



Guida per l'utente

AWS App Mesh



AWS App Mesh: Guida per l'utente

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

I marchi e l'immagine commerciale di Amazon non possono essere utilizzati in relazione a prodotti o servizi che non siano di Amazon, in una qualsiasi modalità che possa causare confusione tra i clienti o in una qualsiasi modalità che denigri o discrediti Amazon. Tutti gli altri marchi non di proprietà di Amazon sono di proprietà delle rispettive aziende, che possono o meno essere associate, collegate o sponsorizzate da Amazon.

Table of Contents

Cosa è AWS App Mesh?	1
Aggiungere App Mesh a un'applicazione di esempio	1
Componenti di App Mesh	2
Come iniziare	4
Accesso a App Mesh	4
Nozioni di base	6
App Mesh e Amazon ECS	6
Scenario	6
Prerequisiti	7
Fase 1: creare una mesh e un servizio virtuale	7
Fase 2: creare un nodo virtuale	8
Fase 3: creare un router virtuale e una route	10
Fase 4: Revisione e creazione	12
Fase 5: creare risorse aggiuntive	13
Fase 6: aggiornare i servizi	18
Argomenti avanzati	35
App Mesh e Kubernetes	35
Prerequisiti	36
Fase 1: installazione dei componenti di integrazione	37
Fase 2: Implementazione delle risorse App Mesh	43
Fase 3: creazione o aggiornamento dei servizi	57
Fase 4: pulizia	64
App Mesh e Amazon EC2	64
Scenario	6
Prerequisiti	7
Fase 1: creare una mesh e un servizio virtuale	66
Fase 2: creare un nodo virtuale	67
Fase 3: creare un router virtuale e una route	10
Fase 4: Revisione e creazione	12
Fase 5: creare risorse aggiuntive	13
Fase 6: aggiornare i servizi	18
App Mesh	87
Concetti	88
Mesh	88

Creazione di una mesh di servizi	88
Eliminazione di una rete	91
Servizi virtuali	92
Creazione di un servizio virtuale	93
Eliminazione di un servizio virtuale	95
Gateway virtuali virtuale virtuale virtuale virtuale virtuale	97
Creazione di un gateway virtuale virtuale virtuale virtuale	98
Implementazione di gateway virtuale virtuale virtuale	103
Eliminazione di un gateway virtuale virtuale virtuale virtuale virtuale virtuale	104
route del gateway	105
Nodi virtuali	112
Creazione di un nodo virtuale	113
Eliminazione di un nodo virtuale	124
Router virtuali	125
Creazione di un router virtuale	126
Eliminazione di un router virtuale	128
Route	130
Envoy	142
Varianti dell'immagine Envoy	142
Variabili di configurazione di Envoy	147
Variabili obbligatorie	147
Variabili opzionali	148
Impostazioni predefinite di Envoy impostate da App Mesh	155
Politica predefinita per i nuovi tentativi di routing	155
Interruttore automatico predefinito	156
Aggiornamento/migrazione a Envoy 1.17	157
Secret Discovery Service con SPIRE	157
Modifiche alle espressioni regolari	157
Riferimenti all'indietro	160
Agente dell'ambasciatore	161
Osservabilità	163
Registrazione	163
Firelens e Cloudwatch	165
Metriche Envoy	165
Parametri dell'applicazione di esempio	168
Esportazione di parametri	172

Tracciamento	180
X-Ray	180
Jaeger	183
Datadog per il tracciamento	151
Utensili	185
AWS CloudFormation	185
AWS CDK	185
Controller App Mesh per Kubernetes	185
Terraform	186
Utilizzo di mesh condivise	187
Concessione delle autorizzazioni per condividere le mesh	187
Concessione dell'autorizzazione alla condivisione di una mesh	187
Concessione delle autorizzazioni per una mesh	188
Prerequisiti per la condivisione delle mesh	189
Servizi correlati	190
Condivisione di una rete	190
Annullamento della condivisione di una mesh condivisa	191
Identificare una mesh condivisa	191
Fatturazione e misurazione	192
Quote di istanze	192
Utilizzo di altri servizi	193
Creazione di risorse App Mesh conAWS CloudFormation	193
App Mesh eAWS CloudFormation modelli	193
Ulteriori informazioni su AWS CloudFormation	193
App Mesh sugliAWS Outposts	194
Prerequisiti	194
Restrizioni	194
Considerazioni sulla connettività di rete	194
Creazione di un proxy App Mesh Envoy su un Outpost	195
Best practice	196
Strumenta tutti i percorsi con nuovi tentativi	196
Regola la velocità di implementazione	197
Scalabilità orizzontale prima della scalabilità interna	198
Implementa controlli sullo stato dei container	198
Ottimizza la risoluzione DNS	199
Protezione delle applicazioni	200

Sicurezza a livello di trasporto (TLS)	201
Requisiti del certificato	202
TLScertificati di autenticazione	202
In che modo App Mesh configura gli Envoy per negoziare TLS	205
Verifica la crittografia	206
Rinnovo del certificato	207
Configura i ECS carichi di lavoro Amazon per utilizzare l'TLSautenticazione con AWS App Mesh	207
Configura i carichi di lavoro Kubernetes per utilizzare l'autenticazione con TLS AWS App Mesh	208
Autenticazione TLS reciproca	209
Certificati di autenticazione TLS reciproca	209
Configura gli endpoint mesh	210
Migra i servizi all'autenticazione TLS reciproca	210
Verifica dell'autenticazione TLS reciproca	211
Procedure dettagliate per l'autenticazione TLS reciproca di App Mesh	212
Gestione dell'identità e degli accessi	212
Destinatari	213
Autenticazione con identità	213
Gestione dell'accesso con policy	216
In che modo AWS App Mesh funziona con IAM	219
Esempi di policy basate su identità	223
AWS politiche gestite	228
Utilizzo di ruoli collegati ai servizi	231
Autorizzazione proxy Envoy Proxy	234
Risoluzione dei problemi	239
CloudTrail registri	241
Eventi di gestione App Mesh in CloudTrail	242
Esempi di eventi App Mesh	243
Protezione dei dati	244
Crittografia dei dati	245
Convalida della conformità	245
Sicurezza dell'infrastruttura	246
Endpoint VPC di interfaccia (AWS PrivateLink)	247
Resilienza	249
Disaster recovery inAWS App Mesh	249

Analisi della configurazione e delle vulnerabilità	249
Risoluzione dei problemi	250
Best practice	250
Abilita l'interfaccia di amministrazione del proxy Envoy	250
Abilita l'integrazione con Envoy DogStats D per l'offload metrico	251
Abilitare log di accesso	251
Abilita la registrazione di debug di Envoy negli ambienti di preproduzione	251
Monitora la connettività del proxy Envoy con il piano di controllo App Mesh	252
Installazione	252
Impossibile estrarre l'immagine del contenitore Envoy	252
Impossibile connettersi al servizio di gestione App Mesh Envoy	253
Envoy disconnesso dal servizio di gestione App Mesh Envoy con testo di errore	254
Controllo dello stato del container Envoy, sonda di prontezza o sonda di vivacità guasto	257
Il controllo dello stato di salute dal load balancer all'endpoint mesh non riesce	257
Il gateway virtuale non accetta traffico sulle porte 1024 o inferiori	258
Connettività	259
Impossibile risolvere il nome DNS per un servizio virtuale	259
Impossibile connettersi a un backend di servizio virtuale	260
Impossibile connettersi a un servizio esterno	261
Impossibile connettersi a un server MySQL o SMTP	262
Impossibile connettersi a un servizio modellato come nodo virtuale TCP o router virtuale in App Mesh	263
La connettività riesce al servizio non elencato come backend di servizio virtuale per un nodo virtuale	264
Alcune richieste hanno esito negativo con il codice di stato HTTP 503 quando un servizio virtuale ha un provider di nodi virtuali	265
Impossibile connettersi a un file system Amazon EFS	265
La connettività riesce correttamente al servizio, ma la richiesta in arrivo non viene visualizzata nei log di accesso, nelle tracce o nelle metriche di accesso di Envoy	266
L'impostazione delle variabili di HTTPS_PROXY ambienteHTTP_PROXY/a livello di contenitore non funziona come previsto.	266
I timeout delle richieste upstream vengono interrotti anche dopo aver impostato il timeout per le rotte.	267
Envoy risponde con una richiesta HTTP Bad.	268
Impossibile configurare correttamente il timeout.	269
Dimensionamento	269

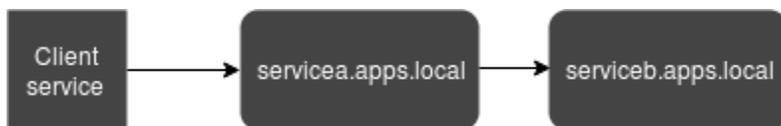
La connettività fallisce e i controlli di integrità dei container falliscono quando si scalano oltre 50 repliche per un nodo virtuale/gateway virtuale	269
Le richieste hanno esito negativo 503 quando il backend di un servizio virtuale viene scalato orizzontalmente o verticalmente	270
Il contenitore Envoy si blocca con segfault in caso di carico aumentato	270
L'aumento delle risorse predefinite non si riflette nei limiti del servizio	271
L'applicazione si arresta in modo anomalo a causa di un numero enorme di chiamate per i controlli sanitari.	271
Osservabilità	271
Impossibile visualizzare le AWS X-Ray tracce delle mie applicazioni	272
Impossibile visualizzare i parametri di Envoy per le mie applicazioni nei parametri di Amazon CloudWatch	272
Impossibile configurare regole di campionamento personalizzate per le tracce AWS X-Ray	273
Sicurezza	274
Impossibile connettersi a un servizio virtuale di backend con una politica client TLS	275
Impossibile connettersi a un servizio virtuale di backend quando l'applicazione sta originando TLS	276
Impossibile affermare che la connettività tra i proxy Envoy utilizzi TLS	276
Risoluzione dei problemi relativi al TLS con Elastic Load Balancing	278
Kubernetes	279
Le risorse App Mesh create in Kubernetes non possono essere trovate in App Mesh	279
Dopo l'iniezione del sidecar Envoy, i pod non sono stati sottoposti ai controlli di prontezza e vivacità	280
I pod non vengono registrati o annullati come istanze AWS Cloud Map	280
Impossibile determinare dove è in esecuzione un pod per una risorsa App Mesh	282
Impossibile determinare su quale risorsa App Mesh sia in esecuzione un pod	282
I Client Envoy non sono in grado di comunicare con App Mesh Envoy Management Service con IMDSv1 disabilitato	283
IRSA non funziona sul contenitore dell'applicazione quando App Mesh è abilitato e Envoy viene iniettato	283
Anteprima del canale	285
Service Quotas	290
Cronologia dei documenti	291
.....	ccxcviii

Cosa è AWS App Mesh?

AWS App Mesh è un mesh di servizio che facilita il monitoraggio e il controllo dei servizi. Una service mesh è un livello di infrastruttura dedicato alla gestione delle service-to-service comunicazioni, in genere tramite una serie di proxy di rete leggeri distribuiti insieme al codice dell'applicazione. App Mesh di App azione di App azione di comunicazione tra i servizi, offrendo end-to-end visibilità e contribuendo a garantire elevata disponibilità per le applicazioni. App Mesh offre visibilità coerente e controlli del traffico di rete per ogni servizio in un'applicazione.

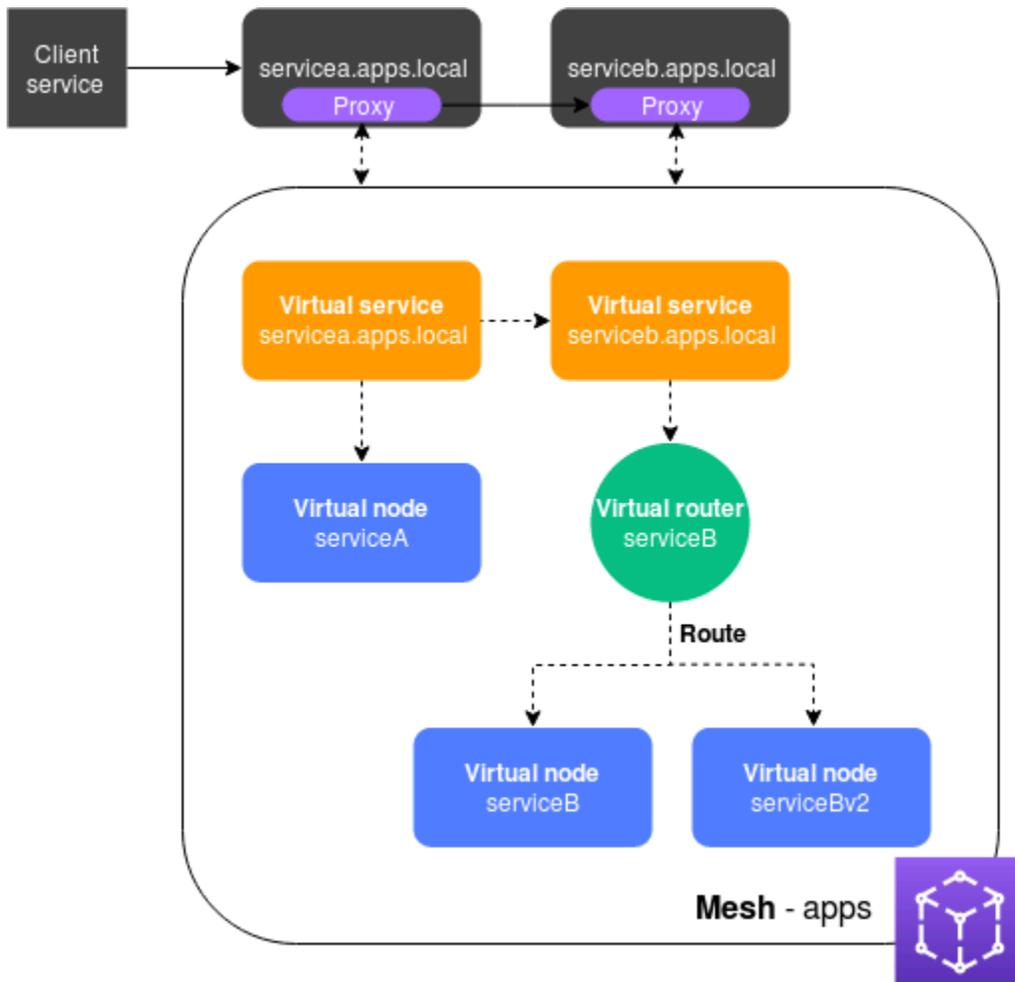
Aggiungere App Mesh a un'applicazione di esempio

Considera la seguente semplice applicazione di esempio che non utilizza App Mesh. I due servizi possono essere eseguiti su Amazon Elastic Container Intainer Intainer Service (Amazon EKS)AWS Fargate, Kubernetes su istanze Amazon Elastic Compute Cloud (Amazon EKS), Kubernetes su istanze Amazon Elastic Compute Cloud (Amazon EKS), Kubernetes su istanze Amazon Elastic Compute Cloud (Amazon EKS), Kubernetes su istanze Amazon Elastic Compute Cloud (Amazon EKS), Kubernetes su istanze Amazon Elastic Compute Cloud (Amazon EKS).



In questa illustrazione, entrambi `serviceA` e `serviceB` sono individuabili tramite il namespace `apps.local`. Supponiamo, ad esempio, che tu decida di distribuire una nuova versione di `serviceb.apps.local` denominata `servicebv2.apps.local`. Successivamente, desideri indirizzare una percentuale del traffico da `servicea.apps.local` a `serviceb.apps.local` e una percentuale verso `servicebv2.apps.local`. Quando sei sicuro che `servicebv2` stia funzionando bene, vuoi inviargli il 100 per cento del traffico.

App Mesh può aiutarti a farlo senza modificare il codice dell'applicazione o i nomi dei servizi registrati. Se usi App Mesh con questa applicazione di esempio, la tua mesh potrebbe assomigliare alla seguente illustrazione.



In questa configurazione, i servizi non comunicano più direttamente tra loro. Invece, comunicano tra loro tramite un proxy. Il proxy distribuito con il servizio `servicea.apps.local` legge la configurazione App Mesh e invia il traffico verso `serviceb.apps.local` o `servicebv2.apps.local` in base alla configurazione.

Componenti di App Mesh

App Mesh è composta dai seguenti componenti, illustrazione dell'esempio precedente:

- **Service azione di servizio:** una mesh dei servizi è un limite logico per il traffico di rete tra i servizi che si trovano al suo interno. Nell'esempio, la rete è denominata `apps` e contiene tutte le altre risorse per la mesh. Per ulteriori informazioni, consulta [Resh di servizio](#).
- **Servizi virtuali:** un servizio virtuale è un'astrazione di un servizio effettivo fornita direttamente o indirettamente da un nodo virtuale, direttamente o indirettamente, tramite un router virtuale. Nell'illustrazione, due servizi virtuali rappresentano i due servizi effettivi. I nomi dei servizi virtuali

sono i nomi individuabili dei servizi effettivi. Quando un servizio virtuale e un servizio effettivo hanno lo stesso nome, più servizi possono comunicare tra loro utilizzando gli stessi nomi utilizzati prima dell'implementazione di App Mesh. Per ulteriori informazioni, consulta [Servizi virtuali](#).

- **Nodi virtuali:** un nodo virtuale agisce come un puntatore logico a un servizio rilevabile, ad esempio un servizio Amazon ECS o Kubernetes. Per ogni servizio virtuale, avrai almeno un nodo virtuale. Nell'illustrazione, il servizio `servicea.apps.local` virtuale ottiene le informazioni di configurazione per il nodo virtuale denominato `serviceA`. Il nodo `serviceA` virtuale è configurato con il servizio `servicea.apps.local` nome per l'individuazione del servizio. Il servizio `serviceb.apps.local` virtuale è configurato per indirizzare il traffico verso il nodo `serviceBv2` virtuale e attraverso un router virtuale denominato `serviceB`. Per ulteriori informazioni, consulta [Nodi virtuali](#).
- **Router e percorsi virtuali:** i router virtuali gestiscono il traffico per uno o più servizi virtuali all'interno della mesh. Un percorso è associato a un router virtuale. Il percorso viene utilizzato per abbinare le richieste per il router virtuale e per distribuire il traffico ai nodi virtuali associati. Nell'illustrazione precedente, il router `serviceB` virtuale ha un percorso che indirizza una percentuale di traffico verso il nodo `serviceB` virtuale e una percentuale di traffico verso il nodo `serviceBv2` virtuale. Puoi impostare la percentuale di traffico indirizzato a un particolare nodo virtuale e modificarla nel tempo. È possibile indirizzare il traffico in base a criteri quali intestazioni HTTP, percorsi URL o nomi di servizi e metodi gRPC. È possibile configurare i criteri di ripetizione per riprovare una connessione in caso di errore nella risposta. Ad esempio, nell'illustrazione, la politica di ripetizione del percorso può specificare che una connessione a `serviceb.apps.local` venga ritentata cinque volte, con dieci secondi tra i tentativi, se `serviceb.apps.local` restituisce tipi di errori specifici. Per ulteriori informazioni, consultare [Router virtuali](#) e [Route](#).
- **Proxy:** configuri i tuoi servizi per utilizzare il proxy dopo aver creato la mesh e le relative risorse. Il proxy legge la configurazione di App Mesh e indirizza il traffico in modo appropriato. Nell'illustrazione, tutte le comunicazioni da `servicea.apps.local` a `serviceb.apps.local` passano attraverso il proxy distribuito con ciascun servizio. I servizi comunicano tra loro utilizzando gli stessi nomi di ricerca dei servizi utilizzati prima di introdurre App Mesh. Poiché il proxy legge la configurazione App Mesh, puoi controllare come i due servizi comunicano tra loro. Quando si desidera modificare la configurazione di App Mesh, non è necessario modificare o ridistribuire i servizi stessi o i proxy. Per ulteriori informazioni, consulta [Immagine dell'inviato](#).

Come iniziare

Per utilizzare App Mesh devi avere un servizio esistente in esecuzione su AWS Fargate Amazon ECS, Amazon EKS, Kubernetes su Amazon EC2 o Amazon EC2 con Docker.

Per iniziare a usare App Mesh, consulta una delle seguenti guide:

- [Guida introduttiva introduttiva App Mesh e Amazon ECS](#)
- [Guida introduttiva introduttiva App Mesh e Kubernetes](#)
- [Guida introduttiva introduttiva App Mesh e Amazon EC2](#)

Accesso a App Mesh

Puoi utilizzare App Mesh nei modi seguenti:

AWS Management Console

La console è un'interfaccia basata sul browser che può essere utilizzata per gestire le risorse di App Mesh. Puoi aprire la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).

AWS CLI

Fornisce comandi per un ampio kit di prodotti AWS ed è supportata su Windows, Mac e Linux. Per iniziare, consulta la [Guida per l'utente di AWS Command Line Interface](#). Per ulteriori informazioni sui comandi per App Mesh, consulta di [appesh](#) nella Documentazione di [riferimento per iAWS CLI comandi](#).

AWS Tools for Windows PowerShell

Fornisce comandi per un ampio kit di AWS prodotti per coloro che eseguono script nell'PowerShell ambiente. Per iniziare, consulta la [AWS Tools for Windows PowerShell Guida per l'utente di](#). Per ulteriori informazioni sui cmdlet per App Mesh, vedere [App Mesh](#) in [AWSTools for PowerShell Cmdlet Reference](#).

AWS CloudFormation

Consente di creare un modello che descrive tutte le AWS risorse desiderate. Utilizzando il modello, AWS CloudFormation fornisce e configura le risorse per te. Per iniziare, consulta la [\(Guida per l'utente di AWS CloudFormation\)](#). Per ulteriori informazioni sui tipi di risorse App Mesh, consulta [App Mesh Resource Type Reference](#) nel [AWS CloudFormation Template Reference](#).

SDK AWS

Forniamo anche degli SDK che consentono di accedere a App Mesh da una varietà di linguaggi di programmazione. Gli SDK si occupano automaticamente di attività come:

- Firma crittografica delle richieste di servizio
- Nuovi tentativi di richiesta
- Gestione delle risposte di errore

Per ulteriori informazioni sugli SDK disponibili, consultare [Strumenti per Amazon Web Services](#).

Per ulteriori informazioni sulle API di App Mesh, consultazione di riferimento dell'[AWS App MeshAPI di riferimento](#).

Guida introduttiva a App Mesh

Puoi utilizzare App Mesh con applicazioni distribuite su AmazonECS, Kubernetes (che distribuisce sulle tue EC2 istanze Amazon o eseguite su Amazon) e AmazonEKS. Per iniziare a usare App Mesh, seleziona uno dei servizi su cui sono distribuite le applicazioni che desideri utilizzare con App Mesh. Puoi sempre consentire alle applicazioni degli altri servizi di funzionare anche con App Mesh dopo aver completato una delle guide introduttive.

Argomenti

- [Guida introduttiva AWS App Mesh ad Amazon ECS](#)
- [Guida introduttiva a AWS App Mesh Kubernetes](#)
- [Guida introduttiva AWS App Mesh ad Amazon EC2](#)
- [App Mesh](#)

Guida introduttiva AWS App Mesh ad Amazon ECS

Questo argomento ti aiuta a utilizzare AWS App Mesh con un servizio effettivo in esecuzione su AmazonECS. Questo tutorial copre le funzionalità di base di diversi tipi di risorse App Mesh.

Scenario

Per illustrare come utilizzare App Mesh, supponiamo di avere un'applicazione con le seguenti caratteristiche:

- È costituito da due servizi denominati `serviceA` e `serviceB`.
- Entrambi i servizi sono registrati in uno spazio dei nomi denominato `apps.local`.
- `ServiceA` comunica con `serviceB` oltre HTTP /2, porta 80.
- Hai già distribuito la versione 2 di `serviceB` e l'hai registrata con il nome `serviceBv2` nello spazio dei nomi `apps.local`.

I requisiti sono i seguenti:

- Vuoi inviare il 75 per cento del traffico da `serviceA` a `serviceB` e il 25 per cento del traffico a `serviceBv2`. Inviando solo il 25 per cento a `serviceBv2`, puoi verificare che sia privo di bug prima di inviare il 100% del traffico da `serviceA`.

- Vuoi essere in grado di regolare facilmente la ponderazione del traffico in modo che il 100% del traffico vada a `serviceBv2` una volta dimostrato essere affidabile. Una volta inviato tutto il traffico a `serviceBv2`, vuoi `serviceB` interromperlo.
- Per soddisfare i requisiti precedenti non è necessario modificare il codice dell'applicazione esistente o la registrazione di Service Discovery per usufruire dei servizi effettivi.

Per soddisfare le tue esigenze, decidi di creare una mesh di servizi App Mesh con servizi virtuali, nodi virtuali, un router virtuale e un percorso. Dopo aver implementato la mesh, aggiorni i servizi per utilizzare il proxy Envoy. Una volta aggiornati, i servizi comunicano tra loro tramite il proxy Envoy anziché direttamente tra loro.

Prerequisiti

- Una comprensione esistente dei concetti di App Mesh. Per ulteriori informazioni, consulta [Cosa è AWS App Mesh?](#).
- Una comprensione esistente dei concetti di Amazon ECS. Per ulteriori informazioni, consulta [What is Amazon ECS nella Amazon Elastic Container Service Developer Guide](#).
- App Mesh supporta i servizi Linux registrati con DNS o entrambi. AWS Cloud Map Per utilizzare questa guida introduttiva, ti consigliamo di avere tre servizi esistenti registrati presso DNS. Le procedure descritte in questo argomento presuppongono che i servizi esistenti abbiano un nome `serviceA` `serviceBv2` e che tutti i servizi siano individuabili tramite uno spazio dei nomi denominato `serviceB apps.local`

Puoi creare una mesh dei servizi e le relative risorse anche se i servizi non esistono, ma non puoi utilizzare la mesh fino a quando non hai distribuito i servizi effettivi. Per ulteriori informazioni sulla scoperta dei servizi su Amazon ECS, consulta [Service Discovery](#). Per creare un ECS servizio Amazon con service discovery, consulta [Tutorial: Creazione di un servizio utilizzando Service Discovery](#). Se non disponi già di servizi in esecuzione, puoi [creare un ECS servizio Amazon con service discovery](#).

Fase 1: creare una mesh e un servizio virtuale

Una mesh dei servizi è un limite logico per il traffico di rete tra i servizi che si trovano al suo interno. Per ulteriori informazioni, consulta [Resh di servizio](#). Un servizio virtuale è un'astrazione di un servizio effettivo. Per ulteriori informazioni, consulta [Servizi virtuali](#).

Crea le seguenti risorse:

- Una mesh denominata `apps`, perché tutti i servizi nello scenario sono registrati nello spazio dei nomi `apps.local`.
- Un servizio virtuale denominato `serviceb.apps.local` perché il servizio virtuale rappresenta un servizio individuabile tramite questo nome e non vuoi modificare il codice per fare riferimento a un altro nome. Un servizio virtuale denominato `servicea.apps.local` viene aggiunto in una fase successiva.

Puoi utilizzare la AWS CLI versione 1.18.116 AWS Management Console o successiva o 2.0.38 o successiva per completare i seguenti passaggi. Se si utilizza il AWS CLI, utilizzare il `aws --version` comando per verificare la versione installata. AWS CLI Se non è installata la versione 1.18.116 o successiva o la versione 2.0.38 o successiva, è necessario [installare](#) o aggiornare la AWS CLI Seleziona la scheda relativa allo strumento che desideri utilizzare.

AWS Management Console

1. [Apri la procedura guidata per la prima esecuzione della console App Mesh all'inizio](https://console.aws.amazon.com/appmesh/)<https://console.aws.amazon.com/appmesh/>.
2. Per Mesh name (Nome mesh), immettere **apps**.
3. Per Virtual service name (Nome servizio virtuale), immettere **serviceb.apps.local**.
4. Scegliere Next (Avanti) per continuare.

AWS CLI

1. Creare una mesh con il comando [create-mesh](#).

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

2. Creare un servizio virtuale con il comando [create-virtual-service](#).

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

Fase 2: creare un nodo virtuale

Un nodo virtuale funziona come un puntatore logico a un servizio effettivo. Per ulteriori informazioni, consulta [Nodi virtuali](#).

Crea un nodo virtuale denominato `serviceB` perché uno dei nodi virtuali rappresenta il servizio effettivo denominato `serviceB`. Il servizio effettivo rappresentato dal nodo virtuale è individuabile tramite DNS con il nome host `serviceb.apps.local`. In alternativa, puoi scoprire i servizi effettivi utilizzando AWS Cloud Map. Il nodo virtuale ascolterà il traffico utilizzando il protocollo HTTP /2 sulla porta 80. Sono supportati anche altri protocolli, così come i controlli di stato. Verranno creati nodi virtuali per `serviceA` e `serviceBv2` in una fase successiva.

AWS Management Console

1. Per Virtual node name (Nome nodo virtuale), immettere **serviceB**.
2. Per il metodo di individuazione del servizio, scegli DNS e inserisci **serviceb.apps.local** il nome DNS host.
3. In Configurazione del listener, scegli http2 per Protocollo e immetti **80** per Porta.
4. Scegliere Next (Avanti) per continuare.

AWS CLI

1. Crea un file denominato `create-virtual-node-serviceb.json` con i seguenti contenuti:

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

2. Crea il nodo virtuale con il [create-virtual-node](#) comando utilizzando il JSON file come input.

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

Fase 3: creare un router virtuale e una route

I router virtuali gestiscono il traffico per uno o più servizi virtuali all'interno della mesh. Per ulteriori informazioni, consulta [Router virtuali](#) e [Route](#).

Crea le seguenti risorse:

- Un router virtuale denominato `serviceB` perché il servizio virtuale `serviceB.apps.local` non avvia la comunicazione in uscita con altri servizi. Tieni presente che il servizio virtuale creato in precedenza è un'astrazione del servizio `serviceb.apps.local` effettivo. Il servizio virtuale invia il traffico al router virtuale. Il router virtuale ascolta il traffico utilizzando il protocollo HTTP /2 sulla porta 80. Sono supportati anche altri protocolli.
- Una route denominata `serviceB`. Indirizza il 100% del traffico verso il nodo `serviceB` virtuale. Il peso viene calcolato in una fase successiva dopo l'aggiunta del nodo `serviceBv2` virtuale. Sebbene non sia trattato in questa guida, è possibile impostare criteri di filtro aggiuntivi per la route e aggiungere una policy per i nuovi tentativi per far sì che il proxy Envoy effettui più tentativi di invio del traffico a un nodo virtuale quando si verifica un problema di comunicazione.

AWS Management Console

1. Per Virtual router name (Nome router virtuale), immettere **serviceB**.
2. In Configurazione del listener, scegli `http2` per Protocollo e specifica **80** per Porta.
3. Per Route name (Nome route), immettere **serviceB**.
4. Per Tipo di route, scegli `http2`.
5. Per il nome del nodo virtuale nella configurazione Target, seleziona `serviceB` e inserisci **100** Weight.
6. In Match configuration, scegli un metodo.
7. Scegliere Next (Avanti) per continuare.

AWS CLI

1. Creare un router virtuale.
 - a. Crea un file denominato `create-virtual-router.json` con i seguenti contenuti:

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

- b. Crea il router virtuale con il [create-virtual-router](#) comando utilizzando il JSON file come input.

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

2. Creare una route.
 - a. Crea un file denominato `create-route.json` con i seguenti contenuti:

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

```
    },  
    "match" : {  
      "prefix" : "/"  
    }  
  }  
},  
"virtualRouterName" : "serviceB"  
}
```

- b. Crea il percorso con il comando [create-route](#) utilizzando il JSON file come input.

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

Fase 4: Revisione e creazione

Rivedi le impostazioni in relazione alle istruzioni precedenti.

AWS Management Console

Scegli **Modifica** per apportare modifiche in qualsiasi sezione. Una volta soddisfatte le impostazioni, scegli **Crea una mesh**.

La schermata **Stato** mostra tutte le risorse mesh create. Puoi visualizzare le risorse create nella console selezionando **Visualizza mesh**.

AWS CLI

Esaminare le impostazioni della mesh creata con il comando [describe-mesh](#) .

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

Rivedi le impostazioni del servizio virtuale che hai creato con il [describe-virtual-service](#) comando.

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

Rivedi le impostazioni del nodo virtuale che hai creato con il [describe-virtual-node](#) comando.

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

Rivedi le impostazioni del router virtuale che hai creato con il [describe-virtual-router](#) comando.

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

Esaminare le impostazioni del percorso creato con il comando [describe-route](#) .

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

Fase 5: creare risorse aggiuntive

Per completare lo scenario è necessario:

- Creare un nodo virtuale denominato `serviceBv2` e un altro denominato `serviceA`. Entrambi i nodi virtuali ascoltano le richieste sulla porta HTTP /2 80. Per il nodo `serviceA` virtuale, configura un backend di `serviceb.apps.local`. Tutto il traffico in uscita dal nodo `serviceA` virtuale viene inviato al servizio virtuale denominato `serviceb.apps.local`. Sebbene non sia trattato in questa guida, è anche possibile specificare un percorso file in cui scrivere i log degli accessi per un nodo virtuale.
- Crea un servizio virtuale aggiuntivo denominato `servicea.apps.local`, che invia tutto il traffico direttamente al nodo `serviceA` virtuale.
- Aggiornare la route `serviceB` creata in una fase precedente per inviare il 75% del traffico al nodo virtuale `serviceB` e il 25% del traffico al nodo virtuale `serviceBv2`. È possibile continuare a modificare i pesi finché `serviceBv2` riceva il 100% del traffico. Una volta inviato tutto il traffico a `serviceBv2`, puoi chiudere e interrompere il nodo `serviceB` virtuale e il servizio effettivo. Quando si cambiano i pesi, il codice non richiede alcuna modifica perché i nomi dei servizi virtuali ed effettivi `serviceb.apps.local` non cambiano. Tenere presente che il servizio virtuale `serviceb.apps.local` invia il traffico al router virtuale che instrada il traffico ai nodi virtuali. I nomi di individuazione dei servizi per i nodi virtuali possono essere modificati in qualsiasi momento.

AWS Management Console

1. Nel riquadro di navigazione a sinistra, selezionare Meshes (Mesh).
2. Selezionare la mesh `apps` creata in una fase precedente.
3. Nel riquadro di navigazione a sinistra selezionare Virtual nodes (Nodi virtuali).
4. Scegliere Create virtual node (Crea nodo virtuale).

5. Per il nome del nodo virtuale, inserisci **serviceBv2**, per Metodo di rilevamento del servizio DNS, scegli e per DNSil nome host, inserisci. **servicebv2.apps.local**
6. Per Configurazione del listener seleziona http2 per Protocollo e immetti **80** per Porta.
7. Scegliere Create virtual node (Crea nodo virtuale).
8. Scegli Crea un nodo virtuale. Immetti **serviceA** per Nome del nodo virtuale. Per il metodo di individuazione del servizio DNS, scegli e per il DNSnome host, inserisci. **servicea.apps.local**
9. Per Inserisci un nome del servizio virtuale in Nuovo back-end immetti **serviceb.apps.local**.
10. In Configurazione del listener scegli http2 per Protocollo, immetti **80** per Porta e quindi scegli Crea un nodo virtuale.
11. Nel riquadro di navigazione a sinistra selezionare Virtual routers (Router virtuali) quindi selezionare il router virtuale serviceB dall'elenco.
12. In Route, seleziona la route denominata ServiceB creata in una fase precedente e scegli Modifica.
13. In Destinazioni, Nome del nodo virtuale, cambia il valore di Peso per serviceB su **75**.
14. Scegli Aggiungi obiettivo, scegli serviceBv2 dall'elenco a discesa e imposta il valore di Weight su. **25**
15. Seleziona Salva.
16. Nel riquadro di navigazione a sinistra seleziona Servizi virtuali, quindi scegli Crea un servizio virtuale.
17. Immetti **servicea.apps.local** per Nome del servizio virtuale, seleziona Nodo virtuale per Provider, seleziona serviceA per Nodo virtuale e quindi scegli Crea un servizio virtuale.

AWS CLI

1. Creare il nodo virtuale serviceBv2.
 - a. Crea un file denominato create-virtual-node-servicebv2.json con i seguenti contenuti:

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

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

- b. Creare il nodo virtuale.

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

2. Creare il nodo virtuale serviceA.

- a. Crea un file denominato `create-virtual-node-servicea.json` con i seguenti contenuti:

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

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

- b. Creare il nodo virtuale.

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

3. Aggiornare il servizio virtuale `serviceb.apps.local` creato in una fase precedente per inviare il traffico al router virtuale `serviceB`. Quando il servizio virtuale è stato originariamente creato non inviava traffico poiché il router virtuale `serviceB` non era ancora stato creato.

- a. Crea un file denominato `update-virtual-service.json` con i seguenti contenuti:

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

- b. Aggiorna il servizio virtuale con il [update-virtual-service](#) comando.

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

4. Aggiornare la route `serviceB` creata in una fase precedente.

- a. Crea un file denominato `update-route.json` con i seguenti contenuti:

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

- b. Aggiornare la route con il comando [update-route](#).

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

5. Creare il servizio virtuale serviceA.

- a. Crea un file denominato `create-virtual-servicea.json` con i seguenti contenuti:

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

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

- b. Creare il servizio virtuale.

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

Riepilogo della mesh

Prima di creare la mesh dei servizi, erano presenti tre servizi effettivi denominati `servicea.apps.local`, `serviceb.apps.local` e `servicebv2.apps.local`. Oltre ai servizi effettivi, ora hai una mesh dei servizi che contiene le seguenti risorse che rappresentano i servizi effettivi:

- Due servizi virtuali. Il proxy invia tutto il traffico dal servizio virtuale `servicea.apps.local` al servizio virtuale `serviceb.apps.local` tramite un router virtuale.
- Tre nodi virtuali denominati `serviceA`, `serviceB` e `serviceBv2`. Il proxy Envoy utilizza le informazioni di individuazione dei servizi configurate per i nodi virtuali per cercare gli indirizzi IP dei servizi effettivi.
- Un router virtuale con una route che indica al proxy Envoy di indirizzare il 75% del traffico in ingresso al nodo virtuale `serviceB` e il 25% del traffico al nodo virtuale `serviceBv2`.

Fase 6: aggiornare i servizi

Dopo aver creato la mesh, è necessario completare le seguenti attività:

- Autorizza il proxy Envoy che distribuisce con ogni ECS attività Amazon a leggere la configurazione di uno o più nodi virtuali. Per ulteriori informazioni su come autorizzare il proxy, consulta [Autorizzazione del proxy](#).
- Aggiorna ciascuna delle definizioni di ECS attività Amazon esistenti per utilizzare il proxy Envoy.

Credenziali

Il contenitore Envoy richiede AWS Identity and Access Management credenziali per la firma delle richieste inviate al servizio App Mesh. Per ECS le attività Amazon distribuite con il tipo di EC2 avvio Amazon, le credenziali possono provenire dal ruolo dell'[istanza o da un ruolo](#) dell'[attività IAM](#). ECSLe

attività Amazon distribuite con Fargate su contenitori Linux non hanno accesso al server di metadati EC2 Amazon che fornisce IAM le credenziali del profilo dell'istanza. Per fornire le credenziali, è necessario assegnare un ruolo di IAM attività a qualsiasi attività distribuita con il tipo di container Fargate su Linux.

Se un'attività viene distribuita con il tipo di EC2 avvio Amazon e l'accesso è bloccato al server di EC2 metadati Amazon, come descritto nell'annotazione Importante in [IAMRuolo per attività](#), all'attività deve essere associato anche un IAM ruolo di attività. [Al ruolo assegnato all'istanza o all'attività deve essere associata una IAM policy come descritto in Autorizzazione proxy.](#)

Per aggiornare la definizione dell'attività utilizzando il AWS CLI

Usi il ECS AWS CLI comando Amazon [register-task-definition](#). La definizione di attività di esempio riportata di seguito mostra come configurare App Mesh per il servizio.

Note

La configurazione di App Mesh for Amazon ECS tramite la console non è disponibile.

Definizione dell'attività (json)

Configurazione del proxy

Per configurare il tuo ECS servizio Amazon per l'utilizzo di App Mesh, la definizione delle attività del servizio deve avere la seguente sezione di configurazione del proxy. Imposta la configurazione del proxy type su APPMESH e il nome containerName su envoy. Imposta di conseguenza i valori di proprietà indicati di seguito.

IgnoredUID

Il proxy Envoy non instrada il traffico dai processi che utilizzano questo ID utente. Per questa opzione puoi scegliere qualsiasi ID utente, ma l'ID deve essere lo stesso indicato come user ID per il container Envoy nella definizione dell'attività. Questa corrispondenza consente a Envoy di ignorare il proprio traffico senza utilizzare il proxy. I nostri esempi utilizzano **1337** per scopi storici.

ProxyIngressPort

Questa è la porta in entrata per il contenitore proxy Envoy. Imposta questo valore su **15000**.

ProxyEgressPort

Questa è la porta in uscita per il contenitore proxy Envoy. Imposta questo valore su 15001.

AppPorts

Specificate le porte in entrata su cui i contenitori delle applicazioni sono in ascolto. In questo esempio, il container dell'applicazione resta in ascolto sulla porta **9080**. La porta specificata deve corrispondere alla porta configurata nel listener del nodo virtuale.

EgressIgnoredIPs

Envoy non invia traffico proxy a questi indirizzi IP. Imposta questo valore su 169.254.170.2, 169.254.169.254, che ignora il server di EC2 metadati Amazon e l'endpoint di metadati Amazon ECS task. L'endpoint di metadati fornisce ruoli per le credenziali delle attività. IAM È possibile aggiungere ulteriori indirizzi.

EgressIgnoredPorts

Puoi aggiungere un elenco di porte separate da virgole. Envoy non invia traffico proxy a queste porte. Anche se non elenchi delle porte, la porta 22 viene ignorata.

Note

Il numero massimo di porte in uscita che possono essere ignorate è 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"
}
]
}
```

Dipendenza tra Envoy e container dell'applicazione

Prima di potersi avviare, i container delle applicazioni nelle definizioni dell'attività devono attendere il bootstrap e l'avvio del proxy Envoy. Per fare in modo che ciò accada, impostate una `dependsOn` sezione in ogni definizione del contenitore dell'applicazione per attendere che il contenitore Envoy riporti come `HEALTHY`. Il codice seguente mostra un esempio di definizione del container dell'applicazione con questa dipendenza. Tutte le proprietà nell'esempio seguente sono obbligatorie. Anche alcuni valori delle proprietà sono obbligatori, ma altri lo sono *replaceable*.

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

Definizione del container Envoy

Le definizioni delle ECS attività Amazon devono contenere un'immagine del contenitore App Mesh Envoy.

Tutte le regioni [supportate](#) diverse `me-south-1`, `ap-east-1`, `ap-southeast-3`, `eu-south-1`, `il-central-1`, e `af-south-1`. È possibile sostituire *Region-code* con qualsiasi regione diversa da `me-south-1`, `ap-east-1`, `ap-southeast-3`, `eu-south-1`, `il-central-1`, e `af-south-1`.

Standard

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

FIPS-conforme

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

me-south-1

Standard

```
772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

FIPS-conforme

```
772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

ap-east-1

Standard

```
856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

FIPS-conforme

```
856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

ap-southeast-3

Standard

```
909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

FIPS-conforme

```
909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

eu-south-1

Standard

```
422531588944.dkr.ecr.eu-south-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

FIPS-conforme

```
422531588944.dkr.ecr.eu-south-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

il-central-1

Standard

```
564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

FIPS-conforme

```
564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

af-south-1

Standard

```
924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

FIPS-conforme

```
924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

Public repository

Standard

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.29.6.1-prod
```

FIPS-conforme

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

Important

Solo la versione v1.9.0.0-prod o successiva è supportata per l'uso con App Mesh.

È necessario utilizzare l'immagine del contenitore App Mesh Envoy finché il team del progetto Envoy non unisce le modifiche che supportano App Mesh. [Per ulteriori dettagli, consulta il problema relativo alla roadmap. GitHub](#)

Tutte le proprietà nell'esempio seguente sono obbligatorie. Anche alcuni valori delle proprietà sono obbligatori, ma altri lo sono *replaceable*.

Note

- La definizione del container Envoy deve essere contrassegnata come `essential`.
- Si consiglia di allocare 512 CPU unità e almeno 64 MiB di memoria nel contenitore Envoy. Su Fargate il valore minimo che potrai impostare è 1024 MiB di memoria.
- Il nome del nodo virtuale per il ECS servizio Amazon deve essere impostato sul valore della `APPMESH_RESOURCE_ARN` proprietà. Questa proprietà richiede una versione `1.15.0` o successiva dell'immagine Envoy. Per ulteriori informazioni, consulta [Envoy](#).
- Il valore dell'impostazione `user` deve corrispondere al valore `IgnoredUID` della configurazione del proxy di definizione dell'attività. In questo esempio viene utilizzato `1337`.
- Il controllo dello stato mostrato qui attende che il contenitore Envoy si avvii correttamente prima di segnalare ad Amazon ECS che il contenitore Envoy è integro e pronto per l'avvio dei contenitori dell'applicazione.
- Per impostazione predefinita, App Mesh utilizza il nome della risorsa specificata in `APPMESH_RESOURCE_ARN` quando Envoy si riferisce a se stesso nei parametri e nelle tracce. È possibile ignorare questo comportamento impostando la variabile di ambiente `APPMESH_RESOURCE_CLUSTER` con il proprio nome. Questa proprietà richiede una

versione 1.15.0 o successiva dell'immagine Envoy. Per ulteriori informazioni, consulta [Envoy](#).

Il seguente codice mostra un esempio di definizione del container Envoy.

```
{
  "name": "envoy",
  "image": "840364872350.dkr.ecr.us-west-2.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-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"
}
```

Esempi di definizioni di attività

Il seguente esempio di definizioni delle ECS attività di Amazon mostra come unire gli esempi precedenti in una definizione di attività per taskB. Vengono forniti esempi per la creazione di attività per entrambi i tipi di ECS avvio di Amazon con o senza utilizzo AWS X-Ray. Cambia il *replaceable* valori, a seconda dei casi, per creare definizioni di attività per le attività denominate taskBv2 e taskA contenute nello scenario. Sostituisci con il nome della tua mesh e del tuo nodo virtuale il valore APPMESH_RESOURCE_ARN e poi il valore AppPorts con un elenco di porte su cui la tua applicazione resta in ascolto. Per impostazione predefinita, App Mesh utilizza il nome della risorsa specificata in APPMESH_RESOURCE_ARN quando Envoy si riferisce a se stesso nei parametri e nelle tracce. È possibile ignorare questo comportamento impostando la variabile di ambiente APPMESH_RESOURCE_CLUSTER con il proprio nome. Tutte le proprietà degli esempi seguenti sono obbligatorie. Anche alcuni valori delle proprietà sono obbligatori, ma altri lo sono *replaceable*.

Se stai eseguendo un'attività Amazon come descritto nella sezione [Credenziali](#), devi aggiungere un [IAM ruolo di attività](#) esistente, agli esempi.

⚠ Important

Fargate deve utilizzare un valore di porta maggiore di 1024.

Example JSON per la definizione ECS delle attività di Amazon - Fargate su contenitori 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:v1.29.6.1-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 JSON per la definizione ECS delle attività di Amazon con AWS X-Ray - Fargate su contenitori Linux

X-Ray consente di raccogliere dati sulle richieste servite da un'applicazione e fornisce strumenti che è possibile utilizzare per visualizzare il flusso di traffico. L'utilizzo del driver X-Ray per Envoy consente a Envoy di riportare le informazioni di tracciamento a X-Ray. [È possibile abilitare il tracciamento X-Ray utilizzando la configurazione Envoy.](#) In base alla configurazione, Envoy invia i dati di tracciamento al demone X-Ray che funziona come contenitore [sidecar](#) e il daemon inoltra le tracce al servizio X-Ray. Una volta pubblicate le tracce su X-Ray, è possibile utilizzare la console X-Ray per visualizzare il grafico delle chiamate di assistenza e richiedere i dettagli di tracciamento. Quanto segue JSON rappresenta una definizione di attività per abilitare l'integrazione con 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:v1.29.6.1-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 per la definizione ECS delle attività di Amazon: tipo di EC2 avvio

```

{
  "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:v1.29.6.1-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": "awsipc"
}

```

Example JSON per la definizione di ECS attività su Amazon con AWS X-Ray - tipo di EC2 avvio

```

{
  "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:v1.29.6.1-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"
}
```

Argomenti avanzati

Implementazioni Canary utilizzando App Mesh

Le implementazioni e le versioni di Canary ti aiutano a spostare il traffico tra una vecchia versione di un'applicazione e una nuova versione distribuita. Monitora anche lo stato della versione appena distribuita. In caso di problemi con la nuova versione, Canary Deployment può riportare automaticamente il traffico alla versione precedente. Le implementazioni di Canary ti danno la possibilità di spostare il traffico tra le versioni dell'applicazione con un maggiore controllo.

Per ulteriori informazioni su come implementare le distribuzioni Canary per Amazon utilizzando App ECS Mesh, consulta [Creare una pipeline con distribuzioni Canary](#) per Amazon utilizzando App Mesh ECS

Note

Per altri esempi e procedure dettagliate per App Mesh, consulta l'archivio degli esempi di [App Mesh](#).

Guida introduttiva a AWS App Mesh Kubernetes

Quando esegui l'integrazione AWS App Mesh con Kubernetes utilizzando il controller App Mesh per Kubernetes, gestisci le risorse App Mesh, come mesh, servizi virtuali, nodi virtuali, router virtuali e percorsi tramite Kubernetes. Inoltre, aggiungi automaticamente le immagini del contenitore del sidecar App Mesh alle specifiche del pod Kubernetes. Questo tutorial ti guida attraverso l'installazione del controller App Mesh per Kubernetes per abilitare questa integrazione.

Il controller è accompagnato dalla distribuzione delle seguenti definizioni di risorse personalizzate Kubernetes: `meshes`, `virtual services`, `virtual nodes` e `virtual routers`. Il controller controlla la creazione, la modifica e l'eliminazione delle risorse personalizzate e apporta modifiche alle risorse App Mesh [the section called “Mesh”](#), [the section called “Servizi virtuali”](#), [the section called](#)

[“Nodi virtuali”, the section called “Gateway virtuali virtuale virtuale virtuale virtuale virtuale”, the section called “route del gateway”, the section called “Router virtuali” \(inclusethe section called “Route”\)](#) corrispondenti tramite App MeshAPI. Per saperne di più o contribuire al controller, consulta il [GitHubprogetto](#).

Il controller installa anche un webhook che inserisce i seguenti container nei pod Kubernetes etichettandoli con il nome che specifichi.

- Proxy App Mesh Envoy: Envoy utilizza la configurazione definita nel piano di controllo App Mesh per determinare dove inviare il traffico dell'applicazione.
- App Mesh proxy route manager: aggiorna `iptables` le regole nello spazio dei nomi di rete di un pod che instradano il traffico in entrata e in uscita tramite Envoy. Questo container viene eseguito come container init Kubernetes all'interno del pod.

Prerequisiti

- Una comprensione esistente dei concetti di App Mesh. Per ulteriori informazioni, consulta [Cosa è AWS App Mesh?](#).
- Un'adeguata conoscenza dei concetti di Kubernetes Per ulteriori informazioni, consulta [Cos'è Kubernetes nella documentazione di Kubernetes](#).
- Un cluster Kubernetes esistente. Se non disponi di un cluster esistente, consulta la sezione [Getting Started with Amazon EKS](#) nella Amazon EKS User Guide. Se stai eseguendo il tuo cluster Kubernetes su AmazonEC2, assicurati che Docker sia autenticato nel repository ECR Amazon in cui si trova l'immagine Envoy. Per ulteriori informazioni, consulta [Envoy image](#), [Registry authentication](#) in Amazon Elastic Container Registry User Guide e [Pull an Image from a Private Registry](#) nella documentazione di Kubernetes.
- App Mesh supporta i servizi Linux registrati con DNS o entrambi. AWS Cloud Map Per utilizzare questa guida introduttiva, ti consigliamo di avere tre servizi esistenti registrati pressoDNS. Le procedure descritte in questo argomento presuppongono che i servizi esistenti abbiano un nome `serviceA` `serviceBv2` e che tutti i servizi siano individuabili tramite uno spazio dei nomi denominato `serviceB apps.local`

Puoi creare una mesh dei servizi e le relative risorse anche se i servizi non esistono, ma non puoi utilizzare la mesh fino a quando non hai distribuito i servizi effettivi.

- È installata la AWS CLI versione 1.18.116 o successiva o 2.0.38 o successiva. [Per installare o aggiornare il AWS CLI, vedere Installazione di. AWS CLI](#)

- Un client `kubectl` configurato per comunicare con il cluster Kubernetes. Se utilizzi Amazon Elastic Kubernetes Service, puoi utilizzare le istruzioni per [kubectl](#) l'installazione e la configurazione di un file. [kubeconfig](#)
- Helm versione 3.0 o successiva installato. Se non hai installato Helm, consulta [Using Helm with Amazon EKS nella Amazon EKS User Guide](#).
- EKSAI momento Amazon supporta IPv6_ONLY solo IPv4_ONLY e solo le preferenze IP, poiché EKS attualmente Amazon supporta solo pod in grado di servire solo IPv4 traffico o solo IPv6 traffico.

Le fasi rimanenti presuppongono che i servizi effettivi siano denominati `serviceA`, `serviceB` e `serviceBv2` e che tutti i servizi siano individuabili tramite uno spazio dei nomi denominato `apps.local`.

Fase 1: installazione dei componenti di integrazione

Installa i componenti di integrazione una volta in ogni cluster che ospita i pod che desideri utilizzare con App Mesh.

Per installare i componenti di integrazione

1. Le fasi rimanenti di questa procedura richiedono un cluster senza una versione non definitiva del controller installata. Se hai installato una versione non definitiva o non sei sicuro di averla installata, puoi scaricare ed eseguire uno script che verifica se nel tuo cluster è installata una versione non definitiva.

```
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
```

Se lo script restituisce `Your cluster is ready for upgrade. Please proceed to the installation instructions`, puoi procedere alla fase successiva. Se viene restituito un messaggio diverso, è necessario completare la procedura di aggiornamento prima di continuare. [Per ulteriori informazioni sull'aggiornamento di una versione non definitiva, consulta Esegui l'aggiornamento su GitHub](#)

2. Aggiungi il repository `eks-charts` a Helm.

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

3. Installa le definizioni delle risorse personalizzate di App Mesh Kubernetes (). CRD

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

4. Crea uno spazio dei nomi Kubernetes per il controller.

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

5. Imposta le seguenti variabili per utilizzarle nelle prossime fasi. Sostituisci *cluster-name* e *Region-code* con i valori per il cluster esistente.

