permisos. Estos permisos deben permitirte enumerar y ver detalles sobre los recursos de App Mesh de tu AWS cuenta. Si crea una política basada en identidad que sea más restrictiva que el mínimo de permisos necesarios, la consola no funcionará del modo esperado para las entidades (usuarios o roles de IAM) que tengan esa política. Puede asociar la política administrada `AWSAppMeshReadOnly` a los usuarios. Para obtener más información, consulte Adición de permisos a un usuario en la Guía del usuario de IAM.

No necesitas conceder permisos mínimos de consola a los usuarios que solo realizan llamadas a la API AWS CLI o a la AWS API. En su lugar, permite acceso únicamente a las acciones que coincidan con la operación de API que intenta realizar.

Permitir a los usuarios consultar sus propios permisos

En este ejemplo, se muestra cómo podría crear una política que permita a los usuarios de IAM ver las políticas administradas e insertadas que se asocian a la identidad de sus usuarios. Esta política incluye permisos para completar esta acción en la consola o mediante programación mediante la API AWS CLI o 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": "*"
        }
    ]
}
```

Creación de una malla

Este ejemplo muestra cómo crear una política que permita a un usuario crear una malla de una cuenta en cualquier región.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "appmesh:CreateMesh",
            "Resource": "arn:aws:appmesh:*:123456789012:CreateMesh"
        }
```

```
    ]
}
```

Enumeración y descripción de todas las mallas

Este ejemplo muestra cómo puede crear una política que permita a un usuario con acceso de solo lectura enumerar o describir todas las mallas.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "appmesh:DescribeMesh",
                "appmesh:ListMeshes"
            ],
            "Resource": "*"
        }
    ]
}
```

Creación de mallas de App Mesh con etiquetas restringidas

Puede utilizar etiquetas en sus políticas de IAM para controlar qué etiquetas se pueden pasar en la solicitud de IAM. Puede especificar qué pares de clave-valor de etiqueta se pueden añadir, cambiar o eliminar en un rol o usuario de IAM. En este ejemplo se muestra cómo se puede crear una política que permita crear una malla, pero solo si la malla se crea con una etiqueta denominada *teamName* y un valor de. *booksTeam*

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": "appmesh:CreateMesh",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/teamName": "booksTeam"
                }
            }
        }
    ]
}
```

También puede asociar esta política al usuario de IAM en su cuenta. Si un usuario intenta crear una malla, esta deberá incluir una etiqueta denominada `teamName` y un valor de `booksTeam`. Si la malla no incluye esta etiqueta ni este valor, no se podrá crear la malla. Para obtener más información, consulte Elementos de la política de JSON de IAM: Condición en la Guía del usuario de IAM.

AWS políticas gestionadas para App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte a AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Una política AWS administrada es una política independiente creada y administrada por. AWS AWS Las políticas administradas están diseñadas para proporcionar permisos para muchos casos de uso comunes, de modo que pueda empezar a asignar permisos a usuarios, grupos y funciones.

Ten en cuenta que es posible que las políticas AWS administradas no otorguen permisos con privilegios mínimos para tus casos de uso específicos, ya que están disponibles para que los usen todos los AWS clientes. Se recomienda definir políticas administradas por el cliente específicas para sus casos de uso a fin de reducir aún más los permisos.

No puedes cambiar los permisos definidos en AWS las políticas administradas. Si AWS actualiza los permisos definidos en una política AWS administrada, la actualización afecta a todas las identidades

principales (usuarios, grupos y roles) a las que está asociada la política. AWS es más probable que actualice una política AWS administrada cuando Servicio de AWS se lance una nueva o cuando estén disponibles nuevas operaciones de API para los servicios existentes.

Para obtener más información, consulte Políticas administradas por AWS en la Guía del usuario de IAM.

AWS política gestionada: AWSApp MeshServiceRolePolicy

Puede adjuntar `AWSAppMeshServiceRolePolicy` a sus entidades de IAM. Permite el acceso a AWS los servicios y recursos utilizados o gestionados por AWS App Mesh.

Para ver los permisos de esta política, consulte AWSAppMeshServiceRolePolicy en la Referencia de la política administrada de AWS .

Para obtener información sobre los detalles de los permisos de `AWSAppMeshServiceRolePolicy`, consulte Permisos de roles vinculados a servicios de App Mesh.

AWS política gestionada: AWSApp MeshEnvoyAccess

Puede adjuntar `AWSAppMeshEnvoyAccess` a sus entidades de IAM. Política de App Mesh Envoy para obtener acceso a la configuración del nodo virtual.

Para ver los permisos de esta política, consulte AWSAppMeshEnvoyAccess en la Referencia de la política administrada de AWS .

AWS política gestionada: AWSApp MeshFullAccess

Puede adjuntar `AWSAppMeshFullAccess` a sus entidades de IAM. Proporciona acceso completo a la AWS App Mesh APIs y Consola de administración de AWS.

Para ver los permisos de esta política, consulte AWSAppMeshFullAccess en la Referencia de la política administrada de AWS .

AWS política gestionada: AWSApp MeshPreviewEnvoyAccess

Puede adjuntar `AWSAppMeshPreviewEnvoyAccess` a sus entidades de IAM. Política de App Mesh Preview Envoy para obtener acceso a la configuración del nodo virtual.

Para ver los permisos de esta política, consulte AWSAppMeshPreviewEnvoyAccess en la Referencia de la política administrada de AWS .

AWS política gestionada: AWSApp MeshPreviewServiceRolePolicy

Puede adjuntar `AWSAppMeshPreviewServiceRolePolicy` a sus entidades de IAM. Permite el acceso a AWS los servicios y recursos utilizados o gestionados por AWS App Mesh.

Para ver los permisos de esta política, consulte AWSAppMeshPreviewServiceRolePolicy en la Referencia de la política administrada de AWS .

AWS política gestionada: AWSApp MeshReadOnly

Puede adjuntar `AWSAppMeshReadOnly` a sus entidades de IAM. Proporciona acceso de solo lectura a y. AWS App Mesh APIs Consola de administración de AWS

Para ver los permisos de esta política, consulte AWSAppMeshReadOnly en la Referencia de la política administrada de AWS .

AWS App Mesh actualizaciones de las políticas gestionadas AWS

Consulte los detalles sobre las actualizaciones de las políticas AWS administradas AWS App Mesh desde que este servicio comenzó a rastrear estos cambios. Para obtener alertas automáticas cuando se produzcan cambios en esta página, suscríbase a la fuente RSS en la página de historial del documento de AWS App Mesh .

Cambio	Descripción	Fecha
AWSAppMeshFullAccess— Política actualizada.	Actualizado `AWSAppMeshFullAccess` para permitir el acceso a la `TagResource` y `UntagResource` APIs.	24 de abril de 2024
AWSAppMeshServiceRolePolicy, AWSServiceRoleForAppMesh— Política actualizada.	Actualizada `AWSServiceRoleForAppMesh` y `AWSAppMeshServiceRolePolicy` para permitir el acceso a la AWS Cloud Map `DiscoverInstancesRevision` API.	12 de octubre de 2023

Para dar acceso, agregue permisos a los usuarios, grupos o roles:

- Usuarios y grupos en AWS IAM Identity Center:

  Cree un conjunto de permisos. Siga las instrucciones de Creación de un conjunto de permisos en la Guía del usuario de AWS IAM Identity Center .

- Usuarios gestionados en IAM a través de un proveedor de identidades:

  Cree un rol para la federación de identidades. Siga las instrucciones descritas en Creación de un rol para un proveedor de identidad de terceros (federación) en la Guía del usuario de IAM.

- Usuarios de IAM:
  - Cree un rol que el usuario pueda aceptar. Siga las instrucciones descritas en Creación de un rol para un usuario de IAM en la Guía del usuario de IAM.
  - (No recomendado) Adjunte una política directamente a un usuario o agregue un usuario a un grupo de usuarios. Siga las instrucciones descritas en Adición de permisos a un usuario (consola) de la Guía del usuario de IAM.

Uso de roles vinculados a servicios de App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

AWS App Mesh utiliza funciones AWS Identity and Access Management vinculadas al servicio (IAM). Un rol vinculado a un servicio es un tipo único de rol de IAM que está vinculado directamente a App Mesh. App Mesh predefine las funciones vinculadas al servicio e incluyen todos los permisos que el servicio requiere para llamar a otros AWS servicios en tu nombre.

Un rol vinculado a un servicio simplifica la configuración de App Mesh porque ya no tendrá que agregar manualmente los permisos necesarios. App Mesh define los permisos de sus roles vinculados a servicios y, a menos que esté definido de otra manera, solo App Mesh 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.

Solo puede eliminar un rol vinculado a servicios después de eliminar sus recursos relacionados. De esta forma, se protegen los recursos de App Mesh, ya que se evita que se puedan eliminar accidentalmente permisos de acceso a los recursos.

Para obtener información acerca de 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 Rol vinculado a un servicio. Elija una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Permisos de roles vinculados a servicios de App Mesh

App Mesh usa el rol vinculado al servicio denominado AWSServiceRoleForAppMesh: el rol permite a App Mesh llamar a AWS los servicios en tu nombre.

El rol AWSService RoleForAppMesh vinculado al servicio confía en que el `appmesh.amazonaws.com` servicio asuma el rol.

Detalles del permiso

- `servicediscovery:DiscoverInstances`: permite que App Mesh realice acciones en todos los recursos de AWS .
- `servicediscovery:DiscoverInstancesRevision`- Permite que App Mesh complete acciones en todos los AWS recursos.

AWSServiceRoleForAppMesh

Esta política incluye los permisos siguientes:

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
  {
   "Sid": "CloudMapServiceDiscovery",
   "Effect": "Allow",
   "Action": [
    "servicediscovery:DiscoverInstances",
    "servicediscovery:DiscoverInstancesRevision"
```

```
    ],
    "Resource": "*"
   },
   {
    "Sid": "ACMCertificateVerification",
    "Effect": "Allow",
    "Action": [
     "acm:DescribeCertificate"
    ],
    "Resource": "*"
   }
  ]
}
```

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 a un servicio para App Mesh

Si creaste una malla después del 5 de junio de 2019 en la Consola de administración de AWS AWS CLI, la o la AWS API, App Mesh creó el rol vinculado al servicio automáticamente. Para que el rol vinculado al servicio se haya creado para usted, la cuenta de IAM que utilizó para crear la malla debe tener asociada la política de IAM `AWSAppMeshFullAccess` o una política asociada que contenga el permiso `iam:CreateServiceLinkedRole`. 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. Al crear una malla, App Mesh se encarga de crear de nuevo el rol vinculado al servicio por usted. Si su cuenta solo contiene mallas creadas antes del 5 de junio de 2019 y quiere usar el rol vinculado al servicio con esas mallas, puede crear el rol mediante la consola de IAM.

Puede utilizar la consola de IAM para crear un rol vinculado a servicios con el caso de uso de App Mesh. En la AWS CLI o en la AWS API, crea un rol vinculado al servicio con el nombre del servicio. `appmesh.amazonaws.com` Para obtener más información, consulte Creación de un rol vinculado a un servicio en la Guía del usuario de IAM. Si elimina este rol vinculado al servicio, puede utilizar este mismo proceso para volver a crear el rol.

Modificación de un rol vinculado a un servicio de App Mesh

App Mesh no permite editar el rol AWSService RoleForAppMesh vinculado al servicio. 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 Edición de un rol vinculado a servicios en la Guía del usuario de IAM.

Eliminación de un rol vinculado a un servicio de App Mesh

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 App Mesh está utilizando el rol cuando intenta eliminar los recursos, la eliminación podría producir un error. En tal caso, espere unos minutos e intente de nuevo la operación.

Para eliminar los recursos de App Mesh utilizados por AWSService RoleForAppMesh

1. Elimine todas las rutas definidas para todos los enrutadores de la malla.
2. Elimine todos los enrutadores virtuales de la malla.
3. Elimine todos los servicios virtuales de la malla.
4. Elimine todos los nodos virtuales de la malla.
5. Elimine la malla.

Realice los pasos anteriores para todas las mallas de su cuenta.

Para eliminar manualmente el rol vinculado a servicios mediante IAM

Utilice la consola de IAM AWS CLI, la o la AWS API para eliminar la función vinculada al AWSService RoleForAppMesh servicio. 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 a servicios de App Mesh

App Mesh admite el uso de roles vinculados a servicios en todas las regiones en las que el servicio esté disponible. Para obtener más información, consulte Puntos de conexión y cuotas de App Mesh.

Autorización de proxy de Envoy

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

La autorización de proxy autoriza al proxy de Envoy que se ejecuta en una tarea de Amazon ECS, en un pod de Kubernetes que se ejecuta en Amazon EKS o que se ejecuta en una instancia de Amazon EC2 a leer la configuración de uno o varios puntos de conexión de malla desde Envoy Management Service de App Mesh. Para las cuentas de clientes que ya tengan Envoys conectados a su punto de conexión de App Mesh antes del 26/04/2021, se requiere autorización de proxy para los nodos virtuales que utilizan seguridad de la capa de transporte (TLS) y para las puertas de enlace virtuales (con o sin TLS). Para las cuentas de clientes que quieran conectar Envoys a su punto de conexión de App Mesh después del 26/04/2021, se requiere la autorización de proxy para todas las funciones de App Mesh. Se recomienda que todas las cuentas de los clientes habiliten la autorización de proxy para todos los nodos virtuales, incluso si no utilizan TLS, para tener una experiencia segura y coherente al utilizar IAM para la autorización de recursos específicos. La autorización de proxy requiere que se especifique el permiso `appmesh:StreamAggregatedResources` en una política de IAM. La política debe estar asociada a un rol de IAM y dicho rol de IAM debe estar asociado al recurso informático en el que se aloja el proxy.

Creación de una política de IAM

Si desea que todos los puntos de conexión de malla de una malla de servicios puedan leer la configuración de todos los puntos de conexión de malla, vaya a la sección Creación de un rol de IAM. Si desea limitar los puntos de conexión de malla desde los que se puede leer la configuración por puntos de conexión de malla individuales, debe crear una o más políticas de IAM. Se recomienda limitar los puntos de conexión de malla desde los que se puede leer la configuración a solo el proxy de Envoy que se ejecuta en recursos informáticos específicos. Cree una política de IAM y añada el permiso `appmesh:StreamAggregatedResources` a la política. El siguiente ejemplo de política permite configurar los nodos virtuales denominados `serviceBv1` y `serviceBv2` para que se lean en una malla de servicios. No se puede leer la configuración de ningún otro nodo virtual definido en

la malla de servicios. Para obtener más información acerca de la creación o edición de políticas de IAM, consulte Creación de políticas de IAM y Editar políticas de IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "appmesh:StreamAggregatedResources",
            "Resource": [
                "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/
serviceBv1",
                "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/
serviceBv2"
            ]
        }
    ]
}
```

Puede crear varias políticas y que cada una restrinja el acceso a diferentes puntos de conexión de malla.