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

6. (Facoltativo) Se vuoi eseguire il controller su Fargate devi creare un profilo Fargate. Se non lo hai `eksctl` installato, consulta [Installazione o aggiornamento eksctl](#) nella Amazon EKS User Guide. Se preferisci creare il profilo utilizzando la console, consulta la sezione [Creazione di un profilo Fargate](#) nella Amazon EKS User Guide.

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

7. Crea un provider di identità OpenID Connect (OIDC) per il tuo cluster. Se non lo hai `eksctl` installato, puoi installarlo seguendo le istruzioni in [Installazione o aggiornamento eksctl](#) nella Amazon EKS User Guide. Se preferisci creare il provider utilizzando la console, consulta la sezione [Abilitazione IAM dei ruoli per gli account di servizio sul tuo cluster](#) nella Amazon EKS User Guide.

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

8. Crea un IAM ruolo, associa le politiche [AWSCloudMapFullAccess](#) AWS gestite ad esso e associalo all'account del servizio `appmesh-controller` Kubernetes. [AWSAppMeshFullAccess](#) Il ruolo consente al controller di aggiungere, rimuovere e modificare risorse App Mesh.

Note

Il comando crea un AWS IAM ruolo con un nome generato automaticamente. Non è possibile specificare il nome del IAM ruolo creato.

```
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
```

Se preferisci creare l'account di servizio utilizzando AWS Management Console o AWS CLI, consulta la sezione [Creazione di un IAM ruolo e di una politica per il tuo account di servizio](#) nella Amazon EKS User Guide. Se utilizzi AWS Management Console o AWS CLI per creare l'account, devi anche mappare il ruolo a un account di servizio Kubernetes. Per ulteriori informazioni, consulta [Specificare un IAM ruolo per il tuo account di servizio](#) nella Amazon EKS User Guide.

9. Implementa il controller App Mesh. Per un elenco di tutte le opzioni di configurazione, vedi [Configuration](#) on GitHub.

1. Per distribuire il controller App Mesh per un cluster privato, devi prima abilitare App Mesh e gli VPC endpoint Amazon di rilevamento dei servizi nella sottorete privata collegata. È inoltre necessario impostare il `accountId`

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

Per abilitare il tracciamento a raggi X in un cluster privato, abilita gli endpoint X-Ray e Amazon Amazon. ECR VPC Il controller lo utilizza `public.ecr.aws/xray/aws-xray-daemon:latest` per impostazione predefinita, quindi trasferisci questa immagine in locale e [inseriscila nel tuo archivio personale](#). ECR

Note

[VPCGli endpoint Amazon](#) attualmente non supportano gli archivi ECR pubblici di Amazon.

L'esempio seguente mostra l'implementazione del controller con configurazioni per 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
```

Verifica se il demone X-Ray viene iniettato correttamente durante l'associazione della distribuzione dell'applicazione con il nodo o il gateway virtuale.

Per ulteriori informazioni, consulta [Private Clusters](#) nella Amazon EKS User Guide.

2. Implementa il controller App Mesh per altri cluster. Per un elenco di tutte le opzioni di configurazione, vedi [Configuration](#) on. 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

Se la tua famiglia di EKS cluster Amazon lo è IPv6, imposta il nome del cluster durante la distribuzione del controller App Mesh aggiungendo la seguente opzione al comando `--set clusterName=$CLUSTER_NAME` precedente.

Important

Se il tuo cluster si trova nelle,,, o `me-south-1` `af-south-1` Regioni `ap-east-1` `ap-southeast-3` `eu-south-1` `il-central-1`, devi aggiungere la seguente opzione al comando precedente:

Replace (Sostituisci) *account-id* e *Region-code* con uno dei set di valori appropriati.

- Per l'immagine del sidecar:

- ```
--set image.repository=account-id.dkr.ecr.Region-code.amazonaws.com/
amazon/appmesh-controller
```
- `772975370895.dkr.ecr.me-south-1.amazonaws.com /:v1.29.6.1-prod aws-appmesh-envoy`
- `856666278305.dkr.ecr.ap-east-1.amazonaws.com /:v1.29.6.1-prod aws-appmesh-envoy`
- `909464085924.dkr.ecr.ap-southeast-3.amazonaws.com /:v1.29.6.1-prod aws-appmesh-envoy`
- `422531588944.dkr.ecr.eu-south-1.amazonaws.com /:v1.29.6.1-prod aws-appmesh-envoy`
- `564877687649.dkr.ecr.il-central-1.amazonaws.com /:v1.29.6.1-prod aws-appmesh-envoy`
- `924023996002.dkr.ecr.af-south-1.amazonaws.com /:v1.29.6.1-prod aws-appmesh-envoy`
- [L'immagine precedente è disponibile URIs nel registro delle modifiche.](#) GitHub La versione AWS degli account su cui sono presenti le immagini è `aws-1.5.0`. Le versioni precedenti delle immagini sono ospitate su AWS account presenti nei registri di immagini dei container Amazon Elastic Kubernetes [Service](#) di Amazon.

- Per l'immagine del controller:

- ```
--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

- Per l'immagine init 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/-manager:v7-prod aws-appmesh-proxy-route
- 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/-manager:v7-prod aws-appmesh-proxy-route
- 564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-proxy-route-manager:v7-prod
- 924023996002.dkr.ecr.af-south-1.amazonaws.com/-manager:v7-prod aws-appmesh-proxy-route

**⚠ Important**

Solo la versione v1.9.0.0-prod o successiva è supportata per l'uso con App Mesh.

10. Verifica che la versione del controller sia v1.4.0 o successiva. [È possibile rivedere il registro delle modifiche.](#) GitHub

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

**ℹ Note**

Se si visualizza il log per il container in esecuzione, è possibile che venga visualizzata una riga che include il testo seguente, che può essere tranquillamente ignorato.

```
Neither -kubeconfig nor -master was specified. Using the inClusterConfig.
This might not work.
```

## Fase 2: Implementazione delle risorse App Mesh

Quando distribuisce un'applicazione in Kubernetes, crei anche le risorse personalizzate Kubernetes in modo che il controller possa creare le risorse App Mesh corrispondenti. La procedura seguente consente di distribuire le risorse App Mesh con alcune delle relative funzionalità. Puoi trovare esempi di manifest per la distribuzione di altre funzionalità di risorse App Mesh nelle v1beta2 sottocartelle di molte delle cartelle di funzionalità elencate nelle procedure dettagliate di [App Mesh](#) su GitHub

**⚠ Important**

Dopo che il controller ha creato una risorsa App Mesh, ti consigliamo di apportare modifiche o eliminare la risorsa App Mesh solo utilizzando il controller. Se apporti modifiche o elimini la risorsa utilizzando App Mesh, il controller non cambierà o ricreerà la risorsa App Mesh modificata o eliminata per dieci ore, per impostazione predefinita. Puoi configurare un periodo inferiore. Per ulteriori informazioni, consulta [Configurazione](#) su GitHub

## Per distribuire risorse App Mesh

1. Crea uno spazio dei nomi Kubernetes in cui distribuire le risorse App Mesh.
  - a. Salva nel tuo computer i seguenti contenuti in un file denominato `namespace.yaml`.

```
apiVersion: v1
kind: Namespace
metadata:
 name: my-apps
 labels:
 mesh: my-mesh
 appmesh.k8s.aws/sidecarInjectorWebhook: enabled
```

- b. Crea lo spazio dei nomi.

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

2. Crea una mesh di servizi App Mesh.
  - a. Salva nel tuo computer i seguenti contenuti in un file denominato `mesh.yaml`. Il file viene utilizzato per creare una risorsa mesh denominata *my-mesh*. Una mesh dei servizi è un limite logico per il traffico di rete tra i servizi che si trovano al suo interno.

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

- b. Crea la mesh.

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

- c. Visualizza i dettagli della risorsa mesh Kubernetes creata.

```
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-j11k11111m711
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. Visualizza i dettagli sulla mesh del servizio App Mesh creata dal controller.

```
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-j11k11111m711",
 "version": 1
 },
 "spec": {},
 "status": {
 "status": "ACTIVE"
 }
}
}

```

3. Crea un nodo virtuale App Mesh. Un nodo virtuale funge da puntatore logico a una distribuzione Kubernetes.
  - a. Salva nel tuo computer i seguenti contenuti in un file denominato `virtual-node.yaml`. Il file viene utilizzato per creare un nodo virtuale App Mesh denominato *my-service-a* in *le mie app* spazio dei nomi. Il nodo virtuale rappresenta un servizio Kubernetes creato in una fase successiva. Il valore di `hostname` è il DNS nome host completo del servizio effettivo rappresentato da questo nodo virtuale.

```

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

```

I nodi virtuali hanno funzionalità, come la end-to-end crittografia e i controlli dello stato, che non sono trattate in questo tutorial. Per ulteriori informazioni, consulta [the section called “Nodi virtuali”](#). Per visualizzare tutte le impostazioni disponibili per un nodo virtuale che puoi impostare nelle specifiche precedenti, esegui il comando seguente.

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

- b. Distribuisci il nodo virtuale.

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

- c. Visualizza i dettagli della risorsa nodo virtuale Kubernetes creata.

```
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-j11k11111m711
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-j11k11111m711
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. Visualizza i dettagli del nodo virtuale creato dal controller in App Mesh.

#### Note

Anche se il nome del nodo virtuale creato in Kubernetes è *my-service-a*, il nome del nodo virtuale creato in App Mesh è *my-service-a\_my-apps*. Il controller aggiunge il nome dello spazio dei nomi Kubernetes al nome del nodo virtuale App Mesh quando crea la risorsa App Mesh. Il nome dello spazio dei nomi viene aggiunto perché in Kubernetes è possibile creare nodi virtuali con lo stesso nome in diversi namespace, ma in App Mesh un nome di nodo virtuale deve essere univoco all'interno di una mesh.

```
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-j11k11111m711",
 "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. Crea un router virtuale App Mesh. I router virtuali gestiscono il traffico per uno o più servizi virtuali all'interno della mesh.
  - a. Salva nel tuo computer i seguenti contenuti in un file denominato `virtual-router.yaml`. Il file viene utilizzato per creare un router virtuale per indirizzare il traffico verso il nodo virtuale denominato `my-service-a` creato nel passaggio precedente. Il controller crea il router virtuale App Mesh e le risorse di routing. Puoi specificare molte più funzionalità per i percorsi e utilizzare protocolli diversi da `http`. Per ulteriori informazioni, consulta [the section called "Router virtuali"](#) e [the section called "Route"](#). Nota che il nome del nodo virtuale a cui

si fa riferimento è il nome del nodo virtuale Kubernetes, non il nome del nodo virtuale App Mesh creato in App Mesh dal controller.

```
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
```

(Facoltativo) Per visualizzare tutte le impostazioni disponibili per un router virtuale che puoi configurare nelle specifiche precedenti, esegui il comando seguente.

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

Per visualizzare tutte le impostazioni disponibili per un percorso che è possibile impostare nelle specifiche precedenti, esegui il comando seguente.

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

b. Distribuisci il router virtuale.

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

c. Visualizza la risorsa router virtuale Kubernetes creata.

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

## Output abbreviato

```
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-j11k11111m711
 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. Visualizza la risorsa router virtuale creata dal controller in App Mesh. Hai specificato `my-service-a-virtual-router_my-apps` forname, perché quando il controller ha creato il router virtuale in App Mesh, ha aggiunto il nome dello spazio dei nomi Kubernetes al nome del router virtuale.

```
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-j11k11111m711",
 "version": 1
 },
 "spec": {
 "listeners": [
 {
 "portMapping": {
 "port": 80,
 "protocol": "http"
 }
 }
]
 },
 "status": {
 "status": "ACTIVE"
 },
 "virtualRouterName": "my-service-a-virtual-router_my-apps"
 }
}
```

```
}

```

- e. Visualizza la risorsa di percorso creata dal controller in App Mesh. La risorsa route non è stata creata in Kubernetes perché fa parte della configurazione del router virtuale in Kubernetes. Le informazioni sul percorso sono state mostrate nei dettagli della risorsa Kubernetes nella sottofase c. Il controller non ha aggiunto il nome dello spazio dei nomi Kubernetes al nome della route App Mesh quando ha creato la route in App Mesh perché i nomi delle route sono unici per un router virtuale.

```
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-j11k11111m711",
 "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. Crea un servizio virtuale App Mesh. Un servizio virtuale è un'astrazione di un servizio reale fornita direttamente o indirettamente da un nodo virtuale mediante un router virtuale. I servizi dipendenti chiamano il servizio virtuale con il suo nome. Sebbene il nome non sia importante per App Mesh, consigliamo di denominare il servizio virtuale con il nome di dominio completo del servizio effettivo rappresentato dal servizio virtuale. Assegnando un nome al servizio virtuale in questo modo, non hai la necessità di modificare il codice dell'applicazione per fare riferimento a un nome diverso. Le richieste vengono instradate al nodo virtuale o al router virtuale specificato come provider per il servizio virtuale.
  - a. Salva nel tuo computer i seguenti contenuti in un file denominato `virtual-service.yaml`. Il file viene utilizzato per creare un servizio virtuale che utilizza un provider di router virtuale per indirizzare il traffico verso il nodo virtuale denominato `my-service-a` creato in un passaggio precedente. Il valore di `awsName` in `spec` è il nome di dominio completo (FQDN) del servizio Kubernetes effettivo che questo servizio virtuale astrae. Il servizio Kubernetes viene creato in [the section called “Fase 3: creazione o aggiornamento dei servizi”](#). Per ulteriori informazioni, consulta [the section called “Servizi virtuali”](#).

```

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

```

Per visualizzare tutte le impostazioni disponibili per un servizio virtuale che puoi impostare nelle specifiche precedenti, esegui il comando seguente.

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

- b. Creare il servizio virtuale.

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

- c. Visualizzare i dettagli della risorsa servizio virtuale Kubernetes creata.

```
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-j11k11111m711
Spec:
 Aws Name: my-service-a.my-apps.svc.cluster.local
 Mesh Ref:
 Name: my-mesh
 UID: 111a11b1-c11d-1e1f-gh1i-j11k11111m711
 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. Visualizza i dettagli della risorsa del servizio virtuale creata dal controller in App Mesh. Il controller Kubernetes non ha aggiunto il nome dello spazio dei nomi Kubernetes al nome del servizio virtuale App Mesh quando ha creato il servizio virtuale in App Mesh perché il nome del servizio virtuale è univoco. FQDN

```

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-j11k11111m711",
 "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"
 }
 }
}
```

Sebbene non sia trattato in questo tutorial, il controller può anche implementare App Mesh [the section called “Gateway virtuali virtuale virtuale virtuale virtuale virtuale”](#) e [the section called “route del gateway”](#). Per una procedura dettagliata sulla distribuzione di queste risorse con il controller, consulta [Configuring Inbound Gateway](#) o un [manifesto di esempio](#) che include le risorse su. GitHub

### Fase 3: creazione o aggiornamento dei servizi

A tutti i pod che desideri utilizzare con App Mesh devono essere aggiunti i contenitori collaterali App Mesh. L'iniettore aggiunge automaticamente i container sidecar a qualsiasi pod distribuito con l'etichetta che specifichi.

1. Abilitare l'autorizzazione proxy. Ti consigliamo di abilitare ogni distribuzione Kubernetes per lo streaming solo della configurazione per il proprio nodo virtuale App Mesh.
  - a. Salva nel tuo computer i seguenti contenuti in un file denominato `proxy-auth.json`. Assicurati di sostituire il *alternate-colored values* con il tuo.

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

- b. Crea la policy.

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

- c. Crea un IAM ruolo, allega la policy creata nel passaggio precedente, crea un account di servizio Kubernetes e associa la policy all'account del servizio Kubernetes. Il ruolo consente al controller di aggiungere, rimuovere e modificare risorse 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
```

Se preferisci creare l'account di servizio utilizzando AWS Management Console o AWS CLI, consulta la sezione [Creazione di un IAM ruolo e policy per il tuo account di servizio](#) nella Amazon EKS User Guide. Se utilizzi AWS Management Console o AWS CLI per creare l'account, devi anche mappare il ruolo a un account di servizio Kubernetes. Per ulteriori informazioni, consulta [Specificare un IAM ruolo per il tuo account di servizio](#) nella Amazon EKS User Guide.

2. (Facoltativo) Se vuoi eseguire la distribuzione nei pod Fargate devi creare un profilo Fargate. Se non lo hai `eksctl` installato, puoi installarlo seguendo le istruzioni in [Installazione o aggiornamento eksctl](#) nella Amazon EKS User Guide. Se preferisci creare il profilo utilizzando la console, consulta la sezione [Creazione di un profilo Fargate](#) nella Amazon EKS User Guide.

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

3. Creare un servizio e una distribuzione Kubernetes. Se disponi di una distribuzione esistente che desideri utilizzare con App Mesh, devi implementare un nodo virtuale, come hai fatto nella fase secondaria 3 di [the section called “Fase 2: Implementazione delle risorse App Mesh”](#) Aggiorna la distribuzione per assicurarti che la relativa etichetta corrisponda all'etichetta che hai impostato sul nodo virtuale, in modo che i contenitori sidecar vengano aggiunti automaticamente ai pod e i pod vengano ridistribuiti.
  - a. Salva nel tuo computer i seguenti contenuti in un file denominato `example-service.yaml`. Se cambi il nome dello spazio dei nomi e usi i pod Fargate, il nome

dello spazio dei nomi deve corrispondere al nome dello spazio dei nomi definito nel profilo 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**

Il valore per il `app matchLabels selector` nella specifica deve corrispondere al valore che hai specificato quando hai creato il nodo virtuale nella sottofase 3 di [the section called “Fase 2: Implementazione delle risorse App Mesh”](#). In caso contrario i container sidecar non verranno inseriti nel container. Nell'esempio precedente, il valore dell'etichetta è `my-app-1`. Se distribuisce un gateway virtuale, anziché un nodo virtuale, il Deployment manifest deve includere solo il contenitore Envoy. Per ulteriori informazioni sull'immagine da utilizzare, consulta [Envoy](#). Per un esempio di manifesto, vedi [l'esempio di implementazione su GitHub](#).

- b. Distribuire il servizio.

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

- c. Visualizzare il servizio e la distribuzione.

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

**Output**

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

- d. Visualizzare i dettagli di uno dei pod che è stato distribuito.

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

**Output abbreviato**

```
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

```

Nell'output precedente, puoi vedere che i container `proxyinit` e `envoy` sono stati aggiunti al pod dal container. Se hai distribuito il servizio di esempio su Fargate, `envoy` il contenitore è stato aggiunto al pod dal controller, ma `proxyinit` il contenitore no.

4. (Facoltativo) Installa componenti aggiuntivi come Prometheus, Grafana, Jaeger e Datadog. AWS X-Ray Per ulteriori informazioni, consulta [i componenti aggiuntivi di App Mesh](#) GitHub e la sezione [Osservabilità](#) della App Mesh User Guide.

**Note**

Per altri esempi e procedure dettagliate per App Mesh, consulta l'archivio degli esempi di [App Mesh](#).

## Fase 4: pulizia

Rimuovere tutte le risorse di esempio create in questo tutorial. Il controller rimuove anche le risorse create nella mesh del servizio `my-mesh` App Mesh.

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

Se hai creato un profilo Fargate per il servizio di esempio, rimuovilo.

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

Elimina la mesh.

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

(Facoltativo) È possibile rimuovere i componenti di integrazione Kubernetes.

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

(Facoltativo) Se hai distribuito i componenti di integrazione Kubernetes su Fargate, elimina il profilo Fargate.

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

## Guida introduttiva AWS App Mesh ad Amazon EC2

Questo argomento ti aiuta a utilizzare AWS App Mesh con un servizio effettivo in esecuzione su AmazonEC2. Questo tutorial copre le funzionalità di base di diversi tipi di risorse App Mesh.

## Scenario

Per illustrare come utilizzare App Mesh, supponiamo di avere un'applicazione con le seguenti caratteristiche:

- È costituito da due servizi denominati `serviceA` e `serviceB`.
- Entrambi i servizi sono registrati in uno spazio dei nomi denominato `apps.local`.
- `ServiceA` comunica con `serviceB` oltre HTTP /2, porta 80.
- Hai già distribuito la versione 2 di `serviceB` e l'hai registrata con il nome `serviceBv2` nello spazio dei nomi `apps.local`.

I requisiti sono i seguenti:

- Vuoi inviare il 75 per cento del traffico da `serviceA` a `serviceB` e il 25 per cento del traffico a `serviceBv2` First. Inviando solo il 25 per cento a `serviceBv2`, puoi verificare che sia privo di bug prima di inviare il 100% del traffico da `serviceA`.
- Vuoi essere in grado di regolare facilmente la ponderazione del traffico in modo che il 100% del traffico vada a `serviceBv2` una volta dimostrato essere affidabile. Una volta inviato tutto il traffico a `serviceBv2`, vuoi `serviceB` interromperlo.
- Per soddisfare i requisiti precedenti non è necessario modificare il codice dell'applicazione esistente o la registrazione di Service Discovery per usufruire dei servizi effettivi.

Per soddisfare le tue esigenze, decidi di creare una mesh di servizi App Mesh con servizi virtuali, nodi virtuali, un router virtuale e un percorso. Dopo aver implementato la mesh, aggiorni i servizi per utilizzare il proxy Envoy. Una volta aggiornati, i servizi comunicano tra loro tramite il proxy Envoy anziché direttamente tra loro.

## Prerequisiti

App Mesh supporta i servizi Linux registrati con DNS o entrambi. AWS Cloud Map Per utilizzare questa guida introduttiva, ti consigliamo di avere tre servizi esistenti registrati presso DNS. Puoi creare una mesh dei servizi e le relative risorse anche se i servizi non esistono, ma non puoi utilizzare la mesh fino a quando non hai distribuito i servizi effettivi.

Se non disponi già di servizi in esecuzione, puoi avviare EC2 istanze Amazon e distribuirvi applicazioni. Per ulteriori informazioni, consulta [Tutorial: Guida introduttiva alle istanze Amazon EC2 Linux](#) nella Amazon EC2 User Guide. Le fasi rimanenti presuppongono che i servizi effettivi siano

denominati `serviceA`, `serviceB` e `serviceBv2` e che tutti i servizi siano individuabili tramite uno spazio dei nomi denominato `apps.local`.

## Fase 1: creare una mesh e un servizio virtuale

Una mesh dei servizi è un limite logico per il traffico di rete tra i servizi che si trovano al suo interno. Per ulteriori informazioni, consulta [Resh di servizio](#). Un servizio virtuale è un'astrazione di un servizio effettivo. Per ulteriori informazioni, consulta [Servizi virtuali](#).

Crea le seguenti risorse:

- Una mesh denominata `apps`, perché tutti i servizi nello scenario sono registrati nello spazio dei nomi `apps.local`.
- Un servizio virtuale denominato `serviceb.apps.local` perché il servizio virtuale rappresenta un servizio individuabile tramite questo nome e non vuoi modificare il codice per fare riferimento a un altro nome. Un servizio virtuale denominato `servicea.apps.local` viene aggiunto in una fase successiva.

Puoi utilizzare la AWS CLI versione 1.18.116 AWS Management Console o successiva o 2.0.38 o successiva per completare i seguenti passaggi. Se si utilizza il AWS CLI, utilizzare il `aws --version` comando per verificare la versione installata. AWS CLI Se non hai installato la versione 1.18.116 o successiva o 2.0.38 o successiva, devi [installare](#) o aggiornare la. AWS CLI Seleziona la scheda relativa allo strumento che desideri utilizzare.

### AWS Management Console

1. [Apri la procedura guidata per la prima esecuzione della console App Mesh all'inizio](https://console.aws.amazon.com/appmesh/)<https://console.aws.amazon.com/appmesh/>.
2. Per Mesh name (Nome mesh), immettere **apps**.
3. Per Virtual service name (Nome servizio virtuale), immettere **serviceb.apps.local**.
4. Scegliere Next (Avanti) per continuare.

### AWS CLI

1. Creare una mesh con il comando [create-mesh](#).

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

2. Creare un servizio virtuale con il comando [create-virtual-service](#).

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

## Fase 2: creare un nodo virtuale

Un nodo virtuale funziona come un puntatore logico a un servizio effettivo. Per ulteriori informazioni, consulta [Nodi virtuali](#).

Crea un nodo virtuale denominato `serviceB` perché uno dei nodi virtuali rappresenta il servizio effettivo denominato `serviceB`. Il servizio effettivo rappresentato dal nodo virtuale è individuabile tramite DNS con il nome host `serviceb.apps.local`. In alternativa, puoi individuare i servizi effettivi utilizzando AWS Cloud Map. Il nodo virtuale ascolta il traffico utilizzando il protocollo HTTP /2 sulla porta 80. Sono supportati anche altri protocolli, così come i controlli di stato. I nodi virtuali vengono creati per `serviceA` e `serviceBv2` in una fase successiva.

### AWS Management Console

1. Per Virtual node name (Nome nodo virtuale), immettere **serviceB**.
2. Per il metodo di individuazione del servizio, scegli DNS e inserisci **serviceb.apps.local** il DNS nome host.
3. In Configurazione del listener, scegli `http2` per Protocollo e immetti **80** per Porta.
4. Scegliere Next (Avanti) per continuare.

### AWS CLI

1. Crea un file denominato `create-virtual-node-serviceb.json` con i seguenti contenuti:

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

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

2. Crea il nodo virtuale con il [create-virtual-node](#) comando utilizzando il JSON file come input.

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

## Fase 3: creare un router virtuale e una route

I router virtuali gestiscono il traffico per uno o più servizi virtuali all'interno della mesh. Per ulteriori informazioni, consulta [Router virtuali](#) e [Route](#).

Crea le seguenti risorse:

- Un router virtuale denominato `serviceB` perché il servizio virtuale `serviceB.apps.local` non avvia la comunicazione in uscita con altri servizi. Tieni presente che il servizio virtuale creato in precedenza è un'astrazione del servizio `serviceb.apps.local` effettivo. Il servizio virtuale invia il traffico al router virtuale. Il router virtuale ascolta il traffico utilizzando il protocollo HTTP /2 sulla porta 80. Sono supportati anche altri protocolli.
- Una route denominata `serviceB`. Indirizza il 100% del traffico verso il nodo `serviceB` virtuale. Il peso viene calcolato in una fase successiva dopo l'aggiunta del nodo `serviceBv2` virtuale. Sebbene non sia trattato in questa guida, è possibile impostare criteri di filtro aggiuntivi per la route e aggiungere una policy per i nuovi tentativi per far sì che il proxy Envoy effettui più tentativi di invio del traffico a un nodo virtuale quando si verifica un problema di comunicazione.

### AWS Management Console

1. Per Virtual router name (Nome router virtuale), immettere **serviceB**.
2. In Configurazione del listener, scegli http2 per Protocollo e specifica **80** per Porta.

3. Per Route name (Nome route), immettere **serviceB**.
4. Per Tipo di route, scegli http2.
5. Per il nome del nodo virtuale nella configurazione Target, seleziona serviceB e inserisci **100** Weight.
6. In Match configuration, scegli un metodo.
7. Scegliere Next (Avanti) per continuare.

## AWS CLI

1. Creare un router virtuale.
  - a. Crea un file denominato `create-virtual-router.json` con i seguenti contenuti:

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

- b. Crea il router virtuale con il [create-virtual-router](#) comando utilizzando il JSON file come input.

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

2. Creare una route.
  - a. Crea un file denominato `create-route.json` con i seguenti contenuti:

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

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

- b. Crea il percorso con il comando [create-route](#) utilizzando il JSON file come input.

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

## Fase 4: Revisione e creazione

Rivedi le impostazioni in relazione alle istruzioni precedenti.

### AWS Management Console

Scegli Modifica per apportare modifiche in qualsiasi sezione. Una volta soddisfatte le impostazioni, scegli Crea una mesh.

La schermata Stato mostra tutte le risorse mesh create. Puoi visualizzare le risorse create nella console selezionando Visualizza mesh.

### AWS CLI

Esaminare le impostazioni della mesh creata con il comando [describe-mesh](#) .

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

Rivedi le impostazioni del servizio virtuale che hai creato con il [describe-virtual-service](#) comando.

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

Rivedi le impostazioni del nodo virtuale che hai creato con il [describe-virtual-node](#) comando.

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

Rivedi le impostazioni del router virtuale che hai creato con il [describe-virtual-router](#) comando.

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

Esaminare le impostazioni del percorso creato con il comando [describe-route](#) .

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

## Fase 5: creare risorse aggiuntive

Per completare lo scenario è necessario:

- Creare un nodo virtuale denominato `serviceBv2` e un altro denominato `serviceA`. Entrambi i nodi virtuali ascoltano le richieste sulla porta HTTP /2 80. Per il nodo `serviceA` virtuale, configura un backend di `serviceb.apps.local`. Tutto il traffico in uscita dal nodo `serviceA` virtuale viene inviato al servizio virtuale denominato `serviceb.apps.local`. Sebbene non sia trattato in questa guida, è anche possibile specificare un percorso file in cui scrivere i log degli accessi per un nodo virtuale.
- Crea un servizio virtuale aggiuntivo denominato `servicea.apps.local`, che invia tutto il traffico direttamente al nodo `serviceA` virtuale.
- Aggiornare la route `serviceB` creata in una fase precedente per inviare il 75% del traffico al nodo virtuale `serviceB` e il 25% del traffico al nodo virtuale `serviceBv2`. È possibile continuare a modificare i pesi finché `serviceBv2` riceve il 100% del traffico. Una volta inviato tutto il traffico a `serviceBv2`, puoi chiudere e interrompere il nodo `serviceB` virtuale e il servizio effettivo. Quando si cambiano i pesi, il codice non richiede alcuna modifica perché i nomi dei servizi virtuali ed effettivi `serviceb.apps.local` non cambiano. Tenere presente che il servizio virtuale `serviceb.apps.local` invia il traffico al router virtuale che instrada il traffico ai nodi virtuali. I nomi di individuazione dei servizi per i nodi virtuali possono essere modificati in qualsiasi momento.

## AWS Management Console

1. Nel riquadro di navigazione a sinistra, selezionare Meshes (Mesh).
2. Selezionare la mesh apps creata in una fase precedente.
3. Nel riquadro di navigazione a sinistra selezionare Virtual nodes (Nodi virtuali).
4. Scegliere Create virtual node (Crea nodo virtuale).
5. Per il nome del nodo virtuale, inserisci **serviceBv2**, per Metodo di rilevamento del servizio DNS, scegli e per DNSil nome host, inserisci **servicebv2.apps.local**
6. Per Configurazione del listener seleziona http2 per Protocollo e immetti **80** per Porta.
7. Scegliere Create virtual node (Crea nodo virtuale).
8. Scegli Crea un nodo virtuale. Immetti **serviceA** per Nome del nodo virtuale. Per il metodo di individuazione del servizio DNS, scegli e per il DNSnome host, inserisci **servicea.apps.local**
9. Per Inserisci un nome del servizio virtuale in Nuovo back-end immetti **serviceb.apps.local**.
10. In Configurazione del listener scegli http2 per Protocollo, immetti **80** per Porta e quindi scegli Crea un nodo virtuale.
11. Nel riquadro di navigazione a sinistra selezionare Virtual routers (Router virtuali) quindi selezionare il router virtuale serviceB dall'elenco.
12. In Route, seleziona la route denominata ServiceB creata in una fase precedente e scegli Modifica.
13. In Destinazioni, Nome del nodo virtuale, cambia il valore di Peso per serviceB su **75**.
14. Scegli Aggiungi obiettivo, scegli serviceBv2 dall'elenco a discesa e imposta il valore di Weight su **25**
15. Seleziona Salva.
16. Nel riquadro di navigazione a sinistra seleziona Servizi virtuali, quindi scegli Crea un servizio virtuale.
17. Immetti **servicea.apps.local** per Nome del servizio virtuale, seleziona Nodo virtuale per Provider, seleziona serviceA per Nodo virtuale e quindi scegli Crea un servizio virtuale.

## AWS CLI

1. Creare il nodo virtuale serviceBv2.

- a. Crea un file denominato `create-virtual-node-servicebv2.json` con i seguenti contenuti:

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

- b. Creare il nodo virtuale.

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

## 2. Creare il nodo virtuale serviceA.

- a. Crea un file denominato `create-virtual-node-servicea.json` con i seguenti contenuti:

```
{
 "meshName" : "apps",
 "spec" : {
 "backends" : [
 {
 "virtualService" : {
 "virtualServiceName" : "serviceb.apps.local"
 }
 }
]
 }
}
```

```

],
 "listeners" : [
 {
 "portMapping" : {
 "port" : 80,
 "protocol" : "http2"
 }
 }
],
 "serviceDiscovery" : {
 "dns" : {
 "hostname" : "servicea.apps.local"
 }
 }
 },
 "virtualNodeName" : "serviceA"
}

```

- b. Creare il nodo virtuale.

```

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

```

3. Aggiornare il servizio virtuale `serviceb.apps.local` creato in una fase precedente per inviare il traffico al router virtuale `serviceB`. Quando il servizio virtuale è stato originariamente creato non inviava traffico poiché il router virtuale `serviceB` non era ancora stato creato.

- a. Crea un file denominato `update-virtual-service.json` con i seguenti contenuti:

```

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

```

- b. Aggiorna il servizio virtuale con il [update-virtual-service](#) comando.

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

4. Aggiornare la route `serviceB` creata in una fase precedente.

a. Crea un file denominato `update-route.json` con i seguenti contenuti:

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

b. Aggiornare la route con il comando [update-route](#).

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

5. Creare il servizio virtuale `serviceA`.

a. Crea un file denominato `create-virtual-servicea.json` con i seguenti contenuti:

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

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

- b. Creare il servizio virtuale.

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

## Riepilogo della mesh

Prima di creare la mesh dei servizi, erano presenti tre servizi effettivi denominati `servicea.apps.local`, `serviceb.apps.local` e `servicebv2.apps.local`. Oltre ai servizi effettivi, ora hai una mesh dei servizi che contiene le seguenti risorse che rappresentano i servizi effettivi:

- Due servizi virtuali. Il proxy invia tutto il traffico dal servizio virtuale `servicea.apps.local` al servizio virtuale `serviceb.apps.local` tramite un router virtuale.
- Tre nodi virtuali denominati `serviceA`, `serviceB` e `serviceBv2`. Il proxy Envoy utilizza le informazioni di individuazione dei servizi configurate per i nodi virtuali per cercare gli indirizzi IP dei servizi effettivi.
- Un router virtuale con una route che indica al proxy Envoy di indirizzare il 75% del traffico in ingresso al nodo virtuale `serviceB` e il 25% del traffico al nodo virtuale `serviceBv2`.

## Fase 6: aggiornare i servizi

Dopo aver creato la mesh, è necessario completare le seguenti attività:

- Autorizza il proxy Envoy che distribuisce con ogni servizio a leggere la configurazione di uno o più nodi virtuali. Per ulteriori informazioni su come autorizzare il proxy, vedere [Autorizzazione proxy Envoy Proxy](#)
- Per aggiornare il servizio esistente, completa i passaggi seguenti.

## Per configurare un'EC2istanza Amazon come membro di un nodo virtuale

1. Crea un IAM ruolo.
  - a. Crea un file denominato `ec2-trust-relationship.json` con i seguenti contenuti.

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

- b. Crea un IAM ruolo con il seguente comando.

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

2. Allega IAM politiche al ruolo che gli consentano di leggere da Amazon ECR e solo la configurazione di uno specifico nodo virtuale App Mesh.
  - a. Crea un file denominato `virtual-node-policy.json` con il seguente contenuto. `apps` è il nome della mesh creata in [the section called “Fase 1: creare una mesh e un servizio virtuale”](#) e `serviceB` è il nome del nodo virtuale creato in [the section called “Fase 2: creare un nodo virtuale”](#). Replace (Sostituisci) `111122223333` con l'ID del tuo account e `us-west-2` con la regione in cui hai creato la mesh.

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

```

]
 }
]
}

```

- b. Creare la policy con il seguente comando.

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

- c. Allega la policy che hai creato nel passaggio precedente al ruolo in modo che il ruolo possa leggere la configurazione solo per il nodo serviceB virtuale da 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. Allega la policy AmazonEC2ContainerRegistryReadOnly gestita al ruolo in modo che possa estrarre l'immagine del contenitore Envoy da Amazon. ECR

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

3. [Avvia un'EC2istanza Amazon con il IAM ruolo](#) che hai creato.
4. Connect alla tua istanza tramiteSSH.
5. Installa Docker and the AWS CLI sulla tua istanza in base alla documentazione del tuo sistema operativo.
6. Effettua l'autenticazione nel repository ECR Amazon Envoy nella regione da cui desideri che il client Docker estragga l'immagine.
  - Tutte le regioni tranne me-south-1,,,ap-east-1, ap-southeast-3 e eu-south-1 il-central-1 af-south-1 Puoi sostituire **us-west-2** con qualsiasi [regione supportata](#) tranne me-south-1ap-east-1,ap-southeast-3,eu-south-1,il-central-1, eaf-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
```

- Regione 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
```

- Regione 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. Esegui uno dei seguenti comandi per avviare il contenitore App Mesh Envoy sulla tua istanza, a seconda della regione da cui desideri estrarre l'immagine. Il *apps* e *serviceB* i valori sono i nomi della mesh e dei nodi virtuali definiti nello scenario. Queste informazioni indicano al proxy quale configurazione del nodo virtuale leggere da App Mesh. Per completare lo scenario, devi anche completare questi passaggi per le EC2 istanze Amazon che ospitano i servizi *serviceBv2* rappresentati dai nodi *serviceA* virtuali. Per la propria applicazione, sostituire questi valori con quelli appropriati.
- Tutte le regioni tranne me-south-1, ap-east-1, ap-southeast-3, eu-south-1, il-central-1, eaf-south-1. Puoi sostituire *Region-code* con qualsiasi [regione supportata](#) ad eccezione di me-south-1, ap-east-1, ap-southeast-3, eu-south-1, il-central-1, e af-south-1 Regioni. Puoi sostituire *1337* con qualsiasi valore compreso tra 0 e 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:v1.29.6.1-prod
```

- Regione me-south-1. Puoi sostituire *1337* con qualsiasi valore compreso tra 0 e 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:v1.29.6.1-prod
```

- Regione ap-east-1. Puoi sostituire **1337** con qualsiasi valore compreso tra 0 e 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:v1.29.6.1-prod
```

### Note

La APPMESH\_RESOURCE\_ARN proprietà richiede una versione 1.15.0 o successiva dell'immagine Envoy. Per ulteriori informazioni, consulta [Envoy](#).

### Important

Solo la versione v1.9.0.0-prod o successiva è supportata per l'uso con App Mesh.

8. Seleziona Show more di seguito. Crea un file denominato envoy-networking.sh nell'istanza con i seguenti contenuti. Replace (Sostituisci) **8000** con la porta utilizzata dal codice dell'applicazione per il traffico in entrata. Puoi modificare il valore per APPMESH\_IGNORE\_UID, ma il valore deve essere uguale a quello specificato nel passaggio precedente, ad esempio 1337. Se necessario puoi aggiungere altri indirizzi in APPMESH\_EGRESS\_IGNORED\_IP. Non modificare altre righe.

```
#!/bin/bash -e

#
Start of configurable options
#

#APPMESH_START_ENABLED=""
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 processes 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. Per configurare le regole `iptables` per instradare il traffico dell'applicazione al proxy Envoy, esegui lo script creato nel passaggio precedente.

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

10. Avviare il codice dell'applicazione del nodo virtuale.

#### Note

Per altri esempi e procedure dettagliate per App Mesh, consulta l'archivio degli esempi di [App Mesh](#).

## App Mesh

Puoi trovare end-to-end procedure dettagliate che mostrano AWS App Mesh azioni ed esempi di codice per l'integrazione con vari AWS servizi nel seguente repository:

[Esempi di App Mesh](#)

# App Mesh

App Mesh è composta dai seguenti concetti.

- [Resh di servizio](#)
- [Servizi virtuali](#)
- [Gateway virtuali virtuale virtuale virtuale virtuale virtuale](#)
- [Nodi virtuali](#)
- [Router virtuali](#)

## Resh di servizio

Una mesh dei servizi è un limite logico per il traffico di rete tra i servizi che si trovano al suo interno. Dopo aver creato la mesh dei servizi, puoi creare servizi virtuali, nodi virtuali, router virtuali e route per distribuire il traffico tra le applicazioni nella mesh.

## Creazione di una mesh di servizi

### Note

Quando si crea una rete Mesh, è necessario aggiungere un selettore di namespace. Se il selettore dello spazio dei nomi è vuoto, seleziona tutti i namespace. Per limitare i namespace, usa un'etichetta per associare le risorse App Mesh alla mesh creata.

## AWS Management Console

Come creare una mesh di servizi tramite AWS Management Console

1. Apri la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).
2. Scegliere Create mesh (Crea mesh).
3. Per Mesh name (Nome mesh), specificare un nome per la mesh dei servizi.
4. (Facoltativo) Scegli Consenti traffico esterno. Per impostazione predefinita, i proxy nella mesh inoltrano solo il traffico tra loro. Se consenti il traffico esterno, i proxy nella mesh inoltrano anche il traffico TCP direttamente ai servizi che non sono distribuiti con un proxy definito nella mesh.

**Note**

Se specifichi qualche back-end su un nodo virtuale quando utilizzi `ALLOW_ALL`, devi specificare tutte le uscite per quel nodo virtuale come back-end. In caso contrario, `ALLOW_ALL` non funzionerà più per quel nodo virtuale.

## 5. Preferenza versione IP

Controlla quale versione IP deve essere utilizzata per il traffico all'interno della mesh attivando Ignora il comportamento della versione IP predefinita. Per impostazione predefinita, App Mesh utilizza una varietà di versioni IP.

**Note**

La mesh applica la preferenza IP a tutti i nodi virtuali e i gateway virtuali all'interno di una mesh. Questo comportamento può essere ignorato su un singolo nodo virtuale impostando la preferenza IP quando si crea o si modifica il nodo. La preferenza IP non può essere sovrascritta su un gateway virtuale perché la configurazione dei gateway virtuali che consente loro di ascoltare il traffico IPv4 e IPv6 è la stessa indipendentemente dalla preferenza impostata sulla mesh.

- Default (Predefinito)
  - Il resolver DNS di Envoy preferisce IPv6 e utilizza IPv4.
  - Utilizziamo l'IPv4 indirizzo restituito da AWS Cloud Map se disponibile, e torniamo a utilizzare l'IPv6 indirizzo.
  - L'endpoint creato per l'app locale utilizza un IPv4 indirizzo.
  - Gli ascoltatori di Envoy si collegano a tutti IPv4 gli indirizzi.
- IPv6 preferito
  - Il resolver DNS di Envoy preferisce IPv6 e utilizza IPv4.
  - L'IPv6 indirizzo restituito da AWS Cloud Map viene utilizzato se disponibile e torna a utilizzare l'IPv4 indirizzo
  - L'endpoint creato per l'app locale utilizza un IPv6 indirizzo.
  - Gli ascoltatori di Envoy si legano a tutti IPv4 gli IPv6 indirizzi.
- IPv4 preferito

- Il resolver DNS di Envoy preferisce IPv4 e utilizza IPv6.
  - Utilizziamo l'IPv4 indirizzo restituito da AWS Cloud Map se disponibile, e torniamo a utilizzare l'IPv6 indirizzo.
  - L'endpoint creato per l'app locale utilizza un IPv4 indirizzo.
  - Gli ascoltatori di Envoy si legano a tutti IPv4 e IPv6 indirizzi.
- Solo IPv6
    - Il resolver DNS di Envoy utilizza solo IPv6.
    - Viene utilizzato solo l'IPv6 indirizzo AWS Cloud Map restituito da. Se AWS Cloud Map restituisce un IPv4 indirizzo, non viene utilizzato alcun indirizzo IP e i risultati vuoti vengono restituiti a Envoy.
    - L'endpoint creato per l'app locale utilizza un IPv6 indirizzo.
    - Gli ascoltatori di Envoy si legano a tutti IPv4 e IPv6 indirizzi.
  - Solo IPv4
    - Il resolver DNS di Envoy utilizza solo IPv4.
    - Viene utilizzato solo l'IPv4 indirizzo AWS Cloud Map restituito da. Se AWS Cloud Map restituisce un IPv6 indirizzo, non viene utilizzato alcun indirizzo IP e i risultati vuoti vengono restituiti a Envoy.
    - L'endpoint creato per l'app locale utilizza un IPv4 indirizzo.
    - Gli ascoltatori di Envoy si legano a tutti IPv4 e IPv6 indirizzi.
6. Scegliere Create mesh (Crea mesh) per terminare.
  7. Condividere la mesh con altri account (facoltativo). Una mesh condivisa consente alle risorse create da account diversi di comunicare tra loro nella stessa mesh. Per ulteriori informazioni, consulta [Utilizzo di mesh condivise](#).

## AWS CLI

Per creare una rete utilizzando il AWS CLI.

Crea una service mesh usando il seguente comando (sostituisci i valori *rossi* con i tuoi):

1. 

```
aws appmesh create-mesh --mesh-name meshName
```

2. Output di esempio:

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

Per ulteriori informazioni sulla creazione di una mesh con AWS CLI for App Mesh, vedete il comando [create-mesh](#) nel [AWS CLI riferimento](#).

## Eliminazione di una rete

### AWS Management Console

Come eliminare un gateway virtuale tramite AWS Management Console

1. Apri la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).
2. Scegliere la mesh da eliminare. Vengono elencate tutte le mesh che possiedi e che sono state [condivise](#) con te.
3. Nella casella di conferma, digita **delete** e fai clic su Elimina.

### AWS CLI

Per eliminare una mesh utilizzando il AWS CLI

1. Usa il seguente comando per eliminare la tua mesh (sostituisci i valori *rossi* con i tuoi):

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

## 2. Output di esempio:

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

Per ulteriori informazioni sull'eliminazione di una mesh con AWS CLI for App Mesh, vedete il comando [delete-mesh](#) nel [AWS CLI riferimento](#).

## Servizi virtuali

Un servizio virtuale è un'astrazione di un servizio reale fornita direttamente o indirettamente da un nodo virtuale mediante un router virtuale. I servizi dipendenti chiamano il servizio virtuale mediante il relativo `virtualServiceName` e tali richieste vengono instradate al nodo virtuale o al router virtuale specificato come provider per il servizio virtuale.

# Creazione di un servizio virtuale

## AWS Management Console

Per creare un servizio virtuale utilizzando AWS Management Console

1. Apri la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).
2. Scegliere la mesh in cui creare il servizio virtuale. Vengono elencate tutte le mesh che possiedi e che sono state [condivise](#) con te.
3. Nel riquadro di navigazione sinistro, scegliere Virtual services (Servizi virtuali).
4. Scegliere Create virtual service (Crea servizio virtuale).
5. In Virtual service name (Nome servizio virtuale), scegliere un nome per il servizio virtuale. Puoi scegliere qualsiasi nome, ma il nome di individuazione del servizio reale a cui ti rivolgi, ad esempio `my-service.default.svc.cluster.local`, è consigliato per semplificare la correlazione dei servizi virtuali con i servizi reali. In questo modo non è necessario modificare il codice per fare riferimento a un nome diverso rispetto a quello a cui fa attualmente riferimento il codice. Il nome specificato deve essere riconducibile a un indirizzo IP senza loopback perché il contenitore dell'app deve essere in grado di risolvere correttamente il nome prima che la richiesta venga inviata al proxy Envoy. Puoi utilizzare qualsiasi indirizzo IP senza loopback perché né l'app né i contenitori proxy comunicano con questo indirizzo IP. Il proxy comunica con altri servizi virtuali tramite i nomi che hai configurato per essi in App Mesh, non tramite gli indirizzi IP a cui i nomi si riferiscono.
6. Per Provider, scegliere il tipo di provider per il servizio virtuale:
  - Se si desidera che il servizio virtuale distribuisca il traffico su più nodi virtuali, selezionare Virtual router (Router virtuale) e quindi scegliere il router virtuale da utilizzare dal menu a discesa.
  - Se desideri che il servizio virtuale raggiunga un nodo virtuale direttamente senza un router virtuale, seleziona Nodo virtuale e quindi scegli il nodo virtuale da utilizzare dal menu a discesa.

### Note

App Mesh può creare automaticamente una politica di ripetizione del percorso Envoy predefinita per ogni provider di nodi virtuali definito a partire dal 29 luglio

2020, anche se non è possibile definire tale politica tramite l'API App Mesh. Per ulteriori informazioni, consulta [Politica predefinita per i nuovi tentativi di routing](#).

- Se al momento non si desidera che il servizio virtuale instradi il traffico (ad esempio, se il router virtuale o i nodi virtuali non esistono ancora), scegliere None (Nessuno). È possibile aggiornare il provider per questo servizio virtuale in un secondo momento.

7. Scegliere Create virtual service (Crea servizio virtuale) per terminare.

## AWS CLI

Per creare un servizio virtuale utilizzando ilAWS CLI.

Crea un servizio virtuale con un provider di nodi virtuali utilizzando il seguente comando e un file JSON di input (sostituisci i valori *rossi* con i tuoi):

1. 

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

2. Contenuto dell'esempio create-virtual-service-virtual -node.json:

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

3. Output di esempio:

```
{
 "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"
}
```

Per ulteriori informazioni sulla creazione di un servizio virtuale con AWS CLI for App Mesh, consulta il [create-virtual-service](#) comando nel [AWS CLI riferimento](#).

## Eliminazione di un servizio virtuale

### Note

Non è possibile eliminare un servizio virtuale a cui fa riferimento un percorso gateway. È necessario prima eliminare il percorso del gateway.

### AWS Management Console

Per eliminare un servizio virtuale tramite AWS Management Console

1. Apri la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).
2. Scegliere la mesh da cui si desidera eliminare un servizio virtuale. Vengono elencate tutte le mesh che possiedi e che sono state [condivise](#) con te.
3. Nel riquadro di navigazione sinistro, scegliere Virtual services (Servizi virtuali).

- Scegli il servizio virtuale che desideri eliminare e fai clic su Elimina nell'angolo in alto a destra. Puoi eliminare solo un gateway virtuale in cui il tuo account è elencato come proprietario della risorsa.
- Nella casella di conferma, digita **delete** e fai clic su Elimina.

## AWS CLI

Per eliminare un servizio virtuale tramite AWS CLI

- Usa il seguente comando per eliminare il tuo servizio virtuale (sostituisci i valori *rossi* con i tuoi):

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

- Output di esempio:

```
{
 "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"
 }
}
```

Per ulteriori informazioni sull'eliminazione di un servizio virtuale con AWS CLI for App Mesh, consulta il [delete-virtual-service](#) comando nel [AWS CLI riferimento](#).

## Gateway virtuali virtuale virtuale virtuale virtuale virtuale

Un gateway virtuale consente alle risorse esterne alla mesh di comunicare con le risorse che si trovano all'interno della mesh. Il gateway virtuale rappresenta un proxy Envoy in esecuzione in un servizio Amazon ECS, in un servizio Kubernetes o in un'istanza Amazon EC2. A differenza di un nodo virtuale, che rappresenta Envoy in esecuzione con un'applicazione, un gateway virtuale rappresenta Envoy distribuito autonomamente.

Le risorse esterne devono essere in grado di risolvere un nome DNS in un indirizzo IP assegnato al servizio o all'istanza che esegue Envoy. Envoy può quindi accedere a tutta la configurazione App Mesh per le risorse che si trovano all'interno della mesh. La configurazione per la gestione delle richieste in arrivo sul gateway virtuale viene specificata utilizzando [Gateway Routes](#).

### Important

Un gateway virtuale con un listener HTTP o HTTP2 riscrive il nome host della richiesta in entrata nel nome del servizio virtuale di destinazione del percorso del gateway e il prefisso corrispondente della Route del gateway viene riscritto/, per impostazione predefinita. Ad esempio, se hai configurato il prefisso di corrispondenza del percorso del gateway su `e/chapter`, se la richiesta in arrivo è `/chapter/1`, la richiesta verrà riscritta in `/1`. Per configurare le riscritture, consulta la sezione [Creazione di un percorso gateway](#) di Gateway Routes.

Quando si crea un gateway virtuale `proxyConfiguration` e `nonuser` deve essere configurato.

Per completare una end-to-end procedura dettagliata, consulta [Configurazione del gateway in entrata](#).



- d. (Facoltativo) Immettere un nome alternativo per un messaggio Oggetto. Per aggiungere altre SAN, selezionare Aggiungi SAN. Le SAN devono essere formattate con FQDN o URI.
- e. (Facoltativo) Seleziona Fornisci certificato client e una delle opzioni seguenti per fornire un certificato client quando un server lo richiede e abilitare l'autenticazione TLS reciproca. Per ulteriori informazioni sul protocollo TLS reciproco, consulta i documenti App Mesh [Mutual TLS Authentication](#).
  - Hosting Envoy Secret Discovery Service (SDS): inserisci il nome del segreto che Envoy recupera utilizzando il Secret Discovery Service.
  - Hosting locale dei file: specifica il percorso del file della catena di certificati, nonché la chiave privata, nel file system in cui è distribuito Envoy. Per una descrizione completa della distribuzione di una mesh con un'applicazione di esempio che utilizza la crittografia con file locali, consulta [Configurazione di TLS con certificati TLS forniti da file](#) su GitHub. end-to-end
7. (Facoltativo) Per configurare la registrazione, seleziona Registrazione. Inserisci il percorso dei log di accesso HTTP che desideri utilizzare in Envoy. Consigliamo il /dev/stdout percorso in modo da poter utilizzare i driver di registro Docker per esportare i log di Envoy in un servizio come Amazon CloudWatch Logs.

 Note

I log devono ancora essere integrati da un agente nell'applicazione e inviati a una destinazione. Questo percorso file indica a Envoy dove inviare i log.

8. Configura il listener.
  - a. Seleziona un protocollo e specifica la porta su cui Envoy ascolta il traffico. Il listener http consente la transizione della connessione ai websocket. È possibile fare clic su Aggiungi ascoltatore per aggiungere più ascoltatori. Il pulsante Rimuovi rimuoverà quel listener.
  - b. (Facoltativo) Abilita pool di connessioni

Il pool di connessioni limita il numero di connessioni che Virtual Gateway Envoy può stabilire contemporaneamente. Ha lo scopo di proteggere la tua istanza Envoy dal sovraccarico di connessioni e ti consente di adattare la modellazione del traffico alle esigenze delle tue applicazioni.

È possibile configurare le impostazioni del pool di connessioni sul lato di destinazione per un listener gateway virtuale. App Mesh imposta le impostazioni del pool di connessioni lato client su infinite per impostazione predefinita, semplificando la configurazione mesh.

 Note

I protocolli `connectionPool` e `connectionPool` `PortMapping` devono essere uguali. Se il protocollo del listener è `grpc` o `http2`, specificare `maxRequests` solo. Se il protocollo del tuo listener è `http`, puoi specificare entrambi `maxConnections` e `maxPendingRequests`.

- In Numero massimo di connessioni, specificare il numero massimo di connessioni in uscita.
  - In Numero massimo di richieste parallel che possono essere stabilite con Virtual Gateway Envoy.
  - (Facoltativo) Per Numero massimo di richieste in sospeso, specifica il numero di richieste in eccesso dopo Numero massimo di connessioni che un Envoy mette in coda. Il valore di default è 2147483647.
- c. (Facoltativo) Se desideri configurare un controllo dello stato di salute per il tuo ascoltatore, seleziona Abilita controllo dello stato.

Una politica di controllo dello stato di Health è facoltativa, ma se si specifica un valore per una politica Health, è necessario specificare i valori per la soglia di integrità, l'intervallo di controllo dello stato, il protocollo di controllo dello stato, il periodo di timeout e la soglia non salutare.

- Per il protocollo Health Check, scegli un protocollo. Se si seleziona `grpc`, il servizio deve essere conforme a [GRPC Health Checking Health Checking](#).
- Per Health check port (Porta del controllo dello stato), specificare la porta su cui deve essere eseguito il controllo dello stato.
- Per Healthy threshold (Soglia di integrità), specificare il numero di controlli dello stato andati a buon fine consecutivi che devono verificarsi prima di dichiarare il listener integro.
- Per Health check interval (Intervallo del controllo dello stato), specificare il periodo di tempo in millisecondi tra ogni esecuzione del controllo dello stato.

- Per Path (Percorso), specificare il percorso di destinazione per la richiesta di controllo dello stato. Questo valore viene utilizzato solo se il protocollo Health check è `http` o `https`. Il valore viene ignorato per gli altri protocolli.
  - Come Periodo di timeout, specificare il periodo di tempo di attesa in millisecondi quando si riceve una risposta dal controllo dello stato in millisecondi.
  - Per Unhealthy threshold (Soglia di mancata integrità), specificare il numero di controlli dello stato non andati consecutivamente a buon fine che sono necessari per dichiarare il listener non integro.
- d. (Facoltativo) Se desideri specificare se i client comunicano con questo gateway virtuale tramite TLS, seleziona Abilita terminazione TLS.
- In Modalità, seleziona la modalità per cui desideri configurare TLS sul listener.
  - Per Metodo certificato, seleziona una delle opzioni seguenti. Il certificato deve soddisfare specifici requisiti. Per ulteriori informazioni, consulta [Requisiti del certificato](#).
    - AWS Certificate Managerhosting: seleziona un certificato esistente.
    - Hosting Envoy Secret Discovery Service (SDS): inserisci il nome del segreto che Envoy recupera utilizzando il Secret Discovery Service.
    - Hosting di file locale: specifica il percorso della catena di certificati e dei file di chiave privata sul file system in cui è distribuito Envoy.
  - (Facoltativo) Seleziona Richiedi certificato client e una delle opzioni seguenti per abilitare l'autenticazione TLS reciproca se il client fornisce un certificato. Per ulteriori informazioni sul protocollo TLS reciproco, consulta i documenti App Mesh [Mutual TLS Authentication](#).
    - Hosting Envoy Secret Discovery Service (SDS): inserisci il nome del segreto che Envoy recupera utilizzando il Secret Discovery Service.
    - Hosting di file locale: specifica il percorso del file della catena di certificati sul file system in cui è distribuito Envoy.
  - (Facoltativo) Immettere un nome alternativo per un messaggio Oggetto. Per aggiungere altre SAN, selezionare Aggiungi SAN. Le SAN devono essere formattate con FQDN o URI.

## 9. Scegli Crea gateway virtuale per terminare.

## AWS CLI

Per creare un gateway virtuale utilizzando ilAWS CLI.

Crea un gateway virtuale usando il seguente comando e inserisci JSON (sostituisci i valori *rossi* con i tuoi):

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

2. Contenuto dell'esempio create-virtual-gateway .json:

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

3. Output di esempio:

```
{
 "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"
 }
}

```

Per ulteriori informazioni sulla creazione di un gateway virtuale con AWS CLI for App Mesh, consulta il [create-virtual-gateway](#) comando nel [AWS CLI riferimento](#).

## Implementazione di gateway virtuale virtuale virtuale

Implementa un servizio Amazon ECS o Kubernetes che contenga solo il [contenitore Envoy](#). È inoltre possibile distribuire il container Envoy in un'istanza Amazon EC2. Per ulteriori informazioni, consulta [Nozioni di base su App Mesh e Amazon EC2](#). Per ulteriori informazioni su come distribuire su Amazon ECS, consulta [Guida introduttiva ad App Mesh e Amazon ECS](#) o [Guida introduttiva ad AWS App Mesh e Kubernetes](#) per la distribuzione su Kubernetes. È necessario impostare la variabile di ambiente `APPMESH_RESOURCE_ARN` ambiente `sumesh/mesh-name/virtualGateway/virtual-gateway-name` e non specificare la configurazione del proxy in modo che il traffico del proxy non venga reindirizzato a se stesso. Per impostazione predefinita, App Mesh utilizza il nome della risorsa specificata in `APPMESH_RESOURCE_ARN` quando Envoy si riferisce a se stesso nei parametri e nelle tracce. È possibile ignorare questo comportamento impostando la variabile di ambiente `APPMESH_RESOURCE_CLUSTER` con il proprio nome.

Ti consigliamo di distribuire più istanze del contenitore e configurare un Network Load Balancer per bilanciare il carico del traffico verso le istanze. Il nome del service discovery del load balancer è il nome che si desidera che i servizi esterni utilizzino per accedere alle risorse presenti nella mesh, ad esempio `myapp.example.com`. Per ulteriori informazioni, consulta [Creating a Network Load Balancer](#) (Amazon ECS), [Creazione di un Load Balancer esterno](#) (Kubernetes) o [Tutorial: Incrementa](#)



```
 "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"
 }
}
```

Per ulteriori informazioni sull'eliminazione di un gateway virtuale con AWS CLI for App Mesh, consulta il [delete-virtual-gateway](#) comando nel [AWS CLI riferimento](#).

## route del gateway

Una route gateway è collegata a un gateway virtuale e instrada il traffico a un servizio virtuale esistente. Se una route corrisponde a una richiesta, può distribuire il traffico a un servizio virtuale di destinazione. Questo argomento ti aiuta a lavorare con gateway in una mesh di servizio.

## Creazione di gateway

### AWS Management Console

Per creare una route gateway mediante l'AWS Management Console

1. Apri la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).
2. Scegli la mesh in cui creare l'gateway. Vengono elencate tutte le mesh che possiedi e che sono state [condivise](#) con te.
3. Scegli Gateway virtuali nella barra di navigazione a sinistra.
4. Scegli il gateway virtuale a cui desideri associare un nuovo percorso del gateway. Se l'elenco risulta vuoto, è necessario [crearne uno](#) prima. Puoi creare un percorso gateway solo per un gateway virtuale di cui il tuo account è elencato come proprietario della risorsa.
5. Nella tabella Percorsi gateway, scegli Crea percorso gateway.
6. Per il nome del percorso del gateway, specifica il nome da utilizzare per il percorso del gateway.
7. Per il tipo di percorso Gateway scegli http, http2 o grpc.
8. Seleziona un nome di servizio virtuale esistente. Se l'elenco risulta vuoto, è necessario [crearne uno](#) prima.
9. Scegli la porta che corrisponde alla destinazione per la porta del provider di servizi virtuali. La porta del provider di servizi virtuali è necessaria quando il provider (router o nodo) del servizio virtuale selezionato ha più listener.
10. (Facoltativo) In Priorità, specifica la priorità per questa rotta del gateway.
11. Per la configurazione Match, specifica:
  - Se http/http2 è il tipo selezionato:
    - (Facoltativo) Metodo - Specifica l'intestazione del metodo da abbinare nelle richieste http /http2 in entrata.
    - (Facoltativo) Corrispondenza porta - Corrisponde alla porta per il traffico in entrata. La corrispondenza delle porte è necessaria se questo gateway virtuale ha più listener.
    - (Facoltativo) Exact/Suffix hostname - Specifica il nome host che deve essere abbinato alla richiesta in entrata da indirizzare al servizio virtuale di destinazione.
    - (Facoltativo) Percorso prefisso/esatto/regex - Il metodo di corrispondenza del percorso dell'URL.

- **Corrispondenza del prefisso** - Una richiesta corrispondente di una rotta gateway viene riscritta con il nome del servizio virtuale di destinazione e il prefisso corrispondente viene riscritto/, per impostazione predefinita. A seconda di come configuri il servizio virtuale, potrebbe utilizzare un router virtuale per indirizzare la richiesta a diversi nodi virtuali, in base a prefissi o intestazioni specifici.

 **Important**

- Non è possibile specificare né l'uno `/aws-appmesh*` né `/aws-app-mesh*` l'altro per la corrispondenza del prefisso. Questi prefissi sono riservati per il future uso interno di App Mesh.
- Se sono definiti più percorsi gateway, una richiesta viene abbinata alla rotta con il prefisso più lungo. Ad esempio, se esistessero due percorsi gateway, uno con il prefisso `di/chapter` e l'altro con il prefisso `di/`, una richiesta `diwww.example.com/chapter/` verrebbe abbinata alla rotta del gateway con il `/chapter` prefisso.

 **Note**

Se abiliti la corrispondenza basata su Path/Prefix, App Mesh abilita la normalizzazione del percorso ([normalize\\_path](#) e [merge\\_slashes](#)) per ridurre al minimo la probabilità di vulnerabilità legate alla confusione dei percorsi. Le vulnerabilità legate alla confusione dei percorsi si verificano quando le parti che partecipano alla richiesta utilizzano rappresentazioni di percorso diverse.

- **Corrispondenza esatta** - Il parametro esatto disabilita la corrispondenza parziale per un percorso e si assicura che restituisca il percorso solo se il percorso corrisponde **ESATTAMENTE** all'URL corrente.
- **Regex match** - Utilizzato per descrivere modelli in cui più URL possono effettivamente identificare una singola pagina del sito Web.
- (Facoltativo) **Parametri di interrogazione** - Questo campo consente di confrontare i parametri dell'interrogazione.
- (Facoltativo) **Headers** - Specifica le intestazioni per http e http2. Dovrebbe corrispondere alla richiesta in entrata da indirizzare al servizio virtuale di destinazione.
- Se `grpc` è il tipo selezionato:

- Tipo di corrispondenza del nome host e (facoltativo) Corrispondenza esatta/suffisso - Specifica il nome host che deve essere abbinato alla richiesta in entrata da indirizzare al servizio virtuale di destinazione.
- nome servizio grpc - Il servizio grpc funge da API per l'applicazione ed è definito con ProtoBuf.

 Important

Non è possibile specificare `/aws . app-mesh* oaws . appmesh` per il nome del servizio. Questi nomi di servizio sono riservati per il future uso interno di App Mesh.

- (Facoltativo) Metadati - Specifica i metadati per grpc. Dovrebbe corrispondere alla richiesta in entrata da indirizzare al servizio virtuale di destinazione.

## 12. (Facoltativo) Per la configurazione di riscrittura:

- Se http/http2 è il tipo selezionato:
  - Se Prefix è il tipo di corrispondenza selezionato:
    - Sostituisci la riscrittura automatica del nome host - Per impostazione predefinita, il nome host viene riscritto con il nome del servizio virtuale di destinazione.
    - Sostituisci la riscrittura automatica del prefisso - Quando è attivata, Prefix rewrite specifica il valore del prefisso riscritto.
  - Se Exact Path è il tipo di corrispondenza selezionato:
    - Sostituisci la riscrittura automatica del nome host: per impostazione predefinita, il nome host viene riscritto con il nome del servizio virtuale di destinazione.
    - Riscrittura del percorso - Specifica il valore del percorso riscritto. Nessun percorso predefinito.
  - Se Regex Path è il tipo di corrispondenza selezionato:
    - Sostituisci la riscrittura automatica del nome host: per impostazione predefinita, il nome host viene riscritto con il nome del servizio virtuale di destinazione.
    - Riscrittura del percorso - Specifica il valore del percorso riscritto. Nessun percorso predefinito.
- Se grpc è il tipo selezionato:

- Sostituisci la riscrittura automatica del nome host - Per impostazione predefinita, il nome host viene riscritto con il nome del servizio virtuale di destinazione.

13. Scegli Crea percorso gateway per terminare.

## AWS CLI

Per creare un percorso gateway utilizzando ilAWS CLI.

Crea un percorso gateway usando il seguente comando e inserisci JSON (sostituisci i valori *rossi* con i tuoi):

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. Contenuto dell'esempio create-gateway-route .json:

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

3. Output di esempio:

```
{
 "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"
 }
}
```

Per ulteriori informazioni sulla creazione di un percorso gateway con AWS CLI for App Mesh, consulta il [create-gateway-route](#) comando nel [AWS CLI riferimento](#).

## Eliminazione gateway

### AWS Management Console

Per eliminare una route gateway mediante l'AWS Management Console

1. Apri la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).
2. Scegliere la mesh da cui si desidera eliminare una route gateway. Vengono elencate tutte le mesh che possiedi e che sono state [condivise](#) con te.
3. Scegli Gateway virtuali nella barra di navigazione a sinistra.
4. Scegli il gateway virtuale da cui desideri eliminare un percorso gateway.
5. Nella tabella Percorsi del gateway, scegli il percorso del gateway che desideri eliminare e seleziona Elimina. Puoi eliminare un percorso gateway solo se il tuo account è indicato come proprietario della risorsa.
6. Nella casella di conferma, digita **delete** e fai clic su Elimina.

### AWS CLI

Per eliminare una route gateway mediante l'AWS CLI

1. Usa il seguente comando per eliminare il percorso del gateway (sostituisci i valori *rossi* con i tuoi):

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

2. Output di esempio:

```
{
 "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"
}
}

```

Per ulteriori informazioni sull'eliminazione di un percorso gateway con AWS CLI for App Mesh, consulta il [delete-gateway-route](#) comando nel [AWS CLI riferimento](#).

## Nodi virtuali

Un nodo virtuale agisce come un puntatore logico a un particolare gruppo di attività, ad esempio un servizio Amazon ECS o una distribuzione Kubernetes. Quando si crea un nodo virtuale, è necessario specificare un metodo di rilevamento dei servizi per il gruppo di attività. Qualsiasi traffico in entrata previsto dal nodo virtuale viene specificato come listener. Qualsiasi servizio virtuale a cui un nodo virtuale invia traffico in uscita viene specificato come backend.

I metadati di risposta per il tuo nuovo nodo virtuale contengono l'Amazon Resource Name (ARN) associato al nodo virtuale. Imposta questo valore come variabile di APPMESH\_RESOURCE\_ARN ambiente per il contenitore proxy Envoy del tuo gruppo di attività nella definizione

dell'attività di Amazon ECS o nella specifica del pod Kubernetes. Ad esempio, il valore potrebbe essere `arn:aws:appmesh:us-west-2:111122223333:mesh/myMesh/virtualNode/myVirtualNode`. Questo è quindi mappato ai parametri Envoy `node.id` e `node.cluster`. È necessario utilizzare `1.15.0` o una versione successiva dell'immagine Envoy quando si imposta questa variabile. Per ulteriori informazioni sulle variabili App Mesh Envoy, vedere [Envoy](#).

### Note

Per impostazione predefinita, App Mesh utilizza il nome della risorsa specificata in `APPMESH_RESOURCE_ARN` quando Envoy si riferisce a se stesso nei parametri e nelle tracce. È possibile ignorare questo comportamento impostando la variabile di ambiente `APPMESH_RESOURCE_CLUSTER` con il proprio nome.

## Creazione di un nodo virtuale

### AWS Management Console

Per creare un nodo virtuale utilizzando AWS Management Console

1. Apri la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).
2. Scegli la mesh in cui vuoi creare il nodo virtuale. Sono elencate tutte le mesh che possiedi e che sono state [condivise](#) con te.
3. Scegliere Virtual nodes (Nodi virtuali) nel riquadro di navigazione sinistro.
4. Scegli Crea nodo virtuale e quindi specifica le impostazioni per il tuo nodo virtuale.
5. Per Nome del nodo virtuale, inserisci un nome per il tuo nodo virtuale.
6. Per il metodo di individuazione del servizio, scegli una delle seguenti opzioni:
  - DNS: specifica il nome host DNS del servizio effettivo rappresentato dal nodo virtuale. Il proxy Envoy viene distribuito in un Amazon VPC. Il proxy invia le richieste di risoluzione dei nomi al server DNS configurato per il VPC. Se il nome host viene risolto, il server DNS restituisce uno o più indirizzi IP. Per ulteriori informazioni sulle impostazioni DNS del VPC, consulta [Uso del DNS con il VPC](#). Per il tipo di risposta DNS (opzionale), specifica i tipi di endpoint restituiti dal resolver DNS. Load Balancer significa che il resolver DNS restituisce un set di endpoint con carico bilanciato. Endpoints significa che il resolver DNS

sta restituendo tutti gli endpoint. Per impostazione predefinita, si presume che il tipo di risposta sia Load Balancer.

 Note

Se usi Route53, dovrai usare Load Balancer.

- **AWS Cloud Map**— Specificare un nome di servizio e uno spazio dei nomi HTTP esistenti. Facoltativamente, puoi anche specificare gli attributi che App Mesh può AWS Cloud Map richiedere selezionando Aggiungi riga e specificando una chiave e un valore. Vengono restituite solo le istanze che corrispondono a tutte le coppie chiave/valore specificate. Per poter essere utilizzato AWS Cloud Map, l'account deve avere il ruolo collegato al `AWSServiceRoleForAppMesh` [servizio](#). Per ulteriori informazioni su AWS Cloud Map, consulta la [Guida per sviluppatori di AWS Cloud Map](#).
- **Nessuno**: seleziona se il tuo nodo virtuale non prevede traffico in entrata.

## 7. Preferenza della versione IP

Controlla quale versione IP deve essere utilizzata per il traffico all'interno della mesh attivando il comportamento Override della versione IP predefinita. Per impostazione predefinita, App Mesh utilizza una varietà di versioni IP.

 Note

L'impostazione della preferenza IP sul nodo virtuale sostituisce solo la preferenza IP impostata per la mesh su questo nodo specifico.

- **Default (Predefinito)**
  - Il resolver DNS di Envoy IPv6 preferisce e ricorre a. IPv4
  - Utilizziamo l'IPv4 indirizzo restituito da, AWS Cloud Map se disponibile, e riprendiamo a utilizzare l'indirizzo. IPv6
  - L'endpoint creato per l'app locale utilizza un IPv4 indirizzo.
  - I listener Envoy si collegano a tutti gli indirizzi. IPv4
- **IPv6 (preferito)**
  - Il resolver DNS di Envoy preferisce e ricorre a. IPv6 IPv4

- L'IPv6 indirizzo restituito da AWS Cloud Map viene utilizzato se disponibile e si torna a utilizzare l'indirizzo IPv4
  - L'endpoint creato per l'app locale utilizza un IPv6 indirizzo.
  - Gli ascoltatori di Envoy si collegano a tutti gli indirizzi. IPv4 IPv6
  - IPv4 (preferito)
    - Il resolver DNS di Envoy preferisce e ricorre a. IPv4 IPv6
    - Utilizziamo l'IPv4 indirizzo restituito da, AWS Cloud Map se disponibile, e riprendiamo a utilizzare l'indirizzo. IPv6
    - L'endpoint creato per l'app locale utilizza un IPv4 indirizzo.
    - Gli ascoltatori di Envoy si collegano a tutti gli indirizzi. IPv4 IPv6
  - Solo IPv6
    - Il resolver DNS di Envoy utilizza solo. IPv6
    - Viene utilizzato solo l'IPv6 indirizzo restituito da. AWS Cloud Map Se AWS Cloud Map restituisce un IPv4 indirizzo, non viene utilizzato alcun indirizzo IP e all'Envoy vengono restituiti risultati vuoti.
    - L'endpoint creato per l'app locale utilizza un indirizzo. IPv6
    - Gli ascoltatori di Envoy si collegano a tutti gli indirizzi. IPv4 IPv6
  - Solo IPv4
    - Il resolver DNS di Envoy utilizza solo. IPv4
    - Viene utilizzato solo l'IPv4 indirizzo restituito da. AWS Cloud Map Se AWS Cloud Map restituisce un IPv6 indirizzo, non viene utilizzato alcun indirizzo IP e all'Envoy vengono restituiti risultati vuoti.
    - L'endpoint creato per l'app locale utilizza un indirizzo. IPv4
    - Gli ascoltatori di Envoy si collegano a tutti gli indirizzi. IPv4 IPv6
8. (Facoltativo) Impostazioni predefinite delle politiche del client: configura i requisiti predefiniti per la comunicazione con i servizi virtuali di backend.

 Note

- Se desideri abilitare Transport Layer Security (TLS) per un nodo virtuale esistente, ti consigliamo di creare un nuovo nodo virtuale, che rappresenti lo stesso servizio del nodo virtuale esistente, su cui abilitare TLS. Quindi sposta gradualmente il

traffico verso il nuovo nodo virtuale utilizzando un router e un percorso virtuali. Per ulteriori informazioni sulla creazione di un percorso e sulla regolazione dei pesi per la transizione, consulta [Route](#). Se aggiorni un nodo virtuale esistente che serve traffico con TLS, è possibile che i proxy Envoy del client downstream ricevano il contesto di convalida TLS prima che il proxy Envoy per il nodo virtuale che hai aggiornato riceva il certificato. Ciò può causare errori di negoziazione TLS sui proxy Envoy a valle.

- [L'autorizzazione proxy](#) deve essere abilitata per il proxy Envoy distribuito con l'applicazione rappresentata dai nodi virtuali del servizio di backend. Quando si abilita l'autorizzazione proxy, si consiglia di limitare l'accesso solo ai nodi virtuali con cui il nodo virtuale comunica.

- (Facoltativo) Seleziona Enforce TLS se desideri richiedere al nodo virtuale di comunicare con tutti i backend utilizzando Transport Layer Security (TLS).
- (Facoltativo) Se desideri richiedere l'uso di TLS solo per una o più porte specifiche, inserisci un numero in Porte. Per aggiungere porte aggiuntive, seleziona Aggiungi porta. Se non specifichi alcuna porta, TLS viene applicato per tutte le porte.
- Per Metodo di convalida, selezionate una delle seguenti opzioni. Il certificato specificato deve già esistere e soddisfare requisiti specifici. Per ulteriori informazioni, consulta [Requisiti del certificato](#).
  - AWS Private Certificate Authorityhosting: seleziona uno o più certificati esistenti. Per una descrizione completa e dettagliata end-to-end della distribuzione di una mesh con un'applicazione di esempio che utilizza la crittografia con un certificato ACM, consulta [Configurazione di TLS con Certificate Manager AWS on GitHub](#).
  - Hosting Envoy Secret Discovery Service (SDS): inserisci il nome del server segreto che Envoy recupererà utilizzando Secret Discovery Service.
  - Hosting di file locale: specifica il percorso del file della catena di certificati sul file system in cui è distribuito Envoy. Per una descrizione completa dell'implementazione di una mesh con un'applicazione di esempio che utilizza la crittografia con file locali, consulta [Configurazione di TLS con certificati TLS forniti da file su end-to-end GitHub](#).
- (Facoltativo) Inserisci un nome alternativo per il soggetto. Per aggiungere altre SAN, selezionare Aggiungi SAN. Le SAN devono essere in formato FQDN o URI.
- (Facoltativo) Seleziona Fornisci certificato client e una delle opzioni seguenti per fornire un certificato client quando un server lo richiede e abilitare l'autenticazione TLS reciproca.

Per ulteriori informazioni su Mutual TLS, consulta i documenti di [autenticazione TLS Mutual TLS](#) di App Mesh.

- Hosting Envoy Secret Discovery Service (SDS): inserisci il nome del segreto che Envoy recupererà utilizzando il Secret Discovery Service.
  - Hosting di file locale: specifica il percorso del file della catena di certificati, nonché la chiave privata, sul file system in cui è distribuito Envoy.
9. (Facoltativo) Backend del servizio: specifica il servizio virtuale App Mesh con cui comunicherà il nodo virtuale.
- Inserisci il nome del servizio virtuale App Mesh o il nome completo di Amazon Resource Name (ARN) per il servizio virtuale con cui comunica il tuo nodo virtuale.
  - (Facoltativo) Se desideri configurare impostazioni TLS uniche per un backend, seleziona Impostazioni TLS e quindi seleziona Ignora impostazioni predefinite.
  - (Facoltativo) Seleziona Enforce TLS se desideri richiedere al nodo virtuale di comunicare con tutti i backend tramite TLS.
  - (Facoltativo) Se desideri richiedere l'uso di TLS solo per una o più porte specifiche, inserisci un numero in Porte. Per aggiungere porte aggiuntive, seleziona Aggiungi porta. Se non specifichi alcuna porta, TLS viene applicato per tutte le porte.
  - Per Metodo di convalida, selezionate una delle seguenti opzioni. Il certificato specificato deve già esistere e soddisfare requisiti specifici. Per ulteriori informazioni, consulta [Requisiti del certificato](#).
    - AWS Private Certificate Authorityhosting: seleziona uno o più certificati esistenti.
    - Hosting Envoy Secret Discovery Service (SDS): inserisci il nome dell'inviato segreto che Envoy recupererà utilizzando il Secret Discovery Service.
    - Hosting di file locale: specifica il percorso del file della catena di certificati sul file system in cui è distribuito Envoy.
  - (Facoltativo) Inserite un nome alternativo per il soggetto. Per aggiungere altre SAN, selezionare Aggiungi SAN. Le SAN devono essere in formato FQDN o URI.
  - (Facoltativo) Seleziona Fornisci certificato client e una delle opzioni seguenti per fornire un certificato client quando un server lo richiede e abilitare l'autenticazione TLS reciproca. Per ulteriori informazioni su Mutual TLS, consulta i documenti di [autenticazione TLS Mutual TLS](#) di App Mesh.
    - Hosting Envoy Secret Discovery Service (SDS): inserisci il nome del segreto che Envoy recupererà utilizzando il Secret Discovery Service.

- **Hosting di file locale:** specifica il percorso del file della catena di certificati, nonché la chiave privata, sul file system in cui è distribuito Envoy.
- Per aggiungere backend aggiuntivi, seleziona **Aggiungi backend**.

## 10. (Facoltativo) Registrazione

Per configurare la registrazione, immettere il percorso dei log di accesso HTTP che deve essere utilizzato da Envoy. Ti consigliamo il `/dev/stdout` percorso in modo da poter utilizzare i driver di registro Docker per esportare i log di Envoy su un servizio come Amazon Logs. CloudWatch

### Note

I log devono ancora essere integrati da un agente nell'applicazione e inviati a una destinazione. Questo percorso file indica a Envoy dove inviare i log.

## 11. Configurazione del listener

Supporto HTTP per gli ascoltatori e HTTP/2 GRPC TCP protocolli. HTTPSnon è supportato.

- a. Se il nodo virtuale prevede traffico in entrata, specifica una porta e un protocollo per il listener. Il listener http consente la transizione della connessione ai websocket. È possibile fare clic su **Aggiungi ascoltatore** per aggiungere più ascoltatori. Il pulsante **Rimuovi** rimuoverà quel listener.
- b. (Facoltativo) **Abilita il pool di connessioni**

Il pool di connessioni limita il numero di connessioni che un Envoy può stabilire contemporaneamente con il cluster di applicazioni locale. Ha lo scopo di proteggere l'applicazione locale dal sovraccarico di connessioni e consente di adattare la modellazione del traffico alle esigenze delle applicazioni.

È possibile configurare le impostazioni del pool di connessioni sul lato di destinazione per un listener di nodi virtuali. App Mesh imposta le impostazioni del pool di connessioni lato client su `infinite` per impostazione predefinita, semplificando la configurazione della mesh.

**Note**

I protocolli ConnectionPool e PortMapping devono essere gli stessi. Se il protocollo del listener è tcp, specifica solo maxConnections. Se il protocollo del listener è grpc o http2, specifica solo maxRequests. Se il protocollo del listener è http, puoi specificare sia MaxConnections che maxPendingRequests

- Per Numero massimo di connessioni, specificare il numero massimo di connessioni in uscita.
  - (Facoltativo) Per il numero massimo di richieste in sospeso, specifica il numero di richieste in eccesso dopo il numero massimo di connessioni che un Envoy metterà in coda. Il valore di default è 2147483647.
- c. (Facoltativo) Abilita il rilevamento dei valori anomali

Il rilevamento dei valori anomali applicato al client Envoy consente ai client di intraprendere azioni quasi immediate sulle connessioni con guasti anomali noti osservati. È una forma di implementazione di un interruttore automatico che tiene traccia dello stato di salute dei singoli host nel servizio upstream.

Il rilevamento dei valori anomali determina dinamicamente se gli endpoint in un cluster upstream offrono prestazioni diverse dagli altri e li rimuove dal set di bilanciamento del carico integro.

**Note**

Per configurare in modo efficace il rilevamento dei valori anomali per un nodo virtuale del server, il metodo di individuazione dei servizi di quel nodo virtuale può essere uno AWS Cloud Map o il DNS con il campo del tipo di risposta impostato su. ENDPOINTS Se si utilizza il metodo di rilevamento del servizio DNS con tipo di risposta asLOADBALANCER, il proxy Envoy sceglierà un solo indirizzo IP per il routing verso il servizio upstream. Ciò annulla il comportamento di rilevamento dei valori anomali che comporta l'espulsione di un host non integro da un set di host. Consulta la sezione Metodo di rilevamento del servizio per maggiori dettagli sul comportamento del proxy Envoy in relazione al tipo di rilevamento del servizio.

- Per gli errori del server, specifica il numero di 5xx errori consecutivi necessari per l'espulsione.
  - Per Intervallo di rilevamento dei valori anomali, specificate l'intervallo di tempo e l'unità tra l'analisi del flusso di espulsione.
  - Per Durata dell'espulsione di Base, specificate la quantità di tempo e l'unità di base per cui viene espulso un host.
  - Per Percentuale di espulsione, specifica la percentuale massima di host nel pool di bilanciamento del carico che possono essere espulsi.
- d. (Facoltativo) Abilita il controllo sanitario: configura le impostazioni per una politica di controllo dello stato di salute.

Una politica di controllo dello stato di salute è facoltativa, ma se si specificano valori per una politica sanitaria, è necessario specificare i valori per Healthy threshold, Health check interval, Health check protocol, Timeout period e Unhealthy threshold.

- Per il protocollo Health check, scegli un protocollo. Se si seleziona grpc, il servizio deve essere conforme al [GRPC Health](#) Checking Protocol.
- Per Health check port (Porta del controllo dello stato), specificare la porta su cui deve essere eseguito il controllo dello stato.
- Per Healthy threshold (Soglia di integrità), specificare il numero di controlli dello stato andati a buon fine consecutivi che devono verificarsi prima di dichiarare il listener integro.
- Per Health check interval (Intervallo del controllo dello stato), specificare il periodo di tempo in millisecondi tra ogni esecuzione del controllo dello stato.
- Per Path (Percorso), specificare il percorso di destinazione per la richiesta di controllo dello stato. Questo valore viene utilizzato solo se il protocollo Health check è http o http2. Il valore viene ignorato per gli altri protocolli.
- In Timeout period (Periodo di timeout), specificare il periodo di tempo di attesa quando si riceve una risposta dal controllo dello stato, in millisecondi.
- Per Unhealthy threshold (Soglia di mancata integrità), specificare il numero di controlli dello stato non andati consecutivamente a buon fine che sono necessari per dichiarare il listener non integro.

- e. (Facoltativo) Abilita la terminazione TLS: configura il modo in cui altri nodi virtuali comunicano con questo nodo virtuale utilizzando TLS.
- Per Modalità, seleziona la modalità per cui desideri configurare TLS sul listener.
  - Per Metodo Certificate, selezionate una delle seguenti opzioni. Il certificato deve soddisfare requisiti specifici. Per ulteriori informazioni, consulta [Requisiti del certificato](#).
    - AWS Certificate Managerhosting: seleziona un certificato esistente.
    - Hosting Envoy Secret Discovery Service (SDS): inserisci il nome dell'Envoy segreto che Envoy recupererà utilizzando il Secret Discovery Service.
    - Hosting di file locale: specifica il percorso del file della catena di certificati, nonché la chiave privata, sul file system in cui è distribuito il proxy Envoy.
  - (Facoltativo) Seleziona Richiedi certificati client e una delle opzioni seguenti per abilitare l'autenticazione TLS reciproca quando un client fornisce un certificato. Per ulteriori informazioni su Mutual TLS, consulta i documenti di [autenticazione TLS Mutual TLS](#) di App Mesh.
    - Hosting Envoy Secret Discovery Service (SDS): inserisci il nome del segreto che Envoy recupererà utilizzando il Secret Discovery Service.
    - Hosting di file locale: specifica il percorso del file della catena di certificati sul file system in cui è distribuito Envoy.
  - (Facoltativo) Inserite un nome alternativo per il soggetto. Per aggiungere altre SAN, selezionare Aggiungi SAN. Le SAN devono essere in formato FQDN o URI.
- f. (Facoltativo) Timeout

 Note

Se specifichi un timeout maggiore di quello predefinito, assicurati di configurare un router virtuale e un percorso con un timeout maggiore di quello predefinito. Tuttavia, se riduci il timeout a un valore inferiore a quello predefinito, è facoltativo aggiornare i timeout su Route. [Per ulteriori informazioni, consulta Routes.](#)

- Timeout della richiesta: puoi specificare un timeout per la richiesta se hai selezionato grpc, http o http2 come protocollo del listener. L'impostazione predefinita è 15 secondi. Il valore di 0 disabilita il timeout.

- Durata di inattività: è possibile specificare una durata di inattività per qualsiasi protocollo di ascolto. Il valore predefinito è 300 secondi.

12. Scegli Crea nodo virtuale per terminare.

## AWS CLI

Per creare un nodo virtuale utilizzando AWS CLI.

Crea un nodo virtuale che utilizzi DNS per l'individuazione dei servizi utilizzando il seguente comando e un file JSON di input (sostituisci i valori *rossi* con i tuoi):

1. 

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

2. Contenuto dell'esempio `.json` `create-virtual-node-dns`:

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

3. Output di esempio:

```
{
 "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"
 }
}
```

Per ulteriori informazioni sulla creazione di un nodo virtuale con AWS CLI for App Mesh, vedere il [create-virtual-node](#) comando nel AWS CLI riferimento.

## Eliminazione di un nodo virtuale

### Note

Non è possibile eliminare un nodo virtuale se è specificato come destinazione in qualsiasi [route](#) o come provider in qualsiasi [servizio virtuale](#).

### AWS Management Console

Per eliminare un nodo virtuale utilizzando il AWS Management Console

1. Apri la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).
2. Scegli la mesh da cui vuoi eliminare un nodo virtuale. Sono elencate tutte le mesh che possiedi e che sono state [condivise](#) con te.
3. Scegliere Virtual nodes (Nodi virtuali) nel riquadro di navigazione sinistro.
4. Nella tabella Nodi virtuali, scegli il nodo virtuale che desideri eliminare e seleziona Elimina. Per eliminare un nodo virtuale, l'ID dell'account deve essere elencato nelle colonne Proprietario della rete Mesh o Proprietario della risorsa del nodo virtuale.
5. Nella casella di conferma, digita **delete** e seleziona Elimina.

### AWS CLI

Per eliminare un nodo virtuale utilizzando AWS CLI

1. Usa il seguente comando per eliminare il tuo nodo virtuale (sostituisci i valori **rossi** con i tuoi):

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

2. Output di esempio:

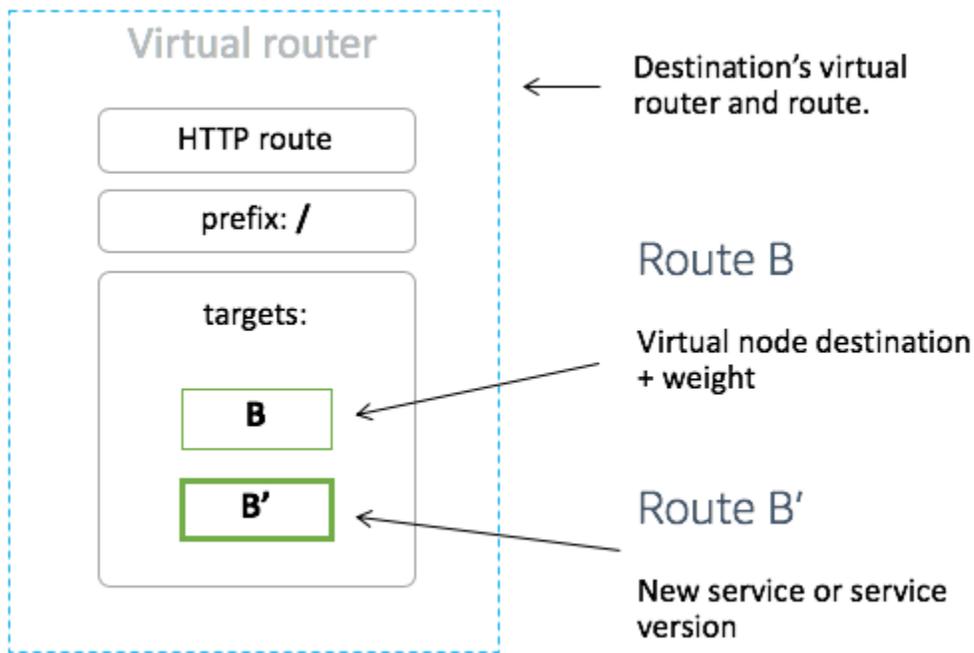
```
{
 "virtualNode": {
 "meshName": "meshName",
 "metadata": {
```

```
 "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"
}
}
```

Per ulteriori informazioni sull'eliminazione di un nodo virtuale con AWS CLI for App Mesh, vedere il [delete-virtual-node](#) comando nel AWS CLI riferimento.

## Router virtuali

I router virtuali gestiscono il traffico per uno o più servizi virtuali all'interno della mesh. Dopo aver creato un router virtuale, è possibile creare e associarvi route che indirizzano le richieste in entrata a diversi nodi virtuali.



Qualsiasi traffico in entrata previsto dal router virtuale deve essere specificato come listener.

## Creazione di un router virtuale

### AWS Management Console

Per creare un router virtuale utilizzando AWS Management Console

#### Note

Quando si crea un router virtuale, è necessario aggiungere un selettore di namespace con un'etichetta per identificare l'elenco dei namespace per associare i percorsi al router virtuale creato.

1. Apri la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).
2. Scegliere la mesh in cui si desidera creare il router virtuale. Vengono elencate tutte le mesh che possiedi e che sono state [condivise](#) con te.
3. Nel riquadro di navigazione sinistro, scegliere Virtual routers (Router virtuali).
4. Scegliere Create virtual router (Crea router virtuali).

5. In Virtual router name (Nome router virtuale), specificare un nome per il router virtuale. Il nome può contenere un massimo di 255 lettere, numeri, trattini e caratteri di sottolineatura.
6. (Facoltativo) Per la configurazione del listener, specifica una porta e un protocollo per il router virtuale. Il `http` listener consente la transizione della connessione ai websocket. È possibile fare clic su Aggiungi ascoltatore per aggiungere più ascoltatori. Il pulsante Rimuovi rimuoverà quel listener.
7. Scegliere Create virtual router (Crea router virtuale) per terminare.

## AWS CLI

Per creare un router virtuale utilizzando ilAWS CLI.

Crea un router virtuale usando il seguente comando e inserisci JSON (sostituisci i valori *rossi* con i tuoi):

1. 

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

2. Contenuto dell'esempio create-virtual-router .json

3. 

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

4. Output di esempio:

```
{
 "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"
}
```

Per ulteriori informazioni sulla creazione di un router virtuale con AWS CLI for App Mesh, consulta il [create-virtual-router](#) comando nel [AWS CLI riferimento](#).

## Eliminazione di un router virtuale

### Note

Non è possibile eliminare un router virtuale se ha [percorsi](#) o se è specificato come provider per qualsiasi [servizio virtuale](#).

## AWS Management Console

Per eliminare un router virtuale utilizzando AWS Management Console

1. Apri la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).
2. Scegli la mesh da cui desideri eliminare un router virtuale. Vengono elencate tutte le mesh che possiedi e che sono state [condivise](#) con te.
3. Nel riquadro di navigazione sinistro, scegliere Virtual routers (Router virtuali).
4. Nella tabella Router virtuali, scegli il router virtuale che desideri eliminare e seleziona Elimina nell'angolo in alto a destra. Per eliminare un router virtuale, l'ID dell'account deve essere elencato nelle colonne Mesh owner o Resource owner del router virtuale.
5. Nella casella di conferma, digita **delete** e fai clic su Elimina.

## AWS CLI

Per eliminare un router virtuale utilizzando AWS CLI

1. Usa il seguente comando per eliminare il tuo router virtuale (sostituisci i valori *rossi* con i tuoi):

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

2. Output di esempio:

```
{
 "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"
 }
}
```

Per ulteriori informazioni sull'eliminazione di un router virtuale con AWS CLI for App Mesh, consulta il [delete-virtual-router](#) comando nel [AWS CLI riferimento](#).

## Route

Una route è associata a un router virtuale. Il percorso viene utilizzato per abbinare le richieste del router virtuale e per distribuire il traffico ai nodi virtuali associati. Se una route corrisponde a una richiesta, può distribuire il traffico a uno o più nodi virtuali di destinazione. È possibile specificare la ponderazione relativa per ogni nodo virtuale. Questo argomento consente di utilizzare route in una mesh dei servizi.

### Creazione di un routing

#### AWS Management Console

Per creare un route mediante l'AWS Management Console

1. Apri la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).
2. Scegliere la mesh in cui si desidera creare il routing. Vengono elencate tutte le mesh che possiedi e che sono state [condivise](#) con te.
3. Nel riquadro di navigazione sinistro, scegliere Virtual routers (Router virtuali).
4. Scegliere il router virtuale a cui si desidera associare un nuovo routing. Se l'elenco risulta vuoto, è necessario [creare prima un router virtuale](#).

5. Nella tabella Routes (Route), scegliere Create route (Crea route). Per creare un percorso, l'ID dell'account deve essere indicato come proprietario della risorsa del percorso.
6. In Route name (Nome route), specificare il nome da utilizzare per la route.
7. Per Tipo di percorso, scegli il protocollo che desideri indirizzare. Il protocollo selezionato deve corrispondere al protocollo del listener selezionato per il router virtuale e al nodo virtuale verso cui indirizzare il traffico.
8. (Facoltativo) Per la priorità del percorso, specifica una priorità compresa tra 0 e 1000 da utilizzare per il percorso. Le route vengono messe in corrispondenza in base al valore specificato, dove 0 è la priorità più alta.
9. (Facoltativo) Scegli Configurazione aggiuntiva. Dai protocolli in basso, scegli il protocollo selezionato per il tipo di percorso e specifica le impostazioni nella console come desideri.
10. Per la configurazione di Target, seleziona il nodo virtuale App Mesh esistente verso cui indirizzare il traffico e specifica un peso. Puoi scegliere Aggiungi obiettivo per aggiungere altri obiettivi. Sommati, la percentuale per tutti gli obiettivi deve raggiungere fino a 100. Se l'elenco risulta vuoto, è necessario [crearne](#) uno prima. Se il nodo virtuale selezionato ha più listener, è necessaria la porta Target.
11. Per la configurazione Match, specifica:

La configurazione della partita non è disponibile per *tcp*

- Se http/http2 è il tipo selezionato:
  - (Facoltativo) Metodo - specifica l'intestazione del metodo da abbinare nelle richieste http/http2 in entrata.
  - (Facoltativo) Corrispondenza porta - Corrisponde alla porta per il traffico in entrata. La corrispondenza delle porte è necessaria se questo router virtuale ha più listener.
  - (Facoltativo) Percorso prefisso/esatto/regex - metodo di corrispondenza del percorso dell'URL.
    - Prefix match: una richiesta corrispondente da parte di un gateway viene riscritta con il nome del servizio virtuale di destinazione e il prefisso corrispondente viene riscritto/, per impostazione predefinita. A seconda di come configuri il servizio virtuale, potrebbe utilizzare un router virtuale per indirizzare la richiesta a diversi nodi virtuali, in base a prefissi o intestazioni specifici.

**Note**

Se abiliti la corrispondenza basata su Path/Prefix, App Mesh abilita la normalizzazione del percorso ([normalize\\_path](#) e [merge\\_slashes](#)) per ridurre al minimo la probabilità di vulnerabilità legate alla confusione dei percorsi. Le vulnerabilità legate alla confusione dei percorsi si verificano quando le parti che partecipano alla richiesta utilizzano rappresentazioni di percorso diverse.

- Corrispondenza esatta - il parametro esatto disabilita la corrispondenza parziale per un percorso e si assicura che restituisca il percorso solo se il percorso corrisponde ESATTAMENTE all'URL corrente.
- Regex match - utilizzato per descrivere modelli in cui più URL possono effettivamente identificare una singola pagina del sito Web.
- (Facoltativo) Parametri di interrogazione: questo campo consente di confrontare i parametri dell'interrogazione.
- (Facoltativo) Headers - specifica le intestazioni per http e http2. Dovrebbe corrispondere alla richiesta in entrata da indirizzare al servizio virtuale di destinazione.
- Se grpc è il tipo selezionato:
  - Nome del servizio - il servizio di destinazione per il quale soddisfare la richiesta.
  - Nome del metodo: il metodo di destinazione per il quale soddisfare la richiesta.
  - (Facoltativo) Metadati - specifica inMatch base alla presenza di metadati. Tutti devono corrispondere affinché la richiesta possa essere elaborata.

## 12. Seleziona Crea percorso.

### AWS CLI

Per creare un percorso utilizzando ilAWS CLI.

Crea un percorso gRPC usando il seguente comando e inserisci JSON (sostituisci i valori *rossi* con i tuoi):

1. 

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

## 2. Contenuto dell'esempio 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. Output di esempio:

```

{
 "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"
}
}

```

Per ulteriori informazioni sulla creazione di un percorso con AWS CLI for App Mesh, consulta il comando [create-route](#) nel [AWS CLI riferimento](#).

## gRPC

### (Facoltativo) Partita

- (Facoltativo) Inserire il nome del servizio di destinazione per cui corrispondere alla richiesta. Se non specifichi un nome, le richieste a qualsiasi servizio corrispondono.
- (Facoltativo) Inserire il nome del metodo di destinazione per il quale corrispondere alla richiesta. Se non specifichi un nome, le richieste a qualsiasi metodo corrispondono. Se si specifica un nome di metodo, è necessario specificare un nome di servizio.

### (Facoltativo) Metadati

Seleziona Add metadata (Aggiungi metadati).

- (Facoltativo) Inserisci il nome dei metadati in base al quale desideri indirizzare, seleziona un tipo di corrispondenza e inserisci un valore di corrispondenza. Selezionando Inverti si otterrà l'effetto

opposto. Ad esempio, se specifichi un nome di metadati `dimyMetadata`, un tipo `Match Exact`, un valore `Match` di `123` e selezioni `Inverti`, il percorso viene confrontato per ogni richiesta il cui nome di metadati inizia con qualcosa di diverso da `123`.

- (Facoltativo) Seleziona `Aggiungi metadati` per aggiungere fino a dieci elementi di metadati.

#### (Facoltativo) Politica relativa ai tentativi

Una policy per i nuovi tentativi consente ai client di proteggersi da guasti di rete intermittenti o da guasti lato server intermittenti. Una policy per la ripetizione dei tentativi è facoltativa, ma è consigliata. I valori di timeout dei tentativi definiscono il timeout per tentativo di nuovo tentativo (incluso il tentativo iniziale). Se non definisci una politica per i nuovi tentativi, App Mesh può creare automaticamente una politica predefinita per ciascuno dei tuoi percorsi. Per ulteriori informazioni, consulta [Politica predefinita per i nuovi tentativi di routing](#).

- Per il timeout `Riprova`, inserisci il numero di unità per la durata del timeout. È necessario un valore se si seleziona un evento di nuovo tentativo del protocollo.
- Per l'unità di timeout `Retry`, selezionare un'unità. È necessario un valore se si seleziona un evento di nuovo tentativo del protocollo.
- Per `Numero massimo di tentativi`, inserisci il numero massimo di tentativi quando la richiesta ha esito negativo. È necessario un valore se si seleziona un evento di nuovo tentativo del protocollo. Consigliamo un valore minimo di due.
- Seleziona uno o più eventi HTTP di nuovo tentativo. Si consiglia di selezionare almeno `stream-error` e `gateway-error`.
- Seleziona un evento di nuovo tentativo TCP.
- Seleziona uno o più eventi di riprova gRPC. Ti consigliamo di selezionare almeno le opzioni `cancellate` e `non disponibili`.

#### Timeout (facoltativo)

- Il valore predefinito è 15 secondi. Se hai specificato un criterio `Riprova`, la durata specificata qui deve essere sempre maggiore o uguale alla durata dei tentativi moltiplicata per il numero massimo di tentativi che hai definito nel criterio `Riprova` in modo che il criterio dei tentativi possa essere completato. Se specifichi una durata superiore a 15 secondi, assicurati che anche il timeout specificato per il listener di qualsiasi nodo virtuale `Target` sia maggiore di 15 secondi. Per ulteriori informazioni, consulta [Nodi virtuali](#).
- Il valore di `0` disabilita il timeout.

- Il tempo massimo durante il quale il routing può rimanere inattivo.

## HTTP e HTTP/2

### (Facoltativo) Partita

- Specifica il prefisso a cui deve corrispondere il percorso. Ad esempio, se il nome del servizio virtuale è `service-b.local` e desideri che la route corrisponda alle richieste in `service-b.local/metrics`, il prefisso deve essere `/metrics`. Specificare i percorsi di tutto il traffico.
- (Facoltativo) Seleziona un metodo.
- (Facoltativo) Seleziona uno schema. Applicabile solo per le route HTTP2.

### Intestazioni (facoltative)

- (Facoltativo) Seleziona Aggiungi intestazione. Inserisci il nome dell'intestazione in base al quale desideri indirizzare, seleziona un tipo di corrispondenza e inserisci un valore di corrispondenza. Selezionando Inverti si otterrà l'effetto opposto. Ad esempio, se si specifica un'intestazione denominata `clientRequestId` con il prefisso di `123` e si seleziona Inverti, la rotta viene trovata per qualsiasi richiesta la cui intestazione inizia con un valore diverso da `123`.
- (Facoltativo) Seleziona Aggiungi intestazione. È possibile aggiungere fino a dieci intestazioni.

### (Facoltativo) Politica relativa ai tentativi

Una policy per i nuovi tentativi consente ai client di proteggersi da guasti di rete intermittenti o da guasti lato server intermittenti. Una policy per la ripetizione dei tentativi è facoltativa, ma è consigliata. I valori di timeout dei tentativi definiscono il timeout per tentativo di nuovo tentativo (incluso il tentativo iniziale). Se non definisci una politica per i nuovi tentativi, App Mesh può creare automaticamente una politica predefinita per ciascuno dei tuoi percorsi. Per ulteriori informazioni, consulta [Politica predefinita per i nuovi tentativi di routing](#).

- Per il timeout Riprova, inserisci il numero di unità per la durata del timeout. È necessario un valore se si seleziona un evento di nuovo tentativo del protocollo.
- Per l'unità di timeout Retry, selezionare un'unità. È necessario un valore se si seleziona un evento di nuovo tentativo del protocollo.

- Per Numero massimo di tentativi, inserisci il numero massimo di tentativi quando la richiesta ha esito negativo. È necessario un valore se si seleziona un evento di nuovo tentativo del protocollo. Consigliamo un valore minimo di due.
- Seleziona uno o più eventi HTTP di nuovo tentativo. Si consiglia di selezionare almeno stream-error e gateway-error.
- Seleziona un evento di nuovo tentativo TCP.

#### Timeout (facoltativo)

- Timeout della richiesta: l'impostazione predefinita è 15 secondi. Se hai specificato un criterio Riprova, la durata specificata qui deve essere sempre maggiore o uguale alla durata dei tentativi moltiplicata per il numero massimo di tentativi che hai definito nel criterio Riprova in modo che il criterio dei tentativi possa essere completato.
- Durata inattività: il valore predefinito è 300 secondi.
- Il valore di 0 disabilita il timeout.

#### Note

Se specifichi un timeout maggiore di quello predefinito, assicurati che anche il timeout specificato per il listener per tutti i partecipanti al nodo virtuale sia maggiore di quello predefinito. Tuttavia, se riduci il timeout a un valore inferiore a quello predefinito, è facoltativo aggiornare i timeout nei nodi virtuali. Per ulteriori informazioni, consulta [Nodi virtuali](#).

## TCP

#### Timeout (facoltativo)

- Durata inattività: il valore predefinito è 300 secondi.
- Il valore di 0 disabilita il timeout.

## Eliminazione di un routing

### AWS Management Console

Per eliminare una route mediante l'AWS Management Console

1. Apri la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).
2. Scegliere la mesh da cui si desidera eliminare un routing. Vengono elencate tutte le mesh che possiedi e che sono state [condivise](#) con te.
3. Nel riquadro di navigazione sinistro, scegliere Virtual routers (Router virtuali).
4. Scegliere il router da cui si desidera eliminare un routing.
5. Nella tabella Percorsi, scegli il percorso che desideri eliminare e seleziona Elimina nell'angolo in alto a destra.
6. Nella casella di conferma, digita **delete** e fai clic su Elimina.

### AWS CLI

Per eliminare una route mediante l'AWS CLI

1. Usa il seguente comando per eliminare il tuo percorso (sostituisci i valori *rossi* con i tuoi):

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

2. Output di esempio:

```
{
 "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
```

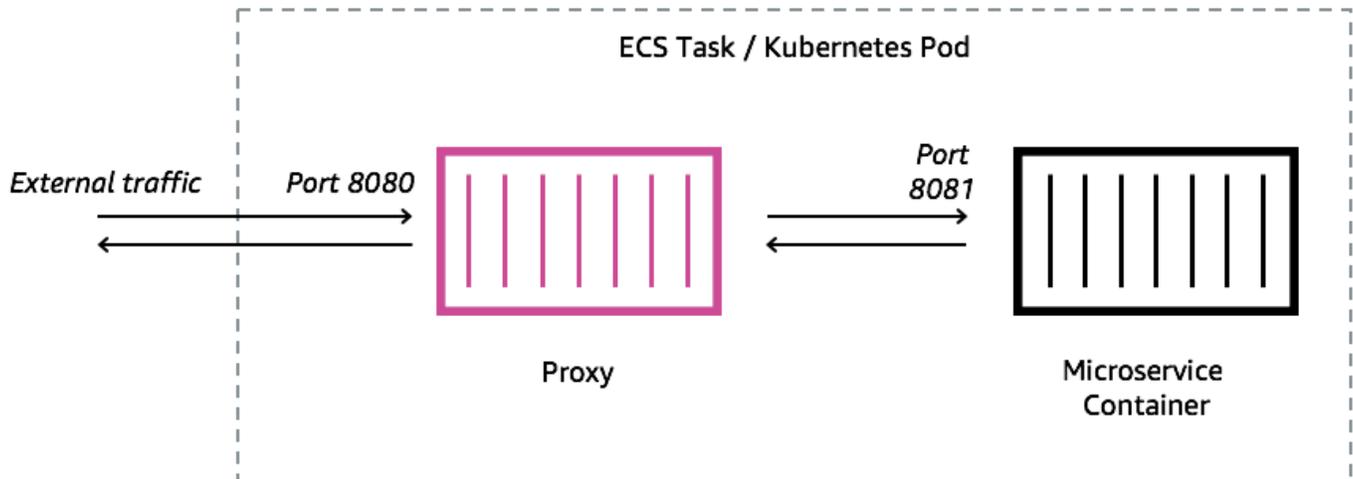
```
},
"routeName": "routeName",
"spec": {
 "grpcRoute": {
 "action": {
 "weightedTargets": [
 {
 "virtualNode": "nodeName",
 "weight": 100
 }
]
 },
 "match": {
 "metadata": [
 {
 "invert": false,
 "match": {
 "prefix": "123"
 },
 "name": "myMetadata"
 }
],
 "methodName": "methodName",
 "serviceName": "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"
 }
}
```

Per ulteriori informazioni sull'eliminazione di un percorso con AWS CLI for App Mesh, consulta il comando [delete-route](#) nel [AWS CLI riferimento](#).

# Immagine dell'invitato

AWS App Mesh è una service mesh basata sul proxy [Envoy](#).



Devi aggiungere un proxy Envoy all'ECSattività Amazon, al pod Kubernetes o all'EC2istanza Amazon rappresentata dal tuo endpoint App Mesh, ad esempio un nodo virtuale o un gateway virtuale. App Mesh fornisce un'immagine del contenitore proxy Envoy corredata con gli ultimi aggiornamenti di vulnerabilità e prestazioni. App Mesh verifica ogni nuova versione del proxy Envoy rispetto al set di funzionalità App Mesh prima di rendere disponibile una nuova immagine.

## Varianti dell'immagine Envoy

App Mesh fornisce due varianti dell'immagine del contenitore proxy Envoy. Le differenze tra i due riguardano il modo in cui il proxy Envoy comunica con il piano dati App Mesh e il modo in cui i proxy Envoy comunicano tra loro. Una è un'immagine standard, che comunica con gli endpoint del servizio App Mesh standard. L'altra variante è FIPS -compliant, che comunica con gli endpoint del FIPS servizio App Mesh e applica la crittografia FIPS nelle comunicazioni TLS tra i servizi App Mesh.

[Puoi scegliere un'immagine regionale dall'elenco seguente o un'immagine dal nostro archivio pubblico denominato `aws-appmesh-envoy`](#)

**⚠ Important**

- A partire dal 30 giugno 2023, solo l'immagine Envoy v1.17.2.0-prod o successiva è compatibile per l'uso con App Mesh. Per i clienti attuali che utilizzavano un'immagine Envoy in precedenza v1.17.2.0, anche se gli envoy esistenti continueranno a essere compatibili, consigliamo vivamente di migrare alla versione più recente.
- Come best practice, si consiglia vivamente di aggiornare regolarmente la versione di Envoy alla versione più recente. Solo l'ultima versione di Envoy è convalidata con le patch di sicurezza, le versioni di funzionalità e i miglioramenti delle prestazioni più recenti.
- La versione 1.17 era un aggiornamento significativo di Envoy. Vedi [Aggiornamento/migrazione a Envoy 1.17 per maggiori dettagli](#).
- La versione o successiva è compatibile. 1.20.0.1 ARM64
- Per il IPv6 supporto, è richiesta la versione Envoy 1.20 o successiva.

Tutte le regioni [supportate](#) diverse da me-south-1, ap-east-1, ap-southeast-3, eu-south-1, il-central-1, e af-south-1. È possibile sostituire *Region-code* con qualsiasi regione diversa da me-south-1, ap-east-1, ap-southeast-3, eu-south-1, il-central-1, e af-south-1.

**Standard**

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

**FIPS-conforme**

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

**me-south-1****Standard**

```
772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

**FIPS-conforme**

```
772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

## ap-east-1

### Standard

```
856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

### FIPS-conforme

```
856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

## ap-southeast-3

### Standard

```
909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

### FIPS-conforme

```
909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

## eu-south-1

### Standard

```
422531588944.dkr.ecr.eu-south-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

### FIPS-conforme

```
422531588944.dkr.ecr.eu-south-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

## il-central-1

### Standard

```
564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

### FIPS-conforme

```
564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

## af-south-1

### Standard

```
924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

### FIPS-conforme

```
924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

## Public repository

### Standard

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.29.6.1-prod
```

### FIPS-conforme

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.29.6.1-prod-fips
```

#### Note

Si consiglia di allocare 512 CPU unità e almeno 64 MiB di memoria nel contenitore Envoy. Su Fargate la quantità di memoria più bassa che è possibile impostare è di 1024 MiB di memoria. L'allocazione delle risorse nel contenitore Envoy può essere aumentata se Container Insights o altre metriche indicano risorse insufficienti a causa di un carico maggiore.

#### Note

Tutte le versioni di rilascio `aws-appmesh-envoy` dell'immagine a partire da `v1.22.0.0` sono create come immagini Docker senza distribuzione. Abbiamo apportato questa modifica in modo da poter ridurre le dimensioni dell'immagine e ridurre la nostra esposizione alla

vulnerabilità nei pacchetti inutilizzati presenti nell'immagine. Se stai costruendo sulla base dell' `aws-appmesh-envoy` immagine e ti affidi ad alcuni pacchetti di AL2 base (ad esempio `yum`) e funzionalità, ti suggeriamo di copiare i binari dall'interno di un'`aws-appmesh-envoy` immagine per creare una nuova immagine Docker con base AL2. Esegui questo script per generare un'immagine docker personalizzata con il tag `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 .
```

L'accesso a questa immagine del contenitore in Amazon ECR è controllato da AWS Identity and Access Management (IAM). Di conseguenza, devi utilizzare IAM per verificare di avere accesso in lettura ad Amazon ECR. Ad esempio, quando usi Amazon ECS, puoi assegnare un ruolo di esecuzione delle attività appropriato a un'attività Amazon ECS. Se utilizzi IAM politiche che limitano l'accesso a risorse Amazon specifiche, assicurati di consentire l'accesso all'Amazon Resource Name (ARN) specifico della regione che identifica il `aws-appmesh-envoy` repository. Ad esempio, nella `us-west-2` regione, consenti l'accesso alla seguente risorsa: `arn:aws:ecr:us-west-2:840364872350:repository/aws-appmesh-envoy`. Per ulteriori informazioni, consulta [Amazon ECR Managed Policies](#). Se utilizzi Docker su un'istanza Amazon EC2, autentica Docker nel repository. Per ulteriori informazioni, consultare [Autenticazione dei registri](#).