Creación de un rol de IAM

Si desea que todos los puntos de conexión de malla de una malla de servicios puedan leer la configuración de todos los puntos de conexión de malla, solo tiene que crear un rol de IAM. Si desea limitar los puntos de conexión de malla cuya configuración puedan leer los puntos de conexión de malla individuales, debe crear un rol para cada política que creó en el paso anterior. Siga las instrucciones del recurso informático en el que se ejecuta el proxy.

- Amazon EKS: si quiere usar un único rol, puede usar el rol existente que se creó y asignó a los nodos de trabajo cuando creó el clúster. Para utilizar varios roles, el clúster debe cumplir los requisitos definidos en la sección Habilitar roles de IAM para las cuentas de servicio de su clúster. Cree los roles de IAM y asócielos a las cuentas de servicio de Kubernetes. Para obtener más información, consulte Creación de una política y un rol de IAM para su cuenta de servicio y Especificación de un rol de IAM para su cuenta de servicio.

- Amazon ECS: seleccione Servicio de AWS , elija Elastic Container Service y, a continuación, elija el caso de uso Tarea de Elastic Container Service al crear su rol de IAM.
- Amazon EC2: seleccione Servicio de AWS , elija EC2 y, a continuación, elija el caso de uso EC2 al crear su rol de IAM. Esto se aplica tanto si aloja el proxy directamente en una instancia de Amazon EC2 como si se ejecuta en una instancia de Kubernetes.

Para obtener más información sobre cómo crear un rol de IAM, consulte Creación de un rol para un AWS servicio.

Asociar una política de IAM

Si desea que todos los puntos de conexión de malla de una malla de servicios puedan leer la configuración de todos los puntos de conexión de malla, asocie la política de IAM administrada `AWSAppMeshEnvoyAccess` al rol de IAM que creó en el paso anterior. Si desea limitar los puntos de conexión de malla cuya configuración puedan leer los puntos de conexión de malla individuales, asocie cada política que haya creado a cada rol que haya creado. Para obtener más información sobre cómo asociar una política de IAM personalizada o administrada a un rol de IAM, consulte Adición de permisos de identidad de IAM.

Asociación de un rol de IAM

Asocie cada rol de IAM al recurso informático correspondiente:

- Amazon EKS: si ha asociado la política al rol asociado a sus nodos de trabajo, puede omitir este paso. Si ha creado roles independientes, asigne cada rol a una cuenta de servicio de Kubernetes independiente y asigne cada cuenta de servicio a una especificación de implementación de un pod de Kubernetes individual que incluya el proxy de Envoy. Para obtener más información, consulte Especificación de un rol de IAM para su cuenta de servicio en la Guía del usuario de Amazon EKS y Configurar cuentas de servicio para pods en la documentación de Kubernetes.
- Amazon ECS: asocie un rol de tarea de Amazon ECS a la definición de la tarea que incluya el proxy de Envoy. La tarea se puede implementar con el tipo de lanzamiento de EC2 o Fargate. Para obtener más información sobre cómo crear un rol de tarea de Amazon ECS y asociarlo a una tarea, consulte Especificar un rol de IAM para sus tareas.
- Amazon EC2: el rol de IAM se debe asociar a la instancia de Amazon EC2 que aloja el proxy de Envoy. Para obtener más información sobre cómo asociar un rol a una instancia de Amazon EC2, consulte He creado un rol de IAM y ahora quiero asignarlo a una instancia de EC2.

Confirmación del permiso

Confirme que el permiso `appmesh:StreamAggregatedResources` está asignado al recurso informático en el que aloja el proxy seleccionando uno de los nombres del servicio de computación.

Amazon EKS

Se puede asignar una política personalizada al rol asignado a los nodos de trabajo, a los pods individuales o a ambos. Sin embargo, se recomienda que asigne la política solo a los pods individuales, de modo que pueda restringir el acceso de los pods individuales a los puntos de conexión de malla individuales. Si la política está asociada al rol asignado a los nodos de trabajo, seleccione la pestaña Amazon EC2 y realice los pasos indicados para sus instancias de nodo de trabajo. Para determinar qué rol de IAM se asigna a un pod de Kubernetes, realice los pasos siguientes.

1. Consulte los detalles de una implementación de Kubernetes que incluya el pod al que quiere confirmar que está asignada una cuenta de servicio de Kubernetes. El siguiente comando muestra los detalles de una implementación denominada*my-deployment*.

   ```
   kubectl describe deployment my-deployment
   ```

   En el resultado devuelto, anote el valor a la derecha de `Service Account:`. Si no existe una línea que comience por `Service Account:`, significa que actualmente no hay una cuenta de servicio personalizada de Kubernetes asignada a la implementación. Deberá asignar una. Para obtener más información, consulte Configurar cuentas de servicio de pods en la documentación de Kubernetes.
2. Consulte los detalles de la cuenta de servicio que obtuvo en el paso anterior. El siguiente comando muestra los detalles de una cuenta de servicio denominada*my-service-account*.

   ```
   kubectl describe serviceaccount my-service-account
   ```

   Siempre que la cuenta de servicio de Kubernetes esté asociada a un AWS Identity and Access Management rol, una de las líneas devueltas tendrá un aspecto similar al del ejemplo siguiente.

   ```
   Annotations:         eks.amazonaws.com/role-arn=arn:aws:iam::123456789012:role/
   my-deployment
   ```

En el ejemplo anterior, `my-deployment` es el nombre del rol de IAM al que está asociada la cuenta de servicio. Si el resultado de la cuenta de servicio no contiene una línea similar a la del ejemplo anterior, entonces la cuenta de servicio de Kubernetes no está asociada a ninguna AWS Identity and Access Management cuenta y debes asociarla a una. Para obtener más información, consulte Especificación de un rol de IAM para su cuenta de servicio.

3. Inicie sesión en la consola de Consola de administración de AWS IAM y ábrala en. https://console.aws.amazon.com/iam/
4. En el panel de navegación izquierdo, seleccione Roles. Seleccione el nombre del rol de IAM que anotó en un paso anterior.
5. Confirme que aparece en la lista la política personalizada que creó anteriormente o la política administrada `AWSAppMeshEnvoyAccess`. Si ninguna de las políticas está asociada, deberá asociar una política de IAM al rol de IAM. Si desea asociar una política de IAM personalizada pero no tiene ninguna, debe crear una política de IAM personalizada con los permisos necesarios. Si hay asociada una política de IAM personalizada, selecciónela y confirme que contiene `"Action": "appmesh:StreamAggregatedResources"`. Si no es así, tendrá que añadir ese permiso a su política de IAM personalizada. También puede confirmar que aparece en la lista el Nombre de recurso de Amazon (ARN) adecuado de un punto de conexión de malla específico. ARNs Si no aparece ninguno, puede editar la política para añadir, eliminar o cambiar lo que aparece en la lista ARNs. Para obtener más información, consulte Editar políticas de IAM y Creación de una política de IAM.
6. Repita los pasos anteriores para cada pod de Kubernetes que contenga el proxy de Envoy.

Amazon ECS

1. En la consola de Amazon ECS, elija Definiciones de tareas.
2. Seleccione su tarea de Amazon ECS.
3. En la página Nombre de la definición de tarea, seleccione la definición de la tarea.
4. En la página Definición de tarea, seleccione el enlace del nombre del rol de IAM que está a la derecha de Rol de tarea. Si un rol de IAM no aparece en la lista, debe crear un rol de IAM y asociarlo a su tarea actualizando la definición de la tarea.
5. En la página Resumen, en la pestaña Permisos, confirme que aparece la política personalizada que creó anteriormente o la política administrada `AWSAppMeshEnvoyAccess`. Si ninguna de las dos políticas está asociada, deberá asociar una política de IAM

al rol de IAM. Si desea asociar una política de IAM personalizada pero no tiene ninguna, deberá crear la política de IAM personalizada. Si hay asociada una política de IAM personalizada, selecciónela y confirme que contiene `"Action": "appmesh:StreamAggregatedResources"`. Si no es así, tendrá que añadir ese permiso a su política de IAM personalizada. También puede confirmar que aparece en la lista el Nombre de recurso de Amazon (ARN) adecuado de los puntos de conexión de una malla específica. ARNs Si no aparece ninguna, puede editar la política para añadir, eliminar o cambiar lo que aparece en la lista ARNs. Para obtener más información, consulte Editar políticas de IAM y Creación de una política de IAM.

6. Repita los pasos anteriores para cada definición de tarea que contenga el proxy de Envoy.

Amazon EC2

1. En la consola de Amazon EC2, seleccione Instancias en el panel de navegación de la izquierda.
2. Seleccione una de las instancias que aloje el proxy de Envoy.
3. En la pestaña Descripción, seleccione el enlace del nombre del rol de IAM que está a la derecha del rol de IAM. Si un rol de IAM no aparece en la lista, deberá crear un rol de IAM.
4. En la página Resumen, en la pestaña Permisos, confirme que aparece la política personalizada que creó anteriormente o la política administrada `AWSAppMeshEnvoyAccess`. Si ninguna de las dos políticas está asociada, deberá asociar la política de IAM al rol de IAM. Si desea asociar una política de IAM personalizada pero no tiene ninguna, deberá crear la política de IAM personalizada. Si hay asociada una política de IAM personalizada, selecciónela y confirme que contiene `"Action": "appmesh:StreamAggregatedResources"`. Si no es así, tendrá que añadir ese permiso a su política de IAM personalizada. También puede confirmar que aparece en la lista el Nombre de recurso de Amazon (ARN) adecuado de los puntos de conexión de una malla específica. ARNs Si no aparece ninguna, puede editar la política para añadir, eliminar o cambiar lo que aparece en la lista ARNs. Para obtener más información, consulte Editar políticas de IAM y Creación de una política de IAM.
5. Repita los pasos anteriores para cada instancia en la que aloje el proxy de Envoy.

Solución de problemas AWS App Mesh de identidad y acceso

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS se suspenderá el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Utilice la siguiente información para diagnosticar y solucionar los problemas comunes que puedan surgir cuando trabaje con App Mesh e IAM.

Temas

- No tengo autorización para realizar una acción en App Mesh
- Quiero permitir que personas ajenas a mi AWS cuenta accedan a mis recursos de App Mesh

No tengo autorización para realizar una acción en App Mesh

Si Consola de administración de AWS le indica que no está autorizado a realizar una acción, debe ponerse en contacto con su administrador para obtener ayuda. El administrador es la persona que le proporcionó las credenciales de inicio de sesión.