Di tanto in tanto rilasciamo nuove funzionalità di App Mesh che dipendono dalle modifiche di Envoy che non sono ancora state unite alle immagini di Envoy upstream. Per utilizzare queste

nuove funzionalità di App Mesh prima che le modifiche di Envoy vengano unite a monte, è necessario utilizzare l'immagine del contenitore App Mesh venduto con Envoy. Per un elenco delle modifiche, consulta i [problemi della GitHub roadmap di App Mesh](#) con l'Envoy Upstream etichetta. Ti consigliamo di utilizzare l'immagine del contenitore App Mesh Envoy come opzione meglio supportata.

## Variabili di configurazione di Envoy

Utilizza le seguenti variabili di ambiente per configurare i contenitori Envoy per i gruppi di attività del nodo virtuale App Mesh.

### Note

App Mesh Envoy 1.17 non supporta gli XD v2 di Envoy. API. Se utilizzi [variabili di configurazione Envoy che accettano i file di configurazione di Envoy](#), devono essere aggiornate agli XD v3 più recenti. API

## Variabili obbligatorie

La seguente variabile di ambiente è richiesta per tutti i contenitori App Mesh Envoy. Questa variabile può essere utilizzata solo con la versione 1.15.0 o successiva dell'immagine Envoy. Se stai usando una versione precedente dell'immagine, devi invece impostare la APPMESH\_VIRTUAL\_NODE\_NAME variabile.

### APPMESH\_RESOURCE\_ARN

Quando aggiungi il contenitore Envoy a un gruppo di attività, imposta questa variabile ARN di ambiente sul nodo virtuale o sul gateway virtuale rappresentato dal gruppo di attività. L'elenco seguente contiene esempi: ARNs

- Nodo virtuale — `arn:aws:appmesh:Region-code:111122223333:mesh/meshName/virtualNode/virtualNodeName`
- Gateway virtuale — `arn:aws:appmesh:Region-code:111122223333:mesh/meshName/virtualGateway/virtualGatewayName`

Quando si utilizza l'[App Mesh Preview Channel](#), ARNs è necessario utilizzare il `us-west-2` Regione e `utilizzoappmesh-preview`, anziché `appmesh`. Ad esempio, il valore ARN di

un nodo virtuale nell'App Mesh Preview Channel è `arn:aws:appmesh-preview:us-west-2:111122223333:mesh/meshName/virtualNode/virtualNodeName`.

## Variabili opzionali

La seguente variabile di ambiente è facoltativa per i contenitori App Mesh Envoy.

### ENVOY\_LOG\_LEVEL

Specifica il livello di registro per il contenitore Envoy.

Valori validi: `trace`, `debug`, `info`, `warn`, `error`, `critical`, `off`

Impostazione predefinita: `info`

### ENVOY\_INITIAL\_FETCH\_TIMEOUT

Specifica il periodo di attesa da parte di Envoy per la prima risposta di configurazione dal server di gestione durante il processo di inizializzazione.

Per ulteriori informazioni, vedere Fonti di [configurazione](#) nella documentazione di Envoy. Se impostato su `0`, non c'è alcun timeout.

Impostazione predefinita: `0`

### ENVOY\_CONCURRENCY

Imposta l'opzione della `--concurrency` riga di comando all'avvio di Envoy. Questa opzione non è impostata di default. Questa opzione è disponibile dalla versione Envoy `v1.24.0.0-prod` o successiva.

Per ulteriori informazioni, vedere [Opzioni della riga di comando nella documentazione](#) di Envoy.

## Variabili di amministrazione

Usa queste variabili di ambiente per configurare l'interfaccia amministrativa di Envoy.

### ENVOY\_ADMIN\_ACCESS\_PORT

Specificate una porta di amministrazione personalizzata su cui Envoy possa ascoltare. Default: `9901`.

**Note**

La porta di amministrazione di Envoy deve essere diversa da qualsiasi porta listener sul gateway virtuale o sul nodo virtuale

**ENVOY\_ADMIN\_ACCESS\_LOG\_FILE**

Specificate un percorso personalizzato su cui scrivere i log di accesso di Envoy. Default: `/tmp/envoy_admin_access.log`.

**ENVOY\_ADMIN\_ACCESS\_ENABLE\_IPV6**

Attiva l'interfaccia di amministrazione di Envoy per accettare il traffico, il che consente a questa interfaccia di accettare sia IPv6 il traffico che il traffico IPv4. Per impostazione predefinita, questo flag è impostato su `false` e Envoy ascolta solo il traffico IPv4. Questa variabile può essere utilizzata solo con la versione 1.22.0 o successiva dell'immagine di Envoy.

## Variabili dell'agente

Utilizzate queste variabili di ambiente per configurare l' AWS App Mesh Agent for Envoy. Per ulteriori informazioni, consulta App Mesh [Agent for Envoy](#).

**APPNET\_ENVOY\_RESTART\_COUNT**

Specifica il numero di volte in cui l'agente riavvia il processo proxy Envoy all'interno di un'attività o di un pod in esecuzione, se viene chiuso. L'agente registra inoltre lo stato di uscita ogni volta che Envoy esce per facilitare la risoluzione dei problemi. Il valore predefinito di questa variabile è `0`. Quando viene impostato il valore predefinito, l'agente non tenta di riavviare il processo.

Impostazione predefinita: `0`

Massimo: `10`

**PID\_POLL\_INTERVAL\_MS**

Specifica l'intervallo in millisecondi in cui lo stato del processo del proxy Envoy viene verificato dall'agente. Il valore predefinito è `100`.

Impostazione predefinita: `100`

Minimo: `100`

Massimo: 1000

#### LISTENER\_DRAIN\_WAIT\_TIME\_S

Specifica il periodo di tempo in secondi in cui il proxy Envoy attende la chiusura delle connessioni attive prima della chiusura del processo.

Impostazione predefinita: 20

Minimo: 5

Massimo: 110

#### APPNET\_AGENT\_ADMIN\_MODE

Avvia il server dell'interfaccia di gestione di Agent e lo associa a un indirizzo tcp o a un socket unix.

Valori validi: tcp, uds

#### APPNET\_AGENT\_HTTP\_PORT

Specificare una porta da utilizzare per associare l'interfaccia di gestione di Agent in modalità tcp. Assicurati che il valore della porta sia > 1024 e != 0. Assicurati che la porta sia inferiore a 65535.

Impostazione predefinita: 9902

#### APPNET\_AGENT\_ADMIN\_UDS\_PATH

Specificare il percorso del socket del dominio unix per l'interfaccia di gestione dell'agente in uds modalità.

Impostazione predefinita: /var/run/ecs/appnet\_admin.sock

## Variabili di tracciamento

È possibile configurare nessuno o uno dei seguenti driver di tracciamento.

### AWS X-Ray variabili

Usa le seguenti variabili di ambiente per configurare App Mesh con AWS X-Ray. Per ulteriori informazioni, consulta la [Guida per gli sviluppatori di AWS X-Ray](#).

## ENABLE\_ENVOY\_XRAY\_TRACING

Abilita il tracciamento a raggi X utilizzando `127.0.0.1:2000` come endpoint demone predefinito. Per abilitare, imposta il valore su `1`. Il valore predefinito è `0`.

## XRAY\_DAEMON\_PORT

Specificate un valore di porta per sovrascrivere la porta demone X-Ray predefinita: `2000`

## XRAY\_SAMPLING\_RATE

Specificate una frequenza di campionamento per sostituire la frequenza di campionamento predefinita del tracciante X-Ray (5%). `0.05` Specificate il valore come valore decimale compreso tra `0` e `1.00` (100%). Questo valore viene sovrascritto se specificato.

`XRAY_SAMPLING_RULE_MANIFEST` Questa variabile è supportata con le immagini di Envoy della versione e successive. `v1.19.1.1-prod`

## XRAY\_SAMPLING\_RULE\_MANIFEST

Specificate un percorso del file nel file system del contenitore Envoy per configurare le regole di campionamento personalizzate localizzate per il tracciante X-Ray. Per ulteriori informazioni, consulta le regole di [campionamento](#) nella Guida per gli sviluppatori AWS X-Ray. Questa variabile è supportata con le immagini di Envoy della versione `v1.19.1.0-prod` e successive.

## XRAY\_SEGMENT\_NAME

Specificate un nome di segmento per le tracce per sovrascrivere il nome predefinito del segmento X-Ray. Per impostazione predefinita, questo valore sarà impostato come `mesh/resourceName`. Questa variabile è supportata con la versione dell'immagine di Envoy `v1.23.1.0-prod` o successiva.

## Variabili di tracciamento Datadog

Le seguenti variabili di ambiente ti aiutano a configurare App Mesh con l'agent tracer Datadog. Per ulteriori informazioni, consulta [Agent Configuration](#) nella documentazione di Datadog.

## ENABLE\_ENVOY\_DATADOG\_TRACING

Abilita la raccolta di tracce Datadog utilizzando `127.0.0.1:8126` come endpoint predefinito dell'agente Datadog. Per abilitare, imposta il valore su `1` (il valore predefinito è `0`).

## DATADOG\_TRACER\_PORT

Specificate un valore di porta per sovrascrivere la porta predefinita dell'agente Datadog: `8126`

## DATADOG\_TRACER\_ADDRESS

Specificare un indirizzo IP per sovrascrivere l'indirizzo predefinito dell'agente Datadog:.

127.0.0.1

## DD\_SERVICE

Specificate un nome di servizio per le tracce per sovrascrivere il nome di servizio Datadog predefinito:/. envoy-meshName virtualNodeName Questa variabile è supportata con le immagini di Envoy della versione e successive. v1.18.3.0-prod

## Variabili di tracciamento Jaeger

Usa le seguenti variabili di ambiente per configurare App Mesh con Jaeger tracing. Per ulteriori informazioni, consulta [Getting Started nella documentazione](#) di Jaeger. Queste variabili sono supportate con le immagini di Envoy della versione e successive. 1.16.1.0-prod

## ENABLE\_ENVOY\_JAEGER\_TRACING

Abilita la raccolta di tracce Jaeger utilizzando 127.0.0.1:9411 come endpoint Jaeger predefinito. Per abilitarla, impostate il valore su 1 (il valore predefinito è). 0

## JAEGER\_TRACER\_PORT

Specificate un valore di porta per sovrascrivere la porta Jaeger predefinita: 9411

## JAEGER\_TRACER\_ADDRESS

Specificate un indirizzo IP per sostituire l'indirizzo Jaeger predefinito: 127.0.0.1

## JAEGER\_TRACER\_VERSION

Specificate se il raccogliatore necessita di tracce in JSON formato codificato. PROTO Per impostazione predefinita, questo valore sarà impostato su. PROTO Questa variabile è supportata con la versione dell'immagine di Envoy v1.23.1.0-prod o successiva.

## Variabile di tracciamento Envoy

Imposta la seguente variabile di ambiente per utilizzare la tua configurazione di tracciamento.

## ENVOY\_TRACING\_CFG\_FILE

Specificate il percorso del file nel file system del contenitore Envoy. Per ulteriori informazioni, consulta la documentazione [config.trace.v3.Tracing](#) di Envoy.

**Note**

Se la configurazione di tracciamento richiede la specifica di un cluster di tracciamento, assicurati di configurare la configurazione del cluster associata `static_resources` nello stesso file di configurazione di tracciamento. Ad esempio, Zipkin ha un [collector\\_cluster](#) campo per il nome del cluster che ospita i raccoglitori di tracce e quel cluster deve essere definito staticamente.

## DogStatsVariabili D

Usa le seguenti variabili di ambiente per configurare App Mesh con DogStats D. Per ulteriori informazioni, consulta la documentazione [DogStatsD](#).

### ENABLE\_ENVOY\_DOG\_STATSD

Abilita le statistiche DogStats D utilizzando `127.0.0.1:8125` come endpoint demone predefinito. Per abilitare, imposta il valore su `1`.

### STATSD\_PORT

Specificate un valore di porta per sostituire la porta demone DogStats D predefinita.

### STATSD\_ADDRESS

Specificare un valore di indirizzo IP per sovrascrivere l'indirizzo IP del demone DogStats D predefinito. Default: `127.0.0.1`. Questa variabile può essere utilizzata solo con la versione `1.15.0` o successiva dell'immagine Envoy.

### STATSD\_SOCKET\_PATH

Specificate un socket di dominio unix per il demone D. DogStats Se questa variabile non è specificata e DogStats D è abilitato, il valore predefinito è la porta dell'indirizzo IP del demone DogStats D di `127.0.0.1:8125`. Se la `ENVOY_STATS_SINKS_CFG_FILE` variabile viene specificata contenente una configurazione stats sinks, sostituisce tutte le variabili D. DogStats Questa variabile è supportata con la versione dell'immagine Envoy o successiva. `v1.19.1.0-prod`

## Variabili App Mesh

Le seguenti variabili consentono di configurare App Mesh.

## APPMESH\_PREVIEW

Imposta il valore su `1` per connetterti all'endpoint App Mesh Preview Channel. Per ulteriori informazioni sull'utilizzo dell'App Mesh Preview Channel, consulta [Canale di anteprima App Mesh](#).

## APPMESH\_RESOURCE\_CLUSTER

Per impostazione predefinita, App Mesh utilizza il nome della risorsa specificato in `APPMESH_RESOURCE_ARN` quando Envoy si riferisce a se stesso nelle metriche e nelle tracce. È possibile ignorare questo comportamento impostando la variabile di ambiente `APPMESH_RESOURCE_CLUSTER` con il proprio nome. Questa variabile può essere utilizzata solo con la versione `1.15.0` o successiva dell'immagine Envoy.

## APPMESH\_METRIC\_EXTENSION\_VERSION

Imposta il valore su `1` per abilitare l'estensione delle metriche App Mesh. Per ulteriori informazioni sull'utilizzo dell'estensione delle metriche App Mesh, consulta [Estensione dei parametri per App Mesh](#).

## APPMESH\_DUALSTACK\_ENDPOINT

Imposta il valore su `1` per connetterti all'endpoint App Mesh Dual Stack. Quando questo flag è impostato, Envoy utilizza il nostro dominio con funzionalità dual stack. Per impostazione predefinita, questo flag è impostato su `false` e si connette solo al nostro dominio. IPv4 Questa variabile può essere utilizzata solo con Envoy image versione `1.22.0` o successiva.

## Variabili di stato di Envoy

Usa le seguenti variabili di ambiente per configurare App Mesh con Envoy Stats. Per ulteriori informazioni, consulta la documentazione di [Envoy Stats](#).

### ENABLE\_ENVOY\_STATS\_TAGS

Abilita l'uso di tag definiti da App Mesh `appmesh.mesh` e `eappmesh.virtual_node`. Per ulteriori informazioni, consulta [config.metrics.v3. TagSpecifiern](#) nella documentazione di Envoy. Per abilitare, imposta il valore su `1`.

### ENVOY\_STATS\_CONFIG\_FILE

Specificate un percorso di file nel file system del contenitore Envoy per sostituire il file di configurazione dei tag Stats predefinito con il vostro. [Per ulteriori informazioni, vedete config.metrics.v3. StatsConfig](#).

**Note**

L'impostazione di una configurazione personalizzata delle statistiche che includa i filtri delle statistiche potrebbe portare Envoy a entrare in uno stato in cui non si sincronizzerà più correttamente con lo stato mondiale dell'App Mesh. Questo è un [bug](#) di Envoy. La nostra raccomandazione è di non eseguire alcun filtraggio delle statistiche in Envoy. [Se il filtraggio è assolutamente necessario, abbiamo elencato un paio di soluzioni alternative in questo numero sulla nostra tabella di marcia.](#)

## ENVOY\_STATS\_SINKS\_CFG\_FILE

Specificate un percorso di file nel file system del contenitore Envoy per sostituire la configurazione predefinita con la vostra. [Per ulteriori informazioni, vedete config.metrics.v3. StatsSink](#) nella documentazione di Envoy.

## Variabili obsolete

Le variabili di ambiente APPMESH\_VIRTUAL\_NODE\_NAME e non APPMESH\_RESOURCE\_NAME sono più supportate nella versione di Envoy o successiva. 1.15.0 Tuttavia, sono ancora supportate per le mesh esistenti. Invece di utilizzare queste variabili con la versione di Envoy 1.15.0 o successiva, usate APPMESH\_RESOURCE\_ARN per tutti gli endpoint App Mesh.

## Impostazioni predefinite di Envoy impostate da App Mesh

Le sezioni seguenti forniscono informazioni sulle impostazioni predefinite di Envoy per la politica di riprova del percorso e l'interruttore automatico impostati da App Mesh.

### Politica predefinita per i nuovi tentativi di routing

Se non avevi mesh nel tuo account prima del 29 luglio 2020, App Mesh crea automaticamente una politica Envoy route retry predefinita per tutte le RPC richieste HTTP, HTTP /2 e g in qualsiasi mesh del tuo account a partire dal 29 luglio 2020. Se avevi delle mesh nel tuo account prima del 29 luglio 2020, non è stata creata alcuna politica predefinita per i percorsi Envoy esistenti prima, il o dopo il 29 luglio 2020. Questo a meno che tu non [apra un ticket con](#) l'assistenza. AWS Dopo che l'assistenza ha elaborato il ticket, viene creata la policy predefinita per tutte le future route Envoy create da App Mesh a partire dalla data di elaborazione del ticket. [Per ulteriori informazioni sulle politiche di riavvio delle rotte di Envoy, consulta config.route.v3. RetryPolicy](#) nella documentazione di Envoy.

App Mesh crea una route Envoy quando si crea una [route](#) App Mesh o si definisce un provider di nodi virtuali per un servizio [virtuale](#) App Mesh. Sebbene sia possibile creare un criterio di ripetizione del percorso App Mesh, non è possibile creare un criterio di ripetizione del percorso App Mesh per un provider di nodi virtuali.

La politica predefinita non è visibile tramite App MeshAPI. La politica predefinita è visibile solo tramite Envoy. Per visualizzare la configurazione, [abilita l'interfaccia di amministrazione](#) e invia una richiesta a Envoy per un. `config_dump` La politica predefinita include le seguenti impostazioni:

- Numero massimo di tentativi: 2
- g) RPC Riprova gli eventi — UNAVAILABLE
- HTTPriprova gli eventi — 503

#### Note

Non è possibile creare una politica di ripetizione del percorso App Mesh che cerchi un codice di HTTP errore specifico. Tuttavia, una politica di ripetizione del percorso App Mesh può cercare `server-error` o `gateway-error`. Entrambi includono 503 errori. Per ulteriori informazioni, consulta [Route](#).

- TCPriprova un evento e `connect-failure refused-stream`

#### Note

Non è possibile creare una politica di ripetizione del percorso App Mesh che cerchi uno di questi eventi. Tuttavia, è possibile cercare una politica di ripetizione del percorso App Mesh `connection-error`, che è equivalente a `connect-failure`. Per ulteriori informazioni, consulta [Route](#).

- Reimpostazione: Envoy tenta un nuovo tentativo se il server upstream non risponde affatto (timeout di disconnessione/ripristinazione/lettura).

## Interruttore automatico predefinito

Quando si distribuisce un Envoy in App Mesh, i valori predefiniti di Envoy vengono impostati per alcune impostazioni dell'interruttore automatico. [Per ulteriori informazioni, consulta cluster.CircuitBreakers.Soglie](#) nella documentazione di Envoy. Queste impostazioni non sono visibili tramite

App MeshAPI. Le impostazioni sono visibili solo tramite Envoy. Per visualizzare la configurazione, [abilita l'interfaccia di amministrazione](#) e invia una richiesta a Envoy per un. `config_dump`

Se non avevi mesh nel tuo account prima del 29 luglio 2020, per ogni Envoy che distribuisce in una mesh creata il o dopo il 29 luglio 2020, App Mesh disabilita efficacemente gli interruttori automatici modificando i valori predefiniti di Envoy per le impostazioni che seguono. Se avevi delle mesh nel tuo account prima del 29 luglio 2020, i valori predefiniti di Envoy sono impostati per tutti gli Envoy che distribuisce in App Mesh il 29 luglio 2020 o dopo, a meno che [tu](#) non apra un ticket con l'assistenza. AWS Una volta che il supporto ha elaborato il ticket, i valori predefiniti di App Mesh per le seguenti impostazioni di Envoy vengono impostati da App Mesh su tutti gli Envoy distribuiti dopo la data di elaborazione del ticket:

- **max\_requests** – 2147483647
- **max\_pending\_requests** – 2147483647
- **max\_connections** – 2147483647
- **max\_retries** – 2147483647

#### Note

Indipendentemente dal fatto che i tuoi Envoy abbiano i valori di interruttore predefiniti di Envoy o App Mesh, non puoi modificare i valori.

## Aggiornamento/migrazione a Envoy 1.17

### Secret Discovery Service con SPIRE

Se utilizzi SPIRE (SPIFFERuntime Environment) con App Mesh per distribuire certificati di fiducia ai tuoi servizi, verifica di utilizzare almeno la versione 0.12.0 dell'[SPIREagente](#) (rilasciata a dicembre 2020). Questa è la prima versione in grado di supportare le versioni di Envoy successive. 1.16

### Modifiche alle espressioni regolari

A partire da Envoy1.17, App Mesh configura Envoy per utilizzare il motore di espressioni [RE2](#) regolari per impostazione predefinita. Questa modifica è evidente alla maggior parte degli utenti, ma le corrispondenze in Routes o Gateway Routes non consentono più riferimenti in avanti o indietro nelle espressioni regolari.

## Aspetto prospettico positivo e negativo

Positivo: una prospettiva positiva è un'espressione tra parentesi che inizia con: ?=

```
(?=example)
```

Queste sono le più utili quando si effettua la sostituzione di stringhe perché consentono di abbinare una stringa senza consumare i caratteri come parte della corrispondenza. Poiché App Mesh non supporta la sostituzione di stringhe regex, ti consigliamo di sostituirle con corrispondenze regolari.

```
(example)
```

Negativo: una previsione negativa è un'espressione tra parentesi che inizia con. ?!

```
ex(?!amp)le
```

Le espressioni tra parentesi vengono utilizzate per affermare che una parte dell'espressione non corrisponde a un determinato input. Nella maggior parte dei casi, è possibile sostituirle con un quantificatore zero.

```
ex(amp){0}le
```

Se l'espressione stessa è una classe di caratteri, potete negare l'intera classe e contrassegnarla come facoltativa utilizzando. ?

```
prefix(?![0-9])suffix => prefix[^0-9]?suffix
```

A seconda del caso d'uso, potresti anche essere in grado di modificare i percorsi per gestirlo.

```
{
 "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 prima corrispondenza del percorso cerca un'intestazione che inizia con «prefisso» ma non seguita da «suffisso». La seconda route fa in modo che corrisponda a tutte le altre intestazioni che iniziano con «prefisso», incluse quelle che terminano con «suffisso». Invece, possono anche essere invertiti per rimuovere il look-ahead negativo.

```

{
 "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"
 }
 }
]
 }
 }
 }
}
}

```

Questo esempio inverte le rotte per dare maggiore priorità alle intestazioni che terminano con «suffix» e tutte le altre intestazioni che iniziano con «prefix» corrispondono alla route con priorità più bassa.

## Riferimenti all'indietro

Un riferimento a ritroso è un modo per scrivere espressioni più brevi ripetendole in un precedente gruppo tra parentesi. Hanno questa forma.

```
(group1)(group2)\1
```

Una barra rovesciata \ seguita da un numero funge da segnaposto per l'ennesimo gruppo tra parentesi nell'espressione. In questo esempio, \1 viene utilizzato come metodo alternativo per scrivere una seconda volta. (group1)

```
(group1)(group2)(group1)
```

Questi possono essere rimossi semplicemente sostituendo il riferimento all'indietro con il gruppo a cui si fa riferimento, come nell'esempio.

## Agente dell'ambasciatore

L'agente è un gestore di processi all'interno dell'immagine Envoy venduta per App Mesh. L'agente garantisce che Envoy rimanga operativo, integro e riduce i tempi di inattività. Filtra le statistiche e i dati ausiliari di Envoy per fornire una visione sintetica del funzionamento del proxy Envoy in App Mesh. Questo può aiutarti a risolvere più rapidamente gli errori correlati.

È possibile utilizzare l'agente per configurare il numero di volte in cui si desidera riavviare il proxy Envoy nel caso in cui il proxy non funzioni correttamente. Se si verifica un errore, l'agente registra lo stato di uscita conclusivo all'uscita di Envoy. È possibile utilizzarlo per risolvere il problema. L'agente facilita inoltre l'esaurimento della connessione Envoy, il che aiuta a rendere le applicazioni più resistenti ai guasti.

Configura l'Agent for Envoy utilizzando queste variabili:

- `APPNET_ENVOY_RESTART_COUNT`— Quando questa variabile è impostata su un valore diverso da zero, l'agente tenta di riavviare il processo proxy Envoy fino al numero impostato quando ritiene che lo stato del processo proxy non sia integro durante il polling. Ciò consente di ridurre i tempi di inattività fornendo un riavvio più rapido rispetto alla sostituzione di un'attività o di un pod da parte del container orchestrator in caso di errori nel controllo dello stato del proxy.
- `PID_POLL_INTERVAL_MS`— Quando si configura questa variabile, l'impostazione predefinita viene mantenuta invariata a 100. Se impostato su questo valore, consentite un rilevamento e un riavvio più rapidi del processo Envoy all'uscita rispetto alla sostituzione di task o pod tramite controlli di integrità del container orchestrator.
- `LISTENER_DRAIN_WAIT_TIME_S`— Quando configuri questa variabile, considera il timeout del container orchestrator impostato per interrompere l'operazione o il pod. Ad esempio, se questo valore è maggiore del timeout dell'orchestratore, il proxy Envoy può esaurirsi solo per la durata fino a quando l'orchestratore non interrompe forzatamente l'attività o il pod.
- `APPNET_AGENT_ADMIN_MODE`— Quando questa variabile è impostata su `cp ouds`, l'agente fornisce un'interfaccia di gestione locale. Questa interfaccia di gestione funge da endpoint sicuro per interagire con il proxy Envoy e fornisce le seguenti API per i controlli di integrità, i dati di telemetria e riassume le condizioni operative del proxy.
  - `GET /status`— Interroga le statistiche di Envoy e restituisce le informazioni sul server.
  - `POST /drain_listeners`— Prosciuga tutti gli ascoltatori in entrata.
  - `POST /enableLogging?level=<desired_level>`— Cambia il livello di registrazione di Envoy su tutti i logger.

- GET /stats/prometheus— Mostra le statistiche di Envoy in formato Prometheus.
- GET /stats/prometheus?usedonly— Mostra solo le statistiche aggiornate da Envoy.

Per ulteriori informazioni sulle variabili di configurazione dell'agente, consulta [Variabili di configurazione di Envoy](#).

Il nuovo AWS App Mesh Agent è incluso nelle immagini Envoy ottimizzate per App Mesh a partire dalla versione 1.21.0.0 e non richiede l'allocazione di risorse aggiuntive nelle attività o nei pod dei clienti.

# App Mesh

Uno dei vantaggi derivanti dall'utilizzo di App Mesh è una maggiore visibilità delle applicazioni di microservizi. App Mesh è in grado di funzionare con molte diverse soluzioni di registrazione, metriche e tracciamento.

Il proxy Envoy e l'App Mesh offrono i seguenti tipi di strumenti per aiutarti a ottenere una visione più chiara delle tue applicazioni e dei tuoi proxy:

- [Registrazione di log](#)
- [Parametri](#)
- [Tracciamento](#)

## Registrazione

Quando crei i tuoi nodi virtuali e i tuoi gateway virtuali, hai la possibilità di configurare i registri di accesso di Envoy. Nella console, si trova nella sezione Logging del nodo virtuale e del gateway virtuale per creare o modificare i flussi di lavoro.

### Logging

#### HTTP access logs path - *optional*

The path used to send logging information for the virtual node. App Mesh recommends using the standard out I/O stream.

`/dev/stdout`

 Logs must still be ingested by an agent in your application and sent to a destination. This file path only instructs Envoy where to send the logs.

L'immagine precedente mostra un percorso di registrazione dei `/dev/stdout` registri di accesso di Envoy.

Specificate `format` infatti uno dei due formati possibili, `json` oppure `text`, e il `pattern`. `json` prende coppie di chiavi e le trasforma in strutture JSON prima di passarle a Envoy.

Il seguente blocco di codice mostra la rappresentazione JSON che è possibile utilizzare in 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

Assicurati di verificare che il modello di input sia valido per Envoy, altrimenti Envoy rifiuterà l'aggiornamento e memorizzerà le ultime modifiche nello stato `error`.

Quando invii i log di accesso di Envoy a `/dev/stdout`, questi vengono mescolati con i registri del contenitore Envoy. Puoi esportarli in un servizio di archiviazione ed elaborazione dei log come CloudWatch Logs utilizzando driver di registro Docker standard come `awslogs`. Per ulteriori informazioni, consulta [Utilizzo del driver di log `awslogs`](#) nella Guida per gli sviluppatori di Amazon ECS. Per esportare solo i log di accesso di Envoy (e ignorare gli altri registri del contenitore Envoy), puoi impostare `ENVOY_LOG_LEVEL` su `off`. È possibile registrare la richiesta senza stringa di query includendo la stringa di formato `%REQ_WITHOUT_QUERY(X?Y):Z%`. Per esempi, vedere [ReqWithoutQueryFormatter](#). Per ulteriori informazioni, consulta la [pagina Accesso](#) nella documentazione di Envoy.

### Abilitare log di accesso su Kubernetes

Quando si utilizza l'[App Mesh Controller per Kubernetes](#), è possibile configurare nodi virtuali con registrazione degli accessi aggiungendo la configurazione di registrazione alle specifiche del nodo virtuale, come mostrato nell'esempio seguente.

```

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"
```

Il cluster deve disporre di un server di inoltro dei log per raccogliere questi log, ad esempio Fluentd. Per ulteriori informazioni, vedere [Configurare Fluentd come DaemonSet a per inviare i log a CloudWatch Logs](#).

Envoy scrive anche vari registri di debug dai suoi filtri `stdout`. Questi registri sono utili per ottenere informazioni sia sulla comunicazione di Envoy con App Mesh che service-to-service sul traffico. Il livello di registrazione specifico può essere configurato utilizzando la variabile di ambiente `ENVOY_LOG_LEVEL`. Ad esempio, il testo seguente è tratto da un registro di debug di esempio che mostra il cluster che Envoy ha trovato per una particolare richiesta HTTP.

```
[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 e Cloudwatch

[Firelens](#) è un router di log per container che puoi utilizzare per raccogliere i log per Amazon ECS e AWS Fargate. Puoi trovare un esempio di utilizzo di Firelens nel nostro [archivio di AWS campioni](#).

È possibile utilizzare CloudWatch per raccogliere informazioni di registrazione e metriche. Puoi trovare maggiori informazioni CloudWatch nella nostra sezione [Esportazione delle metriche](#) dei documenti di App Mesh.

## Monitoraggio dell'applicazione utilizzando le metriche Envoy

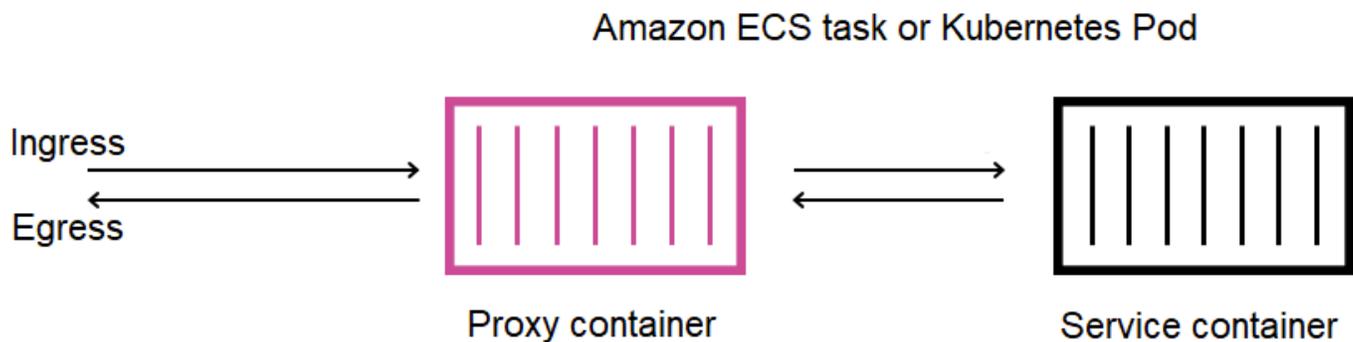
Envoy classifica le sue metriche nelle seguenti categorie principali:

- **Downstream:** metriche relative alle connessioni e alle richieste che arrivano al proxy.
- **Upstream:** metriche relative alle connessioni e alle richieste in uscita effettuate dal proxy.
- **Server:** metriche che descrivono lo stato interno di Envoy. Queste includono metriche come l'uptime o la memoria allocata.

In App Mesh, il proxy intercetta il traffico a monte e a valle. Ad esempio, le richieste ricevute dai tuoi clienti e le richieste effettuate dal tuo container di servizio sono classificate come traffico downstream da Envoy. Per distinguere tra questi diversi tipi di traffico a monte e a valle, App Mesh classifica ulteriormente le metriche Envoy in base alla direzione del traffico relativa al servizio:

- **Ingress:** metriche e risorse relative alle connessioni e alle richieste che fluiscono verso il contenitore di servizi.
- **Uscita:** metriche e risorse relative alle connessioni e alle richieste che fluiscono dal contenitore di servizi e, infine, dall'attività Amazon ECS o dal pod Kubernetes.

L'immagine seguente mostra la comunicazione tra il proxy e i contenitori di servizio.



### Convenzioni di denominazione delle risorse

È utile capire in che modo Envoy visualizza la tua mesh e in che modo le sue risorse si ricollegano alle risorse che definisci in App Mesh. Queste sono le principali risorse Envoy configurate da App Mesh:

- **Listener:** gli indirizzi e le porte su cui il proxy ascolta le connessioni downstream. Nell'immagine precedente, App Mesh crea un listener di ingresso per il traffico in entrata nel task Amazon ECS o nel pod Kubernetes e un listener in uscita per il traffico in uscita dal container di servizi.

- **Cluster:** un gruppo denominato di endpoint upstream a cui il proxy si connette e a cui indirizza il traffico. In App Mesh, il contenitore di servizi è rappresentato come un cluster, così come tutti gli altri nodi virtuali a cui il servizio può connettersi.
- **Percorsi:** corrispondono ai percorsi definiti nella rete. Contengono le condizioni in base alle quali il proxy soddisfa una richiesta e il cluster di destinazione a cui viene inviata una richiesta.
- **Endpoint e assegnazioni di carico dei cluster:** gli indirizzi IP dei cluster upstream. Quando viene utilizzato AWS Cloud Map come meccanismo di rilevamento dei servizi per i nodi virtuali, App Mesh invia le istanze di servizio rilevate come risorse endpoint al proxy.
- **Segreti:** includono, a titolo esemplificativo ma non esaustivo, le chiavi di crittografia e i certificati TLS. Quando viene utilizzato AWS Certificate Manager come fonte per i certificati client e server, App Mesh invia certificati pubblici e privati al proxy come risorse segrete.

App Mesh utilizza uno schema coerente per denominare le risorse di Envoy che puoi utilizzare per ricollegarti alla tua mesh.

Comprendere lo schema di denominazione per ascoltatori e cluster è importante per comprendere le metriche di Envoy in App Mesh.

## Nomi degli ascoltatori

Gli ascoltatori sono denominati utilizzando il formato seguente:

```
lds_<traffic direction>_<listener IP address>_<listening port>
```

In genere vedrai i seguenti ascoltatori configurati in Envoy:

- `lds_ingress_0.0.0.0_15000`
- `lds_egress_0.0.0.0_15001`

Utilizzando un plug-in Kubernetes CNI o le regole delle tabelle IP, il traffico nel task Amazon ECS o nel pod Kubernetes viene indirizzato alle porte 15000 e 15001. App Mesh configura Envoy con questi due ascoltatori per accettare il traffico in ingresso (in entrata) e in uscita (in uscita). Se non hai un listener configurato sul tuo nodo virtuale, non dovresti vedere un listener di ingresso.

## Nomi dei cluster

La maggior parte dei cluster utilizza il formato seguente:

```
cds_<traffic direction>_<mesh name>_<virtual node name>_<protocol>_<port>
```

I nodi virtuali con cui comunicano i tuoi servizi hanno ciascuno il proprio cluster. Come accennato in precedenza, App Mesh crea un cluster per il servizio in esecuzione accanto a Envoy in modo che il proxy possa inviargli traffico in ingresso.

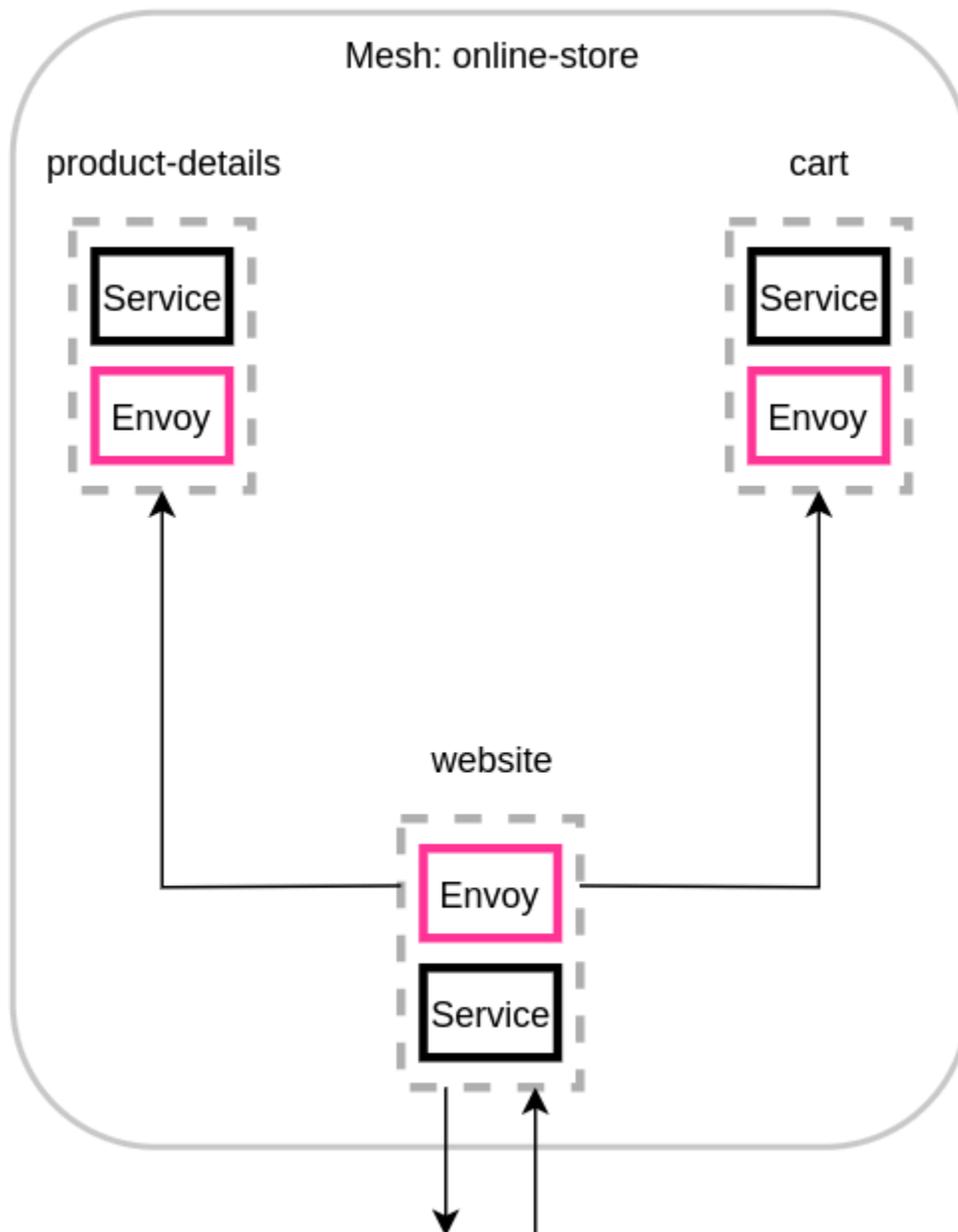
Ad esempio, se hai un nodo virtuale denominato `my-virtual-node` che ascolta il traffico http sulla porta `8080` e quel nodo virtuale si trova in una mesh denominata `my-mesh`, App Mesh crea un cluster denominato `cds_ingress_my-mesh_my-virtual-node_http_8080`. Questo cluster funge da destinazione per il traffico verso `my-virtual-node` il contenitore di servizio.

App Mesh può anche creare i seguenti tipi di cluster speciali aggiuntivi. Questi altri cluster non corrispondono necessariamente alle risorse definite in modo esplicito nella mesh.

- Cluster utilizzati per raggiungere altri AWS servizi. Questo tipo consente alla tua mesh di raggiungere la maggior parte dei AWS servizi per impostazione predefinita: `cds_egress_<mesh name>_amazonaws`.
- Cluster utilizzato per eseguire il routing per i gateway virtuali. Questo può generalmente essere tranquillamente ignorato:
  - Per ascoltatori singoli: `cds_ingress_<mesh name>_<virtual gateway name>_self_redirect_<protocol>_<port>`
  - Per più ascoltatori: `cds_ingress_<mesh name>_<virtual gateway name>_self_redirect_<ingress_listener_port>_<protocol>_<port>`
- Il cluster il cui endpoint puoi definire, ad esempio TLS, quando recuperi segreti utilizzando Envoy's Secret Discovery Service: `static_cluster_sds_unix_socket`.

## Parametri dell'applicazione di esempio

Per illustrare le metriche disponibili in Envoy, la seguente applicazione di esempio dispone di tre nodi virtuali. I servizi virtuali, i router virtuali e i percorsi nella mesh possono essere ignorati poiché non si riflettono nelle metriche di Envoy. In questo esempio, tutti i servizi ascoltano il traffico http sulla porta `8080`.



Ti consigliamo di aggiungere la variabile `ENABLE_ENVOY_STATS_TAGS=1` di ambiente ai contenitori proxy Envoy in esecuzione nella tua mesh. Questo aggiunge le seguenti dimensioni dei parametri a tutti i parametri emessi dal proxy:

- `appmesh.mesh`
- `appmesh.virtual_node`
- `appmesh.virtual_gateway`

Questi tag sono impostati sul nome della mesh, del nodo virtuale o del gateway virtuale per consentire di filtrare le metriche utilizzando i nomi delle risorse nella mesh.

## Nomi delle risorse

Il proxy del nodo virtuale del sito Web dispone delle seguenti risorse:

- Due ascoltatori per il traffico in entrata e in uscita:
  - `lds_ingress_0.0.0.0_15000`
  - `lds_egress_0.0.0.0_15001`
- Due cluster di uscita, che rappresentano i due backend dei nodi virtuali:
  - `cds_egress_online-store-product-details_http_8080`
  - `cds_egress_online-store-cart_http_8080`
- Un cluster di ingresso per il contenitore dei servizi del sito Web:
  - `cds_ingress_online-store-website_http_8080`

## Metriche di esempio per gli ascoltatori

- `listener.0.0.0.0_15000.downstream_cx_active`—Numero di connessioni di rete in ingresso attive a Envoy.
- `listener.0.0.0.0_15001.downstream_cx_active`—Numero di connessioni di rete in uscita attive verso Envoy. Le connessioni effettuate dall'applicazione a servizi esterni sono incluse in questo conteggio.
- `listener.0.0.0.0_15000.downstream_cx_total`—Numero totale di connessioni di rete in ingresso a Envoy.
- `listener.0.0.0.0_15001.downstream_cx_total`—Numero totale di connessioni di rete in uscita verso Envoy.

Per il set completo di metriche degli ascoltatori, consulta [le statistiche](#) nella documentazione di Envoy.

## Esempio di metriche del cluster

- `cluster_manager.active_clusters`—Il numero totale di cluster a cui Envoy ha stabilito almeno una connessione.

- `cluster_manager.warming_clusters`—Il numero totale di cluster a cui Envoy deve ancora connettersi.

Le seguenti metriche del cluster utilizzano il formato di `cluster.<cluster name>.<metric name>`. Questi nomi di metriche sono univoci per l'esempio dell'applicazione e vengono emessi dal contenitore Envoy del sito Web:

- `cluster.cds_egress_online-store_product-details_http_8080.upstream_cx_total`—Numero totale di connessioni tra il sito Web e i dettagli del prodotto.
- `cluster.cds_egress_online-store_product-details_http_8080.upstream_cx_connect_fail`—Numero totale di connessioni non riuscite tra il sito Web e i dettagli del prodotto.
- `cluster.cds_egress_online-store_product-details_http_8080.health_check.failure`—Numero totale di controlli sanitari non riusciti tra il sito Web e i dettagli del prodotto.
- `cluster.cds_egress_online-store_product-details_http_8080.upstream_rq_total`—Numero totale di richieste effettuate tra il sito Web e i dettagli del prodotto.
- `cluster.cds_egress_online-store_product-details_http_8080.upstream_rq_time`—Tempo impiegato dalle richieste effettuate tra il sito Web e i dettagli del prodotto.
- `cluster.cds_egress_online-store_product-details_http_8080.upstream_rq_2xx`—Numero di risposte HTTP 2xx ricevute dal sito Web dai dettagli del prodotto.

Per il set completo di metriche HTTP, consulta [le statistiche](#) nella documentazione di Envoy.

## Metriche del server di gestione

Envoy emette anche metriche relative alla sua connessione al piano di controllo App Mesh, che funge da server di gestione di Envoy. Ti consigliamo di monitorare alcune di queste metriche per avvisarti quando i tuoi proxy vengono desincronizzati dal piano di controllo per lunghi periodi di tempo. La perdita di connettività al piano di controllo o gli aggiornamenti non riusciti impediscono ai proxy di ricevere nuove configurazioni da App Mesh, comprese le modifiche alla mesh apportate tramite le API App Mesh.

- `control_plane.connected_state`—Questa metrica è impostata su 1 quando il proxy è connesso ad App Mesh, altrimenti è 0.
- `*.update_rejected`—Numero totale di aggiornamenti di configurazione rifiutati da Envoy. Questi sono in genere dovuti a una configurazione errata dell'utente. Ad esempio, se configuri App Mesh per leggere un certificato TLS da un file che non può essere letto da Envoy, l'aggiornamento contenente il percorso di quel certificato viene rifiutato.
  - Se il Listener aggiornato viene rifiutato, le statistiche saranno `listener_manager.lds.update_rejected`.
  - Se il Cluster aggiornato viene rifiutato, le statistiche saranno `cluster_manager.cds.update_rejected`.
- `*.update_success`—Numero di aggiornamenti di configurazione effettuati con successo da App Mesh al tuo proxy. Questi includono il payload di configurazione iniziale inviato all'avvio di un nuovo contenitore Envoy.
  - Per il successo dell'aggiornamento di Listener, le statistiche saranno `listener_manager.lds.update_success`.
  - Affinché l'aggiornamento del Cluster abbia successo, le statistiche saranno `cluster_manager.cds.update_success`.