El siguiente error se produce cuando el usuario de `mateojackson` IAM intenta utilizar la consola para crear un nodo virtual denominado *`my-virtual-node`* en la malla denominada*`my-mesh`*, pero no tiene el `appmesh:CreateVirtualNode` permiso.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to
 perform: appmesh:CreateVirtualNode on resource: arn:aws:appmesh:us-
east-1:123456789012:mesh/my-mesh/virtualNode/my-virtual-node
```

En este caso, Mateo pide a su administrador que actualice sus políticas para permitirle crear un nodo virtual mediante la acción `appmesh:CreateVirtualNode`.

> **Note**
>
> Como un nodo virtual se crea dentro de una malla, la cuenta de Mateo también necesita las acciones `appmesh:DescribeMesh` y `appmesh:ListMeshes` para crear el nodo virtual en la consola.

Quiero permitir que personas ajenas a mi AWS cuenta accedan a mis recursos de App Mesh

Se puede crear un rol que los usuarios de otras cuentas o las personas externas a la organización puedan utilizar para acceder a sus recursos. Se puede especificar una persona de confianza para que asuma el rol. En el caso de los servicios que admiten políticas basadas en recursos o listas de control de acceso (ACLs), puedes usar esas políticas para permitir que las personas accedan a tus recursos.

Para obtener más información, consulte lo siguiente:

- Para saber si App Mesh admite estas características, consulte ¿Cómo AWS App Mesh funciona con IAM.
- Para obtener información sobre cómo proporcionar acceso a los recursos de su Cuentas de AWS propiedad, consulte Proporcionar acceso a un usuario de IAM en otro de su propiedad en la Cuenta de AWS Guía del usuario de IAM.
- Para obtener información sobre cómo proporcionar acceso a tus recursos a terceros Cuentas de AWS, consulta Cómo proporcionar acceso a recursos que Cuentas de AWS son propiedad de terceros en la Guía del usuario de IAM.
- Para obtener información sobre cómo proporcionar acceso mediante una federación de identidades, consulte 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.

Registro de llamadas a la AWS App Mesh API mediante AWS CloudTrail

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

AWS App Mesh está integrado con AWS CloudTrailun servicio que proporciona un registro de las acciones realizadas por un usuario, rol o un Servicio de AWS. CloudTrail captura todas las llamadas a la API de App Mesh como eventos. Las llamadas capturadas incluyen las llamadas desde la consola de App Mesh y las llamadas de código a las operaciones de la API de App Mesh. Con la información recopilada por CloudTrail, puedes determinar la solicitud que se realizó a App Mesh, la dirección IP desde la que se realizó la solicitud, cuándo se realizó y detalles adicionales.

Cada entrada de registro o evento contiene información sobre quién generó la solicitud. La información de identidad del usuario le ayuda a determinar lo siguiente:

- Si la solicitud se realizó con las credenciales del usuario raíz o del usuario.
- Si la solicitud se realizó en nombre de un usuario de IAM Identity Center.
- Si la solicitud se realizó con credenciales de seguridad temporales de un rol o fue un usuario federado.
- Si la solicitud la realizó otro Servicio de AWS.

CloudTrail está activa en tu cuenta Cuenta de AWS cuando creas la cuenta y tienes acceso automáticamente al historial de CloudTrail eventos. El historial de CloudTrail eventos proporciona un registro visible, consultable, descargable e inmutable de los últimos 90 días de eventos de gestión registrados en un. Región de AWSPara obtener más información, consulte Cómo trabajar con el historial de CloudTrail eventos en la Guía del usuario.AWS CloudTrail La visualización del historial de eventos no conlleva ningún CloudTrail cargo.

Para tener un registro continuo de los eventos de Cuenta de AWS los últimos 90 días, crea un almacén de datos de eventos de senderos o CloudTraillagos.

CloudTrail senderos

Un rastro permite CloudTrail entregar archivos de registro a un bucket de Amazon S3. Todos los senderos creados con él Consola de administración de AWS son multirregionales. Puede crear un registro de seguimiento de una sola región o multirregionales mediante la AWS CLI. Se recomienda crear un sendero multirregional, ya que puedes capturar toda la actividad de tu Regiones de AWS cuenta. Si crea un registro de seguimiento de una sola región, solo podrá ver los eventos registrados en la Región de AWS del registro de seguimiento. Para obtener más información acerca de los registros de seguimiento, consulte Creación de un registro de seguimiento para su Cuenta de AWS y Creación de un registro de seguimiento para una organización en la Guía del usuario de AWS CloudTrail .

Puede enviar una copia de sus eventos de administración en curso a su bucket de Amazon S3 sin coste alguno CloudTrail mediante la creación de una ruta; sin embargo, hay cargos por almacenamiento en Amazon S3. Para obtener más información sobre CloudTrail los precios, consulte AWS CloudTrail Precios. Para obtener información acerca de los precios de Amazon S3, consulte Precios de Amazon S3.

CloudTrail Almacenes de datos de eventos en Lake

CloudTrail Lake le permite ejecutar consultas basadas en SQL en sus eventos. CloudTrail Lake convierte los eventos existentes en formato JSON basado en filas al formato Apache ORC. 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 sobre CloudTrail Lake, consulte Cómo trabajar con AWS CloudTrail Lake en la Guía del AWS CloudTrail usuario.

CloudTrail Los almacenes de datos y las consultas sobre eventos de Lake conllevan costes. 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 CloudTrail los precios, consulte AWS CloudTrail Precios.

Eventos de gestión de App Mesh 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. De forma predeterminada, CloudTrail registra los eventos de administración.

AWS App Mesh registra todas las operaciones del plano de control de App Mesh como eventos de administración. Para obtener una lista de las operaciones del plano de AWS App Mesh control en las que App Mesh registra CloudTrail, consulta la referencia de la AWS App Mesh API.

Ejemplos de eventos de App Mesh

Un evento representa una solicitud única de cualquier fuente e incluye información sobre la operación de API solicitada, la fecha y la hora de la operación, los parámetros de la solicitud, etc. CloudTrail Los archivos de registro no son un registro ordenado de las llamadas a la API pública, por lo que los eventos no aparecen en ningún orden específico.

En el siguiente ejemplo, se muestra una entrada de CloudTrail registro que demuestra la `StreamAggregatedResources` acción.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAIOSFODNN7EXAMPLE:d060be4ac3244e05aca4e067bfe241f8",
        "arn": "arn:aws:sts::123456789012:assumed-role/Application-TaskIamRole-
C20GBLBRLBXE/d060be4ac3244e05aca4e067bfe241f8",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "invokedBy": "appmesh.amazonaws.com"
    },
    "eventTime": "2021-06-09T23:09:46Z",
    "eventSource": "appmesh.amazonaws.com",
    "eventName": "StreamAggregatedResources",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "appmesh.amazonaws.com",
    "userAgent": "appmesh.amazonaws.com",
    "eventID": "e3c6f4ce-EXAMPLE",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
```

```
    "serviceEventDetails": {
        "connectionId": "e3c6f4ce-EXAMPLE",
        "nodeArn": "arn:aws:appmesh:us-west-2:123456789012:mesh/CloudTrail-Test/
virtualNode/cloudtrail-test-vn",
        "eventStatus": "ConnectionEstablished",
        "failureReason": ""
    },
    "eventCategory": "Management"
}
```

Para obtener información sobre el contenido de los CloudTrail registros, consulte el contenido de los CloudTrail registros en la Guía del AWS CloudTrail usuario.

Protección de datos en AWS App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

El modelo de responsabilidad compartida de AWS se aplica a la protección de datos en AWS App Mesh. Como se describe en este modelo, AWS es responsable de proteger la infraestructura global en la que se ejecutan todos los Nube de AWS. Eres responsable de mantener el control sobre el contenido alojado en esta infraestructura. También eres responsable de las tareas de administración y configuración de seguridad para los Servicios de AWS que utiliza. Para obtener más información sobre la privacidad de los datos, consulte las Preguntas frecuentes sobre privacidad de datos y los . Para obtener más información sobre la protección de datos en Europa, consulte el Centro del Reglamento General de Protección de Datos (RGPD).

Para fines de protección de datos, le recomendamos que proteja Cuenta de AWS las credenciales y configure usuarios individuales con AWS IAM Identity Center o AWS Identity and Access Management (IAM). De esta manera, solo se otorgan a cada usuario los permisos necesarios para cumplir sus obligaciones laborales. También recomendamos proteger sus datos de la siguiente manera:

- Utiliza la autenticación multifactor (MFA) en cada cuenta.

- Se utiliza SSL/TLS para comunicarse con AWS los recursos. Exigimos TLS 1.2 y recomendamos TLS 1.3.
- Configure la API y el registro de actividad de los usuarios con AWS CloudTrail. Para obtener información sobre el uso de CloudTrail senderos para capturar AWS actividades, consulte Cómo trabajar con CloudTrail senderos en la Guía del AWS CloudTrail usuario.
- Utilice soluciones de AWS cifrado, junto con todos los controles de seguridad predeterminados Servicios de AWS.
- Utiliza servicios de seguridad administrados avanzados, como Amazon Macie, que lo ayuden a detectar y proteger la información confidencial almacenada en Amazon S3.
- Si necesita módulos criptográficos validados por FIPS 140-3 para acceder a AWS través de una interfaz de línea de comandos o una API, utilice un punto final FIPS. Para obtener más información sobre los puntos de conexión de FIPS disponibles, consulte Estándar de procesamiento de la información federal (FIPS) 140-3.

Se recomienda encarecidamente no introducir nunca información confidencial o sensible, como por ejemplo, direcciones de correo electrónico de clientes, en etiquetas o campos de formato libre, tales como el campo Nombre. Esto incluye cuando trabajas con App Mesh u otro tipo de aplicaciones Servicios de AWS mediante la consola, la API o AWS los SDK. AWS CLI Cualquier dato que introduzca en etiquetas o campos de formato libre utilizados para los nombres se pueden emplear para los registros de facturación o diagnóstico. Si proporciona una URL a un servidor externo, recomendamos encarecidamente que no incluya información de credenciales en la URL a fin de validar la solicitud para ese servidor.

Cifrado de datos

Sus datos se cifran cuando utiliza App Mesh.

Cifrado en reposo

De forma predeterminada, las configuraciones de App Mesh que cree se cifran en reposo.

Cifrado en tránsito

Los puntos finales del servicio App Mesh utilizan el protocolo HTTPS. Todas las comunicaciones entre el proxy de Envoy y el servicio de administración de App Mesh Envoy están cifradas. Si necesitas un cifrado compatible con FIPS para la comunicación entre el proxy de Envoy y el App

Mesh Envoy Management Service, puedes usar una variante FIPS de la imagen del contenedor de proxy de Envoy. Para obtener más información, consulte Imagen de Envoy.

La comunicación entre contenedores dentro de los nodos virtuales no está cifrada, pero este tráfico no sale del espacio de nombres de la red.

Validación de conformidad para AWS App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Para saber si uno Servicio de AWS está dentro del ámbito de aplicación de programas de conformidad específicos, consulte Servicios de AWS Alcance por programa de conformidad Servicios de AWS y elija el programa de conformidad que le interese. Para obtener información general, consulte Programas de AWS cumplimiento > Programas AWS .

Puede descargar informes de auditoría de terceros utilizando AWS Artifact. Para obtener más información, consulte Descarga de informes en AWS Artifact .

Su responsabilidad de cumplimiento al Servicios de AWS utilizarlos viene determinada por la confidencialidad de sus datos, los objetivos de cumplimiento de su empresa y las leyes y reglamentos aplicables. Para obtener más información sobre su responsabilidad de conformidad al utilizarlos Servicios de AWS, consulte AWS la documentación de seguridad.

Seguridad de infraestructura en AWS App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Como servicio gestionado, AWS App Mesh está protegido por la seguridad de la red AWS global. Para obtener información sobre los servicios AWS de seguridad y cómo se AWS protege la infraestructura, consulte Seguridad AWS en la nube. Para diseñar su AWS entorno utilizando las mejores prácticas de seguridad de la infraestructura, consulte Protección de infraestructuras en un marco de buena AWS arquitectura basado en el pilar de la seguridad.

Utiliza las llamadas a la API AWS publicadas para acceder a App Mesh 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.

Puede mejorar la posición de seguridad de su VPC configurando App Mesh para que utilice un punto de conexión de VPC de interfaz. Para obtener más información, consulte Puntos finales de VPC de la interfaz App Mesh ()AWS PrivateLink.

Puntos finales de VPC de la interfaz App Mesh ()AWS PrivateLink

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Puede mejorar la posición de seguridad de Amazon VPC si configura App Mesh para que utilice un punto de conexión de VPC de interfaz. Los puntos finales de la interfaz funcionan con una tecnología que te permite acceder de forma privada a App Mesh APIs mediante direcciones IP privadas. AWS PrivateLink PrivateLinkrestringe todo el tráfico de red entre tu Amazon VPC y App Mesh a la red de Amazon.

No es necesario que lo configure PrivateLink, pero le recomendamos que lo haga. Para obtener más información sobre los puntos finales de la VPC PrivateLink y su interfaz, consulte Acceder a los servicios mediante. AWS PrivateLink

Consideraciones sobre los puntos finales de VPC de la interfaz App Mesh

Antes de configurar los puntos de conexión de VPC de interfaz para App Mesh, debe tener en cuenta las siguientes consideraciones:

- Si su Amazon VPC no tiene una puerta de enlace a Internet y sus tareas utilizan el controlador de registro para enviar la información de `awslogs` registro a CloudWatch Logs, debe crear un punto de enlace de VPC de interfaz para Logs. CloudWatch Para obtener más información, consulte Uso de CloudWatch registros con puntos de enlace de VPC de interfaz en la Guía del usuario de Amazon CloudWatch Logs.
- Los puntos de enlace de VPC no admiten AWS solicitudes entre regiones. Asegúrese de crear su punto de conexión en la misma región en la que tiene previsto realizar llamadas a la API de App Mesh.
- 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 VPC debe permitir las conexiones entrantes en el puerto 443 desde la subred privada de Amazon VPC.

> **Note**
>
> Una conexión de Envoy no admite el control del acceso a App Mesh asociando una política de punto de conexión al punto de conexión de VPC (por ejemplo, utilizando el nombre del servicio `com.amazonaws.Region.appmesh-envoy-management`).

Para conocer otras consideraciones y limitaciones, consulte Consideraciones sobre la zona de disponibilidad de los puntos de conexión de interfaz y Propiedades y limitaciones de los puntos de conexión de interfaz.

Cree el punto final de VPC de la interfaz para App Mesh

Para crear el punto de conexión de VPC de interfaz para el servicio de App Mesh, siga el procedimiento Creación de un punto de conexión de interfaz en la Guía del usuario de Amazon VPC. Especifique `com.amazonaws.Region.appmesh-envoy-management` como nombre de servicio para que su proxy de Envoy se conecte al servicio público de administración de Envoy de App Mesh y especifique `com.amazonaws.Region.appmesh` para las operaciones de malla.

> **Note**
>
> *Region* representa el identificador de región de una AWS región compatible con App Mesh, como `us-east-2` la región EE.UU. Este (Ohio).

Aunque puede definir un punto de conexión de VPC de interfaz para App Mesh en cualquier región en la que se admita App Mesh, es posible que no pueda definir un punto de conexión para todas las zonas de disponibilidad de cada región. Para saber qué zonas de disponibilidad son compatibles con los puntos finales de la interfaz de VPC en una región, utilice el describe-vpc-endpoint-services comando o utilice el. Consola de administración de AWS Por ejemplo, los siguientes comandos devuelven las zonas de disponibilidad en las que puede implementar un punto de conexión de VPC de interfaz de App Mesh dentro de la región Este de EE. UU. (Ohio):

```
aws --region us-east-2 ec2 describe-vpc-endpoint-services --query 'ServiceDetails[?
ServiceName==`com.amazonaws.us-east-2.appmesh-envoy-management`].AvailabilityZones[]'
```

```
aws --region us-east-2 ec2 describe-vpc-endpoint-services --query 'ServiceDetails[?
ServiceName==`com.amazonaws.us-east-2.appmesh`].AvailabilityZones[]'
```

Resiliencia en AWS App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

La infraestructura AWS global se basa en AWS regiones y zonas de disponibilidad. AWS Las regiones proporcionan varias zonas de disponibilidad aisladas y separadas físicamente, que están conectadas mediante redes de baja latencia, alto rendimiento y alta redundancia. Con las zonas de disponibilidad, puedes diseñar y utilizar aplicaciones y bases de datos que realizan una conmutación por error automática entre zonas de disponibilidad sin interrupciones. Las zonas

de disponibilidad tienen una mayor disponibilidad, tolerancia a errores y escalabilidad que las infraestructuras tradicionales de centros de datos únicos o múltiples.

App Mesh ejecuta sus instancias del plano de control en varias zonas de disponibilidad para garantizar una alta disponibilidad. App Mesh detecta y reemplaza automáticamente las instancias del plano de control en mal estado y proporciona actualizaciones de versiones y parches automatizados para ellas.

Recuperación ante desastres en AWS App Mesh

El servicio de App Mesh gestiona las copias de seguridad de los datos de los clientes. No hay nada que deba hacer para administrar las copias de seguridad. Los datos de la copia de seguridad están cifrados.