Per il set di metriche del server di gestione, vedere [Management Server](#) nella documentazione di Envoy.

## Esportazione di parametri

Envoy emette molte statistiche sia sul proprio funzionamento che su varie dimensioni sul traffico in entrata e in uscita. Per ulteriori informazioni sulle statistiche di Envoy, consulta [Statistiche](#) nella documentazione di Envoy. Queste metriche sono disponibili tramite `/stats` endpoint sulla porta di amministrazione del proxy, che in genere è 9901.

Il `stats` prefisso sarà diverso a seconda che si utilizzino ascoltatori singoli o multipli. Di seguito sono riportati alcuni esempi per illustrare le differenze.

### Warning

Se aggiorni il tuo listener singolo alla funzione multi-listener, potresti riscontrare un cambiamento radicale a causa del prefisso statistico aggiornato illustrato nella tabella seguente.

Ti consigliamo di utilizzare l'immagine di Envoy1.22.2.1-prod o una versione successiva. Ciò consente di visualizzare nomi di metriche simili nell'endpoint Prometheus.

| Single Listener (SL) /Statistiche esistenti con prefisso listener «ingresso» | Ascoltatori multipli (ML) / Nuove statistiche con «ingresso». <protocol>. <port>"prefisso listener                                                    |  |
|------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|--|
| <pre>http.*ingresso*.rds .rds_ingress_http_ 5555.version_text</pre>          | <pre>http.*ingresso.http .5555*.rds.rds_ing ress_http_5555.ver sion_text  http.*ingresso.http .6666*.rds.rds_ing ress_http_6666.ver sion_text</pre>   |  |
| <pre>listener.0.0.0.0_1 5000.http.*ingress *.downstream_rq_2xx</pre>         | <pre>listener.0.0.0.0_1 5000.http.*ingress .http.5555*.downst ream_rq_2xx  listener.0.0.0.0_1 5000.http.*ingress .http.6666*.downst ream_rq_2xx</pre> |  |
| <pre>http.*ingresso*.dow nstream_cx_length_ ms</pre>                         | <pre>http.*ingresso.http .5555*.downstream_ cx_length_ms  http.*ingresso.http .6666*.downstream_ cx_length_ms</pre>                                   |  |

Per ulteriori informazioni sull'endpoint stats, consulta l'[endpoint Statistics](#) nella documentazione di Envoy. Per ulteriori informazioni sull'interfaccia di amministrazione, consulta [Abilita l'interfaccia di amministrazione del proxy Envoy](#).

## Prometheus per App Mesh con Amazon EKS

Prometheus è un kit di strumenti di monitoraggio e avvisi open source. Una delle sue funzionalità consiste nello specificare un formato per l'emissione di metriche che possono essere utilizzate da altri sistemi. Per ulteriori informazioni su Prometheus, consulta [Panoramica](#) nella documentazione di Prometheus. Envoy può emettere le sue metriche tramite il suo endpoint stats inserendo il parametro `/stats?format=prometheus`.

Per i clienti che utilizzano Envoy image build v1.22.2.1-prod, sono disponibili due dimensioni aggiuntive per indicare le statistiche specifiche del listener di ingresso:

- `appmesh.listener_protocol`
- `appmesh.listener_port`

Di seguito è riportato un confronto tra le statistiche esistenti di Prometheus e le nuove statistiche.

- Statistiche esistenti con prefisso listener «ingresso»

```
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="ingresso"} 931433
```

- Nuove statistiche con «ingresso». `<protocol>. <port>`+ Appmesh Envoy Image v1.22.2.1-prod o versione successiva

```
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="55520"}
```

- Nuove statistiche con «ingresso». `<protocol>. <port>`+ Envoy Imagebuild personalizzato

```
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="ingresso"} 15983
```

Per più ascoltatori, il cluster `cds_ingress_<mesh name>_<virtual gateway name>_self_redirect_<ingress_listener_port>_<protocol>_<port>` speciale sarà specifico per l'ascoltatore.

- Statistiche esistenti con prefisso listener «ingress»

```
envoy_cluster_assignment_stale{appmesh_mesh="multiple-listeners-mesh",appmesh_virtual_gateway="telligateway-vg",Mesh="multiple-listeners-mesh",VirtualGateway="telligateway-vg",envoy_cluster_name="cds_ingress_multiple-listeners-mesh_telligateway-vg_self_redirect_http_15001"} 0
```

- Nuove statistiche con «ingresso». `<protocol>.<port>`

```
envoy_cluster_assignment_stale{appmesh_mesh="multiple-listeners-mesh",appmesh_virtual_gateway="telligateway-vg",envoy_cluster_name="cds_ingress_multiple-listeners-mesh_telligateway-vg_self_redirect_1111_http_15001"} 0
envoy_cluster_assignment_stale{appmesh_mesh="multiple-listeners-mesh",appmesh_virtual_gateway="telligateway-vg",envoy_cluster_name="cds_ingress_multiple-listeners-mesh_telligateway-vg_self_redirect_2222_http_15001"} 0
```

## Installazione di Prometheus

1. Aggiungi il repository EKS a Helm:

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

2. Installazione di App Mesh Prometheus

```
helm upgrade -i appmesh-prometheus eks/appmesh-prometheus \
--namespace appmesh-system
```

## Esempio di Prometheus

Di seguito è riportato un esempio di creazione di un archivio persistente `PersistentVolumeClaim` per Prometheus.

```
helm upgrade -i appmesh-prometheus eks/appmesh-prometheus \
--namespace appmesh-system \
--set retention=12h \
--set persistentVolumeClaim.claimName=prometheus
```

Procedura dettagliata per l'utilizzo di Prometheus

- [App Mesh con EKS: osservabilità: Prometheus](#)

Per saperne di più su Prometheus e Prometheus con Amazon EKS

- [Documentazione Prometheus](#)
- EKS - [Metriche del piano di controllo con Prometheus](#)

## CloudWatch per App Mesh

Emissione delle statistiche di Envoy CloudWatch da Amazon EKS

Puoi installare l' CloudWatch agente nel tuo cluster e configurarlo per raccogliere un sottoinsieme di metriche dai tuoi proxy. Se non disponi già di un cluster Amazon EKS, puoi crearne uno seguendo la procedura descritta in [Walkthrough: App Mesh with Amazon EKS](#) attivo GitHub. È possibile installare un'applicazione di esempio nel cluster seguendo la stessa procedura dettagliata.

Per impostare le autorizzazioni IAM appropriate per il cluster e installare l'agente, segui i passaggi in [Installare l' CloudWatch agente con Prometheus Metrics Collection](#). L'installazione predefinita contiene una configurazione di scrape di Prometheus che estrae un utile sottoinsieme di statistiche di Envoy. Per ulteriori informazioni, consulta [Prometheus Parametri per App Mesh](#).

Per creare una CloudWatch dashboard personalizzata di App Mesh configurata per visualizzare le metriche che l'agente sta raccogliendo, segui i passaggi del tutorial [Visualizzazione delle metriche di Prometheus](#). I grafici inizieranno a essere compilati con le metriche corrispondenti non appena il traffico entra nell'applicazione App Mesh.

Applicazione di filtri ai parametri per CloudWatch

L'[estensione delle metriche](#) App Mesh fornisce un sottoinsieme di metriche utili che forniscono informazioni sul comportamento delle risorse che definisci nella tua mesh. Poiché

L'CloudWatch agente supporta lo scraping delle metriche di Prometheus, puoi fornire una configurazione di scrape per selezionare le metriche che desideri estrarre da Envoy e inviarle CloudWatch.

Puoi trovare un esempio di metriche di scraping utilizzando Prometheus nella nostra guida dettagliata di [Metrics Extension](#).

### CloudWatch Esempio

Puoi trovare una configurazione di esempio di CloudWatch nel nostro [archivio di AWS campioni](#).

Procedure dettagliate per l'utilizzo CloudWatch

- [Aggiungi funzionalità di monitoraggio e registrazione](#) nel nostro [workshop App Mesh](#).
- [App Mesh con EKS: osservabilità: CloudWatch](#)
- [Utilizzo dell'estensione metrica di App Mesh su ECS](#)

## Estensione dei parametri per App Mesh

Envoy genera centinaia di metriche suddivise in alcune dimensioni diverse. Le metriche non sono semplici nel modo in cui si riferiscono ad App Mesh. Nel caso dei servizi virtuali, non esiste alcun meccanismo per sapere con certezza quale servizio virtuale sta comunicando con un determinato nodo virtuale o gateway virtuale.

L'estensione delle metriche App Mesh migliora i proxy Envoy in esecuzione nella tua mesh. Questo miglioramento consente ai proxy di emettere metriche aggiuntive che sono consapevoli delle risorse definite. Questo piccolo sottoinsieme di metriche aggiuntive ti aiuterà a comprendere meglio il comportamento delle risorse che hai definito in App Mesh.

Per abilitare l'estensione delle metriche App Mesh, imposta la variabile `APPMESH_METRIC_EXTENSION_VERSION` di ambiente su 1.

```
APPMESH_METRIC_EXTENSION_VERSION=1
```

Per ulteriori informazioni sulle variabili di configurazione Envoy, consulta [Variabili di configurazione di Envoy](#).

Metriche relative al traffico in entrata

- **ActiveConnectionCount**

- `envoy.appmesh.ActiveConnectionCount`— Numero di connessioni TCP attive.
- Dimensioni: maglia `VirtualNode`, `VirtualGateway`
- **NewConnectionCount**
  - `envoy.appmesh.NewConnectionCount`— Numero totale di connessioni TCP.
  - Dimensioni: maglia `VirtualNode`, `VirtualGateway`
- **ProcessedBytes**
  - `envoy.appmesh.ProcessedBytes`— Byte TCP totali inviati e ricevuti dai client downstream.
  - Dimensioni: maglia `VirtualNode`, `VirtualGateway`
- **RequestCount**
  - `envoy.appmesh.RequestCount`— Il numero di richieste HTTP elaborate.
  - Dimensioni: maglia `VirtualNode`, `VirtualGateway`
- **GrpcRequestCount**
  - `envoy.appmesh.GrpcRequestCount`— Il numero di richieste gRPC elaborate.
  - Dimensioni: maglia `VirtualNode`, `VirtualGateway`

Metriche relative al traffico in uscita

Vedrai dimensioni diverse nelle metriche in uscita in base al fatto che provengano da un nodo virtuale o da un gateway virtuale.

- **TargetProcessedBytes**
  - `envoy.appmesh.TargetProcessedBytes`— Byte TCP totali inviati e ricevuti da destinazioni a monte di Envoy.
  - Dimensioni:
    - Dimensioni del nodo virtuale: `mesh`, `VirtualNode`, `TargetVirtualService`, `TargetVirtualNode`
    - Dimensioni del gateway virtuale: `Mesh`, `VirtualGateway`, `TargetVirtualService`, `TargetVirtualNode`
- **HTTPCode\_Target\_2XX\_Count**
  - `envoy.appmesh.HTTPCode_Target_2XX_Count`— Il numero di richieste HTTP indirizzate a un target a monte di Envoy che hanno prodotto una risposta HTTP 2xx.
  - Dimensioni:

- Dimensioni del gateway virtuale: Mesh, VirtualGateway, TargetVirtualService, TargetVirtualNode
- **HTTPCode\_Target\_3XX\_Count**
  - `envoy.appmesh.HTTPCode_Target_3XX_Count`— Il numero di richieste HTTP indirizzate a un target a monte di Envoy che hanno portato a una risposta HTTP 3xx.
  - Dimensioni:
    - Dimensioni del nodo virtuale: mesh, VirtualNode, TargetVirtualService, TargetVirtualNode
    - Dimensioni del gateway virtuale: Mesh, VirtualGateway, TargetVirtualService, TargetVirtualNode
- **HTTPCode\_Target\_4XX\_Count**
  - `envoy.appmesh.HTTPCode_Target_4XX_Count`— Il numero di richieste HTTP indirizzate a un target a monte di Envoy che hanno prodotto una risposta HTTP 4xx.
  - Dimensioni:
    - Dimensioni del nodo virtuale: mesh, VirtualNode, TargetVirtualService, TargetVirtualNode
    - Dimensioni del gateway virtuale: Mesh, VirtualGateway, TargetVirtualService, TargetVirtualNode
- **HTTPCode\_Target\_5XX\_Count**
  - `envoy.appmesh.HTTPCode_Target_5XX_Count`— Il numero di richieste HTTP indirizzate a un target a monte di Envoy che hanno prodotto una risposta HTTP 5xx.
  - Dimensioni:
    - Dimensioni del nodo virtuale: mesh, VirtualNode, TargetVirtualService, TargetVirtualNode
    - Dimensioni del gateway virtuale: Mesh, VirtualGateway, TargetVirtualService, TargetVirtualNode
- **RequestCountPerTarget**
  - `envoy.appmesh.RequestCountPerTarget`— Il numero di richieste inviate a un target a monte di Envoy.
  - Dimensioni:
    - Dimensioni del nodo virtuale: mesh, VirtualNode, TargetVirtualService, TargetVirtualNode
    - Dimensioni del gateway virtuale: Mesh, VirtualGateway, TargetVirtualService, TargetVirtualNode

- **TargetResponseTime**

- `envoy.appmesh.TargetResponseTime`— Il tempo trascorso da quando viene effettuata una richiesta a un target a monte di Envoy a quando viene ricevuta la risposta completa.
- Dimensioni:
  - Dimensioni del nodo virtuale: `mesh`, `VirtualNode`, `TargetVirtualService`, `TargetVirtualNode`
  - Dimensioni del gateway virtuale: `Mesh`, `VirtualGateway`, `TargetVirtualService`, `TargetVirtualNode`

## Datadog per App Mesh

Datadog è un'applicazione di monitoraggio e sicurezza per il monitoraggio completo, le metriche e la registrazione delle applicazioni cloud. Datadog rende l'infrastruttura, le applicazioni e le applicazioni di terze parti completamente osservabili.

### Installazione di Datadog

- EKS - Per configurare Datadog con EKS, segui questi passaggi dai [documenti Datadog](#).
- ECS EC2 - Per configurare Datadog con ECS EC2, segui questi passaggi dai [documenti Datadog](#).

Per ulteriori informazioni su Datadog

- [Documentazione Datadog](#)

## Tracciamento

### Important

Per implementare completamente il tracciamento, dovrai aggiornare l'applicazione. Per visualizzare tutti i dati disponibili dal servizio scelto, è necessario strumentare l'applicazione utilizzando le librerie applicabili.

## Monitora App Mesh con AWS X-Ray

AWS X-Ray è un servizio che fornisce strumenti che consentono di visualizzare, filtrare e analizzare i dati sulle richieste che l'applicazione serve. Queste informazioni ti aiutano a identificare problemi

e opportunità per ottimizzare la tua app. Puoi visualizzare informazioni dettagliate su richieste e risposte e le chiamate a valle che l'applicazione effettua verso altri AWS servizi.

X-Ray si integra con App Mesh per gestire i tuoi microservizi Envoy. I dati di tracciamento di Envoy vengono inviati al demone X-Ray in esecuzione nel contenitore.

Implementa X-Ray nel codice dell'applicazione utilizzando la guida [SDK](#) specifica per la tua lingua.

## Abilitare il monitoraggio di X-Ray tramite App Mesh

- A seconda del tipo di servizio:
  - ECS - Nella definizione del contenitore proxy Envoy, imposta la variabile di ambiente `ENABLE_ENVOY_XRAY_TRACING` su `1` e la variabile di ambiente `XRAY_DAEMON_PORT` su `2000`.
  - EKS - Nella configurazione App Mesh Controller, includi `--set tracing.enabled=true` e `--set tracing.provider=x-ray`.
- Nel tuo contenitore X-Ray, esponi la porta `2000` ed esegui come utente `1337`.

## Esempi di X-Ray

### Una definizione di contenitore Envoy per 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
}
```

## Aggiornamento del controller App Mesh per 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
```

## Procedure dettagliate per l'uso di X-Ray

- [Monitor con AWS X-Ray](#)
- [App Mesh con Amazon EKS - Osservabilità: X-Ray](#)
- [Tracciamento distribuito con X-Ray](#) in AWS App Mesh [officina](#)

## Per saperne di più su AWS X-Ray

- [AWS Documentazione radiografica](#)

## Risoluzione dei problemi di AWS X-Ray con App Mesh

- [Impossibile visualizzare le tracce AWS radiografiche per le mie applicazioni.](#)

## Jaeger per App Mesh con Amazon EKS

Jaeger è un sistema di tracciamento distribuito completo e open source. Può essere utilizzato per profilare le reti e per il monitoraggio. Jaeger può agevolare la risoluzione dei problemi relativi ad applicazioni cloud che possono essere utilizzate in modo da agevolare la risoluzione dei problemi relativi ai

Per implementare Jaeger nel codice della tua applicazione, puoi trovare la guida specifica per la tua lingua nelle [librerie di tracciamento](#) della documentazione Jaeger.

### Installazione di Jaeger tramite Helm

1. Aggiungi il repository EKS a Helm:

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

2. Installazione di App Mesh Installazione di App

```
helm upgrade -i appmesh-jaeger eks/appmesh-jaeger \
--namespace appmesh-system
```

### Esempio Jaeger

Di seguito è riportato un esempio di creazione di un archivio persistente `PersistentVolumeClaim` per 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
```

### Procedura dettagliata per l'utilizzo dello Jaeger

- [App Mesh con EKS: osservabilità: Jaeger](#)

## Per ulteriori informazioni su Jaeger

- [Documentazione Jaeger](#)

## Datadog per il tracciamento

Datadog può essere utilizzato sia per il tracciamento che per le metriche. Per ulteriori informazioni e istruzioni di installazione, trova la guida specifica per la lingua dell'applicazione nella [documentazione di Datadog](#).

# Strumenti App Mesh

App Mesh offre ai clienti la possibilità di interagire indirettamente con le sue API utilizzando strumenti come:

- AWS CloudFormation
- AWS Cloud Development Kit (AWS CDK)
- App Mesh Controller per Kubernetes
- Terraform

## App Mesh e AWS CloudFormation

AWS CloudFormation è un servizio che ti consente di creare un modello con tutte le risorse di cui hai bisogno per la tua applicazione, quindi AWS CloudFormation configurerà e fornirà le risorse per te. Inoltre configurerà tutte le dipendenze, in modo che possiate concentrarvi maggiormente sull'applicazione e meno sulla gestione delle risorse.

Per ulteriori informazioni ed esempi sull'utilizzo AWS CloudFormation con App Mesh, consulta la [AWS CloudFormation documentazione](#).

## App Mesh e AWS CDK

AWS CDK è un framework di sviluppo che consente di utilizzare il codice per definire l'infrastruttura cloud e utilizzarla AWS CloudFormation per il provisioning. AWS CDK supporta diversi linguaggi di programmazione tra cui Python TypeScript JavaScript, Java e C#/.Net.

Per ulteriori informazioni sull'utilizzo AWS CDK con App Mesh, consulta la [AWS CDK documentazione](#).

## Controller App Mesh per Kubernetes

Il controller App Mesh per Kubernetes ti aiuta a gestire le risorse App Mesh per un cluster Kubernetes e a inserire sidecar nei pod. Questo controller è specifico per l'uso con Amazon EKS e ti consente di gestire le tue risorse in un modo nativo di Kubernetes.

Per ulteriori informazioni sul controller App Mesh, consulta la [documentazione dell'App Mesh Controller](#).

# App Mesh e Terraform

[Terraform](#) è un'infrastruttura open source come strumento software di codice. Terraform può gestire i servizi cloud utilizzando la propria CLI e interagisce con le API utilizzando file di configurazione dichiarativi.

Per ulteriori informazioni sull'utilizzo di App Mesh con Terraform, consulta la documentazione di [Terraform](#).

## Utilizzo di mesh condivise

Puoi condividere le tue mesh App Mesh tra più AWS account utilizzando il AWS Resource Access Manager servizio. Una mesh condivisa consente alle risorse create da AWS account diversi di comunicare tra loro nella stessa mesh.

Un AWS account può essere il proprietario di una risorsa mesh, un consumatore mesh o entrambi. I consumatori possono creare risorse in una mesh condivisa con il proprio account. I proprietari possono creare risorse in qualsiasi mesh di proprietà dell'account. Il proprietario di una mesh può condividere una mesh con i seguenti tipi di consumatori di mesh.

- AWS Account specifici all'interno o all'esterno della sua organizzazione in AWS Organizations
- Un'unità organizzativa all'interno della propria organizzazione in AWS Organizations
- La sua intera organizzazione in AWS Organizations

Per una guida end-to-end dettagliata sulla condivisione di una mesh, consulta la [guida sulla rete mesh su GitHub più account](#).

## Concessione delle autorizzazioni per condividere le mesh

Quando si condividono le mesh tra account, sono necessarie le autorizzazioni per il principale IAM che condivide la mesh e le autorizzazioni a livello di risorsa necessarie per la mesh stessa.

### Concessione dell'autorizzazione alla condivisione di una mesh

È richiesto un set minimo di autorizzazioni affinché un principale IAM condivida una mesh. Ti consigliamo di utilizzare `AWSAppMeshFullAccess` e `AWSResourceAccessManagerFullAccess` gestire le policy IAM per garantire che i tuoi responsabili IAM dispongano delle autorizzazioni necessarie per condividere e utilizzare mesh condivise.

Se utilizzi una policy IAM personalizzata, sono necessarie le `appmesh:PutMeshPolicy` `appmesh>DeleteMeshPolicy` azioni e `appmesh:GetMeshPolicy` le relative azioni. Si tratta di azioni IAM che richiedono solo l'autorizzazione. Se a un principale IAM non sono concesse queste autorizzazioni, si verificherà un errore durante il tentativo di condividere la mesh utilizzando il servizio.

#### AWS RAM

Per ulteriori informazioni sul modo in cui il AWS Resource Access Manager servizio utilizza IAM, consulta [How AWS RAM uses IAM nella Guida](#) per l'AWS Resource Access Manager utente.

## Concessione delle autorizzazioni per una mesh

Una mesh condivisa dispone delle seguenti autorizzazioni.

- I consumatori possono elencare e descrivere tutte le risorse in una mesh condivisa con l'account.
- I proprietari possono elencare e descrivere tutte le risorse in qualsiasi mesh di proprietà dell'account.
- I proprietari e i consumatori possono modificare le risorse in una mesh creata dall'account, ma non possono modificare le risorse create da altri account.
- I consumatori possono eliminare qualsiasi risorsa in una mesh creata dall'account.
- I proprietari possono eliminare qualsiasi risorsa in una mesh creata da qualsiasi account.
- Le risorse del proprietario possono fare riferimento solo ad altre risorse dello stesso account. Ad esempio, un nodo virtuale può fare riferimento AWS Cloud Map solo a un AWS Certificate Manager certificato che si trova nello stesso account del proprietario del nodo virtuale.
- I proprietari e i consumatori possono connettere un proxy Envoy ad App Mesh come nodo virtuale di proprietà dell'account.
- I proprietari possono creare gateway virtuali e percorsi di gateway virtuali.
- I proprietari e i consumatori possono elencare i tag e possono taggare/rimuovere i tag dalle risorse in una mesh creata dall'account. Non possono elencare tag e risorse tag/rimuovere tag in una mesh che non sono state create dall'account.

Le mesh condivise utilizzano un'autorizzazione basata su policy. Una mesh viene condivisa con un set fisso di autorizzazioni. Queste autorizzazioni vengono selezionate per essere aggiunte a una politica delle risorse e una policy IAM opzionale può anche essere selezionata in base all'utente/ruolo IAM. L'intersezione delle autorizzazioni consentite in queste policy, meno le autorizzazioni esplicite negate, determina l'accesso del principale alla rete.

Quando si condivide una mesh, il AWS Resource Access Manager servizio crea una policy gestita denominata `AWSRAMDefaultPermissionAppMesh` e la associa all'App Mesh che fornisce le seguenti autorizzazioni.

- `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`

## Prerequisiti per la condivisione delle mesh

Per condividere una mesh, è necessario soddisfare i seguenti prerequisiti.

- Devi essere il proprietario della mesh del tuo AWS account. Non puoi condividere una mesh che è stata condivisa con te.
- Per condividere una mesh con la propria organizzazione o un'unità organizzativa AWS Organizations, è necessario abilitare la condivisione con AWS Organizations. Per ulteriori informazioni, consulta [Abilita la condivisione con AWS Organizations](#) nella Guida per l'utente AWS IAM.
- I tuoi servizi devono essere distribuiti in un Amazon VPC con connettività condivisa tra gli account che includono le risorse mesh con cui desideri comunicare tra loro. Un modo per condividere la

connettività di rete consiste nel distribuire tutti i servizi che desideri utilizzare nella rete mesh in una sottorete condivisa. Per ulteriori informazioni e limitazioni, consulta [Condivisione di una sottorete](#).

- I servizi devono essere rilevabili tramite DNS o AWS Cloud Map. Per ulteriori informazioni sulla scoperta dei servizi, consulta Nodi [virtuali](#).

## Servizi correlati

La condivisione della rete si integra con AWS Resource Access Manager (AWS RAM). AWS RAM è un servizio che ti consente di condividere AWS le tue risorse con qualsiasi AWS account o tramite AWS Organizations. Con AWS RAM, condividi le risorse di tua proprietà creando una condivisione di risorse. Una condivisione delle risorse specifica le risorse da condividere e gli utenti con cui condividerle. I consumatori possono essere singoli AWS account, unità organizzative o un'intera organizzazione AWS Organizations.

Per ulteriori informazioni in merito AWS RAM, consulta la [Guida AWS RAM per l'utente](#).

## Condivisione di una rete

La condivisione di una mesh consente alle risorse mesh create da account diversi di comunicare tra loro all'interno della stessa mesh. Puoi condividere solo una mesh di tua proprietà. Per condividere una mesh, è necessario aggiungerla a una condivisione di risorse. Una condivisione di risorse è una AWS RAM risorsa che consente di condividere le risorse tra AWS account. Una condivisione di risorse specifica le risorse da condividere e i consumatori con cui vengono condivise. Quando condividi una mesh utilizzando la console Amazon Linux, la aggiungi a una condivisione di risorse esistente. Per aggiungere la mesh a una nuova condivisione di risorse, crea la condivisione di risorse utilizzando la [AWS RAM console](#).

Se fai parte di un'organizzazione AWS Organizations e la condivisione all'interno dell'organizzazione è abilitata, ai consumatori dell'organizzazione può essere concesso automaticamente l'accesso alla rete mesh condivisa. In caso contrario, i consumatori ricevono un invito a partecipare alla condivisione di risorse e ottengono l'accesso alla mesh condivisa dopo aver accettato l'invito.

Puoi condividere una mesh di tua proprietà utilizzando la AWS RAM console o il AWS CLI.

Per condividere una mesh di tua proprietà utilizzando la AWS RAM console

Per istruzioni, consulta [Creazione di una condivisione di risorse](#) nella Guida AWS RAM per l'utente. Quando selezioni un tipo di risorsa, seleziona Mesh, quindi seleziona la mesh che desideri

condividere. Se non è elencata alcuna mesh, create prima una mesh. Per ulteriori informazioni, consulta [Creazione di una mesh di servizi](#).

Per condividere una mesh di tua proprietà, usa il AWS CLI

Utilizza il comando [create-resource-share](#). Per l'--resource-arns opzione, specificate l'ARN della mesh che desiderate condividere.

## Annullamento della condivisione di una mesh condivisa

Quando annulli la condivisione di una mesh, App Mesh disabilita l'ulteriore accesso alla mesh da parte degli ex utenti della mesh. Tuttavia, App Mesh non elimina le risorse create dai consumatori. Dopo che la mesh non è condivisa, solo il proprietario della mesh può accedere alle risorse ed eliminarle. App Mesh impedisce all'account che possedeva le risorse nella mesh di ricevere informazioni di configurazione dopo che la mesh non è stata condivisa. App Mesh impedisce inoltre a qualsiasi altro account con risorse nella mesh di ricevere informazioni di configurazione da una mesh non condivisa. Solo il proprietario della mesh può annullarne la condivisione.

Per annullare la condivisione di una mesh condivisa di tua proprietà, devi rimuoverla dalla condivisione di risorse. Puoi farlo utilizzando la AWS RAM console o il AWS CLI.

Per annullare la condivisione di una mesh condivisa di tua proprietà utilizzando la console AWS RAM

Per istruzioni, consulta [Aggiornamento di una condivisione di risorse](#) nella Guida per l'AWS RAM utente.

Per annullare la condivisione di una mesh condivisa di tua proprietà, utilizza il AWS CLI

Utilizza il comando [disassociate-resource-share](#).

## Identificare una mesh condivisa

Proprietari e consumatori possono identificare mesh e risorse mesh condivise utilizzando la console Amazon Linux e AWS CLI

Per identificare una mesh condivisa utilizzando la console Amazon Linux

1. Apri la console App Mesh all'[indirizzo https://console.aws.amazon.com/appmesh/](https://console.aws.amazon.com/appmesh/).
2. Dalla barra di navigazione a sinistra, seleziona Meshes. L'ID dell'account del proprietario della mesh per ogni mesh è elencato nella colonna Proprietario della mesh.

3. Dalla barra di navigazione a sinistra, seleziona Servizi virtuali, Router virtuali o Nodi virtuali. Vengono visualizzati l'ID dell'account del proprietario della rete Mesh e il proprietario della risorsa per ciascuna risorsa.

Per identificare una mesh condivisa utilizzando il AWS CLI

Usate il `aws appmesh list resource` comando, ad esempio `aws appmesh list-meshes`. Il comando restituisce le mesh che possiedi e le mesh condivise con te. La `meshOwner` proprietà mostra l'ID dell' AWS account `meshOwner` e la `resourceOwner` proprietà mostra l'ID dell' AWS account del proprietario della risorsa. Qualsiasi comando eseguito su qualsiasi risorsa mesh restituisce queste proprietà.

I tag definiti dall'utente che alleggi a una mesh condivisa sono disponibili solo per i tuoi Account AWS. Non sono disponibili per gli altri account con cui è condivisa la mesh. Al `aws appmesh list-tags-for-resource` comando per una mesh in un altro account viene negato l'accesso.

## Fatturazione e misurazione

Non sono previsti costi per la condivisione di una rete.

## Quote di istanze

Tutte le quote per una mesh si applicano anche alle mesh condivise, indipendentemente da chi ha creato le risorse nella mesh. Solo il proprietario di una mesh può richiedere aumenti delle quote. Per ulteriori informazioni, consulta [Quotazioni di App Mesh Controller](#). Il AWS Resource Access Manager servizio prevede anche delle quote. Per ulteriori informazioni, consulta [Service Quotas](#).

# AWSservizi integrati con App Mesh

App Mesh offre altriAWS servizi per fornire soluzioni aggiuntive per le sfide aziendali. In questo argomento sono identificati i servizi che utilizzano App Mesh per aggiungere funzionalità, o i servizi utilizzati da App Mesh per eseguire le attività.

## Indice

- [Creazione di risorse App Mesh conAWS CloudFormation](#)
- [App Mesh sugliAWS Outposts](#)

## Creazione di risorse App Mesh conAWS CloudFormation

App Mesh è integrato conAWS CloudFormation, un servizio che consente di modellare e configurare leAWS risorse in modo da dedicare meno tempo alla creazione e alla gestione delle risorse e dell'infrastruttura. È possibile creare un modello che descriva tutte leAWS risorse che desideri, ad esempio una mesh App Mesh. In questoAWS CloudFormation modo gestirà il provisioning e la configurazione delle risorse per te.

Quando usiAWS CloudFormation, puoi riutilizzare il modello per configurare le risorse di App Mesh in modo coerente e continuo. È sufficiente descrivere le risorse una volta e quindi allestire le stesse risorse più volte in più regioni e account AWS.

## App Mesh eAWS CloudFormation modelli

Per eseguire il provisioning e la configurazione delle risorse per App Mesh e i servizi correlati, devi conoscere i [AWS CloudFormationmodelli](#). I modelli sono file di testo formattati in JSON o YAML. Questi modelli descrivono le risorse di cui intendi effettuare il provisioning negli stack AWS CloudFormation. Se non hai familiarità con JSON o YAML, puoi usare AWS CloudFormation Designer per iniziare a utilizzare i modelli AWS CloudFormation. Per ulteriori informazioni, consulta [Che cos'è AWS CloudFormation Designer?](#) nella Guida per l'utente di AWS CloudFormation.

App Mesh supporta la creazione di mesh, percorsi, nodi virtuali, router virtuali e servizi virtuali inAWS CloudFormation. Per ulteriori informazioni, inclusi esempi di modelli JSON e YAML per le risorse App Mesh, [consulRiferimento dei tipi di risorse App Mesh](#) nella Guida per l'AWS CloudFormationutente di.

## Ulteriori informazioni su AWS CloudFormation

Per ulteriori informazioni su AWS CloudFormation, consulta le seguenti risorse:

- [AWS CloudFormation](#)
- [Guida per l'utente di AWS CloudFormation](#)
- [Guida per l'utente dell'interfaccia a riga di comando di AWS CloudFormation](#)

## App Mesh sugliAWS Outposts

AWSOutposts abilitaAWS servizi nativi, infrastrutture e modelli operativi nelle strutture on-premise. Negli ambientiAWS OutpostsAWS è possibile utilizzare le stesse API, strumenti e infrastruttura utilizzati inAWS Cloud. App Mesh suAWS Outposts è ideale per carichi di lavoro a bassa latenza che devono essere eseguiti in prossimità di dati e applicazioni on-premise. Per ulteriori informazioni sugliAWS Outpost, consulta la [Guida per l'AWSOutposts](#).

### Prerequisiti

Di seguito sono riportati i prerequisiti per l'utilizzo di App Mesh suAWS Outposts:

- Devi aver installato e configurato un Outpost nel data center locale.
- É necessaria una connessione di rete affidabile tra l'Outpost e la relativa Regione AWS.
- LaAWS Regione per l'Outpost deve supportareAWS App Mesh. Per un elenco delle regioni supportate, consulta [AWS App MeshEndpoints and Quotas](#) in Riferimenti generali di AWS.

### Restrizioni

Di seguito sono riportate le limitazioni dell'utilizzo di App Mesh suAWS Outposts:

- AWS Identity and Access Management, Application Load Balancer Network Load Balancer, Classic Load Balancer e Amazon Route 53 vengono eseguiti nellaAWS regione, non sugli Outposts. In tal modo si aumenta la latenza tra questi servizi e i container.

### Considerazioni sulla connettività di rete

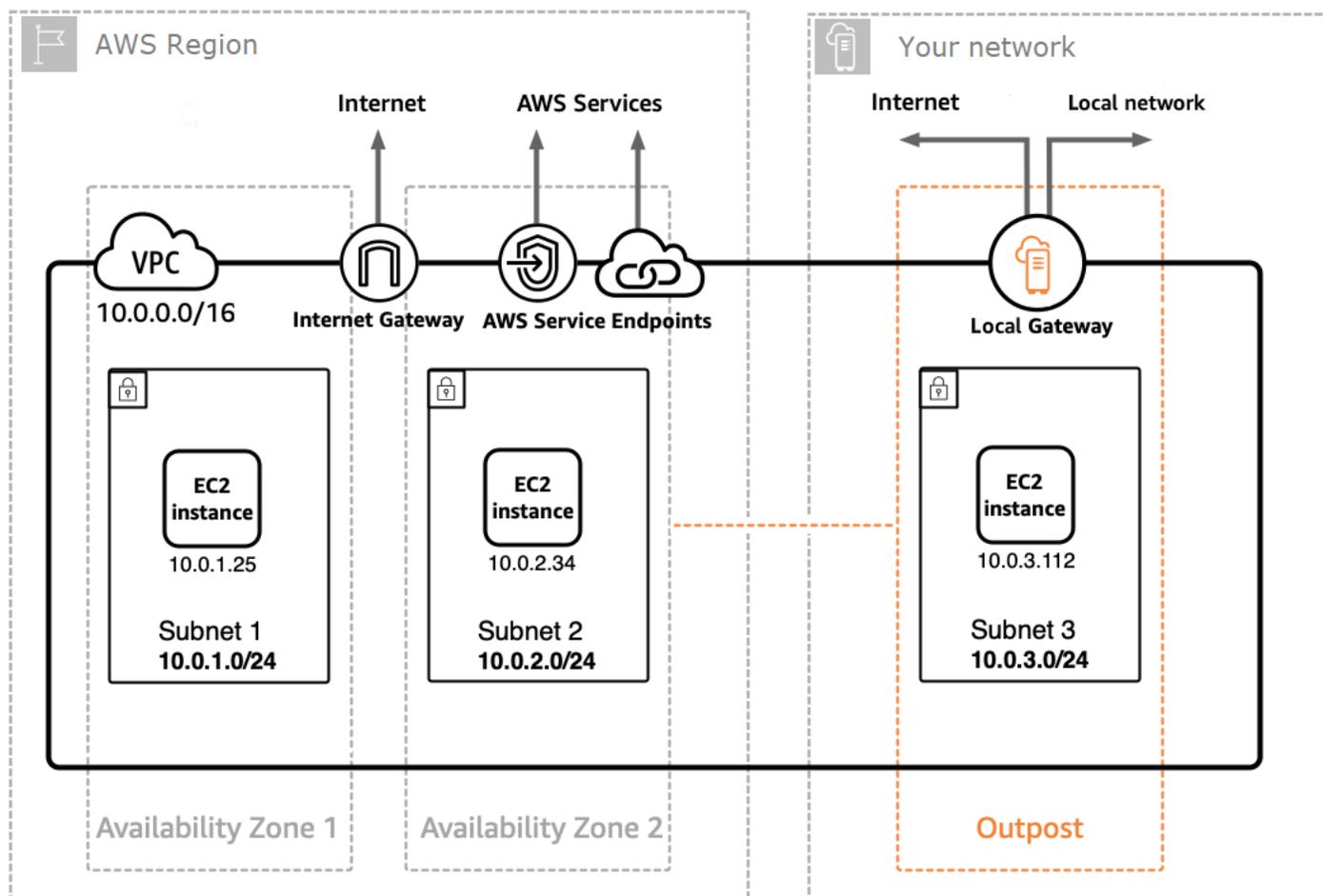
Di seguito sono riportate le considerazioni sulla connettività di rete per Amazon EKSAWS Outposts:

- Se la connettività di rete tra Outpost e la relativaAWS Regione viene persa, i proxy App Mesh Envoy continuano a funzionare. Tuttavia, non sarà possibile modificare la service mesh fino al ripristino della connettività.

- È consigliabile fornire una connettività affidabile, altamente disponibile e a bassa latenza tra l'Outpost e la relativa AWS regione.

## Creazione di un proxy App Mesh Envoy su un Outpost

Un Outpost è un'estensione di una AWS regione e puoi estendere un Amazon VPC in un account per includere più zone di disponibilità e qualsiasi Outpost. Quando si configura l'Outpost, si associa a esso un gruppo di sottoreti per estendere l'ambiente regionale del VPC alla struttura locale. Le istanze di un Outpost appaiono come parte del VPC regionale, simile a una zona di disponibilità con sottoreti associate.



Per creare un proxy App Mesh Envoy su un Outpost, aggiungi l'immagine del contenitore App Mesh Envoy al task Amazon ECS o al pod Amazon EKS in esecuzione su un Outpost. Per ulteriori informazioni, consulta [Amazon Elastic Container Service on AWS Outposts](#) nella Amazon Elastic Container Service Developer Guide e [Amazon Elastic Kubernetes Service on AWS Outposts](#) nella Amazon EKS User Guide.

# Best practice per App Mesh

Per raggiungere l'obiettivo di azzerare le richieste non riuscite durante le implementazioni pianificate e durante la perdita non pianificata di alcuni host, le best practice riportate in questo argomento implementano la seguente strategia:

- Aumentate la probabilità che una richiesta abbia successo dal punto di vista dell'applicazione utilizzando una strategia di riprova predefinita sicura. Per ulteriori informazioni, consulta [Strumenta tutti i percorsi con nuovi tentativi](#).
- Aumentate la probabilità che una nuova richiesta abbia successo massimizzando la probabilità che la richiesta riprovata venga inviata a una destinazione effettiva. Per ulteriori informazioni, consulta [Regola la velocità di implementazione](#), [Scalabilità orizzontale prima della scalabilità interna](#) e [Implementa controlli sullo stato dei container](#).

Per ridurre o eliminare in modo significativo gli errori, si consiglia di implementare i consigli in tutte le seguenti pratiche.

## Strumenta tutti i percorsi con nuovi tentativi

Configura tutti i servizi virtuali per utilizzare un router virtuale e imposta una politica di riprova predefinita per tutte le rotte. Ciò ridurrà le richieste non riuscite rifezionando un host e inviando una nuova richiesta. Per le politiche relative ai nuovi tentativi, consigliamo di impostare un valore di almeno due per `maxRetries` e di specificare le seguenti opzioni per ogni tipo di evento di nuovo tentativo in ogni tipo di percorso che supporta il tipo di evento `Retry`:

- TCP — `connection-error`
- HTTP e HTTP/2 — `stream-error gateway-error`
- gRPC — `cancelled unavailable`

È necessario prendere in considerazione altri casi di tentativi, in case-by-case quanto potrebbero non essere sicuri, ad esempio se la richiesta non è idempotente. Dovrai considerare e testare valori per `perRetryTimeout` che consentano di trovare il giusto compromesso tra la latenza massima di una richiesta (`maxRetries*perRetryTimeout`) `maxRetries` e la maggiore percentuale di successo di più tentativi. Inoltre, quando Envoy tenta di connettersi a un endpoint che non è più presente, dovresti

aspettarti che quella richiesta consumi completamente. `perRetryTimeout` Per configurare una politica di riprova, visualizza [Creazione di un routing](#) e seleziona il protocollo che desideri instradare.

### Note

Se hai implementato un percorso a partire dal 29 luglio 2020 e non hai specificato una politica sui nuovi tentativi, App Mesh potrebbe aver creato automaticamente una politica di nuovo tentativo predefinita simile alla politica precedente per ogni percorso creato a partire dal 29 luglio 2020. Per ulteriori informazioni, consulta [Politica predefinita per i nuovi tentativi di routing](#).

## Regola la velocità di implementazione

Quando utilizzi le distribuzioni in sequenza, riduci la velocità complessiva di implementazione. Per impostazione predefinita, Amazon ECS configura una strategia di distribuzione con almeno il 100% di attività integre e il 200% di attività totali. In fase di implementazione, ciò si traduce in due punti di forte deviazione:

- La dimensione totale della flotta di nuove attività potrebbe essere visibile agli Envoy prima di essere pronti a completare le richieste (vedi [Implementa controlli sullo stato dei container](#) per le mitigazioni).
- La dimensione del 100% della flotta delle vecchie attività potrebbe essere visibile agli Envoy mentre le attività vengono terminate.

Se configurati con questi vincoli di implementazione, gli orchestratori di container possono entrare in uno stato in cui nascondono contemporaneamente tutte le vecchie destinazioni e rendono visibili tutte le nuove destinazioni. Poiché alla fine il piano dati di Envoy è coerente, ciò può comportare periodi in cui l'insieme di destinazioni visibili nel piano dati si discosta dal punto di vista dell'orchestrator. Per mitigare questo problema, consigliamo di mantenere almeno il 100% delle attività integre, ma di ridurre le attività totali al 125%. Ciò ridurrà le divergenze e migliorerà l'affidabilità dei nuovi tentativi. Consigliamo le seguenti impostazioni per i diversi runtime dei contenitori:

### Amazon ECS

Se il servizio ha un conteggio desiderato di due o tre, impostalo `maximumPercent` al 150 per cento. Altrimenti, `maximumPercent` impostalo al 125 per cento.

## Kubernetes

Configura la distribuzione `update strategy`, `maxUnavailable` impostandola sullo 0 per cento e `maxSurge` sul 25 per cento. [Per ulteriori informazioni sulle implementazioni, consulta la documentazione di Kubernetes Deployments.](#)

## Scalabilità orizzontale prima della scalabilità interna

Sia lo scale-out che lo scale-in possono comportare una certa probabilità che le richieste non vadano a buon fine nei nuovi tentativi. Sebbene esistano raccomandazioni sulle attività che riducono la scalabilità orizzontale, l'unico consiglio per la scalabilità orizzontale è ridurre al minimo la percentuale di attività con scalabilità orizzontale in qualsiasi momento. Ti consigliamo di utilizzare una strategia di distribuzione che riduca le nuove attività di Amazon ECS o le implementazioni Kubernetes prima di passare a attività o distribuzioni vecchie. Questa strategia di scalabilità mantiene la percentuale di attività o implementazioni scalabili più bassa, pur mantenendo la stessa velocità. Questa pratica si applica sia alle attività di Amazon ECS che alle distribuzioni Kubernetes.

## Implementa controlli sullo stato dei container

Nello scenario di scalabilità verticale, i contenitori di un'attività Amazon ECS potrebbero non funzionare correttamente e potrebbero non essere inizialmente reattivi. Consigliamo i seguenti suggerimenti per diversi tempi di esecuzione dei container:

### Amazon ECS

Per mitigare questo problema, consigliamo di utilizzare i controlli dello stato dei container e l'ordinamento delle dipendenze dei contenitori per garantire che Envoy sia funzionante e integro prima dell'avvio di qualsiasi contenitore che richieda la connettività di rete in uscita. [Per configurare correttamente un contenitore di applicazioni e un contenitore Envoy in una definizione di attività, consulta Container dependency.](#)

### Kubernetes

[Nessuna, perché le sonde di vivacità e prontezza di Kubernetes non vengono prese in considerazione nella registrazione e nella cancellazione delle istanze nel AWS Cloud Map controller App Mesh per Kubernetes.](#) [Per ulteriori GitHub informazioni, consulta il numero #132.](#)

## Ottimizza la risoluzione DNS

Se utilizzi DNS per l'individuazione dei servizi, è essenziale selezionare il protocollo IP appropriato per ottimizzare la risoluzione DNS durante la configurazione delle mesh. App Mesh supporta entrambi IPv4 e IPv6, e la tua scelta può influire sulle prestazioni e sulla compatibilità del servizio. Se la tua infrastruttura non supporta IPv6, ti consigliamo di specificare un'impostazione IP che sia in linea con l'infrastruttura anziché fare affidamento sul comportamento predefinito IPv6\_PREFERRED. Il comportamento predefinito IPv6\_PREFERRED può ridurre le prestazioni del servizio.

- **IPv6\_Preferred:** questa è l'impostazione predefinita. Envoy esegue prima una ricerca DNS per gli indirizzi IPv6 e torna indietro se non viene trovato alcun indirizzo. IPv4 IPv6 Ciò è utile se l'infrastruttura supporta principalmente ma necessita di compatibilità. IPv6 IPv4
- **IPv4\_Preferred:** Envoy cerca innanzitutto IPv4 gli indirizzi e torna indietro se non sono disponibili indirizzi. IPv6 IPv4 Utilizza questa impostazione se la tua infrastruttura supporta principalmente ma presenta una certa compatibilità. IPv4 IPv6
- **IPv6\_only:** scegli questa opzione se i tuoi servizi supportano esclusivamente il traffico. IPv6 Envoy esegue solo ricerche DNS per gli IPv6 indirizzi, assicurando che tutto il traffico venga instradato. IPv6
- **IPv4\_only:** scegli questa impostazione se i tuoi servizi supportano esclusivamente il traffico. IPv4 Envoy esegue solo ricerche DNS per gli IPv4 indirizzi, assicurando che tutto il traffico venga instradato. IPv4

È possibile impostare le preferenze della versione IP sia a livello di mesh che a livello di nodo virtuale, con le impostazioni del nodo virtuale che hanno la precedenza su quelle a livello di mesh.

Per ulteriori informazioni, consulta [Service Meshes](#) and [Virtual Nodes](#).

# Sicurezza in AWS App Mesh

La sicurezza del cloud AWS è la massima priorità. In qualità di AWS cliente, puoi beneficiare di un data center e di un'architettura di rete progettati per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

La sicurezza è una responsabilità condivisa tra AWS e te. Il [modello di responsabilità condivisa](#) descrive questo come sicurezza del cloud e sicurezza nel cloud:

- **Sicurezza del cloud:** AWS è responsabile della protezione dell'infrastruttura che gestisce AWS i servizi nel AWS cloud. AWS ti fornisce anche servizi che puoi utilizzare in modo sicuro. I revisori di terze parti testano e verificano regolarmente l'efficacia della sicurezza come parte dei [programmi di conformitàAWS](#). Per ulteriori informazioni sui programmi di conformità che si applicano a AWS App Mesh, consulta [Servizi coperti dal programma di conformitàAWS](#). App Mesh è responsabile della fornitura sicura della configurazione ai proxy locali, inclusi segreti come le chiavi private dei TLS certificati.
- **Sicurezza nel cloud:** la tua responsabilità è determinata dal AWS servizio che utilizzi. Sei responsabile anche di altri fattori, tra cui:
  - La sensibilità dei dati, i requisiti aziendali e le leggi e i regolamenti applicabili.
  - La configurazione di sicurezza del piano dati App Mesh, inclusa la configurazione dei gruppi di sicurezza che consentono il passaggio del traffico tra i servizi all'interno dell'utenteVPC.
  - La configurazione delle risorse di calcolo associate ad App Mesh.
  - Le IAM policy associate alle risorse di calcolo e la configurazione che possono recuperare dal piano di controllo App Mesh.

Questa documentazione ti aiuta a capire come applicare il modello di responsabilità condivisa quando usi App Mesh. I seguenti argomenti mostrano come configurare App Mesh per soddisfare i tuoi obiettivi di sicurezza e conformità. Scopri anche come utilizzare altri AWS servizi che ti aiutano a monitorare e proteggere le tue risorse App Mesh.

## Principio di sicurezza App Mesh

I clienti dovrebbero essere in grado di regolare la sicurezza nella misura necessaria. La piattaforma non dovrebbe impedire loro di essere più sicuri. Le funzionalità della piattaforma sono sicure per impostazione predefinita.

## Argomenti

- [Transport Layer Security \(TLS\)](#)
- [Autenticazione TLS reciproca](#)
- [In che modo AWS App Mesh funziona con IAM](#)
- [Registrazione delle chiamate AWS App Mesh API utilizzando AWS CloudTrail](#)
- [Protezione dei dati in AWS App Mesh](#)
- [Convalida della conformità per AWS App Mesh](#)
- [Sicurezza dell'infrastruttura in AWS App Mesh](#)
- [Resilienza in AWS App Mesh](#)
- [Analisi della configurazione e della vulnerabilità in AWS App Mesh](#)

## Transport Layer Security (TLS)

In App Mesh, Transport Layer Security (TLS) crittografa la comunicazione tra i proxy Envoy distribuiti su risorse di calcolo rappresentate in App Mesh da endpoint mesh, come e. [Nodi virtuali Gateway virtuali virtuale virtuale virtuale virtuale virtuale](#) Il proxy negozia e termina. TLS Quando il proxy viene distribuito con un'applicazione, il codice dell'applicazione non è responsabile della negoziazione di una sessione. TLS Il proxy negozia per TLS conto dell'applicazione.

App Mesh consente di fornire il TLS certificato al proxy nei seguenti modi:

- Un certificato privato di AWS Certificate Manager (ACM) rilasciato da un AWS Private Certificate Authority (AWS Private CA).
- Un certificato memorizzato nel file system locale di un nodo virtuale emesso dalla propria autorità di certificazione (CA)
- Un certificato fornito da un endpoint Secrets Discovery Service (SDS) tramite Unix Domain Socket locale.

[Autorizzazione proxy Envoy Proxy](#) deve essere abilitato per il proxy Envoy distribuito rappresentato da un endpoint mesh. Quando abiliti l'autorizzazione proxy, ti consigliamo di limitare l'accesso solo all'endpoint mesh per cui stai abilitando la crittografia.

## Requisiti del certificato

Uno dei nomi alternativi del soggetto (SANs) sul certificato deve soddisfare criteri specifici, a seconda di come viene scoperto il servizio effettivo rappresentato da un endpoint mesh.

- DNS— Uno dei certificati SANs deve corrispondere al valore fornito nelle impostazioni di rilevamento del DNS servizio. Per un'applicazione con il nome di rilevamento del servizio `mesh-endpoint.apps.local`, è possibile creare un certificato corrispondente a quel nome o un certificato con la wild card `*.apps.local`.
- AWS Cloud Map— Uno dei certificati SANs deve corrispondere al valore fornito nelle impostazioni di individuazione del AWS Cloud Map servizio utilizzando il formato `service-name.namespace-name`. Per un'applicazione con le impostazioni di rilevamento del AWS Cloud Map servizio di `serviceName mesh-endpoint` e `namespaceName apps.local`, è possibile creare un certificato corrispondente al nome `mesh-endpoint.apps.local` o un certificato con la wild card `*.apps.local`.

Per entrambi i meccanismi di rilevamento, se nessuno dei certificati SANs corrisponde alle impostazioni di rilevamento del DNS servizio, la connessione tra Envoy fallisce e viene visualizzato il seguente messaggio di errore, visualizzato dal client Envoy.