Análisis de configuración y vulnerabilidad en AWS App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS se interrumpirá el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

App Mesh vende una imagen de contenedor de Docker del proxy de Envoy administrada que puede implementar con sus microservicios. App Mesh garantiza que la imagen del contenedor está parcheada con los parches de vulnerabilidad y rendimiento más recientes. App Mesh prueba las nuevas versiones del proxy de Envoy comparándolas con el conjunto de características de App Mesh antes de poner las imágenes a su disposición.

Debe actualizar sus microservicios para usar la versión actualizada de la imagen del contenedor. La siguiente es la versión más reciente de la imagen.

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.34.13.1-prod
```

Solución de problemas de App Mesh

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

En este capítulo se describen las prácticas recomendadas para solucionar problemas comunes que puedan surgir al utilizar App Mesh. Seleccione una de las siguientes áreas para revisar las prácticas recomendadas y los problemas comunes de dicha área.

Temas

- Solución de problemas y prácticas recomendadas de App Mesh
- Solución de problemas de configuración de App Mesh
- Solución de problemas de conectividad de App Mesh
- Solución de problemas de escalado de App Mesh
- Solución de problemas de observabilidad de App Mesh
- Solución de problemas de seguridad de App Mesh
- Solución de problemas de App Mesh para Kubernetes

Solución de problemas y prácticas recomendadas de App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

Le recomendamos que siga las prácticas recomendadas de este tema para solucionar problemas al utilizar App Mesh.

Habilitar la interfaz de administración del proxy de Envoy

El proxy de Envoy incluye una interfaz de administración que se puede utilizar para detectar la configuración y las estadísticas y para realizar otras funciones administrativas, como el drenaje de conexiones. Para obtener más información, consulte Interfaz de administración en la documentación de Envoy.

Si utiliza la Imagen de Envoy administrada, el punto de conexión de administración está habilitado de forma predeterminada en el puerto 9901. Los ejemplos que se proporcionan en Solución de problemas de configuración de App Mesh muestran la URL del punto de conexión de administración del ejemplo como `http://my-app.default.svc.cluster.local:9901/`.

> **Note**
>
> El punto de conexión de administración nunca debe exponerse públicamente en Internet. Además, recomendamos monitorizar los registros de los puntos de conexión de administración, que la variable de entorno `ENVOY_ADMIN_ACCESS_LOG_FILE` establece en `/tmp/envoy_admin_access.log` de forma predeterminada.

Habilite la integración de Envoy DogStats D para reducir las métricas

El proxy de Envoy se puede configurar para descargar las estadísticas del tráfico de nivel 4 y 7 de OSI y del estado del proceso interno. Si bien en este tema se muestra cómo utilizar estas estadísticas sin descargar las métricas a sumideros como Metrics CloudWatch y Prometheus, disponer de estas estadísticas en una ubicación centralizada para todas sus aplicaciones puede ayudarle a diagnosticar problemas y confirmar el comportamiento más rápidamente. Para obtener más información, consulte Uso de Amazon CloudWatch Metrics y la documentación de Prometheus.

Puede configurar las métricas DogStats D configurando los parámetros definidos en. DogStatsVariables D Para obtener más información sobre DogStats D, consulte la documentación de DogStatsD. Encontrará una demostración de la transferencia de métricas a AWS CloudWatch las métricas en el tutorial básico de App Mesh with Amazon ECS. GitHub

Habilitación de registros de acceso

Recomendamos que habilite los registros de acceso en sus Nodos virtuales y Puertas de enlace virtuales para tener información sobre el tráfico que circula entre sus aplicaciones. Para obtener más información, consulte Acceso al registro en la documentación de Envoy. Los registros proporcionan información detallada sobre el comportamiento del tráfico de las capas 4 y 7 de OSI. Si utilizas el formato predeterminado de Envoy, puedes analizar los registros de acceso con Logs Insights mediante la CloudWatch siguiente sentencia de análisis.

```
parse @message "[*] \"* * *\" * * * * * * * * * * *" as StartTime, Method, Path,
 Protocol, ResponseCode, ResponseFlags, BytesReceived, BytesSent, DurationMillis,
 UpstreamServiceTimeMillis, ForwardedFor, UserAgent, RequestId, Authority, UpstreamHost
```

Habilitación del registro de depuración de Envoy en los entornos de preproducción

Recomendamos establecer el nivel de registro del proxy de Envoy en `debug` en un entorno de preproducción. Los registros de depuración pueden ayudarlo a identificar los problemas antes de trasladar la configuración de App Mesh asociada a su entorno de producción.

Si utiliza la imagen de Envoy, puede establecer el nivel de registro en `debug` mediante la variable de entorno `ENVOY_LOG_LEVEL`.

> **Note**
>
> No recomendamos utilizar el nivel `debug` en entornos de producción. Si se establece este nivel, se `debug` aumenta el registro y puede afectar al rendimiento y al coste total de los registros transferidos a soluciones como CloudWatch Logs.

Si utilizas el formato predeterminado de Envoy, puedes analizar los registros del proceso con CloudWatch Logs Insights mediante la siguiente declaración de análisis:

```
parse @message "[*][*][*][*] [*] *" as Time, Thread, Level, Name, Source, Message
```

Monitorización de la conectividad de Envoy Proxy con el plano de control de App Mesh

Recomendamos monitorizar las métricas de Envoy `control_plane.connected_state` para asegurarse de que el proxy de Envoy se comunica con el plano de control de App Mesh para obtener los recursos de configuración dinámica. Para obtener más información, consulte Servidor de administración.

Solución de problemas de configuración de App Mesh

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

En este tema se explican los problemas comunes que pueden ocurrir con la configuración de App Mesh.

No se puede extraer la imagen del contenedor de Envoy

Síntomas

Recibe el siguiente mensaje de error en una tarea de Amazon ECS. El Amazon ECR *account ID* y *Region* el mensaje siguiente pueden ser diferentes, según el repositorio de Amazon ECR del que haya extraído la imagen del contenedor.

```
CannotPullContainerError: Error response from daemon: pull access denied
 for 840364872350.dkr.ecr.us-west-2.amazonaws.com/aws-appmesh-envoy, repository does
 not exist or may require 'docker login'
```

Resolución

Este error indica que el rol de ejecución de tareas que se utiliza no tiene permiso para comunicar con Amazon ECR y no puede extraer la imagen del contenedor de Envoy del repositorio. El rol

de ejecución de tareas asignado a su tarea de Amazon ECS necesita una política de IAM con las siguientes instrucciones:

```
{
  "Action": [
    "ecr:BatchCheckLayerAvailability",
    "ecr:GetDownloadUrlForLayer",
    "ecr:BatchGetImage"
  ],
  "Resource": "arn:aws:ecr:us-west-2:111122223333:repository/aws-appmesh-envoy",
  "Effect": "Allow"
},
{
  "Action": "ecr:GetAuthorizationToken",
  "Resource": "*",
  "Effect": "Allow"
}
```

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

No se puede conectar con el servicio de administración de Envoy de App Mesh

Síntomas

Su proxy de Envoy no puede conectar con el servicio de administración de Envoy de App Mesh. Está viendo:

- Errores de conexión rechazada
- Tiempos de espera de conexión
- Errores al resolver el punto de conexión del servicio de administración de Envoy de App Mesh
- Errores gRPC

Resolución

Asegúrese de que su proxy de Envoy tenga acceso a Internet o a un punto de conexión de VPC privado y de que sus grupos de seguridad permitan el tráfico saliente en el puerto 443. Los puntos de conexión del servicio de administración de Envoy público de App Mesh siguen el formato de nombre de dominio completo (FQDN).

```
# App Mesh Production Endpoint
appmesh-envoy-management.Region-code.amazonaws.com

# App Mesh Preview Endpoint
appmesh-preview-envoy-management.Region-code.amazonaws.com
```

Puede depurar su conexión a EMS mediante el siguiente comando. Esto envía una solicitud de gRPC válida, pero vacía, al servicio de administración de Envoy.

```
curl -v -k -H 'Content-Type: application/grpc' -X POST https://
appmesh-envoy-management.Region-code.amazonaws.com:443/
envoy.service.discovery.v3.AggregatedDiscoveryService/StreamAggregatedResources
```

Si vuelve a recibir estos mensajes, su conexión con el servicio de administración de Envoy funciona. Para depurar errores relacionados con gRPC, consulte los errores en Envoy se desconectó del servicio de administración de Envoy de App Mesh mostrando un texto de error.

```
grpc-status: 16
grpc-message: Missing Authentication Token
```

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Envoy se desconectó del servicio de administración de Envoy de App Mesh mostrando un texto de error

Síntomas

Su proxy de Envoy no puede conectarse al servicio de administración de Envoy de App Mesh ni recibir su configuración. Sus registros de proxy de Envoy contienen una entrada de registro como la siguiente.

```
gRPC config stream closed: gRPC status code, message
```

Resolución

En la mayoría de los casos, la parte del mensaje del registro debería indicar el problema. En la siguiente tabla se enumeran los códigos de estado de gRPC más comunes que se pueden ver, sus causas y sus resoluciones.

Código de estado de gRPC	Causa	Resolución
0	Desconexión sin problemas del servicio de administración de Envoy.	No hay ningún problema. App Mesh ocasionalmente desconecta los proxies de Envoy con este código de estado. Envoy volverá a conectar y seguirá recibiendo actualizaciones.
3	No se pudo encontrar el punto de conexión de malla (nodo virtual o puerta de enlace virtual) o uno de sus recursos asociados.	Compruebe la configuración de Envoy para asegurarse de que tiene el nombre adecuado del recurso App Mesh que representa. Si tu recurso de App Mesh está integrado con otros AWS recursos, como AWS Cloud Map espacios de nombres o certificados ACM, asegúrate de que esos recursos existan.
7	El proxy de Envoy no está autorizado para realizar una acción, como conectarse al servicio de administración de Envoy o recuperar los recursos asociados.	Asegúrese de crear una política de IAM que contenga las declaraciones de políticas adecuadas para App Mesh y otros servicios y asocie dicha política al rol o usuario de IAM que su proxy de Envoy utiliza para conectarse al servicio de administración de Envoy.
8	La cantidad de proxies de Envoy de un recurso de App Mesh determinado supera la cuota de servicio en el nivel de cuenta.	Consulte Cuotas de servicio de App Mesh para obtener más información acerca de las cuotas de la cuenta predeterm

Código de estado de gRPC	Causa	Resolución
		inado y cómo solicitar un aumento de cuotas.
16	El proxy de Envoy no tiene credenciales de autenticación válidas para AWS.	Asegúrese de que el Envoy tenga las credenciales adecuadas para conectarse a los servicios de AWS a través de un rol o usuario de IAM. Existe un problema conocido, el #24136, en la versión `v1.24` de Envoy y en las versiones anteriores, por el que no se obtienen las credenciales si el proceso de Envoy utiliza más de `1024` descriptores archivos. Esto sucede cuando Envoy atiende un volumen de tráfico elevado. Puede confirmar este problema buscando el texto "`A libcurl function was given a bad argument`" en los registros de Envoy en el nivel de depuración. Para mitigar este problema, actualice a la versión `v1.25.1.0-prod` o posterior de Envoy.

Puedes observar los códigos de estado y los mensajes de tu proxy de Envoy con Amazon CloudWatch Insights mediante la siguiente consulta:

```
filter @message like /gRPC config stream closed/
| parse @message "gRPC config stream closed: *, *" as StatusCode, Message
```

Si el mensaje de error proporcionado no fue útil o tu problema sigue sin resolverse, considera abrir un GitHub problema.

La comprobación de estado del contenedor de Envoy, la sonda de disponibilidad o la sonda de vivacidad producen errores

Síntomas

Su proxy de Envoy no pasa las comprobaciones de estado en una tarea de Amazon ECS, en una instancia de Amazon EC2 o en un pod de Kubernetes. Por ejemplo, consulta la interfaz de administración de Envoy con el siguiente comando y recibe un estado distinto de `LIVE`.

```
curl -s http://my-app.default.svc.cluster.local:9901/server_info | jq '.state'
```

Resolución

A continuación se muestra una lista de pasos de corrección en función del estado devuelto por el proxy de Envoy.

- `PRE_INITIALIZING` o `INITIALIZING`: el proxy de Envoy aún no ha recibido la configuración o no puede conectarse y obtener la configuración del servicio de administración de Envoy de App Mesh. Es posible que el Envoy esté recibiendo un error del servicio de administración de Envoy al intentar conectarse. Para obtener más información, consulte los errores en Envoy se desconectó del servicio de administración de Envoy de App Mesh mostrando un texto de error.
- `DRAINING`: el proxy de Envoy ha empezado a agotar las conexiones en respuesta a una solicitud `/healthcheck/fail` o `/drain_listeners` en la interfaz de administración de Envoy. No recomendamos invocar estas rutas en la interfaz de administración a menos que esté a punto de finalizar su tarea de Amazon ECS, instancia de Amazon EC2 o pod de Kubernetes.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

La comprobación de estado desde el equilibrador de carga hasta el punto de conexión de malla está fallando

Síntomas

La comprobación de estado del contenedor o la sonda de disponibilidad consideran que el punto de conexión de malla está en buen estado, pero la comprobación de estado desde el equilibrador de carga hasta el punto de conexión de malla está fallando.

Resolución

Para resolver el problema, realice las siguientes tareas.

- Asegúrese de que el grupo de seguridad asociado a su punto de conexión de malla acepte el tráfico entrante en el puerto que configuró para la comprobación de estado.
- Asegúrese de que la comprobación de estado se realice correctamente cuando se solicite manualmente; por ejemplo, desde un host bastión de su VPC.
- Si va a configurar una comprobación de estado para un nodo virtual, recomendamos que implemente un punto de conexión de comprobación de estado en su aplicación; por ejemplo, /ping para HTTP. Esto garantiza que tanto el proxy de Envoy como su aplicación se puedan enrutar desde el equilibrador de carga.
- Puede usar cualquier tipo de equilibrador de carga elástico para el nodo virtual, según las características que necesite. Para obtener más información, consulte Características del equilibrador de carga elástico.
- Si va a configurar una comprobación de estado para una puerta de enlace virtual, recomendamos que utilice un equilibrador de carga de red con una comprobación de estado de TCP o TLS en el puerto del oyente de la puerta de enlace virtual. De este modo, se asegura de que el oyente de la puerta de enlace virtual se haya iniciado y esté preparado para aceptar conexiones.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

La puerta de enlace virtual no acepta tráfico en los puertos 1024 o inferiores

Síntomas

Su puerta de enlace virtual no acepta tráfico en el puerto 1024 o inferiores, pero sí acepta tráfico en un número de puerto superior a 1024. Por ejemplo, consulta las estadísticas de Envoy con el siguiente comando y obtiene un valor distinto de cero.