```
TLS error: 268435581:SSL routines:OPENSSL_internal:CERTIFICATE_VERIFY_FAILED
```

## TLS certificati di autenticazione

App Mesh supporta più fonti per i certificati quando si utilizza TLS l'autenticazione.

### AWS Private CA

Il certificato deve essere archiviato ACM nella stessa regione e nello stesso AWS account dell'endpoint mesh che utilizzerà il certificato. Non è necessario che il certificato della CA si trovi nello stesso AWS account, ma deve comunque trovarsi nella stessa regione dell'endpoint mesh. Se non ne hai uno CA privata AWS, devi [crearne uno](#) prima di poterne richiedere un certificato. Per ulteriori informazioni sulla richiesta di un certificato da un utente esistente ACM, AWS Private CA consulta [Richiedere un certificato privato](#). Il certificato non può essere un certificato pubblico.

L'utente privato CAs utilizzato per le politiche TLS dei client deve essere un utente rootCAs.

Per configurare un nodo virtuale con certificati e CAs da AWS Private CA, il principale (come un utente o un ruolo) che usi per chiamare App Mesh deve disporre delle seguenti IAM autorizzazioni:

- Per tutti i certificati aggiunti alla TLS configurazione di un listener, il principale deve disporre dell'`acm:DescribeCertificate` autorizzazione.
- Per qualsiasi CAs configurazione basata su una politica TLS client, il principale deve disporre dell'`acm-pca:DescribeCertificateAuthority` autorizzazione.

#### Important

La condivisione CAs con altri account può conferire a tali account privilegi involontari alla CA. Si consiglia di utilizzare politiche basate sulle risorse per limitare l'accesso solo `acm-pca:DescribeCertificateAuthority` agli account che non devono emettere certificati dalla CA. `acm-pca:GetCertificateAuthorityCertificate`

È possibile aggiungere queste autorizzazioni a una IAM politica esistente associata a un principale o creare un nuovo principale e una nuova politica e allegare la politica al principale. Per ulteriori informazioni, vedere [Modifica dei IAM criteri](#), [Creazione IAM di criteri](#) e [Aggiungere autorizzazioni di IAM identità](#).

#### Note

Paghi una tariffa mensile per il funzionamento di ciascuno di essi AWS Private CA fino a quando non lo elimini. Paghi anche i certificati privati che emetti ogni mese e i certificati privati che esporti. Per ulteriori informazioni, consulta la sezione [Prezzi di AWS Certificate Manager](#).

Quando abiliti [l'autorizzazione proxy](#) per l'Envoy Proxy rappresentato da un endpoint mesh, al IAM ruolo che utilizzi devono essere assegnate le seguenti autorizzazioni: IAM

- Per tutti i certificati configurati sul listener di un nodo virtuale, il ruolo deve disporre dell'autorizzazione. `acm:ExportCertificate`
- Per tutti i ruoli CAs configurati in base a una politica TLS client, il ruolo deve disporre dell'`acm-pca:GetCertificateAuthorityCertificate` autorizzazione.

## File system

È possibile distribuire certificati a Envoy utilizzando il file system. È possibile farlo rendendo disponibili la catena di certificati e la chiave privata corrispondente nel percorso del file. In questo modo, queste risorse sono raggiungibili dal proxy sidecar Envoy.

## Servizio di scoperta segreto di Envoy ( ) SDS

Envoy recupera segreti come i TLS certificati da un endpoint specifico tramite il protocollo Secrets Discovery. [Per ulteriori informazioni su questo protocollo, consulta la documentazione di Envoy.](#)  
[SDS](#)

App Mesh configura il proxy Envoy per utilizzare un Unix Domain Socket locale al proxy per fungere da endpoint Secret Discovery Service (SDS) quando SDS funge da origine per i certificati e le catene di certificati. È possibile configurare il percorso di questo endpoint utilizzando la variabile di ambiente. `APPMESH_SDS_SOCKET_PATH`

### Important

Local Secrets Discovery Service che utilizza Unix Domain Socket è supportato sul proxy App Mesh Envoy versione 1.15.1.0 e successive.

App Mesh supporta il SDS protocollo V2 utilizzando g. RPC

## Integrazione con SPIFFE Runtime Environment ( ) SPIRE

È possibile utilizzare qualsiasi implementazione collaterale di SDSAPI, incluse le toolchain esistenti come [SPIFFERuntime Environment](#) ( ). SPIRE è progettato per consentire l'implementazione dell'autenticazione reciproca tra più carichi di lavoro in sistemi distribuiti. Attesta l'identità dei carichi di lavoro in fase di esecuzione. SPIRE fornisce inoltre chiavi e certificati specifici per i carichi di lavoro, di breve durata e con rotazione automatica direttamente ai carichi di lavoro.

È necessario configurare l'agente come provider per Envoy SPIRE. SDS Consente di fornire direttamente a Envoy il materiale chiave di cui ha bisogno per fornire l'autenticazione reciproca. TLS Esegui SPIRE gli agenti nei sidecar accanto ai proxy Envoy. L'agente si occupa della rigenerazione delle chiavi e dei certificati di breve durata, se necessario. L'agente attesta Envoy e determina quali identità di servizio e certificati CA deve rendere disponibili a Envoy quando Envoy si connette al server esposto dall'agente. SDS SPIRE

Durante questo processo, le identità di servizio e i certificati CA vengono ruotati e gli aggiornamenti vengono trasmessi in streaming a Envoy. Envoy li applica immediatamente alle nuove connessioni senza interruzioni o tempi di inattività e senza che le chiavi private tocchino mai il file system.

## In che modo App Mesh configura gli Envoy per negoziare TLS

App Mesh utilizza la configurazione degli endpoint mesh sia del client che del server per determinare come configurare la comunicazione tra Envoy in una mesh.

### Con le politiche dei clienti

Quando una policy client impone l'uso di TLS e una delle porte nella policy client corrisponde alla porta della policy del server, la policy client viene utilizzata per configurare il contesto di TLS convalida del client. Ad esempio, se la politica client di un gateway virtuale corrisponde alla politica del server di un nodo virtuale, verrà tentata la TLS negoziazione tra i proxy utilizzando le impostazioni definite nella politica client del gateway virtuale. Se la politica del client non corrisponde alla politica della porta del server, è possibile negoziare o meno TLS tra i proxy, a seconda delle impostazioni della politica del server. TLS

### Senza politiche client

Se il client non ha configurato una politica client o la politica del client non corrisponde alla porta del server, App Mesh utilizzerà il server per determinare se negoziare TLS o meno con il client e come. Ad esempio, se un gateway virtuale non ha specificato una policy client e un nodo virtuale non ha configurato la TLS terminazione, non TLS verrà negoziata tra i proxy. Se un client non ha specificato una politica client corrispondente e un server è stato configurato con TLS modalità STRICT o PERMISSIVE, i proxy verranno configurati per la negoziazione. TLS A seconda del modo in cui i certificati sono stati forniti per la TLS terminazione, si applica il seguente comportamento aggiuntivo.

- ACM-managed TLS certificates: quando un server ha configurato la TLS terminazione utilizzando un certificato ACM -managed, App Mesh configura automaticamente i client per negoziare TLS e convalidare il certificato rispetto alla CA dell'utente root a cui il certificato è collegato.
- TLS Certificati basati su file: quando un server ha configurato la TLS terminazione utilizzando un certificato del file system locale del proxy, App Mesh configura automaticamente un client per la negoziazione TLS, ma il certificato del server non viene convalidato.

## Nomi alternativi dei soggetti

Facoltativamente, puoi specificare un elenco di nomi alternativi dei soggetti (SANs) di cui fidarti. SANs deve essere nel URI formato FQDN o. Se SANs forniti, Envoy verifica che il nome alternativo del soggetto del certificato presentato corrisponda a uno dei nomi in questo elenco.

Se non lo specifichi SANs sull'endpoint mesh di terminazione, il proxy Envoy per quel nodo non verifica il certificato su un client peer. SAN Se non lo specifichi SANs sull'endpoint mesh di origine, il certificato fornito dall'endpoint terminante deve corrispondere alla configurazione di rilevamento del servizio di endpoint mesh. SAN

Per ulteriori informazioni, consulta App Mesh [TLS: requisiti del certificato](#).

### Important

È possibile utilizzare i caratteri jolly solo SANs se la policy del client per TLS è impostata su. not enforced Se la policy client per il nodo virtuale o il gateway virtuale del client è configurata per essere applicata TLS, non può accettare caratteri jolly. SAN

## Verifica la crittografia

Una volta abilitata TLS, puoi interrogare il proxy Envoy per confermare che la comunicazione sia crittografata. Il proxy Envoy emette statistiche sulle risorse che possono aiutarti a capire se la tua TLS comunicazione funziona correttamente. Ad esempio, il proxy Envoy registra le statistiche sul numero di TLS handshake riuscite negoziate per un endpoint mesh specificato. Determina quante TLS strette di mano riuscite sono state eseguite per un endpoint mesh denominato con il seguente comando.

*my-mesh-endpoint*

```
curl -s 'http://my-mesh-endpoint.apps.local:9901/stats' | grep ssl.handshake
```

Nell'esempio seguente, l'output restituito è che ci sono state tre strette di mano per l'endpoint mesh, quindi la comunicazione è crittografata.

```
listener.0.0.0.0_15000.ssl.handshake: 3
```

Il proxy Envoy emette statistiche anche quando la negoziazione fallisce. TLS Determina se ci sono stati TLS errori per l'endpoint della mesh.

```
curl -s 'http://my-mesh-endpoint.apps.local:9901/stats' | grep -e "ssl.*\((fail|error\n)"
```

Nell'esempio restituito dall'output, non ci sono stati errori per diverse statistiche, quindi la TLS negoziazione è riuscita.

```
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.fail_verify_san: 0
```

[Per ulteriori informazioni sulle statistiche di Envoy, vedere Envoy TLS Listener Statistics.](#)

## Rinnovo del certificato

### AWS Private CA

Quando rinnovi un certificato con ACM, il certificato rinnovato verrà distribuito automaticamente ai proxy collegati entro 35 minuti dal completamento del rinnovo. Ti consigliamo di utilizzare il rinnovo gestito per rinnovare automaticamente i certificati verso la fine del periodo di validità. Per ulteriori informazioni, consulta la sezione [Rinnovo gestito per i ACM certificati emessi da Amazon nella Guida](#) per l' AWS Certificate Manager utente.

### Il tuo certificato

Quando si utilizza un certificato del file system locale, Envoy non ricaricherà automaticamente il certificato quando viene modificato. È possibile riavviare o ridistribuire il processo Envoy per caricare un nuovo certificato. È inoltre possibile inserire un certificato più recente in un percorso di file diverso e aggiornare la configurazione del nodo virtuale o del gateway con quel percorso di file.

## Configura i ECS carichi di lavoro Amazon per utilizzare l' TLS autenticazione con AWS App Mesh

Puoi configurare la tua mesh per utilizzare l' TLS autenticazione. Assicurati che i certificati siano disponibili per i sidecar proxy di Envoy che aggiungi ai tuoi carichi di lavoro. Puoi allegare un EFS volume EBS o al tuo sidecar Envoy oppure puoi archiviare e recuperare i certificati da Secrets Manager. AWS

- Se utilizzi la distribuzione di certificati basata su file, allega un volume EBS or EFS al sidecar Envoy. Assicurati che il percorso del certificato e della chiave privata corrisponda a quello configurato in. AWS App Mesh
- Se utilizzi la distribuzione SDS basata, aggiungi un sidecar che implementa Envoy's SDS API con accesso al certificato.

#### Note

SPIREnon è supportato su AmazonECS.

## Configura i carichi di lavoro Kubernetes per utilizzare l'autenticazione con TLS AWS App Mesh

Puoi configurare AWS App Mesh Controller for Kubernetes per abilitare l'TLSautenticazione per i backend e i listener dei servizi di nodi e gateway virtuali. Assicurati che i certificati siano disponibili per i sidecar proxy Envoy che aggiungi ai tuoi carichi di lavoro. Puoi vedere un esempio per ogni tipo di distribuzione nella sezione [dettagliata di Autenticazione reciproca](#). TLS

- Se utilizzi la distribuzione di certificati basata su file, allega un EFS volume EBS or al sidecar di Envoy. Assicurati che il percorso del certificato e della chiave privata corrisponda a quello configurato nel controller. In alternativa, puoi utilizzare un Kubernetes Secret montato sul file system.
- Se utilizzi la distribuzione SDS basata, dovresti configurare un SDS provider locale di nodi che implementi Envoy's. SDS API Envoy lo raggiungerà. UDS Per abilitare il TLS supporto SDS based m nel EKS AppMesh controller, imposta il `enable-sds` flag su `true` e fornisci il UDS percorso del SDS provider locale al controller tramite il `sds-uds-path` flag. Se usi helm, li imposti come parte dell'installazione del controller:

```
--set sds.enabled=true
```

#### Note

Non potrai utilizzare SPIRE per distribuire i tuoi certificati se utilizzi Amazon Elastic Kubernetes Service (Amazon) in modalità Fargate. EKS

## Autenticazione TLS reciproca

L'autenticazione Mutual TLS (Transport Layer Security) è un componente opzionale di TLS che offre l'autenticazione peer bidirezionale. L'autenticazione TLS reciproca aggiunge un livello di sicurezza rispetto a TLS e consente ai servizi di verificare il client che effettua la connessione.

Il client nella relazione client-server fornisce anche un certificato X.509 durante il processo di negoziazione della sessione. Il server utilizza questo certificato per identificare e autenticare il client. Questo processo consente di verificare se il certificato è emesso da un'autorità di certificazione (CA) affidabile e se il certificato è valido. Utilizza inoltre il Subject Alternative Name (SAN) sul certificato per identificare il client.

È possibile abilitare l'autenticazione TLS reciproca per tutti i protocolli supportati da AWS App Mesh. Sono TCP, HTTP/1.1, HTTP/2, gRPC.

### Note

Utilizzando App Mesh, puoi configurare l'autenticazione TLS reciproca per le comunicazioni tra i proxy Envoy dei tuoi servizi. Tuttavia, le comunicazioni tra le applicazioni e i proxy Envoy non sono crittografate.

## Certificati di autenticazione TLS reciproca

AWS App Mesh supporta due possibili fonti di certificati per l'autenticazione TLS reciproca. I certificati client in una politica client TLS e la convalida del server in una configurazione TLS del listener possono provenire da:

- **File system:** certificati dal file system locale del proxy Envoy in esecuzione. Per distribuire i certificati a Envoy, devi fornire i percorsi dei file per la catena di certificati e la chiave privata all'API App Mesh.
- **Secret Discovery Service (SDS) di Envoy:** ring-your-own sidecar B che implementano SDS e consentono l'invio di certificati a Envoy. Includono lo SPIFFE Runtime Environment (SPIRE).

**⚠ Important**

App Mesh non archivia i certificati o le chiavi private utilizzati per l'autenticazione TLS reciproca. Invece, Envoy li archivia in memoria.

## Configura gli endpoint mesh

Configura l'autenticazione TLS reciproca per gli endpoint mesh, come nodi o gateway virtuali. Questi endpoint forniscono certificati e specificano autorità affidabili.

A tale scopo, è necessario fornire certificati X.509 sia per il client che per il server e definire in modo esplicito i certificati di autorità attendibili nel contesto di convalida sia della terminazione TLS che dell'origine TLS.

### Fiducia all'interno di una rete

I certificati lato server sono configurati nei listener Virtual Node (terminazione TLS) e i certificati lato client sono configurati nei backend del servizio Virtual Nodes (origine TLS). In alternativa a questa configurazione, è possibile definire una politica client predefinita per tutti i servizi di backend di un nodo virtuale e quindi, se necessario, sovrascrivere questa politica per backend specifici in base alle esigenze. I gateway virtuali possono essere configurati solo con una politica client predefinita che si applica a tutti i relativi backend.

È possibile configurare la fiducia tra diverse mesh abilitando l'autenticazione TLS reciproca per il traffico in entrata sui gateway virtuali per entrambe le mesh.

### Fiducia al di fuori di una rete

Specificate i certificati lato server nel listener Virtual Gateway per la terminazione TLS. Configura il servizio esterno che comunica con il tuo Virtual Gateway per presentare i certificati lato client. I certificati devono essere derivati da una delle stesse autorità di certificazione (CA) utilizzate dai certificati lato server sul listener Virtual Gateway per l'origine del TLS.

## Migra i servizi all'autenticazione TLS reciproca

Segui queste linee guida per mantenere la connettività durante la migrazione dei servizi esistenti all'interno di App Mesh all'autenticazione TLS reciproca.

## Servizi di migrazione che comunicano tramite testo non crittografato

1. Abilita `PERMISSIVE` la modalità per la configurazione TLS sull'endpoint del server. Questa modalità consente al traffico in testo semplice di connettersi all'endpoint.
2. Configura l'autenticazione TLS reciproca sul tuo server, specificando il certificato del server, la catena di fiducia e, facoltativamente, le SAN affidabili.
3. Conferma che la comunicazione avvenga tramite una connessione TLS.
4. Configura l'autenticazione TLS reciproca sui tuoi client, specificando il certificato del client, la catena di fiducia e, facoltativamente, le SAN affidabili.
5. Abilita `STRICT` la modalità per la configurazione TLS sul server.

## Servizi di migrazione che comunicano tramite TLS

1. Configura le impostazioni TLS reciproche sui tuoi client, specificando il certificato del client e, facoltativamente, le SAN affidabili. Il certificato client viene inviato al backend solo dopo che il server di backend lo richiede.
2. Configura le impostazioni TLS reciproche sul tuo server, specificando la catena di fiducia e, facoltativamente, le SAN affidabili. A tal fine, il server richiede un certificato client.

## Verifica dell'autenticazione TLS reciproca

Puoi fare riferimento alla documentazione sulla [crittografia Transport Layer Security: Verify](#) per vedere in che modo esattamente Envoy emette le statistiche relative a TLS. Per l'autenticazione TLS reciproca, è necessario controllare le seguenti statistiche:

- `ssl.handshake`
- `ssl.no_certificate`
- `ssl.fail_verify_no_cert`
- `ssl.fail_verify_san`

I due esempi di statistiche seguenti mostrano insieme che le connessioni TLS riuscite che terminano verso il nodo virtuale hanno tutte avuto origine da un client che ha fornito un certificato.

```
listener.0.0.0.0_15000.ssl.handshake: 3
```

```
listener.0.0.0.0_15000.ssl.no_certificate: 0
```

Il prossimo esempio di statistica mostra che le connessioni da un nodo client virtuale (o gateway) a un nodo virtuale di backend non sono riuscite. Il Subject Alternative Name (SAN) presentato nel certificato del server non corrisponde a nessuna delle SAN considerate affidabili dal client.

```
cluster.cds_egress_my-mesh_my-backend-node_http_9080.ssl.fail_verify_san: 5
```

## Procedure dettagliate per l'autenticazione TLS reciproca di App Mesh

- [Procedura dettagliata di autenticazione TLS reciproca](#): questa procedura dettagliata descrive come utilizzare l'App Mesh CLI per creare un'app a colori con autenticazione TLS reciproca.
- [Procedura dettagliata basata su Mutual TLS SDS di Amazon EKS: questa procedura dettagliata](#) mostra come utilizzare l'autenticazione reciproca basata su TLS SDS con Amazon EKS e SPIFFE Runtime Environment (SPIRE).
- [Procedura dettagliata basata su file TLS reciproco di Amazon EKS: questa procedura dettagliata](#) mostra come utilizzare l'autenticazione reciproca basata su file TLS con Amazon EKS e SPIFFE Runtime Environment (SPIRE).

## In che modo AWS App Mesh funziona con IAM

AWS Identity and Access Management (IAM) è un Servizio AWS che aiuta un amministratore a controllare in modo sicuro l'accesso a AWS risorse. IAM gli amministratori controllano chi può essere autenticato (effettuato l'accesso) e autorizzato (dispone delle autorizzazioni) a utilizzare le risorse App Mesh. IAM è un Servizio AWS che puoi utilizzare senza costi aggiuntivi.

### Argomenti

- [Destinatari](#)
- [Autenticazione con identità](#)
- [Gestione dell'accesso con policy](#)
- [In che modo AWS App Mesh funziona con IAM](#)
- [AWS App Mesh esempi di politiche basate sull'identità](#)
- [AWS politiche gestite per App Mesh](#)
- [Utilizzo di ruoli collegati ai servizi per App Mesh](#)

- [Autorizzazione proxy Envoy Proxy](#)
- [Risoluzione dei problemi AWS App Mesh identità e accesso](#)

## Destinatari

Come si usa AWS Identity and Access Management (IAM) differisce, a seconda del lavoro svolto in App Mesh.

Utente del servizio: se utilizzi il servizio App Mesh per svolgere il tuo lavoro, l'amministratore ti fornisce le credenziali e le autorizzazioni necessarie. Man mano che utilizzi più funzionalità di App Mesh per svolgere il tuo lavoro, potresti aver bisogno di autorizzazioni aggiuntive. La comprensione della gestione dell'accesso ti consente di richiedere le autorizzazioni corrette all'amministratore. Se non riesci ad accedere a una funzionalità in App Mesh, consulta [Risoluzione dei problemi AWS App Mesh identità e accesso](#).

Amministratore del servizio: se sei responsabile delle risorse App Mesh della tua azienda, probabilmente hai pieno accesso ad App Mesh. È tuo compito determinare a quali funzionalità e risorse App Mesh devono accedere gli utenti del servizio. È quindi necessario inviare richieste all'IAM amministratore per modificare le autorizzazioni degli utenti del servizio. Consulta le informazioni contenute in questa pagina per comprendere i concetti di base di IAM. Per ulteriori informazioni su come la tua azienda può utilizzare IAM App Mesh, consulta [In che modo AWS App Mesh funziona con IAM](#).

IAM amministratore: se sei un IAM amministratore, potresti voler saperne di più su come scrivere politiche per gestire l'accesso ad App Mesh. Per visualizzare esempi di policy basate sull'identità di App Mesh che puoi utilizzare in IAM, consulta [AWS App Mesh esempi di politiche basate sull'identità](#)

## Autenticazione con identità

L'autenticazione è il modo in cui si accede a AWS utilizzando le tue credenziali di identità. Devi essere autenticato (aver effettuato l'accesso a AWS) come Utente root dell'account AWS, come IAM utente o assumendo un IAM ruolo.

Puoi accedere a AWS come identità federata utilizzando le credenziali fornite tramite una fonte di identità. AWS IAM Identity Center Gli utenti (IAM Identity Center), l'autenticazione Single Sign-On della tua azienda e le tue credenziali Google o Facebook sono esempi di identità federate. Quando accedi come identità federata, l'amministratore aveva precedentemente configurato la federazione delle identità utilizzando i ruoli IAM. Quando accedi AWS utilizzando la federazione, assumi indirettamente un ruolo.

A seconda del tipo di utente che sei, puoi accedere a AWS Management Console o il AWS portale di accesso. Per ulteriori informazioni sull'accesso a AWS, vedi [Come accedere al tuo Account AWS](#) nella Accedi ad AWS Guida per l'utente.

Se accedi AWS programmaticamente, AWS fornisce un kit di sviluppo software (SDK) e un'interfaccia a riga di comando (CLI) per firmare crittograficamente le richieste utilizzando le credenziali dell'utente. Se non usi AWS strumenti, devi firmare tu stesso le richieste. Per ulteriori informazioni sull'utilizzo del metodo consigliato per firmare autonomamente le richieste, vedi [Firma AWS API richieste](#) nella Guida IAM per l'utente.

A prescindere dal metodo di autenticazione utilizzato, potrebbe essere necessario specificare ulteriori informazioni sulla sicurezza. Ad esempio, AWS consiglia di utilizzare l'autenticazione a più fattori (MFA) per aumentare la sicurezza del proprio account. Per ulteriori informazioni, consulta [Autenticazione a più fattori](#) nel AWS IAM Identity Center Guida per l'utente e [utilizzo dell'autenticazione a più fattori \(\) MFA in AWS](#) nella Guida per l'utente di IAM.

## Account AWS utente root

Quando crei un Account AWS, inizi con un'unica identità di accesso con accesso completo a tutti Servizi AWS e le risorse presenti nell'account. Questa identità è denominata Account AWS utente root ed è accessibile effettuando l'accesso con l'indirizzo e-mail e la password utilizzati per creare l'account. Si consiglia vivamente di non utilizzare l'utente root per le attività quotidiane. Conserva le credenziali dell'utente root e utilizzale per eseguire le operazioni che solo l'utente root può eseguire. Per l'elenco completo delle attività che richiedono l'accesso come utente root, consulta [Attività che richiedono le credenziali dell'utente root](#) nella Guida per l'IAMutente.

## IAM users and groups

Un [IAMutente](#) è un'identità all'interno di Account AWS che dispone di autorizzazioni specifiche per una singola persona o applicazione. Laddove possibile, consigliamo di fare affidamento su credenziali temporanee anziché creare IAM utenti con credenziali a lungo termine come password e chiavi di accesso. Tuttavia, se hai casi d'uso specifici che richiedono credenziali a lungo termine con IAM gli utenti, ti consigliamo di ruotare le chiavi di accesso. Per ulteriori informazioni, consulta [Ruotare regolarmente le chiavi di accesso per i casi d'uso che richiedono credenziali a lungo termine](#) nella Guida per l'utente. IAM

Un [IAMgruppo](#) è un'identità che specifica un insieme di utenti. IAM Non è possibile eseguire l'accesso come gruppo. È possibile utilizzare gruppi per specificare le autorizzazioni per più utenti alla volta. I gruppi semplificano la gestione delle autorizzazioni per set di utenti di grandi dimensioni. Ad esempio,

potresti avere un gruppo denominato IAMAdminse concedere a quel gruppo le autorizzazioni per IAM amministrare le risorse.

Gli utenti sono diversi dai ruoli. Un utente è associato in modo univoco a una persona o un'applicazione, mentre un ruolo è destinato a essere assunto da chiunque ne abbia bisogno. Gli utenti dispongono di credenziali a lungo termine permanenti, mentre i ruoli forniscono credenziali temporanee. Per ulteriori informazioni, consulta [Quando creare un IAM utente \(anziché un ruolo\)](#) nella Guida per l'IAMutente.

## IAMruoli

Un [IAMruolo](#) è un'identità all'interno del tuo Account AWS che dispone di autorizzazioni specifiche. È simile a un IAM utente, ma non è associato a una persona specifica. È possibile assumere temporaneamente un IAM ruolo nel AWS Management Console [cambiando ruolo](#). Puoi assumere un ruolo chiamando un AWS CLI oppure AWS APIoperazione o utilizzando un comando personalizzatoURL. Per ulteriori informazioni sui metodi di utilizzo dei ruoli, vedere [Utilizzo IAM dei ruoli](#) nella Guida per l'IAMutente.

IAMI ruoli con credenziali temporanee sono utili nelle seguenti situazioni:

- **Accesso utente federato:** per assegnare le autorizzazioni a una identità federata, è possibile creare un ruolo e definire le autorizzazioni per il ruolo. Quando un'identità federata viene autenticata, l'identità viene associata al ruolo e ottiene le autorizzazioni da esso definite. Per informazioni sui ruoli per la federazione, vedere [Creazione di un ruolo per un provider di identità di terze parti](#) nella Guida per l'IAMutente. Se utilizzi IAM Identity Center, configuri un set di autorizzazioni. Per controllare a cosa possono accedere le identità dopo l'autenticazione, IAM Identity Center correla il set di autorizzazioni a un ruolo in. IAM [Per informazioni sui set di autorizzazioni, consulta Set di autorizzazioni nel](#) AWS IAM Identity Center Guida per l'utente.
- **Autorizzazioni IAM utente temporanee:** un IAM utente o un ruolo può assumere un IAM ruolo per assumere temporaneamente autorizzazioni diverse per un'attività specifica.
- **Accesso su più account:** puoi utilizzare un IAM ruolo per consentire a qualcuno (un responsabile fidato) di un altro account di accedere alle risorse del tuo account. I ruoli sono lo strumento principale per concedere l'accesso multi-account. Tuttavia, con alcuni Servizi AWS, è possibile allegare una policy direttamente a una risorsa (anziché utilizzare un ruolo come proxy). Per conoscere la differenza tra i ruoli e le politiche basate sulle risorse per l'accesso tra più account, consulta l'accesso alle [risorse tra account IAM nella Guida per l'utente](#). IAM
- **Accesso a più servizi:** alcuni Servizi AWS usa le funzionalità in altri Servizi AWS. Ad esempio, quando effettui una chiamata in un servizio, è normale che quel servizio esegua applicazioni

in Amazon EC2 o archivi oggetti in Amazon S3. Un servizio può eseguire questa operazione utilizzando le autorizzazioni dell'entità chiamante, utilizzando un ruolo di servizio o utilizzando un ruolo collegato al servizio.

- **Sessioni di accesso diretto (FAS):** quando utilizzi un IAM utente o un ruolo per eseguire azioni in AWS, sei considerato un preside. Quando si utilizzano alcuni servizi, è possibile eseguire un'operazione che attiva un'altra operazione in un servizio diverso. FAS utilizza le autorizzazioni del principale che chiama un Servizio AWS, in combinazione con la richiesta Servizio AWS per effettuare richieste ai servizi a valle. FAS le richieste vengono effettuate solo quando un servizio riceve una richiesta che richiede interazioni con altri Servizi AWS o risorse da completare. In questo caso è necessario disporre delle autorizzazioni per eseguire entrambe le azioni. Per i dettagli FAS delle politiche relative alle richieste, consulta [Forward access sessions](#).
- **Ruolo di servizio:** un ruolo di servizio è un [IAM ruolo](#) che un servizio assume per eseguire azioni per conto dell'utente. Un IAM amministratore può creare, modificare ed eliminare un ruolo di servizio dall'interno IAM. Per ulteriori informazioni, vedere [Creazione di un ruolo per delegare le autorizzazioni a un Servizio AWS](#) nella Guida per l'utente di IAM.
- **Ruolo collegato al servizio:** un ruolo collegato al servizio è un tipo di ruolo di servizio collegato a un Servizio AWS. Il servizio può assumere il ruolo di eseguire un'azione per conto dell'utente. I ruoli collegati ai servizi vengono visualizzati nel Account AWS e sono di proprietà del servizio. Un IAM amministratore può visualizzare, ma non modificare le autorizzazioni per i ruoli collegati al servizio.
- **Applicazioni in esecuzione su Amazon EC2:** puoi utilizzare un IAM ruolo per gestire le credenziali temporanee per le applicazioni in esecuzione su un'EC2 istanza e in fase di creazione AWS CLI oppure AWS API richieste. Ciò è preferibile alla memorizzazione delle chiavi di accesso all'interno dell'EC2 istanza. Per assegnare un AWS assegnate un ruolo a un'EC2 istanza e renderlo disponibile a tutte le relative applicazioni, create un profilo di istanza collegato all'istanza. Un profilo di istanza contiene il ruolo e consente ai programmi in esecuzione sull'EC2 istanza di ottenere credenziali temporanee. Per ulteriori informazioni, consulta [Usare un IAM ruolo per concedere le autorizzazioni alle applicazioni in esecuzione su EC2 istanze Amazon nella Guida](#) per l'IAM utente.

Per sapere se utilizzare IAM ruoli o IAM utenti, consulta [Quando creare un IAM ruolo \(anziché un utente\)](#) nella Guida per l'IAM utente.

## Gestione dell'accesso con policy

Puoi controllare l'accesso in AWS creando politiche e allegandole a AWS identità o risorse.

Una politica è un oggetto in AWS che, se associato a un'identità o a una risorsa, ne definisce le

autorizzazioni. AWS valuta queste politiche quando un principale (utente, utente root o sessione di ruolo) effettua una richiesta. Le autorizzazioni nelle policy determinano l'approvazione o il rifiuto della richiesta. La maggior parte delle politiche viene archiviata in AWS come JSON documenti. Per ulteriori informazioni sulla struttura e il contenuto dei documenti relativi alle JSON politiche, vedere [Panoramica delle JSON politiche](#) nella Guida per l'IAMutente.

Gli amministratori possono utilizzare AWS JSONpolitiche per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire azioni su quali risorse e in quali condizioni.

Per impostazione predefinita, utenti e ruoli non dispongono di autorizzazioni. Per concedere agli utenti il permesso di eseguire azioni sulle risorse di cui hanno bisogno, un IAM amministratore può creare IAM politiche. L'amministratore può quindi aggiungere le IAM politiche ai ruoli e gli utenti possono assumerli.

IAMle politiche definiscono le autorizzazioni per un'azione indipendentemente dal metodo utilizzato per eseguire l'operazione. Ad esempio, supponiamo di disporre di una policy che consente l'operazione `iam:GetRole`. Un utente con tale criterio può ottenere informazioni sul ruolo da AWS Management Console, il AWS CLI, o AWS API.

## Policy basate su identità

I criteri basati sull'identità sono documenti relativi ai criteri di JSON autorizzazione che è possibile allegare a un'identità, ad esempio un IAM utente, un gruppo di utenti o un ruolo. Tali policy definiscono le azioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. [Per informazioni su come creare una politica basata sull'identità, consulta Creazione di politiche nella Guida per l'utente. IAM IAM](#)

Le policy basate su identità possono essere ulteriormente classificate come policy inline o policy gestite. Le policy inline sono integrate direttamente in un singolo utente, gruppo o ruolo. Le politiche gestite sono politiche autonome che puoi allegare a più utenti, gruppi e ruoli nel tuo Account AWS. Le politiche gestite includono AWS politiche gestite e politiche gestite dai clienti. Per informazioni su come scegliere tra una politica gestita o una politica in linea, consulta [Scelta tra politiche gestite e politiche in linea nella Guida](#) per l'IAMutente.

## Policy basate su risorse

Le politiche basate sulle risorse sono documenti di JSON policy allegati a una risorsa. Esempi di politiche basate sulle risorse sono le policy di trust dei IAM ruoli e le policy dei bucket di Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarli

per controllare l'accesso a una risorsa specifica. Quando è collegata a una risorsa, una policy definisce le azioni che un principale può eseguire su tale risorsa e a quali condizioni. È necessario [specificare un principale](#) in una policy basata sulle risorse. I principali possono includere account, utenti, ruoli, utenti federati o Servizi AWS.

Le policy basate sulle risorse sono policy inline che si trovano in tale servizio. Non puoi usare AWS politiche gestite da IAM una politica basata sulle risorse.

## Liste di controllo degli accessi ( ) ACLs

Le liste di controllo degli accessi (ACLs) controllano quali principali (membri dell'account, utenti o ruoli) dispongono delle autorizzazioni per accedere a una risorsa. ACLs sono simili alle politiche basate sulle risorse, sebbene non utilizzino il formato del documento di policy. JSON

Amazon S3, AWS WAF e Amazon VPC sono esempi di servizi che supportano ACLs. Per ulteriori informazioni ACLs, consulta la [panoramica di Access control list \(ACL\)](#) nella Amazon Simple Storage Service Developer Guide.

## Altri tipi di policy

AWS supporta tipi di policy aggiuntivi e meno comuni. Questi tipi di policy possono impostare il numero massimo di autorizzazioni concesse dai tipi di policy più comuni.

- **Limiti delle autorizzazioni:** un limite di autorizzazioni è una funzionalità avanzata in cui si impostano le autorizzazioni massime che una politica basata sull'identità può concedere a un'entità (utente o ruolo). IAM IAM È possibile impostare un limite delle autorizzazioni per un'entità. Le autorizzazioni risultanti sono l'intersezione delle policy basate su identità dell'entità e i relativi limiti delle autorizzazioni. Le policy basate su risorse che specificano l'utente o il ruolo nel campo `Principal` sono condizionate dal limite delle autorizzazioni. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. [Per ulteriori informazioni sui limiti delle autorizzazioni, consulta Limiti delle autorizzazioni per le entità nella Guida per l'utente. IAM IAM](#)
- **Politiche di controllo del servizio (SCPs):** SCPs sono JSON politiche che specificano le autorizzazioni massime per un'organizzazione o un'unità organizzativa (OU) in AWS Organizations. AWS Organizations è un servizio per il raggruppamento e la gestione centralizzata di più Account AWS di cui è proprietaria la tua azienda. Se abiliti tutte le funzionalità di un'organizzazione, puoi applicare le politiche di controllo del servizio (SCPs) a uno o tutti i tuoi account. I SCP limiti e le autorizzazioni per le entità presenti negli account dei membri, inclusi tutti Utente root dell'account AWS. Per ulteriori informazioni su Organizations and SCPs, vedere [Service control policies](#) nel AWS Organizations Guida per l'utente.

- **Policy di sessione:** le policy di sessione sono policy avanzate che vengono trasmesse come parametro quando si crea in modo programmatico una sessione temporanea per un ruolo o un utente federato. Le autorizzazioni della sessione risultante sono l'intersezione delle policy basate su identità del ruolo o dell'utente e le policy di sessione. Le autorizzazioni possono anche provenire da una policy basata su risorse. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni, consulta [le politiche di sessione](#) nella Guida IAM per l'utente.

## Più tipi di policy

Quando più tipi di policy si applicano a una richiesta, le autorizzazioni risultanti sono più complicate da comprendere. Per scoprire come AWS determina se consentire una richiesta quando sono coinvolti più tipi di policy, consulta [Logica di valutazione delle policy](#) nella Guida per l'IAMutente.

## In che modo AWS App Mesh funziona con IAM

Prima di utilizzare IAM per gestire l'accesso ad App Mesh, è necessario comprendere quali IAM funzionalità sono disponibili per l'uso con App Mesh. Per avere una visione di alto livello di come App Mesh e altro AWS i servizi funzionano conIAM, vedi [AWS Servizi compatibili con IAM](#) la Guida per l'IAMutente.

### Argomenti

- [Criteri basati sull'identità App Mesh](#)
- [Criteri basati sulle risorse App Mesh](#)
- [Autorizzazione basata sui tag App Mesh](#)
- [IAMRuoli App Mesh](#)

## Criteri basati sull'identità App Mesh

Con le politiche IAM basate sull'identità, puoi specificare azioni e risorse consentite o negate, nonché le condizioni in base alle quali le azioni sono consentite o negate. App Mesh supporta azioni, risorse e chiavi di condizione specifiche. Per maggiori informazioni su tutti gli elementi utilizzati in una JSON policy, consulta [IAMJSONPolicy Elements Reference](#) nella Guida per l'IAMutente.

## Azioni

Gli amministratori possono utilizzare AWS JSONpolitiche per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'Actionelemento di una JSON policy descrive le azioni che è possibile utilizzare per consentire o negare l'accesso a una policy. Le azioni politiche in genere hanno lo stesso nome di quelle associate AWS APIoperazione. Esistono alcune eccezioni, come le azioni di sola autorizzazione che non hanno un'operazione corrispondente. API Esistono anche alcune operazioni che richiedono più operazioni in una policy. Queste operazioni aggiuntive sono denominate operazioni dipendenti.

Includi le operazioni in una policy per concedere le autorizzazioni a eseguire l'operazione associata.

Le azioni politiche in App Mesh utilizzano il seguente prefisso prima dell'azione: `appmesh:`.

Ad esempio, per concedere a qualcuno il permesso di elencare le mesh in un account con l'`appmesh:ListMeshes`APIoperazione, includi l'`appmesh:ListMeshes`azione nella sua politica. Le istruzioni della policy devono includere un elemento `Action` o `NotAction`.

Per specificare più operazioni in una singola istruzione, separarle con una virgola come mostrato di seguito.

```
"Action": [
 "appmesh:ListMeshes",
 "appmesh:ListVirtualNodes"
]
```

Puoi specificare più operazioni tramite caratteri jolly (\*). Ad esempio, per specificare tutte le operazioni che iniziano con la parola `Describe`, includi la seguente operazione.

```
"Action": "appmesh:Describe*"
```

Per visualizzare un elenco di azioni App Mesh, consulta [Azioni definite da AWS App Mesh](#) nella Guida per l'utente di IAM.

## Risorse

Gli amministratori possono utilizzare AWS JSONpolitiche per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'elemento `Resource` JSON policy specifica l'oggetto o gli oggetti a cui si applica l'azione. Le istruzioni devono includere un elemento `Resource` o un elemento `NotResource`. Come best

practice, specifica una risorsa utilizzando il relativo [Amazon Resource Name \(ARN\)](#). Puoi eseguire questa operazione per azioni che supportano un tipo di risorsa specifico, note come autorizzazioni a livello di risorsa.

Per le azioni che non supportano le autorizzazioni a livello di risorsa, ad esempio le operazioni di elenco, utilizza un carattere jolly (\*) per indicare che l'istruzione si applica a tutte le risorse.

```
"Resource": "*"

```

La mesh risorsa App Mesh ha quanto segueARN.

```
arn:${Partition}:appmesh:${Region}:${Account}:mesh/${MeshName}

```

Per ulteriori informazioni sul formato diARNs, consulta [Amazon Resource Names \(ARNs\) e AWS Namespace dei](#) servizi.

Ad esempio, per specificare la mesh denominata *apps* nel *Region-code* Regione nella tua dichiarazione, usa quanto segueARN.

```
arn:aws:appmesh:Region-code:111122223333:mesh/apps

```

Per specificare tutte le istanze database che appartengono a un account specifico, utilizza il carattere jolly (\*).

```
"Resource": "arn:aws:appmesh:Region-code:111122223333:mesh/*"

```

Alcune azioni App Mesh, come quelle per la creazione di risorse, non possono essere eseguite su una risorsa specifica. In questi casi, è necessario utilizzare il carattere jolly (\*).

```
"Resource": "*"

```

Molte API azioni App Mesh coinvolgono più risorse. Ad esempio, CreateRoute crea una route con una destinazione di nodo virtuale, quindi un IAM utente deve disporre delle autorizzazioni per utilizzare la route e il nodo virtuale. Per specificare più risorse in una singola istruzione, separale ARNs con virgole.

```
"Resource": [
 "arn:aws:appmesh:Region-code:111122223333:mesh/apps/virtualRouter/serviceB/route/
 *",

```

```
] "arn:aws:appmesh:Region-code:111122223333:mesh/apps/virtualNode/serviceB"
```

Per visualizzare un elenco dei tipi di risorse App Mesh e relativi ARNs, vedi [Risorse definite da AWS App Mesh](#) nella Guida per l'utente di IAM. Per sapere con quali azioni è possibile specificare le ARN risorse, consulta [Azioni definite da AWS App Mesh](#).

## Chiavi di condizione

App Mesh supporta l'utilizzo di alcune chiavi di condizione globali. Per vedere tutto AWS chiavi di condizione globali, vedi [AWS Tasti contestuali della condizione globale](#) nella Guida IAM per l'utente. Per visualizzare un elenco delle chiavi di condizione globali supportate da App Mesh, vedi [Condition Keys per AWS App Mesh](#) nella Guida per l'utente di IAM. Per sapere con quali azioni e risorse puoi utilizzare con una chiave di condizione, vedi [Azioni definite da AWS App Mesh](#).

## Esempi

Per visualizzare esempi di policy basate sull'identità di App Mesh, vedere [AWS App Mesh esempi di politiche basate sull'identità](#)

## Criteri basati sulle risorse App Mesh

App Mesh non supporta le politiche basate sulle risorse. Tuttavia, se si utilizza il AWS Resource Access Manager (AWS RAM) servizio su cui condividere una mesh AWS servizi, una politica basata sulle risorse viene applicata alla rete mesh dal AWS RAM servizio. Per ulteriori informazioni, consulta [Concessione delle autorizzazioni per una mesh](#).

## Autorizzazione basata sui tag App Mesh

Puoi allegare tag alle risorse App Mesh o passare i tag in una richiesta ad App Mesh. Per controllare l'accesso basato su tag, fornisci informazioni sui tag nell'[elemento condizione](#) di una policy utilizzando le chiavi di condizione `appmesh:ResourceTag/key-name`, `aws:RequestTag/key-name` o `aws:TagKeys`. Per ulteriori informazioni sull'etichettatura delle risorse App Mesh, consulta [Tagging AWS Risorse](#).

Per visualizzare una policy basata sulle identità di esempio per limitare l'accesso a una risorsa basata su tag su tale risorsa, consulta [Creazione di mesh App Mesh con tag limitati](#).

## IAM Ruoli App Mesh

Un [IAM ruolo](#) è un'entità all'interno del tuo AWS account con autorizzazioni specifiche.

## Utilizzo di credenziali temporanee con App Mesh

Puoi utilizzare credenziali temporanee per accedere con la federazione, assumere un IAM ruolo o assumere un ruolo tra account. È possibile ottenere credenziali di sicurezza temporanee chiamando AWS STS API operazioni come [AssumeRole](#) o [GetFederationToken](#).

App Mesh supporta l'utilizzo di credenziali temporanee.

### Ruoli collegati ai servizi

[I ruoli collegati ai servizi lo consentono](#) AWS servizi per accedere alle risorse di altri servizi per completare un'azione per conto dell'utente. I ruoli collegati ai servizi vengono visualizzati nell'IAM account e sono di proprietà del servizio. Un IAM amministratore può visualizzare ma non modificare le autorizzazioni per i ruoli collegati al servizio.

App Mesh supporta i ruoli collegati ai servizi. Per informazioni dettagliate sulla creazione o la gestione di ruoli collegati al servizio App Mesh, vedere. [Utilizzo di ruoli collegati ai servizi per App Mesh](#)

### Ruoli dei servizi

Questa caratteristica consente a un servizio di assumere un [ruolo di servizio](#) per conto dell'utente. Questo ruolo consente al servizio di accedere alle risorse in altri servizi per completare un'azione per conto dell'utente. I ruoli di servizio vengono visualizzati nell'IAM account e sono di proprietà dell'account. Ciò significa che un IAM amministratore può modificare le autorizzazioni per questo ruolo. Tuttavia, il farlo potrebbe pregiudicare la funzionalità del servizio.

App Mesh non supporta i ruoli di servizio.

## AWS App Mesh esempi di politiche basate sull'identità

Per impostazione predefinita, IAM gli utenti e i ruoli non dispongono dell'autorizzazione per creare o modificare risorse App Mesh. Inoltre, non possono eseguire attività utilizzando il AWS Management Console, AWS CLI, oppure AWS API. Un IAM amministratore deve creare IAM politiche che concedano a utenti e ruoli l'autorizzazione a eseguire API operazioni specifiche sulle risorse specificate di cui hanno bisogno. L'amministratore deve quindi allegare tali politiche agli IAM utenti o ai gruppi che richiedono tali autorizzazioni.

Per informazioni su come creare una politica IAM basata sull'identità utilizzando questi documenti di esempio JSON, consulta [Creazione di politiche nella JSON scheda nella Guida per l'utente](#). IAM

## Argomenti

- [Best practice per le policy](#)
- [Utilizzo della console App Mesh](#)
- [Consentire agli utenti di visualizzare le loro autorizzazioni](#)
- [Crea una mesh](#)
- [Elenca e descrivi tutte le mesh](#)
- [Creazione di mesh App Mesh con tag limitati](#)

## Best practice per le policy

Le politiche basate sull'identità determinano se qualcuno può creare, accedere o eliminare le risorse App Mesh nel tuo account. Queste azioni possono comportare costi per Account AWS. Quando crei o modifichi politiche basate sull'identità, segui queste linee guida e consigli:

- Inizia con AWS politiche gestite e passaggio alle autorizzazioni con privilegi minimi: per iniziare a concedere autorizzazioni a utenti e carichi di lavoro, utilizza il AWS politiche gestite che concedono autorizzazioni per molti casi d'uso comuni. Sono disponibili nel tuo Account AWS. Si consiglia di ridurre ulteriormente le autorizzazioni definendo AWS politiche gestite dai clienti specifiche per i tuoi casi d'uso. Per ulteriori informazioni, consulta [AWS politiche gestite](#) o [AWS politiche gestite per le funzioni lavorative](#) nella Guida per IAM l'utente.
- Applica le autorizzazioni con privilegi minimi: quando imposti le autorizzazioni con le IAM politiche, concedi solo le autorizzazioni necessarie per eseguire un'attività. Puoi farlo definendo le azioni che possono essere intraprese su risorse specifiche in condizioni specifiche, note anche come autorizzazioni con privilegi minimi. Per ulteriori informazioni sull'utilizzo per applicare le autorizzazioni, consulta [Politiche](#) e autorizzazioni nella Guida IAM per l'utente. IAM IAM
- Utilizza le condizioni nelle IAM politiche per limitare ulteriormente l'accesso: puoi aggiungere una condizione alle tue politiche per limitare l'accesso ad azioni e risorse. Ad esempio, puoi scrivere una condizione di policy per specificare che tutte le richieste devono essere inviate utilizzando SSL. È inoltre possibile utilizzare le condizioni per concedere l'accesso alle azioni di servizio se vengono utilizzate tramite uno specifico Servizio AWS, ad esempio AWS CloudFormation. Per ulteriori informazioni, consulta [Elementi IAM JSON della politica: Condizione](#) nella Guida IAM per l'utente.
- Usa IAM Access Analyzer per convalidare IAM le tue policy e garantire autorizzazioni sicure e funzionali: IAM Access Analyzer convalida le policy nuove ed esistenti in modo che aderiscano al linguaggio delle IAM policy () e alle best practice. JSON IAM IAMAccess Analyzer fornisce

più di 100 controlli delle politiche e consigli pratici per aiutarti a creare policy sicure e funzionali. Per ulteriori informazioni, vedere [Convalida delle policy di IAM Access Analyzer nella Guida per l'utente](#). IAM

- Richiedi l'autenticazione a più fattori (MFA): se si dispone di uno scenario che richiede IAM utenti o un utente root nel Account AWS, attivala MFA per una maggiore sicurezza. Per richiedere MFA quando vengono richiamate API le operazioni, aggiungi MFA delle condizioni alle tue politiche. Per ulteriori informazioni, vedere [Configurazione dell'API accesso MFA protetto nella Guida per l'IAM utente](#).

Per ulteriori informazioni sulle procedure consigliate in IAM, consulta la sezione [Procedure consigliate in materia di sicurezza IAM nella Guida per l'IAM utente](#).

## Utilizzo della console App Mesh

Per accedere a AWS App Mesh console, è necessario disporre di un set minimo di autorizzazioni. Queste autorizzazioni devono consentirti di elencare e visualizzare i dettagli sulle risorse App Mesh nel tuo AWS account. Se crei una politica basata sull'identità più restrittiva delle autorizzazioni minime richieste, la console non funzionerà come previsto per le entità (IAM utenti o ruoli) che applicano tale politica. Puoi allegare la policy gestita agli utenti. [AWSAppMeshReadOnl](#)y Per ulteriori informazioni, consulta [Aggiungere autorizzazioni a un utente](#) nella Guida per l'IAM utente.

Non è necessario consentire autorizzazioni minime per la console agli utenti che effettuano chiamate solo verso AWS CLI o AWS API. Consenti invece l'accesso solo alle azioni che corrispondono all'API operazione che stai cercando di eseguire.

## Consentire agli utenti di visualizzare le loro autorizzazioni

Questo esempio mostra come è possibile creare una politica che consenta IAM agli utenti di visualizzare le politiche in linea e gestite allegate alla loro identità utente. Questa politica include le autorizzazioni per completare questa azione sulla console o utilizzando programmaticamente il AWS CLI oppure AWS API.

```
{
 "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": "*"
 }
]
}

```

## Crea una mesh

Questo esempio mostra come creare una politica che consenta a un utente di creare una mesh per un account, in qualsiasi regione.

```

{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": "appmesh:CreateMesh",
 "Resource": "arn:aws:appmesh:*:123456789012:CreateMesh"
 }
]
}

```

## Elenca e descrivi tutte le mesh

Questo esempio mostra come è possibile creare una politica che consenta a un utente di accedere in sola lettura all'elenco e alla descrizione di tutte le mesh.

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": [
 "appmesh:DescribeMesh",
 "appmesh:ListMeshes"
],
 "Resource": "*"
 }
]
}
```

## Creazione di mesh App Mesh con tag limitati

Puoi utilizzare i tag nelle tue IAM politiche per controllare quali tag possono essere passati nella IAM richiesta. Puoi specificare quali coppie chiave-valore di tag possono essere aggiunte, modificate o rimosse da un IAM utente o da un ruolo. Questo esempio mostra come è possibile creare una politica che consenta la creazione di una mesh, ma solo se la mesh viene creata con un tag denominato *teamName* e un valore di *booksTeam*.