```
curl -s http://my-app.default.svc.cluster.local:9901/stats | grep "update_rejected"
```

Es posible que vea un texto similar al siguiente en sus registros que describe un error al conectarse a un puerto privilegiado:

```
gRPC config for type.googleapis.com/envoy.api.v2.Listener rejected: Error adding/
updating listener(s) lds_ingress_0.0.0.0_port_<port num>: cannot bind '0.0.0.0:<port
 num>': Permission denied
```

Resolución

Para resolver el problema, el usuario especificado para la puerta de enlace debe tener la capacidad `CAP_NET_BIND_SERVICE` de Linux. Para obtener más información, consulte Capacidades en el Manual del programador de Linux, Parámetros de Linux en los parámetros de definición de tareas de ECS y Establecer capacidades para un contenedor en la documentación de Kubernetes.

> **Important**
>
> Fargate debe usar un valor de puerto superior a 1024.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Solución de problemas de conectividad de App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

En este tema se explican los problemas comunes que pueden ocurrir con la conectividad de App Mesh.

No se puede resolver el nombre DNS de un servicio virtual

Síntomas

La aplicación no puede resolver el nombre DNS de un servicio virtual al que intenta conectarse.

Resolución

Se trata de un problema conocido. Para obtener más información, consulte el tema del nombre VirtualServices junto a cualquier nombre de host o FQDN GitHub . Los servicios virtuales de App Mesh pueden tener cualquier nombre. Siempre que haya un registro `A` DNS para el nombre del servicio virtual y la aplicación pueda resolver el nombre del servicio virtual, Envoy redirigirá la solicitud mediante un proxy y la dirigirá al destino correspondiente. Para resolver el problema, añada un registro `A` DNS a cualquier dirección IP que no sea de bucle invertido, por ejemplo `10.10.10.10`, para el nombre del servicio virtual. El registro `A` DNS se puede agregar en las siguientes condiciones:

- En Amazon Route 53, si el nombre tiene el sufijo del nombre de la zona alojada privada
- Dentro del archivo `/etc/hosts` del contenedor de aplicaciones
- En un servidor DNS de terceros que usted administre

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

No se puede conectar a un backend de servicio virtual

Síntomas

Su aplicación no puede establecer una conexión a un servicio virtual definido como backend en su nodo virtual. Al intentar establecer una conexión, esta puede fallar por completo, o la solicitud desde la perspectiva de la aplicación puede fallar mostrando un código de respuesta `HTTP 503`.

Resolución

Si la aplicación no se conecta en absoluto (no se devuelve ningún código de respuesta `HTTP 503`), haga lo siguiente:

- Asegúrese de que su entorno informático esté configurado para funcionar con App Mesh.
  - Para Amazon ECS, asegúrese de que tiene habilitada la configuración de proxy adecuada. Para ver un end-to-end tutorial, consulte Introducción a App Mesh y Amazon ECS.
  - Para Kubernetes, incluido Amazon EKS, asegúrese de que tiene instalado el controlador de App Mesh más reciente a través de Helm. Para obtener más información, consulte Controlador de App Mesh en Helm Hub o Tutorial: Configurar la integración de App Mesh con Kubernetes.

  - En el caso de Amazon EC2, asegúrese de que ha configurado su instancia de Amazon EC2 para el tráfico de App Mesh mediante proxy. Para obtener más información, consulte Actualizar servicios.
- Asegúrese de que el contenedor de Envoy que se ejecuta en su servicio informático se haya conectado correctamente al servicio de administración de Envoy de App Mesh. Puede confirmarlo consultando las estadísticas de Envoy correspondientes al campo `control_plane.connected_state`. Para obtener más información acerca de `control_plane.connected_state`, consulte Monitorizar la conectividad del proxy de Envoy en nuestras Prácticas recomendadas para la resolución de problemas.

  Si Envoy pudo establecer la conexión inicialmente, pero luego se desconectó y nunca se volvió a conectar, consulte Envoy se desconectó del servicio de administración de Envoy de App Mesh mostrando un texto de error para solucionar el motivo de la desconexión.

Si la aplicación se conecta pero la solicitud produce un error y aparece un código de respuesta HTTP `503`, intente lo siguiente:

- Asegúrese de que el servicio virtual al que se está conectando existe en la malla.
- Asegúrese de que el servicio virtual tenga un proveedor (un enrutador virtual o un nodo virtual).
- Cuando utilice Envoy como proxy HTTP, si en las estadísticas de Envoy comprueba que en `cluster.cds_egress_*_mesh-allow-all` entra tráfico de salida en lugar del destino correcto, lo más probable es que Envoy no enrute las solicitudes correctamente a través de `filter_chains`. Esto puede deberse al uso de un nombre de servicio virtual no cualificado. Recomendamos que utilice el nombre de detección de servicios del servicio real como nombre del servicio virtual, ya que el proxy de Envoy se comunica con otros servicios virtuales a través de sus nombres.

  Para obtener más información, consulte servicios virtuales.
- Examine los registros del proxy de Envoy para ver si hay alguno de los siguientes mensajes de error:
  - `No healthy upstream`: el nodo virtual al que el proxy de Envoy intenta enrutar no tiene ningún punto de conexión resuelto o no tiene ningún punto de conexión en buen estado. Asegúrese de que el nodo virtual de destino tenga la configuración correcta de detección de servicios y comprobación de estado.

Si las solicitudes al servicio producen un error durante la implementación o el escalado del servicio virtual de backend, siga las instrucciones que se indican en Algunas solicitudes producen un error con el código de estado HTTP 503 cuando un servicio virtual tiene un proveedor de nodos virtuales.

- `No cluster match for URL`: lo más probable es que esto se deba a que se envía una solicitud a un servicio virtual que no cumple los criterios definidos por ninguna de las rutas definidas por un proveedor de enrutadores virtuales. Asegúrese de que las solicitudes de la aplicación se envíen a una ruta compatible, comprobando que la ruta y los encabezados de las solicitudes HTTP sean correctos.
- `No matching filter chain found`: lo más probable es que esto se deba a que se envía una solicitud a un servicio virtual en un puerto no válido. Asegúrese de que las solicitudes de la aplicación utilicen el mismo puerto especificado en el enrutador virtual.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

No se puede conectar a un servicio externo

Síntomas

Su aplicación no puede conectarse a un servicio fuera de la malla, por ejemplo `amazon.com`.

Resolución

De forma predeterminada, App Mesh no permite el tráfico saliente de las aplicaciones dentro de la malla a ningún destino fuera de la malla. Para habilitar la comunicación con un servicio externo, hay dos opciones:

- Establezca el filtro de salida en el recurso de malla en `ALLOW_ALL`. Esta configuración permitirá que cualquier aplicación dentro de la malla se comunique con cualquier dirección IP de destino dentro o fuera de la malla.
- Modele el servicio externo en la malla mediante un servicio virtual, enrutador virtual, ruta y nodo virtual. Por ejemplo, para modelar el servicio externo `example.com`, puede crear un servicio virtual denominado `example.com` con un enrutador y una ruta virtuales que envíen todo el tráfico a un nodo virtual con un nombre de host de detección de servicios DNS de `example.com`.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

No se puede conectar a un servidor MySQL o SMTP

Síntomas

Al permitir el tráfico saliente a todos los destinos (Malla `EgressFilter type=ALLOW_ALL`), por ejemplo, un servidor SMTP o una base de datos MySQL mediante una definición de nodo virtual, se produce un error en la conexión de la aplicación. A modo de ejemplo, el siguiente es un mensaje de error al intentar conectarse a un servidor MySQL.

```
ERROR 2013 (HY000): Lost connection to MySQL server at 'reading initial communication
 packet', system error: 0
```

Resolución

Se trata de un problema conocido que se resuelve con la versión 1.15.0 o posterior de la imagen de App Mesh. Para obtener más información, consulta el GitHub problema No se puede conectar a MySQL con App Mesh. Este error se produce porque el oyente saliente de Envoy configurado por App Mesh añade el filtro de oyente de TLS Inspector de Envoy. Para obtener más información, consulte TLS Inspector en la documentación de Envoy. Este filtro evalúa si una conexión utiliza o no TLS inspeccionando el primer paquete enviado desde el cliente. Sin embargo, con MySQL y SMTP, el servidor envía el primer paquete después de la conexión. Para obtener más información acerca de MySQL, consulte Protocolo de enlace inicial en la documentación de MySQL. Como el servidor envía el primer paquete, se produce un error en la inspección del filtro.

Solución de este problema según la versión de Envoy:

- Si la versión de Envoy para imágenes de App Mesh es la 1.15.0 o posterior, no modele servicios externos como MySQL, SMTP, MSSQL, etc. como un backend para el nodo virtual de su aplicación.
- Si la versión de Envoy para imágenes de App Mesh es anterior a la 1.15.0, añada el puerto `3306` a la lista de valores de `APPMESH_EGRESS_IGNORED_PORTS` en sus servicios para MySQL y como el puerto que va a utilizar para STMP.

> **Important**
>
> Si bien los puertos SMTP estándar son 25, 587 y465, solo debe añadir el puerto que está utilizando a `APPMESH_EGRESS_IGNORED_PORTS` y no los tres.

Para obtener más información, consulte Servicios de actualización para Kubernetes, Servicios de actualización para Amazon ECS o Servicios de actualización para Amazon EC2.

Si tu problema sigue sin resolverse, puedes proporcionarnos detalles sobre lo que estás experimentando con el GitHub problema existente o ponerte en contacto con AWS Support.

No se puede conectar a un servicio modelado como un nodo virtual TCP o enrutador virtual en App Mesh

Síntomas

Tu aplicación no puede conectarse a un backend que utilice la configuración del protocolo TCP en la PortMappingdefinición de App Mesh.

Resolución

Se trata de un problema conocido. Para obtener más información, consulta Cómo enrutar a varios destinos TCP en el mismo puerto. GitHub Actualmente, App Mesh no permite que varios destinos de backend modelados como TCP compartan el mismo puerto debido a las restricciones de la información proporcionada al proxy de Envoy en la capa 4 de OSI. Para asegurarse de que el tráfico de TCP se pueda enrutar de manera adecuada para todos los destinos de backend, haga lo siguiente:

- Asegúrese de que todos los destinos utilicen un puerto único. Si utiliza un proveedor de enrutadores virtuales para el servicio virtual de backend, puede cambiar el puerto del enrutador virtual sin cambiar el puerto de los nodos virtuales a los que se enruta. Esto permite que las aplicaciones abran conexiones en el puerto del router virtual mientras el proxy de Envoy sigue utilizando el puerto definido en el nodo virtual.
- Si el destino modelado como TCP es un servidor MySQL o cualquier otro protocolo basado en TCP en el que el servidor envía los primeros paquetes después de la conexión, consulte No se puede conectar a un servidor MySQL o SMTP.

Si tu problema sigue sin resolverse, puedes proporcionarnos detalles sobre lo que estás experimentando con el GitHub problema existente o ponerte en contacto con AWS Support.

La conectividad es correcta para un servicio que no figura como backend de servicio virtual de un nodo virtual

Síntomas

Su aplicación puede conectarse y enviar tráfico a un destino que no esté especificado como backend de servicio virtual en su nodo virtual.

Resolución

Si las solicitudes llegan sucesivamente a un destino que no se ha modelado en App Mesh APIs, la causa más probable es que el tipo de filtro de salida de la malla se haya establecido en. `ALLOW_ALL` Si el filtro de salida está configurado en `ALLOW_ALL`, una solicitud de salida de su aplicación que no coincida con un destino modelado (backend) se enviará a la dirección IP de destino establecida por la aplicación.

Si desea impedir el tráfico a destinos no modelados en la malla, considere la posibilidad de establecer el valor del filtro de salida en `DROP_ALL`.

> **Note**
>
> La configuración del valor del filtro de salida de la malla afecta a todos los nodos virtuales de la malla.
> La configuración `egress_filter` `DROP_ALL` y la activación de TLS no están disponibles para el tráfico saliente que no se dirija a un dominio. AWS

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Algunas solicitudes producen un error con el código de estado HTTP **`503`** cuando un servicio virtual tiene un proveedor de nodos virtuales

Síntomas

Una parte de las solicitudes de la aplicación no se envía a un backend de servicios virtuales que utiliza un proveedor de nodos virtuales en lugar de un proveedor de enrutadores virtuales. Cuando

se utiliza un proveedor de enrutadores virtuales para el servicio virtual, las solicitudes no producen errores.

Resolución

Se trata de un problema conocido. Para obtener más información, consulte la política de reintentos del proveedor de nodos virtuales para instalar un servicio virtual en GitHub. Cuando usa un nodo virtual como proveedor de un servicio virtual, no puede especificar la política de reintentos predeterminada que desea que usen los clientes de su servicio virtual. En cambio, los proveedores de enrutadores virtuales permiten especificar políticas de reintentos porque son una propiedad de los recursos de ruta secundarios.

Para reducir los errores en las solicitudes a los proveedores de nodos virtuales, utilice un proveedor de enrutadores virtuales en su lugar y especifique una política de reintentos en sus rutas. Para saber otras formas de reducir los errores en las solicitudes de sus aplicaciones, consulte Prácticas recomendadas para App Mesh.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

No se puede conectar a un sistema de archivos de Amazon EFS

Síntomas

Al configurar una tarea de Amazon ECS con un sistema de archivos de Amazon EFS como volumen, la tarea no se inicia y aparece el siguiente error.