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": "appmesh:CreateMesh",
 "Resource": "*",
 "Condition": {
 "StringEquals": {
 "aws:RequestTag/teamName": "booksTeam"
 }
 }
 }
]
}
```

```
}
```

Puoi allegare questa politica agli IAM utenti del tuo account. Se un utente tenta di creare una mesh, la mesh deve includere un tag denominato `teamName` e un valore `diBooksTeam`. Se la mesh non include questo tag e questo valore, il tentativo di creare la mesh fallisce. Per ulteriori informazioni, vedere [IAMJSONPolicy Elements: Condition](#) nella Guida IAM per l'utente.

## AWS politiche gestite per App Mesh

Una politica AWS gestita è una politica autonoma creata e amministrata da AWS. AWS le politiche gestite sono progettate per fornire autorizzazioni per molti casi d'uso comuni, in modo da poter iniziare ad assegnare autorizzazioni a utenti, gruppi e ruoli.

Tieni presente che le policy AWS gestite potrebbero non concedere le autorizzazioni con il privilegio minimo per i tuoi casi d'uso specifici, poiché sono disponibili per tutti i clienti. AWS Consigliamo pertanto di ridurre ulteriormente le autorizzazioni definendo [policy gestite dal cliente](#) specifiche per i tuoi casi d'uso.

Non è possibile modificare le autorizzazioni definite nelle politiche gestite. AWS Se AWS aggiorna le autorizzazioni definite in una politica AWS gestita, l'aggiornamento ha effetto su tutte le identità principali (utenti, gruppi e ruoli) a cui è associata la politica. AWS è più probabile che aggiorni una policy AWS gestita quando ne Servizio AWS viene lanciata una nuova o quando diventano disponibili nuove operazioni API per i servizi esistenti.

Per ulteriori informazioni, consultare [Policy gestite da AWS](#) nella Guida per l'utente di IAM.

### AWS politica gestita: `AWSAppMeshServiceRolePolicy`

È possibile allegare `AWSAppMeshServiceRolePolicy` alle entità IAM. Consente l'accesso ai AWS servizi e alle risorse utilizzati o gestiti da AWS App Mesh.

Per visualizzare le autorizzazioni per questa politica, consulta [AWSAppMeshServiceRolePolicy](#) il AWS Managed Policy Reference.

Per informazioni sui dettagli delle autorizzazioni per `AWSAppMeshServiceRolePolicy`, vedi [Autorizzazioni ai ruoli collegati ai servizi per App Mesh](#).

### AWS politica gestita: `AWSAppMeshEnvoyAccess`

È possibile allegare `AWSAppMeshEnvoyAccess` alle entità IAM. Policy di App Mesh Envoy per l'accesso alla configurazione dei nodi virtuali.

Per visualizzare le autorizzazioni per questa politica, consulta il AWS Managed Policy [AWSAppMeshEnvoyAccess](#)Reference.

### AWS politica gestita: AWSAppMeshFullAccess

È possibile allegare `AWSAppMeshFullAccess` alle entità IAM. Fornisce accesso completo alle AWS App Mesh API e AWS Management Console.

Per visualizzare le autorizzazioni per questa policy, consulta [AWSAppMeshFullAccess](#)il AWS Managed Policy Reference.

### AWS politica gestita: AWSAppMeshPreviewEnvoyAccess

È possibile allegare `AWSAppMeshPreviewEnvoyAccess` alle entità IAM. Politica di App Mesh Preview Envoy per l'accesso alla configurazione dei nodi virtuali.

Per visualizzare le autorizzazioni per questa politica, consulta il AWS Managed Policy [AWSAppMeshPreviewEnvoyAccess](#)Reference.

### AWS politica gestita: AWSAppMeshPreviewServiceRolePolicy

È possibile allegare `AWSAppMeshPreviewServiceRolePolicy` alle entità IAM. Consente l'accesso ai AWS servizi e alle risorse utilizzati o gestiti da AWS App Mesh.

Per visualizzare le autorizzazioni per questa politica, consulta [AWSAppMeshPreviewServiceRolePolicy](#)il AWS Managed Policy Reference.

### AWS politica gestita: AWSAppMeshReadOnly

È possibile allegare `AWSAppMeshReadOnly` alle entità IAM. Fornisce accesso in sola lettura alle AWS App Mesh API e. AWS Management Console

Per visualizzare le autorizzazioni per questa policy, consulta il Managed Policy [AWSAppMeshReadOnly](#)Reference.AWS

## AWS App Mesh aggiornamenti alle politiche AWS gestite

Visualizza i dettagli sugli aggiornamenti delle politiche AWS gestite AWS App Mesh da quando questo servizio ha iniziato a tenere traccia di queste modifiche. Per gli avvisi automatici sulle

modifiche apportate a questa pagina, sottoscrivi il feed RSS nella pagina della cronologia dei documenti di AWS App Mesh .

| Modifica                                                                                                                                                          | Descrizione                                                                                                                                                              | Data            |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| <a href="#">AWSAppMeshFullAccess</a> —<br>Politica aggiornata.                                                                                                    | Aggiornato <code>AWSAppMeshFullAccess</code> per consentire l'accesso alle <code>UntagResource</code> API <code>TagResource</code> e.                                    | 24 aprile 2024  |
| <a href="#">AWSAppMeshServiceRolePolicy</a> , <a href="#">AWSServiceRoleForAppMesh</a> —<br><a href="#">AWSAppMeshServiceRolePolicy</a> —<br>Politica aggiornata. | Aggiornato <code>AWSAppMeshServiceRolePolicy</code> e <code>AWSServiceRoleForAppMesh</code> per consentire l'accesso all' <code>MapDiscoverInstancesRevision</code> API. | 12 ottobre 2023 |

Per fornire l'accesso, aggiungi autorizzazioni ai tuoi utenti, gruppi o ruoli:

- Utenti e gruppi in AWS IAM Identity Center:

Crea un set di autorizzazioni. Segui le istruzioni riportate nella pagina [Create a permission set](#) (Creazione di un set di autorizzazioni) nella Guida per l'utente di AWS IAM Identity Center .

- Utenti gestiti in IAM tramite un provider di identità:

Crea un ruolo per la federazione delle identità. Segui le istruzioni riportate nella pagina [Creating a role for a third-party identity provider \(federation\)](#) (Creazione di un ruolo per un provider di identità di terze parti [federazione]) nella Guida per l'utente di IAM.

- Utenti IAM:

- Crea un ruolo che l'utente possa assumere. Per istruzioni, consulta la pagina [Creating a role for an IAM user](#) (Creazione di un ruolo per un utente IAM) nella Guida per l'utente di IAM.

- (Non consigliato) Collega una policy direttamente a un utente o aggiungi un utente a un gruppo di utenti. Segui le istruzioni riportate nella pagina [Aggiunta di autorizzazioni a un utente \(console\)](#) nella Guida per l'utente di IAM.

## Utilizzo di ruoli collegati ai servizi per App Mesh

AWS App Mesh utilizza [ruoli collegati al servizio AWS Identity and Access Management \(IAM\)](#).

Un ruolo collegato al servizio è un tipo unico di ruolo IAM collegato direttamente ad App Mesh. I ruoli collegati ai servizi sono predefiniti da App Mesh e includono tutte le autorizzazioni richieste dal servizio per chiamare altri AWS servizi per tuo conto.

Un ruolo collegato al servizio semplifica la configurazione di App Mesh perché non è necessario aggiungere manualmente le autorizzazioni necessarie. App Mesh definisce le autorizzazioni dei suoi ruoli collegati al servizio e, se non diversamente definito, solo App Mesh può assumerne i ruoli. Le autorizzazioni definite includono la policy di attendibilità e la policy delle autorizzazioni che non può essere collegata a nessun'altra entità IAM.

È possibile eliminare un ruolo collegato ai servizi solo dopo aver eliminato le risorse correlate. In questo modo proteggi le tue risorse App Mesh perché non puoi rimuovere inavvertitamente l'autorizzazione ad accedere alle risorse.

Per informazioni sugli altri servizi che supportano i ruoli collegati ai servizi, consulta [Servizi AWS che funzionano con IAM](#) e cerca i servizi che riportano Sì nella colonna Ruolo associato ai servizi. Scegli Yes (Sì) in corrispondenza di un link per visualizzare la documentazione relativa al ruolo collegato ai servizi per tale servizio.

### Autorizzazioni di ruolo collegate al servizio per App Mesh

App Mesh utilizza il ruolo collegato al servizio denominato `AWSServiceRoleForAppMesh`: il ruolo consente a App Mesh di chiamare AWS i servizi per tuo conto.

Il ruolo `AWSServiceRoleForAppMesh` collegato al servizio si fida che il `appmesh.amazonaws.com` servizio assuma il ruolo.

#### Dettagli delle autorizzazioni

- `servicediscovery:DiscoverInstances`- Consente ad App Mesh di completare azioni su tutte le AWS risorse.
- `servicediscovery:DiscoverInstancesRevision`- Consente ad App Mesh di completare azioni su tutte le AWS risorse.

#### `AWSServiceRoleForAppMesh`

Questa policy include le seguenti autorizzazioni:

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "CloudMapServiceDiscovery",
 "Effect": "Allow",
 "Action": [
 "servicediscovery:DiscoverInstances",
 "servicediscovery:DiscoverInstancesRevision"
],
 "Resource": "*"
 },
 {
 "Sid": "ACMCertificateVerification",
 "Effect": "Allow",
 "Action": [
 "acm:DescribeCertificate"
],
 "Resource": "*"
 }
]
}
```

Per consentire a un'entità IAM (come un utente, un gruppo o un ruolo) di creare, modificare o eliminare un ruolo collegato ai servizi devi configurare le relative autorizzazioni. Per ulteriori informazioni, consulta [Autorizzazioni del ruolo collegato ai servizi](#) nella Guida per l'utente di IAM.

## Creazione di un ruolo collegato al servizio per App Mesh

Se hai creato una mesh dopo il 5 giugno 2019 nell'AWSAPIAWS Management Console, App Mesh ha creato il ruolo collegato al servizio per te. AWS CLI Affinché il ruolo collegato al servizio sia stato creato automaticamente, all'account IAM che hai utilizzato per creare la mesh deve essere associata la policy [AWSAppMeshFullAccess](#) IAM o una policy associata che contenesse l'autorizzazione. `iam:CreateServiceLinkedRole` Se elimini questo ruolo collegato ai servizi, puoi ricrearlo seguendo lo stesso processo utilizzato per ricreare il ruolo nell'account. Quando crei una mesh, App Mesh crea nuovamente il ruolo collegato al servizio per te. Se il tuo account contiene solo mesh create prima del 5 giugno 2019 e desideri utilizzare il ruolo collegato al servizio con tali mesh, puoi creare il ruolo utilizzando la console IAM.

Puoi utilizzare la console IAM per creare un ruolo collegato al servizio con lo use case App Mesh. In AWS CLI o in AWS API, crea un ruolo collegato ai servizi con il nome di servizio

appmesh.amazonaws.com. Per ulteriori informazioni, consulta [Creazione di un ruolo collegato ai servizi](#) nella Guida per l'utente IAM. Se elimini il ruolo collegato ai servizi, puoi utilizzare lo stesso processo per crearlo nuovamente.

## Modifica di un ruolo collegato al servizio per App Mesh

App Mesh non consente di modificare il ruolo AWSServiceRoleForAppMesh collegato al servizio. Dopo aver creato un ruolo collegato ai servizi, non potrai modificarne il nome perché varie entità potrebbero farvi riferimento. È possibile tuttavia modificarne la descrizione utilizzando IAM. Per ulteriori informazioni, consulta la sezione [Modifica di un ruolo collegato ai servizi](#) nella Guida per l'utente di IAM.

## Eliminazione di un ruolo collegato al servizio per App Mesh

Se non è più necessario utilizzare una funzionalità o un servizio che richiede un ruolo collegato ai servizi, ti consigliamo di eliminare il ruolo. In questo modo non sarà più presente un'entità non utilizzata che non viene monitorata e gestita attivamente. Tuttavia, è necessario effettuare la pulizia delle risorse associate al ruolo collegato ai servizi prima di poterlo eliminare manualmente.

### Note

Se il servizio App Mesh utilizza il ruolo quando si tenta di eliminare le risorse, l'eliminazione potrebbe non riuscire. In questo caso, attendi alcuni minuti e quindi ripeti l'operazione.

Per eliminare le risorse App Mesh utilizzate da AWSServiceRoleForAppMesh

1. Elimina tutti i [percorsi](#) definiti per tutti i router nella mesh.
2. Elimina tutti i [router virtuali](#) nella mesh.
3. Eliminare tutti i [servizi virtuali](#) nella mesh.
4. Eliminare tutti i [nodi virtuali](#) nella mesh.
5. Eliminare la [mesh](#).

Completa i passaggi precedenti per tutte le mesh del tuo account.

Per eliminare manualmente il ruolo collegato ai servizi utilizzando IAM

Utilizza la console IAMAWS CLI, o l'AWSAPI per eliminare il ruolo collegato al AWSServiceRoleForAppMesh servizio. Per ulteriori informazioni, consulta [Eliminazione del ruolo collegato ai servizi](#) nella Guida per l'utente di IAM.

## Regioni supportate per i ruoli collegati al servizio App Mesh

App Mesh supporta l'utilizzo di ruoli collegati al servizio in tutte le regioni in cui il servizio è disponibile. Per ulteriori informazioni, consulta [App Mesh Endpoints and Quotas](#).

## Autorizzazione proxy Envoy Proxy

L'autorizzazione proxy autorizza il proxy [Envoy](#) in esecuzione all'interno di un'attività Amazon ECS, in un pod Kubernetes in esecuzione su Amazon EKS o in esecuzione su un'istanza Amazon EC2 a leggere la configurazione di uno o più endpoint mesh dall'App Mesh Envoy Management Service. Per gli account dei clienti che hanno già Envoy connesso all'endpoint App Mesh prima del 26/04/2021, l'autorizzazione proxy è richiesta per i nodi virtuali che utilizzano [Transport Layer Security \(TLS\)](#) e per i gateway virtuali (con o senza TLS). Per gli account dei clienti che desiderano connettere Envoy al proprio endpoint App Mesh dopo il 26/04/2021, è necessaria l'autorizzazione proxy per tutte le funzionalità di App Mesh. Si consiglia a tutti gli account dei clienti di abilitare l'autorizzazione proxy per tutti i nodi virtuali, anche se non utilizzano TLS, per avere un'esperienza sicura e coerente nell'uso di IAM per l'autorizzazione a risorse specifiche. L'autorizzazione proxy richiede che l'appmesh:StreamAggregatedResourcesautorizzazione sia specificata in una policy IAM. La policy deve essere associata a un ruolo IAM e quel ruolo IAM deve essere associato alla risorsa di elaborazione su cui si ospita il proxy.

## Creare una policy IAM

Se desideri che tutti gli endpoint mesh in un service mesh siano in grado di leggere la configurazione per tutti gli endpoint mesh, passa a [Creazione di un ruolo IAM](#). Se si desidera limitare gli endpoint mesh da cui è possibile leggere la configurazione ai singoli endpoint mesh, è necessario creare una o più policy IAM. Si consiglia di limitare gli endpoint mesh da cui è possibile leggere la configurazione al solo proxy Envoy in esecuzione su risorse di elaborazione specifiche. Crea una policy IAM e aggiungere l'appmesh:StreamAggregatedResourcesautorizzazione alla policy. La seguente politica di esempio consente la configurazione dei nodi virtuali denominatiserviceBv1 eserviceBv2 la lettura in una mesh di servizio. La configurazione non può essere letta per nessun altro nodo virtuale definito nella service mesh. Per ulteriori informazioni sulla creazione o la modifica di una policy IAM, consulta [Creazione di policy IAM](#) e [Modifica di policy IAM](#).

```
{
```

```
"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"
]
 }
]
```

È possibile creare più policy, ciascuna delle quali limita l'accesso a diversi endpoint mesh.

## Creazione di un ruolo IAM

Se desideri che tutti gli endpoint mesh in una service mesh siano in grado di leggere la configurazione per tutti gli endpoint mesh, devi creare un solo ruolo IAM. Se si desidera limitare gli endpoint mesh dai quali è possibile leggere la configurazione ai singoli endpoint mesh, è necessario creare un ruolo per ogni policy creata nel passaggio precedente. Completa le istruzioni per la risorsa di elaborazione su cui viene eseguito il proxy.

- Amazon EKS: se desideri utilizzare un singolo ruolo, puoi utilizzare il ruolo esistente che è stato creato e assegnato ai nodi di lavoro al momento della creazione del cluster. Per utilizzare più ruoli, il cluster deve soddisfare i requisiti definiti in [Abilitazione dei ruoli IAM per gli account di servizio nel cluster](#). Crea i ruoli IAM e associa i ruoli agli account del servizio Kubernetes. Per ulteriori informazioni, vedi [Creazione di un ruolo e una politica IAM per il tuo account di servizio](#) e [Specificazione di un ruolo IAM per il tuo account di servizio](#).
- Amazon ECS: seleziona il AWSservizio, seleziona Elastic Container Service, quindi seleziona lo use case di Elastic Container Service Task durante la creazione del tuo ruolo IAM.
- Amazon EC2: seleziona il AWSservizio, seleziona EC2 e quindi seleziona lo use case EC2 durante la creazione del tuo ruolo IAM. Questo vale sia che il proxy sia ospitato direttamente su un'istanza di Amazon EC2 o Kubernetes in esecuzione su un'istanza.

Per ulteriori informazioni su come creare un ruolo IAM, consulta [Creazione di un ruolo per unAWS servizio](#).

## Collegamento di policy IAM

Se desideri che tutti gli endpoint mesh di una service mesh siano in grado di leggere la configurazione di tutti gli endpoint mesh, collega la policy IAM [AWSAppMeshEnvoyAccess](#) gestita al ruolo IAM creato in un passaggio precedente. Se desideri limitare gli endpoint mesh da cui è possibile leggere la configurazione ai singoli endpoint mesh, collega ogni policy che hai creato a ogni ruolo che hai creato. Per ulteriori informazioni sull'associazione di una policy IAM personalizzata o gestita a un ruolo IAM, vedi [Aggiungere le autorizzazioni di identità IAM](#).

## Collegamento di un ruolo IAM

Collega ogni ruolo IAM alla risorsa di elaborazione appropriata:

- Amazon EKS: se hai collegato la policy al ruolo associato ai tuoi nodi di lavoro, puoi saltare questo passaggio. Se hai creato ruoli separati, assegna ogni ruolo a un account di servizio Kubernetes separato e assegna ogni account di servizio a una specifica di distribuzione del pod Kubernetes che include il proxy Envoy. Per ulteriori informazioni, consulta [Specificare un ruolo IAM per il tuo account di servizio](#) nella Guida per l'utente di Amazon EKS e [Configurazione degli account di servizio per i Pod](#) nella documentazione di Kubernetes.
- Amazon ECS: allega un ruolo di attività Amazon ECS alla definizione dell'attività che include il proxy Envoy. L'attività può essere implementata con il tipo di avvio EC2 o Fargate. Per ulteriori informazioni su come creare un ruolo di attività Amazon ECS e collegarlo a un'attività, consulta [Specificare un ruolo IAM per le tue attività](#).
- Amazon EC2: il ruolo IAM deve essere associato all'istanza Amazon EC2 che ospita il proxy Envoy. Per ulteriori informazioni su come collegare un ruolo a un'istanza Amazon EC2, vedi [Ho creato un ruolo IAM e ora voglio assegnarlo a un'istanza EC2](#).

## Conferma l'autorizzazione

Verificare che l'appmesh:StreamAggregatedResourcesautorizzazione sia assegnata alla risorsa di elaborazione su cui si ospita il proxy selezionando uno dei nomi dei servizi di elaborazione.

### Amazon EKS

È possibile assegnare una politica personalizzata al ruolo assegnato ai nodi di lavoro, ai singoli pod o a entrambi. Si consiglia tuttavia di assegnare la policy solo ai singoli pod, in modo da poter limitare l'accesso dei singoli pod ai singoli endpoint mesh. Se la policy è associata al ruolo assegnato ai nodi di lavoro, seleziona la scheda Amazon EC2 e completa i passaggi indicati

per le istanze del tuo nodo di lavoro. Per determinare quale ruolo IAM è assegnato a un pod Kubernetes, completa i seguenti passaggi.

1. Visualizza i dettagli di una distribuzione Kubernetes che include il pod a cui desideri confermare l'assegnazione di un account di servizio Kubernetes. Il comando seguente visualizza i dettagli di una distribuzione denominata *my-deployment*.

```
kubectl describe deployment my-deployment
```

Nell'output restituito annota il valore a destra di `Service Account:`. Se `Service Account:` non esiste una riga che inizia con, un account di servizio Kubernetes personalizzato non è attualmente assegnato alla distribuzione. Dovrai assegnarne una. Per ulteriori informazioni, consultare [Configurare gli account di servizio per i pod](#) nella documentazione di Kubernetes.

2. Visualizza i dettagli dell'account di servizio restituito nel passaggio precedente. Il comando seguente visualizza i dettagli di un account di servizio denominato *my-service-account*.

```
kubectl describe serviceaccount my-service-account
```

A condizione che l'account del servizio Kubernetes sia associato a unAWS Identity and Access Management ruolo, una delle righe restituite avrà un aspetto simile all'esempio seguente.

```
Annotations: eks.amazonaws.com/role-arn=arn:aws:iam::123456789012:role/
my-deployment
```

Nell'esempio precedente *my-deployment* c'è il nome del ruolo IAM a cui è associato l'account di servizio. Se l'output dell'account di servizio non contiene una riga simile all'esempio precedente, l'account del servizio Kubernetes non è associato a unAWS Identity and Access Management account e devi associarlo a uno. Per ulteriori informazioni, consulta [Specificare un ruolo IAM per l'account di servizio](#).

3. Accedi alla AWS Management Console e apri la console IAM all'indirizzo <https://console.aws.amazon.com/iam/>.
4. Nel riquadro di navigazione di sinistra seleziona Ruoli. Seleziona il nome del ruolo IAM annotato in una fase precedente.

5. Conferma che la politica personalizzata creata in precedenza o la politica [AWSAppMeshEnvoyAccess](#) gestita sia elencata. Se nessuna delle politiche è associata, [allega una politica IAM](#) al ruolo IAM. Se desideri allegare una politica IAM personalizzata ma non ne hai una, devi [creare una politica IAM personalizzata](#) con le autorizzazioni richieste. Se è allegata una politica IAM personalizzata, seleziona la politica e conferma che contiene "Action": "appmesh:StreamAggregatedResources". In caso contrario, è necessario aggiungere tale autorizzazione alla politica IAM personalizzata. Puoi anche confermare l'Amazon Resource Name (ARN) appropriato per uno specifico endpoint mesh. Se non è elencato alcun ARN, puoi modificare la policy per aggiungere, rimuovere o modificare gli ARN elencati. Per ulteriori informazioni, consulta [Modifica di policy IAM](#) e [Creare una policy IAM](#).
6. Ripeti i passaggi precedenti per ogni pod Kubernetes che contiene il proxy Envoy.

## Amazon ECS

1. Dalla console Amazon ECS, scegli Definizioni delle attività.
2. Seleziona l'attività di Amazon ECS.
3. Nella pagina Nome della definizione dell'attività, seleziona la definizione dell'attività.
4. Nella pagina Definizione dell'attività, seleziona il link del nome del ruolo IAM a destra di Task Role. Se un ruolo IAM non è elencato, devi [creare un ruolo IAM](#) e collegarlo all'attività [aggiornando la definizione dell'attività](#).
5. Nella pagina Riepilogo, nella scheda Autorizzazioni, conferma che sia elencata la politica personalizzata creata in precedenza o la politica [AWSAppMeshEnvoyAccess](#) gestita. Se nessuna delle politiche è associata, [allega una politica IAM](#) al ruolo IAM. Se desideri allegare una politica IAM personalizzata ma non ne hai una, devi [creare la politica IAM personalizzata](#). Se è allegata una politica IAM personalizzata, seleziona la politica e conferma che contiene "Action": "appmesh:StreamAggregatedResources". In caso contrario, è necessario aggiungere tale autorizzazione alla politica IAM personalizzata. Puoi anche confermare l'Amazon Resource Name (ARN) appropriato per uno specifico endpoint mesh. Se non è elencato alcun ARN, puoi modificare la policy per aggiungere, rimuovere o modificare gli ARN elencati. Per ulteriori informazioni, consulta [Modifica di policy IAM](#) e [Creare una policy IAM](#).
6. Ripeti i passaggi precedenti per ogni definizione di attività che contiene il proxy Envoy.

## Amazon EC2

1. Dalla console Amazon EC2, seleziona Istanze nella barra di navigazione a sinistra.
2. Seleziona una delle tue istanze che ospita il proxy Envoy.
3. Nella scheda Descrizione, seleziona il link del nome del ruolo IAM a destra del ruolo IAM. Se un ruolo IAM non è elencato, devi [creare un ruolo IAM](#).
4. Nella pagina Riepilogo, nella scheda Autorizzazioni, conferma che sia elencata la politica personalizzata creata in precedenza o la politica [AWSAppMeshEnvoyAccess](#) gestita. Se nessuna delle politiche è associata, [collega la politica IAM](#) al ruolo IAM. Se desideri allegare una politica IAM personalizzata ma non ne hai una, devi [creare la politica IAM personalizzata](#). Se è allegata una politica IAM personalizzata, seleziona la politica e conferma che contiene "Action": "appmesh:StreamAggregatedResources". In caso contrario, è necessario aggiungere tale autorizzazione alla politica IAM personalizzata. Puoi anche confermare l'Amazon Resource Name (ARN) appropriato per uno specifico endpoint mesh. Se non è elencato alcun ARN, puoi modificare la policy per aggiungere, rimuovere o modificare gli ARN elencati. Per ulteriori informazioni, consulta [Modifica di policy IAM](#) e [Creare una policy IAM](#).
5. Ripeti i passaggi precedenti per ogni istanza in cui ospiti il proxy Envoy.

## Risoluzione dei problemi AWS App Mesh identità e accesso

Utilizza le seguenti informazioni per aiutarti a diagnosticare e risolvere i problemi più comuni che potresti riscontrare quando lavori con App Mesh e IAM.

### Argomenti

- [Non sono autorizzato a eseguire un'azione in App Mesh](#)
- [Voglio consentire l'accesso a persone esterne al mio AWS account per accedere alle mie risorse App Mesh](#)

### Non sono autorizzato a eseguire un'azione in App Mesh

Se il file AWS Management Console ti dice che non sei autorizzato a eseguire un'azione, quindi devi contattare l'amministratore per ricevere assistenza. L'amministratore è colui che ti ha fornito le credenziali di accesso.

Il seguente errore si verifica quando l'utente `mateojacksonIAMutente` tenta di utilizzare la console per creare un nodo virtuale denominato `my-virtual-node` nella mesh denominata `my-mesh` ma non ha il `appmesh:CreateVirtualNode` permesso.

```
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
```

In questo caso, Mateo chiede al suo amministratore di aggiornare le sue politiche per consentirgli di creare un nodo virtuale utilizzando l'azione `appmesh:CreateVirtualNode`.

#### Note

Poiché un nodo virtuale viene creato all'interno di una mesh, l'account di Mateo richiede anche `appmesh:ListMeshes` le azioni `appmesh:DescribeMesh` e per creare il nodo virtuale nella console.

## Voglio consentire l'accesso a persone esterne al mio AWS account per accedere alle mie risorse App Mesh

È possibile creare un ruolo con il quale utenti in altri account o persone esterne all'organizzazione possono accedere alle tue risorse. È possibile specificare chi è attendibile per l'assunzione del ruolo. Per i servizi che supportano politiche basate sulle risorse o liste di controllo degli accessi (ACLs), puoi utilizzare tali politiche per concedere alle persone l'accesso alle tue risorse.

Per ulteriori informazioni, consulta gli argomenti seguenti:

- Per sapere se App Mesh supporta queste funzionalità, consulta [In che modo AWS App Mesh funziona con IAM](#).
- Per scoprire come fornire l'accesso alle tue risorse in tutto il mondo Account AWS di cui sei proprietario, vedi [Fornire l'accesso a un IAM utente in un altro Account AWS che possiedi](#) nella Guida per l'IAMutente.
- Per scoprire come fornire l'accesso alle tue risorse a terze parti Account AWS, vedi [Fornire l'accesso a Account AWS di proprietà di terzi](#) nella Guida per l'IAMutente.
- Per informazioni su come fornire l'accesso tramite la federazione delle identità, consulta [Fornire l'accesso a utenti autenticati esternamente \(federazione delle identità\)](#) nella Guida per l'IAMutente.

- Per conoscere la differenza tra l'utilizzo di ruoli e politiche basate sulle risorse per l'accesso tra account diversi, consulta la sezione Accesso alle [risorse tra account nella Guida per l'utente](#). IAM IAM

## Registrazione delle chiamate AWS App Mesh API utilizzando AWS CloudTrail

AWS App Mesh è integrato con [AWS CloudTrail](#), un servizio che fornisce una registrazione delle azioni intraprese da un utente, ruolo o un Servizio AWS. CloudTrail acquisisce tutte le chiamate API per App Mesh come eventi. Le chiamate acquisite includono chiamate dalla console App Mesh e chiamate di codice alle operazioni dell'API App Mesh. Utilizzando le informazioni raccolte da CloudTrail, è possibile determinare la richiesta effettuata ad App Mesh, l'indirizzo IP da cui è stata effettuata la richiesta, quando è stata effettuata e dettagli aggiuntivi.

Ogni evento o voce di log contiene informazioni sull'utente che ha generato la richiesta. Le informazioni di identità consentono di determinare quanto segue:

- Se la richiesta è stata effettuata con le credenziali utente root o utente.
- Se la richiesta è stata effettuata per conto di un utente IAM Identity Center.
- Se la richiesta è stata effettuata con le credenziali di sicurezza temporanee per un ruolo o un utente federato.
- Se la richiesta è stata effettuata da un altro Servizio AWS.

CloudTrail è attivo nel tuo account Account AWS quando crei l'account e hai automaticamente accesso alla cronologia degli CloudTrail eventi. La cronologia CloudTrail degli eventi fornisce un record visualizzabile, ricercabile, scaricabile e immutabile degli ultimi 90 giorni di eventi di gestione registrati in un. Regione AWS Per ulteriori informazioni, consulta [Lavorare con la cronologia degli CloudTrail eventi](#) nella Guida per l'utente. AWS CloudTrail Non sono CloudTrail previsti costi per la visualizzazione della cronologia degli eventi.

Per una registrazione continua degli eventi degli Account AWS ultimi 90 giorni, crea un trail o un data store di eventi [CloudTrailLake](#).

### CloudTrail sentieri

Un trail consente di CloudTrail inviare file di log a un bucket Amazon S3. Tutti i percorsi creati utilizzando il AWS Management Console sono multiregionali. È possibile creare un percorso a

regione singola o multiregione utilizzando. AWS CLI La creazione di un percorso multiregionale è consigliata in quanto consente di registrare l'intera attività del proprio account Regioni AWS . Se crei un percorso a regione singola, puoi visualizzare solo gli eventi registrati nel percorso. Regione AWS Per ulteriori informazioni sui percorsi, consulta [Creazione di un percorso per te Account AWS](#) e [Creazione di un percorso per un'organizzazione nella Guida](#) per l'AWS CloudTrail utente.

Puoi inviare gratuitamente una copia dei tuoi eventi di gestione in corso al tuo bucket Amazon S3 CloudTrail creando un percorso, tuttavia ci sono costi di storage di Amazon S3. [Per ulteriori informazioni sui CloudTrail prezzi, consulta la pagina Prezzi.AWS CloudTrail](#) Per informazioni sui prezzi di Amazon S3, consulta [Prezzi di Amazon S3](#).

## CloudTrail Archivi di dati sugli eventi di Lake

CloudTrail Lake ti consente di eseguire query basate su SQL sui tuoi eventi. CloudTrail [Lake converte gli eventi esistenti in formato JSON basato su righe in formato Apache ORC](#). ORC è un formato di archiviazione a colonne ottimizzato per il recupero rapido dei dati. Gli eventi vengono aggregati in archivi di dati degli eventi, che sono raccolte di eventi immutabili basate sui criteri selezionati applicando i [selettori di eventi avanzati](#). I selettori applicati a un archivio di dati degli eventi controllano quali eventi persistono e sono disponibili per l'esecuzione della query. Per ulteriori informazioni su CloudTrail Lake, consulta [Working with AWS CloudTrail Lake](#) nella Guida per l'utente.AWS CloudTrail

CloudTrail Gli archivi e le richieste di dati sugli eventi di Lake comportano dei costi. Quando crei un datastore di eventi, scegli l'[opzione di prezzo](#) da utilizzare per tale datastore. L'opzione di prezzo determina il costo per l'importazione e l'archiviazione degli eventi, nonché il periodo di conservazione predefinito e quello massimo per il datastore di eventi. [Per ulteriori informazioni sui CloudTrail prezzi, consulta la sezione Prezzi.AWS CloudTrail](#)

## Eventi di gestione App Mesh in CloudTrail

[Gli eventi](#) di gestione forniscono informazioni sulle operazioni di gestione eseguite sulle risorse di Account AWS. Queste operazioni sono definite anche operazioni del piano di controllo (control-plane). Per impostazione predefinita, CloudTrail registra gli eventi di gestione.

AWS App Mesh registra tutte le operazioni del piano di controllo App Mesh come eventi di gestione. Per un elenco delle operazioni del piano di AWS App Mesh controllo a cui App Mesh accede CloudTrail, consulta l'[AWS App Mesh API Reference](#).

## Esempi di eventi App Mesh

Un evento rappresenta una singola richiesta proveniente da qualsiasi fonte e include informazioni sull'operazione API richiesta, la data e l'ora dell'operazione, i parametri della richiesta e così via. CloudTrail i file di registro non sono una traccia ordinata dello stack delle chiamate API pubbliche, quindi gli eventi non vengono visualizzati in un ordine specifico.

L'esempio seguente mostra una voce di CloudTrail registro che illustra l'StreamAggregatedResourcesazione.

```
{
 "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"
}
```

Per informazioni sul contenuto dei CloudTrail record, consultate il [contenuto dei CloudTrail record](#) nella Guida per l'AWS CloudTrail utente.

## Protezione dei dati in AWS App Mesh

Il AWS modello di [responsabilità condivisa modello](#) di si applica alla protezione dei dati in AWS App Mesh. Come descritto in questo modello, AWS è responsabile della protezione dell'infrastruttura globale che gestisce tutte le Cloud AWS. L'utente è responsabile del mantenimento del controllo sui contenuti ospitati su questa infrastruttura. L'utente è inoltre responsabile delle attività di configurazione e gestione della sicurezza per Servizi AWS che usi. Per ulteriori informazioni sulla privacy dei dati, consulta la sezione [Privacy dei dati FAQ](#). Per informazioni sulla protezione dei dati in Europa, consulta la [AWS Modello di responsabilità condivisa e post sul GDPR](#) blog sul AWS Blog sulla sicurezza.

Ai fini della protezione dei dati, ti consigliamo di proteggere Account AWS credenziali e configura singoli utenti con AWS IAM Identity Center oppure AWS Identity and Access Management (IAM). In tal modo, a ogni utente verranno assegnate solo le autorizzazioni necessarie per svolgere i suoi compiti. Ti suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Usa l'autenticazione a più fattori (MFA) con ogni account.
- Usa SSL/TLS per comunicare con AWS risorse. Richiediamo TLS 1.2 e consigliamo TLS 1.3.
- Configurazione API e registrazione delle attività degli utenti con AWS CloudTrail. Per informazioni sull'utilizzo dei CloudTrail percorsi per l'acquisizione AWS attività, vedi [Lavorare con i CloudTrail sentieri](#) in AWS CloudTrail Guida per l'utente.
- Utilizzo AWS soluzioni di crittografia, insieme a tutti i controlli di sicurezza predefiniti all'interno Servizi AWS.
- Utilizza i servizi di sicurezza gestiti avanzati, come Amazon Macie, che aiutano a individuare e proteggere i dati sensibili archiviati in Amazon S3.
- Se sono necessari FIPS 140-3 moduli crittografici convalidati per l'accesso AWS tramite un'interfaccia a riga di comando o un'API, utilizza un endpoint. FIPS Per ulteriori informazioni sugli FIPS endpoint disponibili, vedere [Federal Information Processing Standard \(FIPS\) 140-3](#).

Ti consigliamo vivamente di non inserire mai informazioni riservate o sensibili, ad esempio gli indirizzi e-mail dei clienti, nei tag o nei campi di testo in formato libero, ad esempio nel campo Nome. Ciò include quando lavori con App Mesh o altro Servizi AWS utilizzando la console API, AWS CLI, oppure AWS SDKs. I dati inseriti nei tag o nei campi di testo in formato libero utilizzati per i nomi possono

essere utilizzati per i la fatturazione o i log di diagnostica. Se fornisci un URL a un server esterno, ti consigliamo vivamente di non includere le informazioni sulle credenziali URL per convalidare la tua richiesta a quel server.

## Crittografia dei dati

I tuoi dati vengono crittografati quando utilizzi App Mesh.

### Crittografia a riposo

Per impostazione predefinita, le configurazioni App Mesh che crei sono crittografate a riposo.

### Crittografia in transito

Gli endpoint del servizio App Mesh utilizzano il HTTPS protocollo. Tutte le comunicazioni tra il proxy Envoy e l'App Mesh Envoy Management Service sono crittografate. Se è necessaria una crittografia FIPS conforme per la comunicazione tra il proxy Envoy e l'App Mesh Envoy Management Service, è possibile utilizzare una FIPS variante dell'immagine del contenitore proxy Envoy. Per ulteriori informazioni, consulta [Immagine dell'inviato](#).

La comunicazione tra contenitori all'interno dei nodi virtuali non è crittografata, ma questo traffico non esce dallo spazio dei nomi di rete.

## Convalida della conformità per AWS App Mesh

Per sapere se un Servizio AWS programma rientra nell'ambito di specifici programmi di conformità, consulta Servizi AWS la sezione [Scope by Compliance Program Servizi AWS](#) e scegli il programma di conformità che ti interessa. Per informazioni generali, consulta Programmi di [AWS conformità Programmi](#) di di .

È possibile scaricare report di audit di terze parti utilizzando AWS Artifact. Per ulteriori informazioni, consulta [Scaricamento dei report in AWS Artifact](#) .

La vostra responsabilità di conformità durante l'utilizzo Servizi AWS è determinata dalla sensibilità dei dati, dagli obiettivi di conformità dell'azienda e dalle leggi e dai regolamenti applicabili. AWS fornisce le seguenti risorse per contribuire alla conformità:

- [Guide introduttive su sicurezza e conformità](#): queste guide all'implementazione illustrano considerazioni sull'architettura e forniscono passaggi per implementare ambienti di base incentrati sulla AWS sicurezza e la conformità.

- [Architettura per la HIPAA sicurezza e la conformità su Amazon Web Services](#): questo white paper descrive in che modo le aziende possono utilizzare AWS per creare applicazioni idonee. HIPAA

#### Note

Non tutte sono idonee. Servizi AWS HIPAA Per ulteriori informazioni, consulta la [Guida ai servizi HIPAA idonei](#).

- [AWS Risorse per la per la conformità](#): questa raccolta di cartelle di lavoro e guide potrebbe riguardare il settore e la località in cui operi.
- [AWS Guide alla conformità dei clienti](#): comprendi il modello di responsabilità condivisa attraverso la lente della conformità. Le guide riassumono le migliori pratiche per la protezione Servizi AWS e mappano le linee guida per i controlli di sicurezza su più framework (tra cui il National Institute of Standards and Technology (NIST), il Payment Card Industry Security Standards Council (PCI) e l'International Organization for Standardization ()). ISO
- [Evaluating Resources with Rules](#) nella Guida per gli AWS Config sviluppatori: il AWS Config servizio valuta la conformità delle configurazioni delle risorse alle pratiche interne, alle linee guida del settore e alle normative.
- [AWS Security Hub](#)— Ciò Servizio AWS fornisce una visione completa dello stato di sicurezza interno. AWS La Centrale di sicurezza utilizza i controlli di sicurezza per valutare le risorse AWS e verificare la conformità agli standard e alle best practice del settore della sicurezza. Per un elenco dei servizi e dei controlli supportati, consulta la pagina [Documentazione di riferimento sui controlli della Centrale di sicurezza](#).
- [Amazon GuardDuty](#): Servizio AWS rileva potenziali minacce ai tuoi carichi di lavoro Account AWS, ai contenitori e ai dati monitorando l'ambiente alla ricerca di attività sospette e dannose. GuardDuty può aiutarti a soddisfare vari requisiti di conformità, ad esempio PCI DSS soddisfacendo i requisiti di rilevamento delle intrusioni imposti da determinati framework di conformità.
- [AWS Audit Manager](#)— Ciò Servizio AWS consente di verificare continuamente AWS l'utilizzo per semplificare la gestione del rischio e la conformità alle normative e agli standard di settore.

## Sicurezza dell'infrastruttura in AWS App Mesh

In quanto servizio gestito, AWS App Mesh è protetto dalla sicurezza di rete AWS globale. Per informazioni sui servizi AWS di sicurezza e su come AWS protegge l'infrastruttura, consulta [AWS Cloud Security](#). Per progettare il tuo AWS ambiente utilizzando le migliori pratiche per la sicurezza dell'infrastruttura, vedi [Infrastructure Protection](#) in Security Pillar AWS Well-Architected Framework.

Utilizzi API le chiamate AWS pubblicate per accedere ad App Mesh attraverso la rete. I client devono supportare quanto segue:

- Transport Layer Security (TLS). Richiediamo TLS 1.2 e consigliamo TLS 1.3.
- Suite di cifratura con Perfect Forward Secrecy (PFS) come (Ephemeral Diffie-Hellman) o DHE (Elliptic Curve Ephemeral Diffie-Hellman). ECDHE La maggior parte dei sistemi moderni, come Java 7 e versioni successive, supporta tali modalità.

Inoltre, le richieste devono essere firmate utilizzando un ID chiave di accesso e una chiave di accesso segreta associata a un principale. IAM O puoi utilizzare [AWS Security Token Service](#) (AWS STS) per generare credenziali di sicurezza temporanee per sottoscrivere le richieste.

Puoi migliorare il livello di sicurezza del tuo dispositivo VPC configurando App Mesh per utilizzare un endpoint di interfacciaVPC. Per ulteriori informazioni, consulta [Endpoint VPC con interfaccia App Mesh \(\)AWS PrivateLink](#).

## Endpoint VPC con interfaccia App Mesh ()AWS PrivateLink

Puoi migliorare il livello di sicurezza del tuo Amazon VPC configurando App Mesh per utilizzare un endpoint VPC di interfaccia. Gli endpoint di interfaccia sono alimentati da AWS PrivateLink, una tecnologia che consente di accedere in modo privato alle API App Mesh utilizzando indirizzi IP privati. PrivateLinklimita tutto il traffico di rete tra Amazon VPC e App Mesh alla rete Amazon.

La configurazione non è obbligatoria PrivateLink, ma la consigliamo. Per ulteriori informazioni sugli endpoint VPC PrivateLink e sull'interfaccia, consulta [Accesso ai servizi tramite](#). AWS PrivateLink

## Considerazioni sugli endpoint VPC dell'interfaccia App Mesh

Prima di configurare gli endpoint VPC dell'interfaccia per App Mesh, tieni presente le seguenti considerazioni:

- Se il tuo Amazon VPC non dispone di un gateway Internet e le tue attività utilizzano il driver di `awslogs` registro per inviare informazioni di log a CloudWatch Logs, devi creare un endpoint VPC di interfaccia per Logs. CloudWatch Per ulteriori informazioni, consulta [Using CloudWatch Logs with Interface VPC Endpoints](#) nella `CloudWatch Amazon Logs User Guide`.
- Gli endpoint VPC non supportano AWS le richieste interregionali. Assicurati di creare l'endpoint nella stessa regione in cui intendi inviare le chiamate API ad App Mesh.

- Gli endpoint VPC supportano solo il DNS fornito da Amazon tramite Amazon Route 53. Se si desidera utilizzare il proprio DNS, è possibile usare l'inoltro condizionale sul DNS. Per ulteriori informazioni, consulta [Set opzioni DHCP](#) nella Guida per l'utente di Amazon VPC.
- Il gruppo di sicurezza collegato all'endpoint VPC deve consentire le connessioni in entrata sulla porta 443 dalla sottorete privata di Amazon VPC.

#### Note

Il controllo dell'accesso ad App Mesh allegando una policy dell'endpoint all'endpoint VPC (ad esempio, utilizzando il nome del servizio `com.amazonaws.Region.appmesh-envoy-management`) non è supportato per la connessione Envoy.

[Per ulteriori considerazioni e limitazioni, consulta \*Interface Endpoint Availability Zone Considerations\* e \*Interface Endpoint Properties and Limitations\*.](#)

## Crea l'endpoint VPC dell'interfaccia per App Mesh

Per creare l'endpoint VPC di interfaccia per il servizio App Mesh, utilizza la procedura [Creating an Interface Endpoint](#) nella Amazon VPC User Guide. Specificate `com.amazonaws.Region.appmesh-envoy-management` il nome del servizio per il vostro proxy Envoy per la connessione al servizio pubblico di gestione Envoy di App Mesh e per le operazioni `mesh.com.amazonaws.Region.appmesh`

#### Note

La *regione* rappresenta l'identificatore della regione per una AWS regione supportata da App Mesh, ad esempio `us-east-2` per la regione Stati Uniti orientali (Ohio).

Sebbene sia possibile definire un endpoint VPC di interfaccia per App Mesh in qualsiasi regione in cui App Mesh è supportato, potresti non essere in grado di definire un endpoint per tutte le zone di disponibilità in ciascuna regione. Per scoprire quali zone di disponibilità sono supportate con gli endpoint VPC di interfaccia in una regione, usa il [describe-vpc-endpoint-services](#) comando o usa il AWS Management Console. Ad esempio, i seguenti comandi restituiscono le zone di disponibilità in cui è possibile distribuire un endpoint VPC con interfaccia App Mesh all'interno della regione Stati Uniti orientali (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[]'
```

## Resilienza in AWS App Mesh

L'infrastruttura globale di AWS è basata su Regioni e zone di disponibilità AWS. AWS Le Regioni forniscono più zone di disponibilità fisicamente separate e isolate che sono connesse tramite reti altamente ridondanti, a bassa latenza e velocità effettiva elevata. Con le zone di disponibilità, è possibile progettare e gestire le applicazioni e database che eseguono il failover automatico tra zone di disponibilità senza interruzioni. Le zone di disponibilità sono più disponibili, fault tolerant e scalabili rispetto alle infrastrutture a data center singolo o multiplo.

App Mesh esegue istanze del piano di controllo basate sul carico, e fornisce loro gli aggiornamenti. App Mesh rileva e sostituisce automaticamente le istanze del piano di controllo non integre e fornisce loro gli aggiornamenti automatici delle versioni e le patch.

## Disaster recovery in AWS App Mesh

Il servizio App Mesh gestisce i backup dei dati dei clienti. Non è necessario fare nulla per gestire i backup. I dati di backup sono crittografati.

## Analisi della configurazione e della vulnerabilità in AWS App Mesh

App Mesh fornisce un'[immagine del contenitore Docker proxy Envoy](#) gestita che distribuisce con i tuoi microservizi. App Mesh garantisce che l'immagine del contenitore sia aggiornata con le ultime patch di vulnerabilità e prestazioni. App Mesh testa le nuove versioni del proxy Envoy rispetto al set di funzionalità App Mesh prima di rendere disponibili le immagini.

È necessario aggiornare i microservizi per utilizzare la versione aggiornata dell'immagine del contenitore. Di seguito è riportata la versione più recente dell'immagine.

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.29.6.1-prod
```

# Risoluzione dei problemi di App Mesh

Questo capitolo illustra le best practice per la risoluzione dei problemi e i problemi più comuni che potresti riscontrare durante l'utilizzo di App Mesh. Seleziona una delle seguenti aree per esaminare le migliori pratiche e i problemi comuni relativi a quell'area.

## Argomenti

- [Best practice per la risoluzione dei problemi di App Mesh](#)
- [Risoluzione dei problemi relativi alla configurazione di App Mesh](#)
- [Risoluzione dei problemi di connettività App Mesh](#)
- [Risoluzione dei problemi di scalabilità dell'App Mesh](#)
- [Risoluzione dei problemi di osservabilità dell'App Mesh](#)
- [Risoluzione dei problemi di sicurezza di App Mesh](#)
- [Risoluzione dei problemi di App Mesh Kubernetes](#)

## Best practice per la risoluzione dei problemi di App Mesh

Ti consigliamo di seguire le best practice riportate in questo argomento per risolvere i problemi relativi all'utilizzo di App Mesh.

### Abilita l'interfaccia di amministrazione del proxy Envoy

Il proxy Envoy viene fornito con un'interfaccia di amministrazione che è possibile utilizzare per rilevare configurazioni e statistiche e per eseguire altre funzioni amministrative come il drenaggio della connessione. Per ulteriori informazioni, vedere [Interfaccia di amministrazione](#) nella documentazione di Envoy.

Se si utilizza l'endpoint gestito [Immagine dell'inviato](#), l'endpoint di amministrazione è abilitato per impostazione predefinita sulla porta 9901. Gli esempi forniti in [Risoluzione dei problemi relativi alla configurazione di App Mesh](#) Visualizza l'URL dell'endpoint di amministrazione di esempio come.  
`http://my-app.default.svc.cluster.local:9901/`

#### Note

L'endpoint di amministrazione non deve mai essere esposto alla rete Internet pubblica. Inoltre, consigliamo di monitorare i log degli endpoint di amministrazione, che sono impostati

dalla variabile di ENVY\_ADMIN\_ACCESS\_LOG\_FILE ambiente come impostazione predefinita. /tmp/envoy\_admin\_access.log

## Abilita l'integrazione con Envoy DogStats D per l'offload metrico

Il proxy Envoy può essere configurato per scaricare le statistiche per il traffico OSI Layer 4 e Layer 7 e per lo stato di salute dei processi interni. Sebbene questo argomento mostri come utilizzare queste statistiche senza scaricare le metriche su sink come CloudWatch Metrics e Prometheus., disporre di queste statistiche in una posizione centralizzata per tutte le applicazioni può aiutarti a diagnosticare i problemi e confermare il comportamento più rapidamente. Per ulteriori informazioni, consulta [Using Amazon CloudWatch Metrics](#) e la documentazione di [Prometheus](#).

Puoi configurare i parametri DogStats D impostando i parametri definiti in. [DogStatsVariabili D](#) Per ulteriori informazioni su DogStats D, consulta la documentazione [DogStatsD](#). Puoi trovare una dimostrazione dell'offload delle metriche alle AWS CloudWatch metriche nella procedura dettagliata App [Mesh with Amazon ECS](#) basics. GitHub

## Abilitare log di accesso

Ti consigliamo di abilitare i log di accesso sulle tue applicazioni e di scoprire dettagli sul traffico in transito tra [Nodi virtuali Gateway virtuali virtuale virtuale virtuale virtuale virtuale](#) le tue applicazioni. Per ulteriori informazioni, consulta la sezione [Registrazione degli accessi nella documentazione di Envoy](#). I log forniscono informazioni dettagliate sul comportamento del traffico OSI Layer 4 e Layer 7. Quando si utilizza il formato predefinito di Envoy, è possibile analizzare i log di accesso con [CloudWatch Logs Insights](#) utilizzando la seguente istruzione di analisi.

```
parse @message "[*] \"* * *\" * * * * * * * * * * *\" as StartTime, Method, Path, Protocol, ResponseCode, ResponseFlags, BytesReceived, BytesSent, DurationMillis, UpstreamServiceTimeMillis, ForwardedFor, UserAgent, RequestId, Authority, UpstreamHost
```

## Abilita la registrazione di debug di Envoy negli ambienti di riproduzione

Consigliamo di impostare il livello di registro del proxy Envoy su un ambiente di riproduzione. debug I log di debug possono aiutarti a identificare i problemi prima di trasferire la configurazione App Mesh associata all'ambiente di produzione.

Se stai usando l'[immagine Envoy](#), puoi impostare il livello di registro debug tramite la variabile di ambiente. ENVY\_LOG\_LEVEL

### Note

Non è consigliabile utilizzare il debug livello negli ambienti di produzione. [L'impostazione del livello su debug aumenta la registrazione e può influire sulle prestazioni e sul costo complessivo dei log scaricati su soluzioni come Logs. CloudWatch](#)

Quando si utilizza il formato predefinito di Envoy, è possibile analizzare i log dei processi con [CloudWatch Logs](#) Insights utilizzando la seguente istruzione di analisi:

```
parse @message "[*][*][*][*] [*] *" as Time, Thread, Level, Name, Source, Message
```

## Monitora la connettività del proxy Envoy con il piano di controllo App Mesh

Ti consigliamo di monitorare le metriche di Envoy `control_plane.connected_state` per assicurarti che il proxy Envoy comunichi con il piano di controllo dell'App Mesh per recuperare le risorse di configurazione dinamica. [Per ulteriori informazioni, vedere Management Server.](#)

## Risoluzione dei problemi relativi alla configurazione di App Mesh

Questo argomento descrive i problemi più comuni che potrebbero verificarsi con la configurazione di App Mesh.

### Impossibile estrarre l'immagine del contenitore Envoy

#### Caratteristiche

Riceverai il seguente messaggio di errore in un'attività Amazon ECS. L'*ID e la regione dell'account* Amazon ECR nel messaggio seguente possono essere diversi, a seconda del repository Amazon ECR da cui hai estratto l'immagine del contenitore.

```
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'
```

#### Risoluzione

Questo errore indica che il ruolo di esecuzione dell'attività utilizzato non dispone dell'autorizzazione per comunicare con Amazon ECR e non può estrarre l'immagine del contenitore Envoy dal

repository. Il ruolo di esecuzione dell'attività assegnato all'attività Amazon ECS richiede una policy IAM con le seguenti istruzioni:

```
{
 "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"
}
```

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Impossibile connettersi al servizio di gestione App Mesh Envoy

### Caratteristiche

Il proxy Envoy non è in grado di connettersi al servizio di gestione App Mesh Envoy. Stai vedendo:

- Errori di connessione rifiutata
- Timeout di connessione
- Errori nella risoluzione dell'endpoint del servizio di gestione App Mesh Envoy
- Errori gRPC

### Risoluzione

Assicurati che il tuo proxy Envoy abbia accesso a Internet o a un [endpoint VPC](#) privato e che i tuoi [gruppi di sicurezza](#) consentano il traffico in uscita sulla porta 443. Gli endpoint del servizio di gestione Envoy pubblico di App Mesh seguono il formato FQDN (Fully Qualified Domain Name).

```
App Mesh Production Endpoint
appmesh-envoy-management.Region-code.amazonaws.com

App Mesh Preview Endpoint
```

```
appmesh-preview-envoy-management.Region-code.amazonaws.com
```

È possibile eseguire il debug della connessione a EMS utilizzando il comando seguente. Questo invia una richiesta gRPC valida ma vuota all'Envoy Management Service.

```
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
```

Se ricevi questi messaggi, la tua connessione all'Envoy Management Service è funzionante. Per il debug degli errori relativi a gRPC, vedere gli errori [in Envoy disconnesso dal servizio di gestione App Mesh Envoy](#) con testo di errore.

```
grpc-status: 16
grpc-message: Missing Authentication Token
```

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Envoy disconnesso dal servizio di gestione App Mesh Envoy con testo di errore

### Caratteristiche

Il proxy Envoy non è in grado di connettersi al servizio di gestione App Mesh Envoy e di riceverne la configurazione. I log del proxy Envoy contengono una voce di registro come la seguente.

```
gRPC config stream closed: gRPC status code, message
```

### Risoluzione

Nella maggior parte dei casi, la parte del registro relativa ai messaggi dovrebbe indicare il problema. La tabella seguente elenca i codici di stato gRPC più comuni che potreste vedere, le loro cause e le relative risoluzioni.

| Codice di stato gRPC | Causa                                               | Risoluzione                                                                     |
|----------------------|-----------------------------------------------------|---------------------------------------------------------------------------------|
| 0                    | Graceful disconnect dal servizio di gestione Envoy. | Non c'è nessun problema. App Mesh disconnette occasionalmente i proxy Envoy con |

| Codice di stato gRPC | Causa                                                                                                                                    | Risoluzione                                                                                                                                                                                                                                                                      |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                      |                                                                                                                                          | questo codice di stato. Envoy si riconnetterà e continuerà a ricevere aggiornamenti.                                                                                                                                                                                             |
| 3                    | L'endpoint mesh (nodo virtuale o gateway virtuale) o una delle risorse associate non è stato trovato.                                    | Ricontrolla la configurazione di Envoy per assicurarti che abbia il nome appropriato della risorsa App Mesh che rappresenta. Se la tua risorsa App Mesh è integrata con altre AWS risorse, come AWS Cloud Map namespace o certificati ACM, assicurati che tali risorse esistano. |
| 7                    | Il proxy Envoy non è autorizzato a eseguire un'azione, come connettersi al servizio di gestione Envoy o recuperare le risorse associate. | Assicurati di <a href="#">creare una policy IAM</a> che contenga le dichiarazioni politiche appropriate per App Mesh e altri servizi e di allegare tale policy all'utente o al ruolo IAM che il tuo proxy Envoy utilizza per connettersi al servizio di gestione Envoy.          |
| 8                    | Il numero di proxy Envoy per una determinata risorsa App Mesh supera la quota di servizio a livello di account.                          | Vedi <a href="#">Quotazioni di App Mesh Controller</a> per informazioni sulle quote predefinite degli account e su come richiedere un aumento della quota.                                                                                                                       |

| Codice di stato gRPC | Causa                                                                       | Risoluzione                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 16                   | Il proxy Envoy non dispone di credenziali di autenticazione valide per. AWS | Assicurati che l'Envoy disponga delle credenziali appropriate per connettersi ai AWS servizi tramite un utente o un ruolo IAM. Un problema noto, <a href="#">#24136</a> , in Envoy per le versioni v1.24 e precedenti non riesce a recuperare le credenziali se il processo Envoy utilizza più descrittori di file. 1024. Ciò accade quando Envoy sta servendo un volume di traffico elevato. Puoi confermare questo problema controllando i log di Envoy a livello di debug per il testo "». A libcurl function was given a bad argument. Per mitigare questo problema, esegui l'aggiornamento alla versione di Envoy o successiva. v1.25.1.0-prod |

Puoi osservare i codici di stato e i messaggi del tuo proxy Envoy con [Amazon CloudWatch Insights](#) utilizzando la seguente query:

```
filter @message like /gRPC config stream closed/
| parse @message "gRPC config stream closed: *, *" as StatusCode, Message
```

[Se il messaggio di errore fornito non è stato utile o il problema persiste, valuta la possibilità di aprire un GitHub problema.](#)

## Controllo dello stato del container Envoy, sonda di prontezza o sonda di vivacità guasto

### Caratteristiche

Il tuo proxy Envoy non supera i controlli di integrità in un'attività Amazon ECS, un'istanza Amazon EC2 o un pod Kubernetes. Ad esempio, esegui una query sull'interfaccia di amministrazione di Envoy con il seguente comando e ricevi uno stato diverso da LIVE

```
curl -s http://my-app.default.svc.cluster.local:9901/server_info | jq '.state'
```

### Risoluzione

Di seguito è riportato un elenco di passaggi di riparazione in base allo stato restituito dal proxy Envoy.