```
ResourceInitializationError: failed to invoke EFS utils commands to set up EFS volumes:
 stderr: mount.nfs4: Connection refused : unsuccessful EFS utils command execution;
 code: 32
```

Resolución

Se trata de un problema conocido. Este error se produce porque la conexión de NFS a Amazon EFS se produce antes de que se inicie ningún contenedor de la tarea. La configuración del proxy enruta este tráfico a Envoy, que no se ejecuta en este momento. Debido al orden de inicio, el cliente NFS no se conecta al sistema de archivos de Amazon EFS y la tarea no se inicia. Para resolver el problema, añada el puerto `2049` a la lista de valores del parámetro `EgressIgnoredPorts` de la configuración del proxy de la definición de la tarea de Amazon ECS. Para más información, consulte Configuración del proxy.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

La conectividad funciona correctamente, pero la solicitud entrante no aparece en los registros, rastreos o métricas de acceso de Envoy

Síntomas

Aunque su aplicación puede conectarse y enviar solicitudes a otra aplicación, no podrá ver las solicitudes entrantes en los registros de acceso ni en la información de rastreo del proxy de Envoy.

Resolución

Se trata de un problema conocido. Para obtener más información, consulte el problema configuración de las reglas de iptables en Github. El proxy de Envoy solo intercepta el tráfico entrante al puerto al que está conectado su nodo virtual correspondiente. Las solicitudes a cualquier otro puerto omitirán el proxy de Envoy y llegarán directamente al servicio que está detrás de él. Para que el proxy de Envoy intercepte el tráfico entrante de su servicio, debe configurar el servicio y el nodo virtual para que escuchen en el mismo puerto.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Establecer las variables de entorno **HTTP_PROXY/HTTPS_PROXY** a nivel de contenedor no funciona como se esperaba.

Síntomas

Cuando HTTP_PROXY/HTTPS_PROXY se establece como variable de entorno en el:

- Contenedor de aplicaciones en la definición de tareas con App Mesh habilitado, las solicitudes que se envíen al espacio de nombres de los servicios de App Mesh recibirán respuestas de error HTTP `500` del sidecar de Envoy.
- Contenedor de Envoy en la definición de tareas con App Mesh habilitado, las solicitudes que salgan del sidecar de Envoy no pasarán por el servidor proxy HTTP/HTTPS y la variable de entorno no funcionará.

Resolución

Para el contenedor de aplicaciones:

App Mesh funciona haciendo que el tráfico de su tarea pase por el proxy de Envoy. La configuración `HTTP_PROXY/HTTPS_PROXY` anula este comportamiento al configurar el tráfico del contenedor para que pase por un proxy externo diferente. Envoy seguirá interceptando el tráfico, pero no es compatible con el envío a través de un proxy externo del tráfico de malla.

Si desea utilizar un proxy para todo el tráfico que no sea de malla, configure `NO_PROXY` para que incluya el CIDR o el espacio de nombres de la malla, el host local y los puntos de conexión de la credencial, como en el siguiente ejemplo.

```
NO_PROXY=localhost,127.0.0.1,169.254.169.254,169.254.170.2,10.0.0.0/16
```

Para el contenedor Envoy:

Envoy no admite un proxy genérico. No recomendamos configurar estas variables.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Se agotan los tiempos de espera de las solicitudes ascendentes incluso después de configurar el tiempo de espera de las rutas.

Síntomas

Ha definido el tiempo de espera de:

- Las rutas, pero sigue apareciendo un error de tiempo de espera en la solicitud ascendente.
- El oyente del nodo virtual y el tiempo de espera de reintento de las rutas, pero sigue apareciendo un error de tiempo de espera de la solicitud ascendente.

Resolución

Para que las solicitudes de alta latencia superiores a 15 segundos se completen correctamente, debe especificar un tiempo de espera tanto en el nivel de ruta como en el de oyente del nodo virtual.

Si especifica un tiempo de espera de la ruta superior al valor predeterminado de 15 segundos, asegúrese de que el tiempo de espera también esté especificado para el oyente en todos los nodos virtuales que participen. Sin embargo, si reduce el tiempo de espera a un valor inferior al predeterminado; opcionalmente, puede actualizar los tiempos de espera en los nodos virtuales. Para

obtener más información sobre las opciones a la hora de configurar nodos virtuales y rutas, consulte nodos virtuales y rutas.

Si especificó una política de reintentos, la duración que indique para el tiempo de espera de la solicitud siempre debe ser mayor o igual al tiempo de espera de reintentos multiplicado por los reintentos máximos que haya definido en la política de reintentos. Esto permite que la solicitud, con todos los reintentos, se complete correctamente. Para obtener más información, consulte rutas.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Envoy responde con una solicitud HTTP incorrecta.

Síntomas

Envoy responde con una solicitud HTTP 400 incorrecta para todas las solicitudes enviadas a través del Equilibrador de carga de red (NLB). Cuando comprobamos los registros de Envoy, vemos lo siguiente:

- Registros de depuración:

```
dispatch error: http/1.1 protocol error: HPE_INVALID_METHOD
```

- Registros de acceso:

```
"- - HTTP/1.1" 400 DPE 0 11 0 - "-" "-" "-" "-" "-"
```

Resolución

La solución consiste en deshabilitar la versión 2 (PPv2) del protocolo proxy en los atributos del grupo objetivo de su NLB.

A día de hoy, no PPv2 es compatible con la puerta de enlace virtual ni el nodo virtual Envoy, que se ejecutan mediante el plano de control App Mesh. Si despliegas NLB mediante un controlador de equilibrio de AWS carga en Kubernetes, desactívalo PPv2 configurando el siguiente atributo en: `false`

```
service.beta.kubernetes.io/aws-load-balancer-target-group-attributes:
 proxy_protocol_v2.enabled
```

Consulte Anotaciones del controlador del equilibrador de carga de AWS para obtener más información sobre los atributos de los recursos del NLB.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

No se ha podido configurar correctamente el tiempo de espera.

Síntomas

Su solicitud agota el tiempo de espera en 15 segundos, incluso después de configurar el tiempo de espera en el oyente del nodo virtual y el tiempo de espera en la ruta hacia el backend del nodo virtual.

Resolución

Asegúrese de que se incluye el servicio virtual correcto en la lista de backend.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Solución de problemas de escalado de App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

En este tema se describen los problemas comunes que pueden ocurrir con el escalado de App Mesh.

La conectividad falla y las comprobaciones de estado de los contenedores fallan al escalar más de 50 réplicas para una puerta de enlace virtual node/virtual

Síntomas

Cuando se amplía el número de réplicas, como tareas de Amazon ECS, pods de Kubernetes o instancias de Amazon EC2, para una node/virtual puerta de enlace virtual superior a 50, las comprobaciones de estado de los contenedores de Envoy para detectar Envoys nuevos y en ejecución comienzan a fallar. Las aplicaciones intermedias que envían tráfico a la node/virtual puerta de enlace virtual comienzan a detectar errores en las solicitudes con el código de estado HTTP. `503`

Resolución

La cuota predeterminada de App Mesh para el número de enviados por node/virtual puerta de enlace virtual es de 50. Cuando el número de Envoys en ejecución supera esta cuota, los Envoys nuevos y los que se están ejecutando actualmente no pueden conectarse al servicio de administración de Envoy de App Mesh con el código de estado gRPC 8 (`RESOURCE_EXHAUSTED`). Esta cuota se puede aumentar. Para obtener más información, consulte Cuotas de servicio de App Mesh.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Las solicitudes producen un error **`503`** cuando el backend de un servicio virtual se escala vertical u horizontalmente

Síntomas

Cuando un servicio virtual de backend se escala vertical u horizontalmente, las solicitudes de las aplicaciones descendentes producen un error con un código de estado `HTTP 503`.

Resolución

App Mesh recomienda varios métodos para mitigar los casos de error y, al mismo tiempo, escalar las aplicaciones horizontalmente. Para obtener información detallada acerca de cómo evitar estos errores, consulte Prácticas recomendadas para App Mesh.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

El contenedor de Envoy se bloquea debido a un error de segmentación al aumentar la carga

Síntomas

Cuando hay mucha carga de tráfico, el proxy de Envoy se bloquea debido a un error de segmentación (código de salida de Linux 139). Los registros del proceso de Envoy contienen una instrucción como la siguiente.

```
Caught Segmentation fault, suspect faulting address 0x0"
```

Resolución

Es probable que el proxy de Envoy haya superado el valor de nofile ulimit predeterminado del sistema operativo, es decir, el límite del número de archivos que un proceso puede tener abiertos a la vez. Esta infracción se debe a que el tráfico genera más conexiones, lo que consume más sockets del sistema operativo. Para resolver este problema, aumente el valor de ulimit nofile en el sistema operativo host. Si utiliza Amazon ECS, este límite se puede cambiar mediante la configuración de Ulimit en la configuración de límites de recursos de la definición de la tarea.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

El aumento de los recursos predeterminados no se refleja en los límites de servicio

Síntomas

Tras aumentar el límite predeterminado de los recursos de App Mesh, el nuevo valor no se refleja al comprobar los límites de servicio.

Resolución

Si bien los nuevos límites no se muestran actualmente, los clientes aún pueden aplicarlos.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

La aplicación se bloquea debido a la gran cantidad de llamadas de comprobación de estado.

Síntomas

Tras habilitar las comprobaciones de estado activas de un nodo virtual, se produce un aumento del número de llamadas de comprobación de estado. La aplicación se bloquea debido al aumento considerable del volumen de llamadas de comprobación de estado que se realizan a la aplicación.

Resolución

Cuando la comprobación de estado activa está habilitada, cada punto de conexión de Envoy del clúster descendente (cliente) envía solicitudes de estado a cada punto de conexión del clúster ascendente (servidor) para tomar decisiones de enrutamiento. Como resultado, el número total de solicitudes de comprobación de estado sería `number of client Envoys * number of server Envoys * active health check frequency`.

Para resolver este problema, modifique la frecuencia del sondeo de comprobación de estado, lo que reduciría el volumen total de los sondeos de comprobación de estado. Además de las comprobaciones de estado activas, App Mesh permite configurar la detección de valores atípicos como un medio de comprobación de estado pasiva. Utilice la detección de valores atípicos para configurar cuándo eliminar un host en particular en función de las respuestas `5xx` consecutivas.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Solución de problemas de observabilidad de App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

En este tema se describen los problemas comunes que pueden ocurrir con la observabilidad de App Mesh.

No puedo ver los AWS X-Ray rastros de mis aplicaciones

Síntomas

Su aplicación en App Mesh no muestra información de rastreo de rayos X en la consola de X-Ray o APIs.

Resolución

Para usar X-Ray en App Mesh, debe configurar correctamente los componentes para permitir la comunicación entre la aplicación, los contenedores sidecar y el servicio de X-Ray. Realice los pasos siguientes para confirmar que X-Ray se ha configurado correctamente:

- Asegúrese de que el protocolo de oyente del nodo virtual de App Mesh no esté establecido en `TCP`.
- Asegúrese de que el contenedor de X-Ray que se implementa con la aplicación muestre el puerto UDP `2000` y se ejecute como usuario `1337`. Para obtener más información, consulte el ejemplo de Amazon ECS X-Ray en GitHub.
- Asegúrese de que el contenedor de Envoy tenga habilitado el rastreo. Si utiliza la imagen de App Mesh Envoy, puede habilitar X-Ray estableciendo la variable de entorno `ENABLE_ENVOY_XRAY_TRACING` en un valor de `1` y la variable de entorno `XRAY_DAEMON_PORT` en un valor de `2000`.
- Si ha instrumentado X-Ray en el código de su aplicación con uno de los idiomas específicos SDKs , asegúrese de que esté configurado correctamente siguiendo las guías de su idioma.
- Si todos los elementos anteriores están configurados correctamente, revise los registros del contenedor de X-Ray para ver si hay errores y siga las instrucciones de Solución de problemas de AWS X-Ray. Puede encontrar una explicación más detallada de la integración de X-Ray en App Mesh en Integración de X-Ray con App Mesh.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

No puedo ver las métricas de Envoy para mis aplicaciones en las CloudWatch métricas de Amazon

Síntomas

Tu aplicación en App Mesh no emite las métricas generadas por el proxy de Envoy a CloudWatch las métricas.

Resolución

Cuando utilizas CloudWatch métricas en App Mesh, debes configurar correctamente varios componentes para permitir la comunicación entre el proxy de Envoy, el sidecar del CloudWatch agente y el servicio de CloudWatch métricas. Siga los siguientes pasos para confirmar que CloudWatch las métricas del proxy de Envoy se han configurado correctamente:

- Asegúrese de utilizar la imagen del CloudWatch agente para App Mesh. Para obtener más información, consulte el CloudWatchagente App Mesh en GitHub.
- Asegúrese de haber configurado el CloudWatch agente para App Mesh de forma adecuada siguiendo las instrucciones de uso específicas de la plataforma. Para obtener más información, consulte el CloudWatchagente App Mesh en GitHub.
- Si todos los elementos anteriores están configurados correctamente, revise los registros del contenedor del CloudWatch agente para ver si hay errores y siga las instrucciones que se proporcionan en Solución de problemas con el CloudWatch agente.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

No se pueden configurar reglas de muestreo personalizadas para las AWS X-Ray trazas

Síntomas

Su aplicación utiliza el rastreo de X-Ray, pero no puede configurar las reglas de muestreo de sus rastreos.

Resolución

Dado que App Mesh Envoy no admite actualmente la configuración de muestreo dinámico de X-Ray, existen las siguientes soluciones alternativas.

Si su versión de Envoy es `1.19.1` o posterior, tiene las siguientes opciones.

- Para establecer únicamente la frecuencia de muestreo, utilice la variable de entorno `XRAY_SAMPLING_RATE` en el contenedor de Envoy. El valor debe especificarse como un decimal entre `0` y `1.00` (100 %). Para obtener más información, consulte AWS X-Ray variables.
- Para configurar las reglas de muestreo personalizadas y localizadas para el rastreador de X-Ray, utilice la variable de entorno `XRAY_SAMPLING_RULE_MANIFEST` para especificar una ruta de archivo del sistema de archivos del contenedor de Envoy. Para obtener más información, consulte Reglas de muestreo en la Guía para desarrolladores de AWS X-Ray .

Si su versión de Envoy es anterior a la `1.19.1`, haga lo siguiente.

- Utilice la variable de entorno `ENVOY_TRACING_CFG_FILE` para cambiar la frecuencia de muestreo. Para obtener más información, consulte Variables de configuración de Envoy. Especifique una configuración de rastreo personalizada y defina las reglas de muestreo locales. Para obtener más información, consulte Configuración de X-Ray para Envoy.
- Ejemplo de configuración de rastreo personalizada para la variable de entorno `ENVOY_TRACING_CFG_FILE`:

```
tracing:
   http:
     name: envoy.tracers.xray
     typedConfig:
       "@type": type.googleapis.com/envoy.config.trace.v3.XRayConfig
       segmentName: foo/bar
       segmentFields:
         origin: AWS::AppMesh::Proxy
         aws:
           app_mesh:
             mesh_name: foo
             virtual_node_name: bar
       daemonEndpoint:
             protocol: UDP
             address: 127.0.0.1
             portValue: 2000
       samplingRuleManifest:
             filename: /tmp/sampling-rules.json
```

- Para obtener más información sobre la configuración del manifiesto de reglas de muestreo en la propiedad `samplingRuleManifest`, consulte Configuración del SDK de X-Ray para Go.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Solución de problemas de seguridad de App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información,

visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

En este tema se describen los problemas comunes que pueden ocurrir con la seguridad de App Mesh.

No se puede conectar a un servicio virtual de backend con una política de cliente TLS

Síntomas

Al agregar una política de cliente TLS a un backend de servicio virtual en un nodo virtual, se produce un error de conectividad de dicho backend. Al intentar enviar tráfico al servicio de backend, las solicitudes producen un error con un código de respuesta `HTTP 503` y el mensaje de error: `upstream connect error or disconnect/reset before headers. reset reason: connection failure`.

Resolución

Para determinar la causa raíz del problema, recomendamos que utilice los registros del proceso del proxy de Envoy para ayudarlo a diagnosticar el problema. Para obtener más información, consulte Habilitación del registro de depuración de Envoy en los entornos de preproducción. Utilice la siguiente lista para determinar la causa del error de conexión:

- Asegúrese de que la conectividad con el backend es correcta descartando los errores mencionados en No se puede conectar a un backend de servicio virtual.
- En los registros del proceso de Envoy, busque los siguientes errores (registrados en el nivel de depuración).

  ```
  TLS error: 268435581:SSL routines:OPENSSL_internal:CERTIFICATE_VERIFY_FAILED
  ```

  Este error se debe a una o varias de las siguientes razones:

  - El certificado no lo firmó ninguna de las autoridades de certificación definidas en la agrupación de confianza de políticas de cliente de TLS.
  - El certificado ya no es válido (ha caducado).
  - El nombre alternativo del asunto (SAN) no coincide con el nombre de host DNS solicitado.

- Asegúrese de que el certificado ofrecido por el servicio de backend sea válido, de que esté firmado por una de las autoridades de certificación de la agrupación de confianza de políticas de cliente de TLS y de que cumpla los criterios definidos en seguridad de la capa de transporte (TLS).
- Si el error que recibe es como el que se muestra a continuación, significa que la solicitud omite el proxy de Envoy y llega directamente a la aplicación. Al enviar tráfico, las estadísticas de Envoy no cambian, lo que indica que Envoy no está en vías de descifrar el tráfico. En la configuración de proxy del nodo virtual, asegúrese de que `AppPorts` contiene el valor correcto que escucha la aplicación.

```
upstream connect error or disconnect/reset before headers. reset reason:
 connection failure, transport failure reason: TLS error: 268435703:SSL
 routines:OPENSSL_internal:WRONG_VERSION_NUMBER
```

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support. Si cree que ha encontrado una vulnerabilidad de seguridad o tiene dudas sobre la seguridad de App Mesh, consulte las Directrices de notificación de vulnerabilidades de AWS.

No se puede conectar a un servicio virtual de backend cuando la aplicación es el origen de TLS

Síntomas

Cuando una aplicación es el origen de una sesión de TLS, en lugar del proxy de Envoy, se produce un error en la conectividad con un servicio virtual de backend.

Resolución

Se trata de un problema conocido. Para obtener más información, consulta la solicitud de función: problema con la negociación de TLS entre la aplicación descendente y el proxy ascendente. GitHub En App Mesh, el origen de TLS se admite actualmente desde el proxy de Envoy, pero no desde la aplicación. Para usar la compatibilidad con el origen de TLS en Envoy, deshabilite el origen de TLS en la aplicación. Esto permite al Envoy leer los encabezados de las solicitudes salientes y reenviar la solicitud al destino correspondiente mediante una sesión de TLS. Para obtener más información, consulte seguridad de la capa de transporte (TLS).

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support. Si cree que ha encontrado una vulnerabilidad de seguridad o tiene dudas sobre la seguridad de App Mesh, consulte las Directrices de notificación de vulnerabilidades de AWS.

No se puede asegurar que la conectividad entre los proxies de Envoy utilice TLS

Síntomas

Su aplicación ha habilitado la terminación de TLS en el nodo virtual o el oyente de puerta de enlace virtual, o el origen de TLS en la política del cliente de TLS del backend, pero no puede asegurar que la conectividad entre los proxies de Envoy se produzca durante una sesión negociada mediante TLS.

Resolución

Los pasos definidos en esta resolución utilizan la interfaz de administración de Envoy y las estadísticas de Envoy. Si necesita ayuda para configurarlos, consulte Habilitar la interfaz de administración del proxy de Envoy y Habilite la integración de Envoy DogStats D para reducir las métricas. Los siguientes ejemplos de estadísticas utilizan la interfaz de administración para simplificar.

- Para el proxy de Envoy que realiza la terminación de TLS:
  - Asegúrese de que el certificado TLS se haya iniciado en la configuración de Envoy con el siguiente comando.

    ```
    curl http://my-app.default.svc.cluster.local:9901/certs
    ```

    En el resultado devuelto debería ver al menos una entrada bajo `certificates[].cert_chain` para el certificado utilizado en la terminación de TLS.
  - Asegúrese de que el número de conexiones entrantes correctas al oyente del proxy sea exactamente el mismo que el número de protocolos de enlace SSL más el número de sesiones SSL reutilizadas, como se muestra en los siguientes ejemplos de comandos y resultados.

    ```
    curl -s http://my-app.default.svc.cluster.local:9901/stats | grep
     "listener.0.0.0.0_15000" | grep downstream_cx_total
    listener.0.0.0.0_15000.downstream_cx_total: 11
    curl -s http://my-app.default.svc.cluster.local:9901/stats | grep
     "listener.0.0.0.0_15000" | grep ssl.connection_error
    listener.0.0.0.0_15000.ssl.connection_error: 1
    ```

```
curl -s http://my-app.default.svc.cluster.local:9901/stats | grep
 "listener.0.0.0.0_15000" | grep ssl.handshake
listener.0.0.0.0_15000.ssl.handshake: 9
curl -s http://my-app.default.svc.cluster.local:9901/stats | grep
 "listener.0.0.0.0_15000" | grep ssl.session_reused
listener.0.0.0.0_15000.ssl.session_reused: 1
# Total CX (11) - SSL Connection Errors (1) == SSL Handshakes (9) + SSL Sessions
 Re-used (1)
```

- Para el proxy de Envoy que establece el origen de TLS:
  - Asegúrese de que el almacén de confianza de TLS se haya iniciado en la configuración de Envoy con el siguiente comando.

    ```
    curl http://my-app.default.svc.cluster.local:9901/certs
    ```

    Debería ver al menos una entrada bajo `certificates[].ca_certs` para los certificados utilizados para validar el certificado del backend al establecer el origen de TLS.
  - Asegúrese de que el número de conexiones salientes correctas al clúster de backend sea exactamente el mismo que el número de protocolos de enlace SSL más el número de sesiones SSL reutilizadas, como se muestra en los siguientes ejemplos de comandos y resultados.

    ```
    curl -s http://my-app.default.svc.cluster.local:9901/stats | grep "virtual-node-
    name" | grep upstream_cx_total
    cluster.cds_egress_mesh-name_virtual-node-name_protocol_port.upstream_cx_total: 11
    curl -s http://my-app.default.svc.cluster.local:9901/stats | grep "virtual-node-
    name" | grep ssl.connection_error
    cluster.cds_egress_mesh-name_virtual-node-name_protocol_port.ssl.connection_error:
     1
    curl -s http://my-app.default.svc.cluster.local:9901/stats | grep "virtual-node-
    name" | grep ssl.handshake
    cluster.cds_egress_mesh-name_virtual-node-name_protocol_port.ssl.handshake: 9
    curl -s http://my-app.default.svc.cluster.local:9901/stats | grep "virtual-node-
    name" | grep ssl.session_reused
    cluster.cds_egress_mesh-name_virtual-node-name_protocol_port.ssl.session_reused: 1
    # Total CX (11) - SSL Connection Errors (1) == SSL Handshakes (9) + SSL Sessions
     Re-used (1)
    ```

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support. Si cree que ha encontrado una vulnerabilidad de seguridad o tiene dudas sobre la seguridad de App Mesh, consulte las Directrices de notificación de vulnerabilidades de AWS.

Solución de problemas de TLS con el equilibrador de carga elástico

Síntomas

Al intentar configurar un Equilibrador de carga de aplicación o Equilibrador de carga de red para cifrar el tráfico a un nodo virtual, las comprobaciones de estado del equilibrador de carga y la conectividad pueden producir errores.

Resolución

Para determinar la causa raíz del problema, debe comprobar lo siguiente:

- Para el proxy de Envoy que realiza la terminación de TLS, debe descartar cualquier error de configuración. Siga los pasos que se indicaron anteriormente en No se puede conectar a un servicio virtual de backend con una política de cliente TLS.
- Para el equilibrador de carga, debe revisar la configuración de `TargetGroup`:
  - Asegúrese de que el puerto `TargetGroup` coincida con el puerto del oyente definido para el nodo virtual.
  - En el caso de los equilibradores de carga de aplicación que son el origen de conexiones TLS a través de HTTP con su servicio, asegúrese de que el protocolo `TargetGroup` esté establecido en `HTTPS`. Si se utilizan comprobaciones de estado, asegúrese de que `HealthCheckProtocol` esté establecido en `HTTPS`.
  - En el caso de los equilibradores de carga de red que son el origen de conexiones TLS a través de TCP con su servicio, asegúrese de que el `TargetGroup` protocolo esté configurado en `TLS`. Si se utilizan comprobaciones de estado, asegúrese de que `HealthCheckProtocol` esté establecido en `TCP`.

    > **Note**
    >
    > Cualquier actualización de `TargetGroup` requiere cambiar el nombre `TargetGroup`.

Con esta configuración correcta, el equilibrador de carga debería proporcionar una conexión segura a su servicio mediante el certificado proporcionado al proxy de Envoy.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support. Si cree que ha encontrado una vulnerabilidad de seguridad o tiene dudas sobre la seguridad de App Mesh, consulte las Directrices de notificación de vulnerabilidades de AWS.

Solución de problemas de App Mesh para Kubernetes

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

En este tema se describen los problemas comunes que pueden ocurrir al usar App Mesh con Kubernetes.

Los recursos de App Mesh creados en Kubernetes no se encuentran en App Mesh

Síntomas

Has creado los recursos de App Mesh con la definición de recursos personalizada (CRD) de Kubernetes, pero los recursos que has creado no están visibles en App Mesh cuando utilizas o. Consola de administración de AWS APIs

Resolución

La causa probable es un error del controlador de Kubernetes para App Mesh. Para obtener más información, consulte Solución de problemas en. GitHub Compruebe los registros del controlador para ver si hay errores o advertencias que indiquen que el controlador no ha podido crear ningún recurso.

```
kubectl logs -n appmesh-system -f \
    $(kubectl get pods -n appmesh-system -o name | grep controller)
```

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Las comprobaciones de disponibilidad y vivacidad de los pods tienen errores después de inyectar el sidecar de Envoy

Síntomas

Los pods de su aplicación se ejecutaban correctamente anteriormente, pero una vez que se inyecta el sidecar de Envoy en un pod, las comprobaciones de disponibilidad y vivacidad comienzan a producir errores.

Resolución

Asegúrese de que el contenedor de Envoy que se inyectó en el pod se haya iniciado con el servicio de administración de Envoy de App Mesh. Puede verificar cualquier error consultando los códigos de error en Envoy se desconectó del servicio de administración de Envoy de App Mesh mostrando un texto de error. Puede usar el siguiente comando para examinar los registros de Envoy del pod correspondiente.

```
kubectl logs -n appmesh-system -f \
    $(kubectl get pods -n appmesh-system -o name | grep controller) \
    | grep "gRPC config stream closed"
```

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Los pods no se registran o se dan de baja como instancias AWS Cloud Map

Síntomas

Tus pods de Kubernetes no se registran ni se cancelan del registro AWS Cloud Map como parte de su ciclo de vida. Es posible que un pod se inicie correctamente y esté listo para enviar tráfico, pero no para recibirlo. Cuando se cierra un pod, es posible que los clientes aún conserven su dirección IP e intenten enviarle tráfico, lo cual no funciona.

Resolución

Se trata de un problema conocido. Para obtener más información, consulta los pods no se vuelven automáticos registered/deregistered en Kubernetes con problemas. AWS Cloud Map GitHub Debido a la relación entre los pods, los nodos virtuales de App Mesh y los AWS Cloud Map recursos, el controlador App Mesh para Kubernetes puede desincronizarse y perder recursos. Por ejemplo,

esto puede suceder si se elimina un recurso de nodo virtual de Kubernetes antes de cerrar los pods asociados.

Para mitigar este problema:

- Asegúrese de ejecutar la última versión del controlador de App Mesh para Kubernetes.
- Asegúrese de que las letras AWS Cloud Map `namespaceName` y `serviceName` sean correctas en la definición de su nodo virtual.
- Asegúrese de eliminar todos los pods asociados antes de eliminar la definición de su nodo virtual. Si necesita ayuda para identificar qué pods están asociados a un nodo virtual, consulte No es posible determinar dónde se ejecuta un pod para un recurso de App Mesh.
- Si el problema persiste, ejecute el siguiente comando para examinar los registros del controlador en busca de errores que puedan ayudar a descubrir el problema subyacente.

```
kubectl logs -n appmesh-system \
    $(kubectl get pods -n appmesh-system -o name | grep appmesh-controller)
```

- Considere la posibilidad de usar el comando siguiente para reiniciar los pods de su controlador. Esto podría solucionar problemas de sincronización.

```
kubectl delete -n appmesh-system \
    $(kubectl get pods -n appmesh-system -o name | grep appmesh-controller)
```

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

No es posible determinar dónde se ejecuta un pod para un recurso de App Mesh

Síntomas

Cuando ejecuta App Mesh en un clúster de Kubernetes, un operador no puede determinar dónde se ejecuta una carga de trabajo, o pod, para un recurso de App Mesh determinado.

Resolución

Los recursos del pod de Kubernetes se anotan con la malla y el nodo virtual a los que están asociados. Puede consultar qué pods se están ejecutando para un nombre de nodo virtual determinado con el siguiente comando.

```
kubectl get pods --all-namespaces -o json | \
    jq '.items[] | { metadata } | select(.metadata.annotations."appmesh.k8s.aws/
virtualNode" == "virtual-node-name")'
```

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

No se puede determinar cómo qué recurso de App Mesh se está ejecutando un pod

Síntomas

Al ejecutar App Mesh en un clúster de Kubernetes, un operador no puede determinar cómo qué recurso de App Mesh se está ejecutando un pod determinado.

Resolución

Los recursos del pod de Kubernetes se anotan con la malla y el nodo virtual a los que están asociados. Puede generar los nombres de la malla y de los nodos virtuales consultando el pod directamente con el siguiente comando.

```
kubectl get pod pod-name -n namespace -o json | \
    jq '{ "mesh": .metadata.annotations."appmesh.k8s.aws/mesh",
 "virtualNode": .metadata.annotations."appmesh.k8s.aws/virtualNode" }'
```

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Los enviados del cliente no pueden comunicarse con App Mesh Envoy Management Service si está deshabilitado IMDSv1

Síntomas

Cuando `IMDSv1` está deshabilitado, los Envoys del cliente no pueden comunicar con el plano de control de App Mesh (servicio de administración de Envoy). `IMDSv2` no es compatible con la versión de App Mesh Envoy anterior a la `v1.24.0.0-prod`.

Resolución

Para resolver este problema, puede elegir una de estas tres soluciones.

- Actualice a la versión `v1.24.0.0-prod` o posterior de App Mesh Envoy, que es compatible con `IMDSv2`.
- Vuelva a habilitar `IMDSv1` en la instancia en la que se esté ejecutando Envoy. Para obtener instrucciones sobre la restauración de `IMDSv1`, consulte Configurar las opciones de metadatos de la instancia.
- Si sus servicios se ejecutan en Amazon EKS, se recomienda utilizar roles de IAM para cuentas de servicio (IRSA) para obtener las credenciales. Para obtener instrucciones sobre cómo habilitar IRSA, consulte Roles de IAM para cuentas de servicio.

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

IRSA no funciona en el contenedor de aplicaciones cuando App Mesh está habilitada y Envoy está inyectado

Síntomas

Cuando App Mesh está habilitada en un clúster de Amazon EKS con la ayuda del controlador App Mesh para Amazon EKS, Envoy y los contenedores `proxyinit` se inyectan en el pod de la aplicación. La aplicación no puede asumir `IRSA` y, en su lugar, asume `node role`. Cuando describimos los detalles del pod, vemos que la variable de entorno `AWS_WEB_IDENTITY_TOKEN_FILE` o `AWS_ROLE_ARN` no está incluida en el contenedor de la aplicación.

Resolución

Si se define alguna de las variables de entorno `AWS_WEB_IDENTITY_TOKEN_FILE` o `AWS_ROLE_ARN`, el webhook omitirá el pod. No proporcione ninguna de estas variables y el webhook se encargará de inyectarlas por usted.

```
reservedKeys := map[string]string{
        "AWS_ROLE_ARN":                "",
        "AWS_WEB_IDENTITY_TOKEN_FILE": "",
    }
    ...
    for _, env := range container.Env {
        if _, ok := reservedKeys[env.Name]; ok {
            reservedKeysDefined = true
```

```
        }
```

Si el problema sigue sin resolverse, considera abrir un GitHub problema o ponte en contacto con AWS Support.

Cuotas de servicio de App Mesh

> **Important**
>
> Aviso de fin de soporte: el 30 de septiembre de 2026, AWS suspenderemos el soporte para AWS App Mesh. Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

AWS App Mesh se ha integrado con Service Quotas, un AWS servicio que le permite ver y gestionar sus cuotas desde una ubicación central. Las cuotas de servicio también se denominan límites. Para obtener más información, consulte ¿Qué son las Service Quotas? en la Guía del usuario de Service Quotas.

Con Service Quotas, resulta más sencillo buscar el valor de todas las cuotas de servicio de App Mesh.

Para ver las cuotas de servicio de App Mesh mediante el Consola de administración de AWS

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 Servicios de AWS , busque y seleccione AWS App Mesh.

   En la lista de cuotas de servicio, puede ver el nombre de la cuota de servicio, el valor aplicado (si está disponible), la cuota AWS predeterminada y si el valor de la cuota es ajustable.
4. Para ver información adicional sobre una cuota de servicio, como, por ejemplo, la descripción, elija el nombre de cuota.

Para solicitar un aumento de cuota, consulte Solicitud de aumento de cuota en la Guía del usuario de Service Quotas.

Para ver las cuotas de servicio de App Mesh mediante el AWS CLI

Ejecute el siguiente comando.

```
aws service-quotas list-aws-default-service-quotas \
```

```
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code appmesh \
    --output table
```

Para trabajar más con las cuotas de servicio mediante el AWS CLI, consulte la Referencia de AWS CLI comandos de Service Quotas.

Historial de documentos de App Mesh

> **Important**
>
> Aviso de fin del soporte: el 30 de septiembre de 2026, AWS dejaremos de ofrecer soporte para. AWS App Mesh Después del 30 de septiembre de 2026, ya no podrás acceder a la AWS App Mesh consola ni a AWS App Mesh los recursos. Para obtener más información, visite esta entrada del blog Migración desde AWS App Mesh a Amazon ECS Service Connect.

En la siguiente tabla se describen las principales actualizaciones y nuevas características de la Guía del usuario de AWS App Mesh . Actualizamos la documentación con frecuencia para dar respuesta a los comentarios que se nos envía.

Cambio	Descripción	Fecha
Se actualizó la política de `AWSAppMeshFullAccess`	Se actualizó `AWSAppMeshFullAccess` para permitir el acceso a `TagResource` y `UntagResource` APIs.	24 de abril de 2024
CloudTrail documentación de integración actualizada	Se ha actualizado la documentación que describe la integración de App Mesh con la actividad de la API CloudTrail para registrar.	28 de marzo de 2024
Políticas actualizadas	Se actualizó `AWSServiceRoleForAppMesh` y `AWSAppMeshServiceRolePolicy` para permitir el acceso a la AWS Cloud Map `DiscoverInstancesRevision` API.	12 de octubre de 2023

Compatibilidad con la política de puntos de conexión de VPC para App Mesh	App Mesh ahora admite políticas de punto de conexión de VPC.	11 de mayo de 2023
Varios oyentes para App Mesh	App Mesh ahora admite varios oyentes.	18 de agosto de 2022
IPv6 para App Mesh	App Mesh ahora es compatible IPv6.	18 de mayo de 2022
CloudTrail soporte de registro para App Mesh Envoy Management Service	App Mesh ahora admite el CloudTrail registro para App Mesh Envoy Management Service.	18 de marzo de 2022
Agente de App Mesh para Envoy	App Mesh ahora es compatible con el Agente para Envoy.	25 de febrero de 2022
Varios oyentes para App Mesh	(Solo en App Mesh Preview Channel) puede implementar varios oyentes para App Mesh.	23 de noviembre de 2021
ARM64 soporte para App Mesh	App Mesh ahora es compatible ARM64.	19 de noviembre de 2021
Extensión de métricas para App Mesh	Puede implementar extensiones de métricas para App Mesh.	29 de octubre de 2021
Implementación de mejoras del tráfico entrante	Puede implementar la coincidencia de nombre de host y encabezado y las reescrituras de la ruta y el nombre del host.	14 de junio de 2021
Implementación de la autenticación TLS mutua	Puede implementar la autenticación TLS mutua.	4 de febrero de 2021

Lanzamiento en la región af-south-1	App Mesh ahora está disponible en la región af-south-1.	22 de enero de 2021
Implementación de la autenticación TLS mutua	(Solo en App Mesh Preview Channel) puede implementar la autenticación TLS mutua.	23 de noviembre de 2020
Implementación de la agrupación de conexiones para un oyente de puerta de enlace virtual	Puede implementar la agrupación de conexiones para un oyente de puerta de enlace virtual.	5 de noviembre de 2020
Implementación de la agrupación de conexiones y la detección de valores atípicos para un oyente de nodo virtual	Puede implementar la agrupación de conexiones y la detección de valores atípicos para un oyente de nodo virtual.	5 de noviembre de 2020
Lanzamiento en la región eu-south-1	App Mesh ahora está disponible en la región eu-south-1.	21 de octubre de 2020
Implementación de la agrupación de conexiones para un oyente de puerta de enlace virtual	(Solo en App Mesh Preview Channel) Puede implementar la agrupación de conexiones para un oyente de puerta de enlace virtual.	28 de septiembre de 2020
Implementación de la agrupación de conexiones y la detección de valores atípicos para un oyente de nodo virtual	(Solo en App Mesh Preview Channel) Puede implementar la agrupación de conexiones y la detección de valores atípicos para un oyente de nodo virtual.	28 de septiembre de 2020
Creación de una puerta de enlace virtual y una ruta de puerta de enlace para la entrada de malla	Permita que los recursos que están dentro de la malla comuniquen con los recursos que están dentro de ella.	10 de julio de 2020

Crear y administrar recursos de App Mesh desde Kubernetes con el controlador de App Mesh para Kubernetes	Puede crear y administrar recursos de App Mesh desde Kubernetes. El controlador también inyecta automáticamente el proxy de Envoy y los contenedores init en los pods que se implementan.	18 de junio de 2020
Adición de un valor de tiempo de espera a un oyente de nodo virtual y una ruta	Puede añadir un valor de tiempo de espera a un oyente de nodo virtual y una ruta.	18 de junio de 2020
Adición de un valor de tiempo de espera a un oyente de nodo virtual	(Solo en App Mesh Preview Channel) Puede añadir un valor de tiempo de espera a un oyente de nodo virtual.	29 mayo de 2020
Creación de una puerta de enlace virtual para la entrada de malla	(Solo en App Mesh Preview Channel) Habilite los recursos fuera de una malla para que se comuniquen con los recursos dentro de una malla.	8 de abril de 2020
Cifrado TLS	(Solo en el canal de vista previa de App Mesh) Utilice certificados de una AWS Private Certificate Authority o de su propia autoridad de certificación para cifrar la comunicación entre nodos virtuales mediante TLS.	17 de enero de 2020

Uso compartido de una malla con otra cuenta	(Solo en App Mesh Preview Channel) Puede compartir una malla con otra cuenta. Los recursos creados por cualquier cuenta se pueden comunicar con otros recursos de la malla.	17 de enero de 2020
Adición de un valor de tiempo de espera a una ruta	(Solo en App Mesh Preview Channel) Puede añadir un valor de tiempo de espera a una ruta.	17 de enero de 2020
Crea un proxy de App Mesh en un AWS Outpost	Puedes crear un proxy de App Mesh Envoy en un AWS Outpost.	3 de diciembre de 2019
Compatibilidad con HTTP/2 y gRPC para rutas, enrutadores virtuales y nodos virtuales	Puede enrutar el tráfico que use los protocolos HTTP/2 y gRPC. También puede agregar un oyente para estos protocolos a nodos virtuales y enrutadores virtuales.	25 de octubre de 2019
Política de reintentos	Una política de reintento s permite a los clientes protegerse de errores intermite ntes de red o errores intermite ntes del lado del servidor. Puede añadir una lógica de reintentos a una ruta.	10 de septiembre de 2019
Cifrado TLS	(Solo en App Mesh Preview Channel) Cifre la comunicac ión entre los nodos virtuales mediante TLS.	6 de septiembre de 2019

Enrutamiento HTTP basado en encabezados	Enrute el tráfico en función de la presencia y los valores de los encabezados HTTP de una solicitud.	15 de agosto de 2019
Disponibilidad de App Mesh Preview Channel	App Mesh Preview Channel es una variante distinta del servicio de App Mesh. Preview Channel muestra las próximas características para que las pruebe a medida que se desarrollan. A medida que utilices las funciones del canal de vista previa, podrás enviar comentarios GitHub para determinar el comportamiento de las funciones.	19 de julio de 2019
Puntos finales de VPC de la interfaz App Mesh ()AWS PrivateLink	Puede mejorar la posición de seguridad de su VPC configurando App Mesh para que utilice un punto de conexión de VPC de interfaz. Los puntos finales de la interfaz funcionan con una tecnología que te permite acceder de forma privada a App Mesh APIs mediante direcciones IP privadas. AWS PrivateLink PrivateLink restringe todo el tráfico de red entre tu VPC y App Mesh a la red de Amazon.	14 de junio de 2019

Se agregó AWS Cloud Map como método de descubrimiento de servicios de nodos virtuales	Puede especificar el DNS o AWS Cloud Map como un método de descubrimiento de servicios de nodos virtuales. Para utilizar AWS Cloud Map para la detección de servicios, su cuenta debe tener el rol vinculado al servicio de App Mesh.	13 de junio de 2019
Creación automática de recursos de App Mesh en Kubernetes	Cree recursos de App Mesh y añada automáticamente las imágenes del contenedor de sidecar de App Mesh a sus implementaciones de Kubernetes cuando cree recursos en Kubernetes.	11 de junio de 2019
Disponibilidad general de App Mesh	El servicio de App Mesh ahora está disponible con carácter general para su uso con fines de producción.	27 de marzo de 2019
Actualización de la API de App Mesh	APIs Se actualizó la App Mesh para mejorar la usabilidad. Para obtener más información, consulte [BUG] Rutas a nodos virtuales de destino con puertos no coincidentes (Blackhole).	7 de marzo de 2019
Versión inicial de App Mesh	Documentación inicial de la versión preliminar pública del servicio	28 de noviembre de 2018

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la version original de inglés, prevalecerá la version en inglés.