- **PRE\_INITIALIZING** oppure **INITIALIZING** — Il proxy Envoy non ha ancora ricevuto la configurazione oppure non è in grado di connettersi e recuperare la configurazione dal servizio di gestione App Mesh Envoy. È possibile che Envoy riceva un errore dal servizio di gestione di Envoy durante il tentativo di connessione. Per ulteriori informazioni, consulta gli errori in [Envoy disconnesso dal servizio di gestione App Mesh Envoy con testo di errore](#)
- **DRAINING** — Il proxy Envoy ha iniziato a prosciugare le connessioni in risposta a una `drain_listeners` richiesta `/healthcheck/fail` or sull'interfaccia di amministrazione di Envoy. Non è consigliabile richiamare questi percorsi sull'interfaccia di amministrazione a meno che tu non stia per terminare l'attività Amazon ECS, l'istanza Amazon EC2 o il pod Kubernetes.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Il controllo dello stato di salute dal load balancer all'endpoint mesh non riesce

### Caratteristiche

L'endpoint mesh è considerato integro dal controllo dello stato del contenitore o dalla sonda di prontezza, ma il controllo dello stato dal load balancer all'endpoint mesh non riesce.

### Risoluzione

Per risolvere il problema, completa le seguenti attività.

- Assicurati che il [gruppo di sicurezza](#) associato al tuo endpoint mesh accetti il traffico in entrata sulla porta che hai configurato per il controllo dello stato.
- Assicurati che il controllo dello stato abbia esito positivo in modo coerente quando richiesto manualmente, ad esempio da un [host bastion all'interno del tuo VPC](#).
- Se stai configurando un controllo dello stato di salute per un nodo virtuale, ti consigliamo di implementare un endpoint per il controllo dello stato nell'applicazione, ad esempio /ping per HTTP. Ciò garantisce che sia il proxy Envoy che l'applicazione siano instradabili dal sistema di bilanciamento del carico.
- È possibile utilizzare qualsiasi tipo di bilanciamento del carico elastico per il nodo virtuale, a seconda delle funzionalità necessarie. Per ulteriori informazioni, consulta le funzionalità di [Elastic Load Balancing](#).
- Se stai configurando un controllo dello stato di integrità per un [gateway virtuale](#), ti consigliamo di utilizzare un [sistema di bilanciamento del carico di rete](#) con un controllo dello stato TCP o TLS sulla porta listener del gateway virtuale. Ciò garantisce che il listener del gateway virtuale sia avviato e pronto ad accettare connessioni.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Il gateway virtuale non accetta traffico sulle porte 1024 o inferiori

### Caratteristiche

Il gateway virtuale non accetta traffico sulla porta 1024 o inferiore, ma accetta traffico su un numero di porta maggiore di 1024. Ad esempio, si interrogano le statistiche di Envoy con il seguente comando e si riceve un valore diverso da zero.

```
curl -s http://my-app.default.svc.cluster.local:9901/stats | grep "update_rejected"
```

Nei log potresti visualizzare un testo simile al seguente che descrive un errore di associazione a una porta privilegiata:

```
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
```

### Risoluzione

Per risolvere il problema, l'utente specificato per il gateway deve disporre della funzionalità linux. CAP\_NET\_BIND\_SERVICE Per ulteriori informazioni, consulta [Capabilities](#) nel Linux Programmer's Manual, [Linux parameters](#) in ECS Task definition parameters e [Set abilities for a container nella documentazione di Kubernetes](#).

#### Important

Fargate deve utilizzare un valore di porta maggiore di 1024.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Risoluzione dei problemi di connettività App Mesh

Questo argomento descrive i problemi più comuni che potrebbero verificarsi con la connettività App Mesh.

### Impossibile risolvere il nome DNS per un servizio virtuale

#### Caratteristiche

L'applicazione non è in grado di risolvere il nome DNS di un servizio virtuale a cui sta tentando di connettersi.

#### Risoluzione

Si tratta di un problema noto. Per ulteriori informazioni, consulta il problema [Name VirtualServices by any hostname o FQDN](#). GitHub I servizi virtuali in App Mesh possono essere denominati con qualsiasi nome. Finché esiste un A record DNS per il nome di servizio virtuale e l'applicazione è in grado di risolvere il nome del servizio virtuale, la richiesta verrà inoltrata da Envoy e indirizzata alla destinazione appropriata. Per risolvere il problema, aggiungi un A record DNS a qualsiasi indirizzo IP non di loopback, ad esempio per il nome del servizio virtuale. 10.10.10.10 Il A record DNS può essere aggiunto nelle seguenti condizioni:

- In Amazon Route 53, se il nome ha il suffisso del nome della tua zona ospitata privata
- All'interno del file del contenitore dell'applicazione /etc/hosts
- In un server DNS di terze parti che gestisci

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Impossibile connettersi a un backend di servizio virtuale

### Caratteristiche

L'applicazione non è in grado di stabilire una connessione a un servizio virtuale definito come backend sul nodo virtuale. Quando si tenta di stabilire una connessione, la connessione potrebbe fallire completamente oppure la richiesta dal punto di vista dell'applicazione potrebbe fallire con un codice di HTTP 503 risposta.

### Risoluzione

Se l'applicazione non riesce affatto a connettersi (non viene restituito alcun codice di HTTP 503 risposta), procedi come segue:

- Assicurati che il tuo ambiente di calcolo sia configurato per funzionare con App Mesh.
  - Per Amazon ECS, assicurati di avere abilitata la [configurazione proxy](#) appropriata. Per una end-to-end procedura dettagliata, consulta [Getting Started with App Mesh e Amazon ECS](#).
  - Per Kubernetes, incluso Amazon EKS, assicurati di avere il controller App Mesh più recente installato tramite Helm. Per ulteriori informazioni, consulta [App Mesh Controller](#) su Helm Hub o [Tutorial: Configura l'integrazione di App Mesh con Kubernetes](#).
  - Per Amazon EC2, assicurati di aver configurato l'istanza Amazon EC2 per il trasferimento tramite proxy del traffico App Mesh. [Per ulteriori informazioni, consulta Update services](#).
- Assicurati che il contenitore Envoy in esecuzione sul tuo servizio di elaborazione si sia connesso correttamente al servizio di gestione App Mesh Envoy. Puoi confermarlo controllando le statistiche di Envoy per il campo `control_plane.connected_state`. Per ulteriori informazioni `control_plane.connected_state`, consulta [Monitorare la connettività del proxy Envoy](#) nelle nostre best practice per la risoluzione dei problemi.

Se l'Envoy è stato in grado di stabilire la connessione inizialmente, ma in seguito è stato disconnesso e non si è mai ricollegato, vedi [Envoy disconnesso dal servizio di gestione App Mesh Envoy con](#) testo di errore per risolvere il motivo della disconnessione.

Se l'applicazione si connette ma la richiesta fallisce con un codice di risposta, prova quanto segue:  
HTTP 503

- Assicurati che il servizio virtuale a cui ti stai connettendo esista nella mesh.

- Assicurati che il servizio virtuale abbia un provider (un router virtuale o un nodo virtuale).
- Quando usi Envoy come proxy HTTP, se vedi che il traffico in uscita arriva `cluster.cds_egress*_mesh-allow-all` invece della destinazione corretta tramite le statistiche di Envoy, molto probabilmente Envoy non sta instradando le richieste correttamente. `filter_chains` Ciò può essere dovuto all'utilizzo di un nome di servizio virtuale non qualificato. Si consiglia di utilizzare il nome di rilevamento del servizio effettivo come nome del servizio virtuale, poiché il proxy Envoy comunica con altri servizi virtuali tramite i rispettivi nomi.

[Per ulteriori informazioni, consulta Servizi virtuali.](#)

- Ispeziona i log del proxy Envoy per verificare la presenza di uno dei seguenti messaggi di errore:
  - `No healthy upstream`— Il nodo virtuale verso cui il proxy Envoy sta tentando di indirizzare non ha endpoint risolti o non ha endpoint integri. Assicurati che il nodo virtuale di destinazione abbia le impostazioni corrette per il rilevamento del servizio e il controllo dello stato di salute.

Se le richieste al servizio non vanno a buon fine durante l'implementazione o il ridimensionamento del servizio virtuale di backend, segui le istruzioni riportate in [Alcune richieste hanno esito negativo con il codice di stato HTTP 503 quando un servizio virtuale ha un provider di nodi virtuali](#)

- `No cluster match for URL`— Ciò è probabilmente causato dall'invio di una richiesta a un servizio virtuale che non corrisponde ai criteri definiti da nessuna delle rotte definite da un provider di router virtuale. Assicurati che le richieste dall'applicazione vengano inviate a un percorso supportato assicurandoti che il percorso e le intestazioni delle richieste HTTP siano corretti.
- `No matching filter chain found`— Ciò è probabilmente causato dall'invio di una richiesta a un servizio virtuale su una porta non valida. Assicurati che le richieste provenienti dall'applicazione utilizzino la stessa porta specificata sul router virtuale.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Impossibile connettersi a un servizio esterno

### Caratteristiche

L'applicazione non è in grado di connettersi a un servizio esterno alla mesh, ad esempio `amazon.com`.

### Risoluzione

Per impostazione predefinita, App Mesh non consente il traffico in uscita dalle applicazioni all'interno della mesh verso alcuna destinazione esterna alla mesh. Per abilitare la comunicazione con un servizio esterno, sono disponibili due opzioni:

- Imposta il [filtro in uscita](#) sulla risorsa mesh su. `ALLOW_ALL`. Questa impostazione consentirà a qualsiasi applicazione all'interno della mesh di comunicare con qualsiasi indirizzo IP di destinazione all'interno o all'esterno della mesh.
- Modella il servizio esterno nella mesh utilizzando un servizio virtuale, un router virtuale, una route e un nodo virtuale. Ad esempio, per modellare il servizio esterno `example.com`, è possibile creare un servizio virtuale denominato `example.com` con un router e un routing virtuali che invii tutto il traffico a un nodo virtuale con un nome host di rilevamento del servizio DNS di `example.com`.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Impossibile connettersi a un server MySQL o SMTP

### Caratteristiche

Quando si consente il traffico in uscita verso tutte le destinazioni (`Mesh EgressFilter type =ALLOW_ALL`), ad esempio un server SMTP o un database MySQL utilizzando una definizione di nodo virtuale, la connessione dall'applicazione non riesce. Ad esempio, il seguente è un messaggio di errore generato dal tentativo di connessione a un server MySQL.

```
ERROR 2013 (HY000): Lost connection to MySQL server at 'reading initial communication packet', system error: 0
```

### Risoluzione

Si tratta di un problema noto che viene risolto utilizzando l'immagine App Mesh versione 1.15.0 o successiva. Per ulteriori informazioni, consulta il [problema Impossibile connettersi a MySQL con App Mesh](#). GitHub Questo errore si verifica perché il listener in uscita in Envoy configurato da App Mesh aggiunge il filtro listener Envoy TLS Inspector. Per ulteriori informazioni, consulta [TLS Inspector](#) nella documentazione di Envoy. Questo filtro valuta se una connessione utilizza o meno TLS ispezionando il primo pacchetto inviato dal client. Con MySQL e SMTP, tuttavia, il server invia il primo pacchetto dopo la connessione. Per ulteriori informazioni su MySQL, [consulta Initial Handshake](#) nella documentazione di MySQL. Poiché il server invia il primo pacchetto, l'ispezione del filtro fallisce.

Per risolvere questo problema a seconda della versione di Envoy in uso:

- Se la versione Envoy dell'immagine App Mesh è 1.15.0 o successiva, non modellare servizi esterni come MySQL, SMTP, MSSQL, ecc. come backend per il nodo virtuale dell'applicazione.
- Se la versione Envoy dell'immagine App Mesh è precedente alla 1.15.0, aggiungi port **3306** all'elenco di valori per i **APPMESH\_EGRESS\_IGNORED\_PORTS** tuoi servizi per MySQL e come porta che stai utilizzando per SMTP.

#### Important

Sebbene le porte SMTP standard siano 25, e 587465, dovresti aggiungere solo la porta che stai utilizzando e non tutte e tre. **APPMESH\_EGRESS\_IGNORED\_PORTS**

Per ulteriori informazioni, consulta [Update services](#) for Kubernetes, [Update services for](#) Amazon ECS o [Update services for Amazon](#) EC2.

Se il problema persiste, puoi fornirci i dettagli su cosa stai riscontrando utilizzando il [GitHub problema](#) esistente o contattare l'[AWS assistenza](#).

## Impossibile connettersi a un servizio modellato come nodo virtuale TCP o router virtuale in App Mesh

### Caratteristiche

L'applicazione non è in grado di connettersi a un backend che utilizza l'impostazione del protocollo TCP nella definizione App Mesh [PortMapping](#).

### Risoluzione

Si tratta di un problema noto. Per ulteriori informazioni, consulta [Routing verso più destinazioni TCP sulla stessa porta su](#). GitHub App Mesh attualmente non consente a più destinazioni di backend modellate come TCP di condividere la stessa porta a causa delle restrizioni nelle informazioni fornite al proxy Envoy su OSI Layer 4. Per assicurarti che il traffico TCP possa essere instradato in modo appropriato per tutte le destinazioni di backend, procedi come segue:

- Assicurati che tutte le destinazioni utilizzino una porta unica. Se si utilizza un provider di router virtuale per il servizio virtuale di backend, è possibile modificare la porta del router virtuale senza

modificare la porta sui nodi virtuali verso cui viene indirizzata. Ciò consente alle applicazioni di aprire connessioni sulla porta del router virtuale mentre il proxy Envoy continua a utilizzare la porta definita nel nodo virtuale.

- Se la destinazione modellata come TCP è un server MySQL o qualsiasi altro protocollo basato su TCP in cui il server invia i primi pacchetti dopo la connessione, vedere. [Impossibile connettersi a un server MySQL o SMTP](#)

Se il problema persiste, puoi fornirci i dettagli su cosa stai riscontrando utilizzando il [GitHub problema](#) esistente o contattare l'[AWS assistenza](#).

## La connettività riesce al servizio non elencato come backend di servizio virtuale per un nodo virtuale

### Caratteristiche

L'applicazione è in grado di connettersi e inviare traffico verso una destinazione che non è specificata come backend di servizio virtuale sul nodo virtuale.

### Risoluzione

Se le richieste hanno esito positivo verso una destinazione che non è stata modellata nelle API App Mesh, la causa più probabile è che il tipo di [filtro in uscita](#) della mesh è stato impostato su. ALLOW\_ALL. Quando il filtro in uscita è impostato su ALLOW\_ALL, una richiesta in uscita dall'applicazione che non corrisponde a una destinazione modellata (backend) verrà inviata all'indirizzo IP di destinazione impostato dall'applicazione.

Se desideri impedire il traffico verso destinazioni non modellate nella mesh, valuta la possibilità di impostare il valore del filtro in uscita su. DROP\_ALL.

#### Note

L'impostazione del valore del filtro in uscita della mesh influisce su tutti i nodi virtuali all'interno della mesh.

La configurazione `egress_filter` come DROP\_ALL e l'abilitazione di TLS non sono disponibili per il traffico in uscita che non è diretto a un dominio. AWS

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Alcune richieste hanno esito negativo con il codice di stato HTTP **503** quando un servizio virtuale ha un provider di nodi virtuali

### Caratteristiche

Una parte delle richieste dell'applicazione non viene inviata a un backend di servizi virtuali che utilizza un provider di nodi virtuali anziché un provider di router virtuale. Quando si utilizza un provider di router virtuale per il servizio virtuale, le richieste non hanno esito negativo.

### Risoluzione

Si tratta di un problema noto. Per ulteriori informazioni, consulta la [politica Riprova sul provider Virtual Node for a Virtual Service](#) on GitHub. Quando si utilizza un nodo virtuale come provider per un servizio virtuale, non è possibile specificare la politica di ripetizione dei tentativi predefinita che si desidera che i client del servizio virtuale utilizzino. In confronto, i provider di router virtuali consentono di specificare le politiche di riprova perché sono una proprietà delle risorse del percorso secondario.

Per ridurre gli errori di richiesta ai provider di nodi virtuali, utilizza invece un provider di router virtuale e specifica una politica di riprova sulle relative rotte. Per altri modi per ridurre gli errori nelle richieste delle applicazioni, consulta [Best practice per App Mesh](#)

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Impossibile connettersi a un file system Amazon EFS

### Caratteristiche

Quando si configura un'attività Amazon ECS con un file system Amazon EFS come volume, l'attività non inizia con il seguente errore.

```
ResourceInitializationError: failed to invoke EFS utils commands to set up EFS volumes:
 stderr: mount.nfs4: Connection refused : unsuccessful EFS utils command execution;
 code: 32
```

### Risoluzione

Si tratta di un problema noto. Questo errore si verifica perché la connessione NFS ad Amazon EFS si verifica prima dell'avvio dei container inclusi nell'attività. Questo traffico viene indirizzato dalla configurazione del proxy a Envoy, che a questo punto non sarà in esecuzione. A causa dell'ordine di avvio, il client NFS non riesce a connettersi al file system Amazon EFS e

l'attività non viene avviata. Per risolvere il problema, aggiungi port 2049 all'elenco di valori per l'EgressIgnoredPorts impostazione nella configurazione proxy della definizione dell'attività Amazon ECS. Per ulteriori informazioni, consulta [Configurazione del proxy](#).

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## La connettività riesce correttamente al servizio, ma la richiesta in arrivo non viene visualizzata nei log di accesso, nelle tracce o nelle metriche di accesso di Envoy

### Caratteristiche

Anche se l'applicazione è in grado di connettersi e inviare richieste a un'altra applicazione, non è possibile visualizzare le richieste in arrivo nei registri di accesso o nelle informazioni di tracciamento del proxy Envoy.

### Risoluzione

Si tratta di un problema noto. Per ulteriori informazioni, consulta il problema di configurazione delle regole di [iptables](#) su Github. Il proxy Envoy intercetta solo il traffico in entrata verso la porta su cui è in ascolto il nodo virtuale corrispondente. Le richieste a qualsiasi altra porta bypasseranno il proxy Envoy e raggiungeranno direttamente il servizio dietro di esso. Per consentire al proxy Envoy di intercettare il traffico in entrata del servizio, è necessario impostare il nodo virtuale e il servizio in modo che vengano ascoltati sulla stessa porta.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## L'impostazione delle variabili di **HTTPS\_PROXY** ambiente **HTTP\_PROXY**/a livello di contenitore non funziona come previsto.

### Caratteristiche

Quando **HTTP\_PROXY/HTTPS\_PROXY** è impostato come variabile di ambiente in:

- Contenitore di app nella definizione dell'attività con App Mesh abilitato, le richieste inviate allo spazio dei nomi dei servizi App Mesh riceveranno risposte di HTTP 500 errore dal sidecar Envoy.
- Contenitore Envoy nella definizione delle attività con App Mesh abilitato, le richieste provenienti dal sidecar Envoy non passeranno attraverso il server HTTPS proxy HTTP/e la variabile di ambiente non funzionerà.

## Risoluzione

Per il contenitore dell'app:

App Mesh funziona facendo sì che il traffico all'interno dell'attività passi attraverso il proxy Envoy. HTTP\_PROXY/HTTPS\_PROXY configuration sovrascrive questo comportamento configurando il traffico del contenitore in modo che passi attraverso un proxy esterno diverso. Il traffico verrà comunque intercettato da Envoy, ma non supporta l'inoltro del traffico mesh tramite proxy esterno.

Se desideri utilizzare come proxy tutto il traffico non mesh, imposta NO\_PROXY in modo da includere il CIDR/namespace della tua mesh, il localhost e gli endpoint della credenziale, come nell'esempio seguente.

```
NO_PROXY=localhost,127.0.0.1,169.254.169.254,169.254.170.2,10.0.0.0/16
```

Per il contenitore Envoy:

Envoy non supporta un proxy generico. Non è consigliabile impostare queste variabili.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## I timeout delle richieste upstream vengono interrotti anche dopo aver impostato il timeout per le rotte.

### Caratteristiche

Hai definito il timeout per:

- I percorsi, ma continui a ricevere un errore di timeout della richiesta upstream.
- Il listener del nodo virtuale e il timeout e il timeout dei tentativi per le rotte, ma si verifica comunque un errore di timeout della richiesta upstream.

### Risoluzione

Affinché le richieste ad alta latenza superiori a 15 secondi vengano completate correttamente, è necessario specificare un timeout sia a livello di routing che a livello di listener del nodo virtuale.

Se specificate un timeout del percorso superiore ai 15 secondi predefiniti, assicuratevi che il timeout sia specificato anche per il listener per tutti i nodi virtuali partecipanti. Tuttavia, se riduci il timeout a

un valore inferiore a quello predefinito, è facoltativo aggiornare i timeout nei nodi virtuali. [Per ulteriori informazioni sulle opzioni per la configurazione di nodi e percorsi virtuali, consulta nodi e percorsi virtuali.](#)

Se hai specificato una politica per i nuovi tentativi, la durata specificata per il timeout della richiesta deve essere sempre maggiore o uguale al timeout dei tentativi moltiplicato per il numero massimo di tentativi definito nella politica di nuovi tentativi. Ciò consente di completare correttamente la richiesta con tutti i tentativi. Per ulteriori informazioni, consulta [percorsi](#).

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Envoy risponde con una richiesta HTTP Bad.

### Caratteristiche

Envoy risponde con HTTP 400 Bad request per tutte le richieste inviate tramite Network Load Balancer (NLB). Quando controlliamo i log di Envoy, vediamo:

- Registri di debug:

```
dispatch error: http/1.1 protocol error: HPE_INVALID_METHOD
```

- Registri di accesso:

```
"- - HTTP/1.1" 400 DPE 0 11 0 - "-" "-" "-" "-" "
```

### Risoluzione

[La soluzione è disabilitare la versione 2 del protocollo proxy \(PPv2\) sugli attributi del gruppo target del vostro NLB.](#)

Ad oggi il PPv2 non è supportato dal gateway virtuale e dal nodo virtuale Envoy che vengono eseguiti utilizzando il piano di controllo App Mesh. Se distribuisce NLB utilizzando il controller di AWS bilanciamento del carico su Kubernetes, disabilita PPv2 impostando il seguente attributo su: `false`

```
service.beta.kubernetes.io/aws-load-balancer-target-group-attributes:
proxy_protocol_v2.enabled
```

Vedi [AWS Load Balancer Controller Annotations](#) per maggiori dettagli sugli attributi delle risorse NLB.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Impossibile configurare correttamente il timeout.

### Caratteristiche

Il timeout della richiesta scade entro 15 secondi anche dopo aver configurato il timeout sul listener del nodo virtuale e il timeout sul percorso verso il backend del nodo virtuale.

### Risoluzione

Assicurati che il servizio virtuale corretto sia incluso nell'elenco dei backend.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Risoluzione dei problemi di scalabilità dell'App Mesh

Questo argomento descrive i problemi più comuni che potrebbero verificarsi con il ridimensionamento di App Mesh.

## La connettività fallisce e i controlli di integrità dei container falliscono quando si scalano oltre 50 repliche per un nodo virtuale/gateway virtuale

### Caratteristiche

Quando si ridimensiona il numero di repliche, ad esempio attività Amazon ECS, pod Kubernetes o istanze Amazon EC2, per un nodo virtuale/gateway virtuale oltre 50, i controlli dello stato dei container Envoy per Envoy nuovi e attualmente in esecuzione iniziano a fallire. Le applicazioni downstream che inviano traffico al nodo virtuale/gateway virtuale iniziano a registrare errori nelle richieste con codice di stato HTTP. 503

### Risoluzione

La quota predefinita di App Mesh per il numero di Envoy per nodo virtuale/gateway virtuale è 50. Quando il numero di Envoy in esecuzione supera questa quota, gli Envoy nuovi e attualmente in esecuzione non riescono a connettersi al servizio di gestione Envoy di App Mesh con il codice di stato gRPC (`RESOURCE_EXHAUSTED`). Questa quota può essere aumentata. Per ulteriori informazioni, consulta [Quotazioni di App Mesh Controller](#).

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Le richieste hanno esito negativo **503** quando il backend di un servizio virtuale viene scalato orizzontalmente o verticalmente

### Caratteristiche

Quando un servizio virtuale di backend viene scalato orizzontalmente o internamente, le richieste provenienti dalle applicazioni downstream hanno esito negativo e viene assegnato un codice di stato. HTTP 503

### Risoluzione

App Mesh consiglia diversi approcci per mitigare i casi di errore scalando le applicazioni orizzontalmente. Per informazioni dettagliate su come prevenire questi errori, consulta. [Best practice per App Mesh](#)

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Il contenitore Envoy si blocca con segfault in caso di carico aumentato

### Caratteristiche

In caso di carico di traffico elevato, il proxy Envoy si blocca a causa di un errore di segmentazione (codice di uscita di Linux). 139 I log dei processi di Envoy contengono un'istruzione come la seguente.

```
Caught Segmentation fault, suspect faulting address 0x0"
```

### Risoluzione

Il proxy Envoy ha probabilmente violato il nofile ulimit predefinito del sistema operativo, il limite al numero di file che un processo può avere aperti alla volta. Questa violazione è dovuta al traffico che causa un maggior numero di connessioni, che consumano socket aggiuntivi del sistema operativo. Per risolvere questo problema, aumentate il valore ulimit nofile sul sistema operativo host. Se utilizzi Amazon ECS, questo limite può essere modificato tramite le impostazioni [Ulimit nelle impostazioni](#) dei limiti delle [risorse](#) della definizione dell'attività.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## L'aumento delle risorse predefinite non si riflette nei limiti del servizio

### Caratteristiche

Dopo aver aumentato il limite predefinito delle risorse App Mesh, il nuovo valore non si riflette quando si esaminano i limiti del servizio.

### Risoluzione

Sebbene i nuovi limiti non siano attualmente indicati, i clienti possono comunque esercitarli.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## L'applicazione si arresta in modo anomalo a causa di un numero enorme di chiamate per i controlli sanitari.

### Caratteristiche

Dopo aver abilitato i controlli sanitari attivi per un nodo virtuale, si verifica un aumento del numero di chiamate ai controlli sanitari. L'applicazione si arresta in modo anomalo a causa del notevole aumento del volume di chiamate per il controllo dello stato di salute effettuate all'applicazione.

### Risoluzione

Quando il controllo attivo dello stato è abilitato, ogni endpoint Envoy del downstream (client) invia richieste di integrità a ciascun endpoint del cluster upstream (server) per prendere decisioni di routing. Di conseguenza, il numero totale di richieste di controllo dello stato sarebbe `* *. number of client Envoys number of server Envoys active health check frequency`

Per risolvere questo problema, modificate la frequenza della sonda per i controlli sanitari, in modo da ridurre il volume totale delle sonde per i controlli sanitari. Oltre ai controlli sanitari attivi, App Mesh consente di configurare il [rilevamento dei valori anomali](#) come mezzo di controllo passivo dello stato. Utilizza il rilevamento dei valori anomali per configurare quando rimuovere un determinato host in base a risposte consecutive. `5xx`

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Risoluzione dei problemi di osservabilità dell'App Mesh

Questo argomento descrive i problemi più comuni che potresti riscontrare con l'osservabilità di App Mesh.

# Impossibile visualizzare le AWS X-Ray tracce delle mie applicazioni

## Caratteristiche

L'applicazione in App Mesh non visualizza le informazioni di tracciamento a raggi X nella console X-Ray o nelle API.

## Risoluzione

Per utilizzare X-Ray in App Mesh, è necessario configurare correttamente i componenti per abilitare la comunicazione tra l'applicazione, i contenitori sidecar e il servizio X-Ray. Effettua le seguenti operazioni per confermare che X-Ray sia stato impostato correttamente:

- Assicurati che il protocollo listener App Mesh Virtual Node non sia impostato come TCP.
- Assicurati che il contenitore X-Ray distribuito con l'applicazione esponga la porta 2000 UDP e funzioni come utente. 1337 Per ulteriori informazioni, consulta l'esempio di [Amazon ECS X-Ray](#) su GitHub
- Assicurati che il contenitore Envoy abbia il tracciamento abilitato. Se si utilizza [l'immagine App Mesh Envoy](#), è possibile abilitare X-Ray impostando la variabile di ambiente `ENABLE_ENVOY_XRAY_TRACING` su un valore di 1 e la `XRAY_DAEMON_PORT` variabile di ambiente su `2000`
- Se hai inserito X-Ray nel codice dell'applicazione con uno degli SDK [specifici della lingua](#), assicurati che sia configurato correttamente seguendo le guide per la tua lingua.
- [Se tutti gli elementi precedenti sono configurati correttamente, esaminate i registri del contenitore X-Ray per individuare eventuali errori e seguire le istruzioni riportate in Risoluzione dei problemi.](#) [AWS X-Ray](#) Una spiegazione più dettagliata dell'integrazione di X-Ray in App Mesh è disponibile in [Integrating X-Ray with](#) App Mesh.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

# Impossibile visualizzare i parametri di Envoy per le mie applicazioni nei parametri di Amazon CloudWatch

## Caratteristiche

La tua applicazione in App Mesh non emette metriche generate dal proxy Envoy nelle metriche CloudWatch

## Risoluzione

Quando utilizzi le CloudWatch metriche in App Mesh, devi configurare correttamente diversi componenti per abilitare la comunicazione tra il proxy Envoy, il sidecar CloudWatch dell'agente e il servizio di metrica. CloudWatch Segui i passaggi seguenti per confermare che le CloudWatch metriche per il proxy Envoy siano state configurate correttamente:

- Assicurati di utilizzare l'immagine dell' CloudWatch agente per App Mesh. Per ulteriori informazioni, consulta [App Mesh CloudWatch agent](#) on GitHub.
- Assicurati di aver configurato l' CloudWatch agente per App Mesh in modo appropriato seguendo le istruzioni d'uso specifiche della piattaforma. Per ulteriori informazioni, consulta [App Mesh CloudWatch agent](#) on GitHub.
- Se tutti gli elementi precedenti sono configurati correttamente, esamina i registri del contenitore dell' CloudWatch agente per individuare eventuali errori e segui le indicazioni fornite in [Risoluzione dei problemi dell' CloudWatch agente](#).

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Impossibile configurare regole di campionamento personalizzate per le tracce AWS X-Ray

### Caratteristiche

L'applicazione utilizza il tracciamento a raggi X, ma non è possibile configurare le regole di campionamento per le tracce.

### Risoluzione

Poiché App Mesh Envoy attualmente non supporta la configurazione del campionamento a raggi X dinamici, sono disponibili le seguenti soluzioni alternative.

Se la tua versione di Envoy è 1.19.1 o successiva, hai le seguenti opzioni.

- Per impostare solo la frequenza di campionamento, usa la variabile di `XRAY_SAMPLING_RATE` ambiente sul contenitore Envoy. Il valore deve essere specificato come decimale tra 0 e (100%). `1.00` Per ulteriori informazioni, consulta [AWS X-Ray variabili](#).
- Per configurare le regole di campionamento personalizzate localizzate per il tracciante X-Ray, utilizzate la variabile di `XRAY_SAMPLING_RULE_MANIFEST` ambiente per specificare un

percorso del file nel file system contenitore Envoy. Per ulteriori informazioni, consulta le regole di [campionamento](#) nella Guida per gli sviluppatori.AWS X-Ray

Se la tua versione di Envoy è precedente alla 1.19.1, procedi come segue.

- Utilizzate la variabile di `ENVOY_TRACING_CFG_FILE` ambiente per modificare la frequenza di campionamento. Per ulteriori informazioni, consulta [Variabili di configurazione di Envoy](#). Specificate una configurazione di tracciamento personalizzata e definite le regole di campionamento locali. Per ulteriori informazioni, vedere [Envoy X-Ray config](#).

- Esempio di configurazione di tracciamento personalizzata per la variabile di ambiente:

`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
```

- Per i dettagli sulla configurazione per il manifesto delle regole di campionamento nella `samplingRuleManifest` proprietà, vedere [Configurazione dell'X-Ray SDK for Go](#).

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Risoluzione dei problemi di sicurezza di App Mesh

Questo argomento descrive i problemi più comuni che potresti riscontrare con la sicurezza App Mesh.

# Impossibile connettersi a un servizio virtuale di backend con una politica client TLS

## Caratteristiche

Quando si aggiunge una policy client TLS a un backend di servizio virtuale in un nodo virtuale, la connettività a quel backend fallisce. Quando si tenta di inviare traffico al servizio di backend, le richieste falliscono con un codice di HTTP 503 risposta e il messaggio di errore: `upstream connect error or disconnect/reset before headers. reset reason: connection failure`

## Risoluzione

Per determinare la causa principale del problema, ti consigliamo di utilizzare i registri dei processi del proxy Envoy per aiutarti a diagnosticare il problema. Per ulteriori informazioni, consulta [Abilita la registrazione di debug di Envoy negli ambienti di preproduzione](#). Utilizza il seguente elenco per determinare la causa dell'errore di connessione:

- Assicurati che la connettività al backend funzioni correttamente escludendo gli errori menzionati in [Impossibile connettersi a un backend di servizio virtuale](#)
- Nei log dei processi di Envoy, cerca i seguenti errori (registrati a livello di debug).

```
TLS error: 268435581:SSL routines:OPENSSL_internal:CERTIFICATE_VERIFY_FAILED
```

Questo errore è causato da uno o più dei seguenti motivi:

- Il certificato non è stato firmato da una delle autorità di certificazione definite nel TLS Client Policy Trust Bundle.
- Il certificato non è più valido (scaduto).
- Il nome alternativo del soggetto (SAN) non corrisponde al nome host DNS richiesto.
- Assicurati che il certificato offerto dal servizio di backend sia valido, che sia firmato da una delle autorità di certificazione incluse nel tuo TLS Client Policies Trust Bundle e che soddisfi i criteri definiti in [Transport Layer Security \(TLS\)](#)
- Se l'errore che ricevi è simile a quello riportato di seguito, significa che la richiesta sta ignorando il proxy Envoy e sta raggiungendo direttamente l'applicazione. Quando si invia traffico, le statistiche su Envoy non cambiano, il che indica che Envoy non è sulla strada giusta per decrittografare il traffico. Nella configurazione proxy del nodo virtuale, assicurati che `AppPort s` contenga il valore corretto che l'applicazione sta ascoltando.

```
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
```

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

Se ritieni di aver trovato una vulnerabilità di sicurezza o hai domande sulla sicurezza di App Mesh, consulta le linee guida per la [segnalazione delle AWS vulnerabilità](#).

## Impossibile connettersi a un servizio virtuale di backend quando l'applicazione sta originando TLS

### Caratteristiche

Quando si origina una sessione TLS da un'applicazione, anziché dal proxy Envoy, la connettività a un servizio virtuale di backend non funziona.

### Risoluzione

Si tratta di un problema noto. Per ulteriori informazioni, consulta il problema relativo alla [richiesta di funzionalità: negoziazione TLS tra l'applicazione downstream e il proxy upstream](#). GitHub In App Mesh, l'origine TLS è attualmente supportata dal proxy Envoy ma non dall'applicazione. Per utilizzare il supporto di origine TLS su Envoy, disabilita l'origine TLS nell'applicazione. Ciò consente all'Envoy di leggere le intestazioni delle richieste in uscita e inoltrare la richiesta alla destinazione appropriata tramite una sessione TLS. Per ulteriori informazioni, consulta [Transport Layer Security \(TLS\)](#).

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

Se ritieni di aver trovato una vulnerabilità di sicurezza o hai domande sulla sicurezza di App Mesh, consulta le linee guida per la [segnalazione delle AWS vulnerabilità](#).

## Impossibile affermare che la connettività tra i proxy Envoy utilizzi TLS

### Caratteristiche

L'applicazione ha abilitato la terminazione TLS sul nodo virtuale o sul listener del gateway virtuale o l'origine TLS sulla policy del client TLS di backend, ma non è possibile affermare che la connettività tra i proxy Envoy avvenga su una sessione negoziata con TLS.

### Risoluzione

Le fasi definite in questa risoluzione utilizzano l'interfaccia di amministrazione di Envoy e le statistiche di Envoy. Per informazioni sulla configurazione di questi, consulta e. [Abilita l'interfaccia di amministrazione del proxy Envoy](#) [Abilita l'integrazione con Envoy DogStats D per l'offload metrico](#) I seguenti esempi di statistiche utilizzano l'interfaccia di amministrazione per semplicità.

- Per il proxy Envoy che esegue la terminazione TLS:
  - Assicurati che il certificato TLS sia stato avviato nella configurazione di Envoy con il seguente comando.

```
curl http://my-app.default.svc.cluster.local:9901/certs
```

Nell'output restituito, dovresti vedere almeno una voce sotto il certificato utilizzato nella terminazione `certificates[[]].cert_chain` TLS.

- Assicurati che il numero di connessioni in entrata riuscite al listener del proxy sia esattamente uguale al numero di handshake SSL più il numero di sessioni SSL riutilizzate, come mostrato dai comandi e dall'output di esempio seguenti.

```
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)
```

- Per il proxy Envoy che esegue l'origine TLS:
  - Assicurati che il trust store TLS sia stato avviato nella configurazione di Envoy con il seguente comando.

```
curl http://my-app.default.svc.cluster.local:9901/certs
```

Dovresti vedere almeno una voce sotto `certificates[[]].ca_certs` per i certificati utilizzati per convalidare il certificato del backend durante l'origine del TLS.

- Assicurati che il numero di connessioni in uscita riuscite al cluster di backend sia esattamente lo stesso del numero di handshake SSL più il numero di sessioni SSL riutilizzate, come mostrato dai seguenti comandi e output di esempio.

```
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)
```

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

Se ritieni di aver trovato una vulnerabilità di sicurezza o hai domande sulla sicurezza di App Mesh, consulta le linee guida per la [segnalazione delle AWS vulnerabilità](#).

## Risoluzione dei problemi relativi al TLS con Elastic Load Balancing

### Caratteristiche

Quando si tenta di configurare un Application Load Balancer o un Network Load Balancer per crittografare il traffico verso un nodo virtuale, i controlli di connettività e di integrità del load balancer possono fallire.

### Risoluzione

Per determinare la causa principale del problema, è necessario verificare quanto segue:

- Affinché il proxy Envoy esegua la terminazione TLS, è necessario escludere eventuali errori di configurazione. Segui i passaggi indicati sopra in. [Impossibile connettersi a un servizio virtuale di backend con una politica client TLS](#)
- Per il bilanciamento del carico, è necessario esaminare la configurazione di TargetGroup :
  - Assicurati che la TargetGroup porta corrisponda alla porta listener definita dal nodo virtuale.
  - Per gli Application Load Balancer che originano connessioni TLS tramite HTTP al tuo servizio, assicurati che il TargetGroup protocollo sia impostato su. HTTPS Se vengono utilizzati i controlli sanitari, assicurati che sia impostato su. HealthCheckProtocol HTTPS
  - Per i Network Load Balancer che originano connessioni TLS tramite TCP al tuo servizio, assicurati che il protocollo sia impostato su. TargetGroup TLS Se vengono utilizzati controlli sanitari, assicurati che sia impostato su. HealthCheckProtocol TCP

#### Note

Eventuali aggiornamenti che TargetGroup richiedono la modifica del TargetGroup nome.

Se configurato correttamente, il sistema di bilanciamento del carico dovrebbe fornire una connessione sicura al servizio utilizzando il certificato fornito al proxy Envoy.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

Se ritieni di aver trovato una vulnerabilità di sicurezza o hai domande sulla sicurezza di App Mesh, consulta le linee guida per la [segnalazione delle AWS vulnerabilità](#).

## Risoluzione dei problemi di App Mesh Kubernetes

Questo argomento descrive i problemi più comuni che potresti riscontrare quando utilizzi App Mesh con Kubernetes.

### Le risorse App Mesh create in Kubernetes non possono essere trovate in App Mesh

#### Caratteristiche

Hai creato le risorse App Mesh utilizzando la definizione di risorsa personalizzata (CRD) di Kubernetes, ma le risorse che hai creato non sono visibili in App Mesh quando utilizzi le API or. AWS Management Console

## Risoluzione

La causa probabile è un errore nel controller Kubernetes per App Mesh. [Per ulteriori informazioni, consulta Risoluzione dei problemi su](#). GitHub Controlla i log del controller per eventuali errori o avvisi che indicano che il controller non è riuscito a creare alcuna risorsa.

```
kubectl logs -n appmesh-system -f \
$(kubectl get pods -n appmesh-system -o name | grep controller)
```

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Dopo l'iniezione del sidecar Envoy, i pod non sono stati sottoposti ai controlli di prontezza e vivacità

### Caratteristiche

I pod della tua applicazione funzionavano correttamente in precedenza, ma dopo l'iniezione del sidecar Envoy in un pod, i controlli di prontezza e vivacità iniziano a fallire.

## Risoluzione

Assicurati che il contenitore Envoy che è stato iniettato nel pod sia stato avviato con il servizio di gestione Envoy di App Mesh. Puoi verificare eventuali errori facendo riferimento ai codici di errore in. [Envoy disconnesso dal servizio di gestione App Mesh Envoy con testo di errore](#) È possibile utilizzare il seguente comando per ispezionare i log di Envoy per il relativo pod.

```
kubectl logs -n appmesh-system -f \
$(kubectl get pods -n appmesh-system -o name | grep controller) \
| grep "gRPC config stream closed"
```

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## I pod non vengono registrati o annullati come istanze AWS Cloud Map

### Caratteristiche

I tuoi pod Kubernetes non vengono registrati o cancellati durante il loro ciclo di vita. AWS Cloud Map Un pod può avviarsi correttamente ed essere pronto a servire il traffico, ma non a riceverne. Quando un pod viene terminato, i client possono comunque conservare il relativo indirizzo IP e tentare di inviargli traffico, ma senza successo.

## Risoluzione

Si tratta di un problema noto. Per ulteriori informazioni, consulta i [Pod non si registrano automaticamente o annullano la registrazione in Kubernetes con problemi. AWS Cloud Map GitHub](#) A causa della relazione tra pod, nodi virtuali App Mesh e AWS Cloud Map risorse, il [controller App Mesh per Kubernetes](#) potrebbe desincronizzarsi e perdere risorse. Ad esempio, ciò può accadere se una risorsa di nodo virtuale viene eliminata da Kubernetes prima di terminare i pod associati.

Per mitigare questo problema:

- Assicurati di utilizzare la versione più recente del controller App Mesh per Kubernetes.
- Assicurati che i AWS Cloud Map namespaceName e serviceName siano corretti nella definizione del nodo virtuale.
- Assicurati di eliminare tutti i pod associati prima di eliminare la definizione del nodo virtuale. Se hai bisogno di aiuto per identificare quali pod sono associati a un nodo virtuale, consulta [Impossibile determinare dove è in esecuzione un pod per una risorsa App Mesh](#)
- Se il problema persiste, esegui il comando seguente per ispezionare i log del controller alla ricerca di errori che potrebbero aiutare a rivelare il problema sottostante.

```
kubectl logs -n appmesh-system \
$(kubectl get pods -n appmesh-system -o name | grep appmesh-controller)
```

- Valuta la possibilità di utilizzare il seguente comando per riavviare i pod del controller. Questo può risolvere i problemi di sincronizzazione.

```
kubectl delete -n appmesh-system \
$(kubectl get pods -n appmesh-system -o name | grep appmesh-controller)
```

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Impossibile determinare dove è in esecuzione un pod per una risorsa App Mesh

### Caratteristiche

Quando esegui App Mesh su un cluster Kubernetes, un operatore non può determinare dove è in esecuzione un carico di lavoro, o pod, per una determinata risorsa App Mesh.

### Risoluzione

Le risorse del pod Kubernetes sono annotate con la mesh e il nodo virtuale a cui sono associate. Puoi interrogare quali pod sono in esecuzione per un determinato nome di nodo virtuale con il seguente comando.

```
kubectl get pods --all-namespaces -o json | \
 jq '.items[] | { metadata } | select(.metadata.annotations."appmesh.k8s.aws/virtualNode" == "virtual-node-name")'
```

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Impossibile determinare su quale risorsa App Mesh sia in esecuzione un pod

### Caratteristiche

Quando si esegue App Mesh su un cluster Kubernetes, un operatore non può determinare su quale risorsa App Mesh è in esecuzione un determinato pod.

### Risoluzione

Le risorse del pod Kubernetes sono annotate con la mesh e il nodo virtuale a cui sono associate. Puoi generare i nomi della mesh e dei nodi virtuali interrogando direttamente il pod utilizzando il seguente 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" }'
```

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

# I Client Envoy non sono in grado di comunicare con App Mesh Envoy Management Service con IMDSv1 disabilitato

## Caratteristiche

Quando IMDSv1 è disabilitato, gli Envoy client non sono in grado di comunicare con il piano di controllo App Mesh (Envoy Management Service). IMDSv2 il supporto non era disponibile nella versione precedente di App Mesh Envoy. `v1.24.0.0-prod`

## Risoluzione

Per risolvere questo problema, puoi fare una di queste tre cose.

- Esegui l'aggiornamento alla versione App Mesh Envoy `v1.24.0.0-prod` o successiva, che IMDSv2 supporta.
- Riattiva IMDSv1 sull'istanza in cui è in esecuzione Envoy. Per istruzioni sul ripristino IMDSv1, consulta [Configurare le opzioni dei metadati dell'istanza](#).
- Se i tuoi servizi sono in esecuzione su Amazon EKS, ti consigliamo di utilizzare i ruoli IAM per gli account di servizio (IRSA) per recuperare le credenziali. Per istruzioni su come abilitare IRSA, consulta i [ruoli IAM](#) per gli account di servizio.

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

# IRSA non funziona sul contenitore dell'applicazione quando App Mesh è abilitato e Envoy viene iniettato

## Caratteristiche

Quando App Mesh è abilitato su un cluster Amazon EKS con l'aiuto del controller App Mesh per Amazon EKS, Envoy e `proxyinit` i contenitori vengono iniettati nel pod dell'applicazione.

L'applicazione non è in grado di assumere IRSA e invece presuppone il `node role`

Quando descriviamo i dettagli del pod, vediamo che la variabile di ambiente `AWS_ROLE_ARN` ambiente `AWS_WEB_IDENTITY_TOKEN_FILE` o la variabile di ambiente non sono incluse nel contenitore dell'applicazione.

## Risoluzione

Se una delle due `AWS_WEB_IDENTITY_TOKEN_FILE` variabili di `AWS_ROLE_ARN` ambiente è definita, il webhook salterà il pod. Non fornite nessuna di queste variabili e il webhook si occuperà di iniettarle per voi.

```
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
 }
}
```

[Se il problema persiste, valuta la possibilità di aprirne uno GitHub o di contattare l'AWS assistenza.](#)

## Canale di anteprima App Mesh

L'App Mesh Preview Channel è una variante distinta del servizio App Mesh fornito nella us-west-2 regione. Il canale di anteprima espone le funzionalità imminenti da provare man mano che vengono sviluppate. Mentre utilizzi le funzionalità nel canale di anteprima, puoi fornire feedback GitHub per definire il comportamento delle funzionalità. Una volta che una funzionalità avrà una funzionalità completa nel canale di anteprima, con tutte le integrazioni e i controlli necessari, passerà al servizio App Mesh di produzione.

Il canale di AWS App Mesh anteprima è un servizio beta e tutte le funzionalità sono anteprime, in quanto tali termini sono definiti nei [Termini del AWS servizio](#). La partecipazione dell'utente al canale di anteprima è regolata dal Contratto AWS e dai Termini del AWS servizio, in particolare dai termini di partecipazione al servizio universale e beta, ed è riservata.

Le seguenti domande vengono poste frequentemente sul canale di anteprima.

### Cos'è il canale di anteprima?

Il canale di anteprima è un endpoint di servizio pubblico che consente di provare e fornire feedback sulle nuove funzionalità del servizio prima che siano generalmente disponibili. L'endpoint di servizio per il canale di anteprima è separato dall'endpoint di produzione standard. Si interagisce con l'endpoint utilizzando il AWS CLI file del modello di servizio per il canale di anteprima e i file di input dei comandi per il AWS CLI. Il canale di anteprima ti consente di provare nuove funzionalità senza influire sulla tua attuale infrastruttura di produzione. Sei incoraggiato a [fornire un feedback](#) al team di App Mesh per garantire che App Mesh soddisfi i requisiti più importanti dei clienti. Il tuo feedback sulle funzionalità mentre sono nel canale di anteprima può contribuire a plasmare le funzionalità di App Mesh in modo da poter offrire il miglior servizio possibile.

### In che modo il canale di anteprima è diverso dall'App Mesh di produzione?

La tabella seguente elenca gli aspetti del servizio App Mesh diversi dal canale di anteprima.

| Aspetto | Servizio di produzione App Mesh | Servizio App Mesh Preview Channel |
|---------|---------------------------------|-----------------------------------|
|---------|---------------------------------|-----------------------------------|

|                                      |                                                          |                                                                                                            |
|--------------------------------------|----------------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| Frontend endpoint                    | appmesh.us-west-2.<br>amazonaws.com                      | appmesh-preview.us-west-2.a<br>mazonaws.com                                                                |
| Envoy management service<br>endpoint | appmesh-envoy-mana<br>gement.us-west-2.a<br>mazonaws.com | appmesh-preview-envoy-<br>management.us-west-2.am<br>azonaws.com                                           |
| CLI                                  | AWS App Mesh list-meshes                                 | AWS App Mesh-preview list-<br>meshes (only available after<br>adding the Preview Channel<br>service model) |
| Signing name                         | appmesh                                                  | appmesh-preview                                                                                            |
| Service principal                    | appmesh.amazonaws.com                                    | appmesh-preview.am<br>azonaws.com                                                                          |

### Note

Sebbene l'esempio nella tabella del servizio di produzione App Mesh elenchi laus-west-2 regione, il servizio di produzione è disponibile nella maggior parte delle regioni. Per un elenco di tutte le regioni in cui è disponibile il servizio di produzione App Mesh, vedi [AWS App MeshEndpoints and Quotas](#). Tuttavia, il servizio App Mesh Preview Channel è disponibile solo nellaus-west-2 regione.

## Come posso usare le funzionalità del canale di anteprima?

1. Aggiungete il modello di servizio Preview Channel che include la funzionalità Preview ChannelAWS CLI con il seguente comando.

```
aws configure add-model \
 --service-name appmesh-preview \
 --service-model https://raw.githubusercontent.com/aws/aws-app-mesh-roadmap/
main/appmesh-preview/service-model.json
```

2. Crea un file JSON che includa la funzionalità, in base all'esempio JSON e alle istruzioni fornite nella [Guida per l'AWS App Meshutente](#) della funzionalità.

3. Implementa la funzionalità con ilAWS CLI comando e il file di input dei comandi appropriati. Ad esempio, il comando seguente crea un percorso con le funzionalità del canale di anteprima utilizzando il file `route.json`.

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

4. Aggiungilo `APPMESH_PREVIEW = 1` come variabile di configurazione per il contenitore Envoy quando lo aggiungi alle definizioni delle attività di Amazon ECS, alle specifiche Kubernetes Pod o alle istanze Amazon EC2. Questa variabile consente al contenitore Envoy di comunicare con gli endpoint del canale di anteprima. Per ulteriori informazioni sull'aggiunta di variabili di configurazione, vedi [Aggiornamento dei servizi in Amazon ECS](#), [Aggiornamento dei servizi in Kubernetes](#) e [Aggiornamento dei servizi su Amazon EC2](#).

## Come si forniscono gli?

Puoi fornire un feedback direttamente sul problema relativo al [GitHub repository della roadmap di App Mesh](#), collegato alla documentazione relativa alla funzionalità.

## Per quanto tempo ho a disposizione per fornire un feedback su una funzionalità del canale di anteprima?

Il periodo di feedback varia a seconda delle dimensioni e della complessità della funzionalità introdotta. Intendiamo concedere un periodo di commento di 14 giorni tra il rilascio di una funzionalità all'endpoint di anteprima e il rilascio della funzionalità alla produzione. Il team di App Mesh può prolungare il periodo di feedback per funzionalità specifiche.

## Qual è il livello di supporto fornito per il canale di anteprima?

Sebbene ti invitiamo a fornire feedback e segnalazioni di bug direttamente sul [problema della GitHub roadmap](#) di App Mesh, comprendiamo che potresti avere dati sensibili da condividere o potresti riscontrare un problema che non ritieni sicuro divulgare pubblicamente. Per questi problemi, puoi contattare AWS Support o fornire un feedback [inviando un'e-mail](#) direttamente al team di App Mesh.

## I miei dati sono al sicuro sull'endpoint del canale di anteprima?

Sì. Al canale di anteprima viene assegnato lo stesso livello di sicurezza dell'endpoint di produzione standard.

## Per quanto tempo sarà disponibile la mia configurazione?

Puoi lavorare con una mesh nel canale di anteprima per trenta giorni. Trenta giorni dopo la creazione di una mesh, puoi solo elencarla, leggerla o eliminarla. Se tenti di creare o aggiornare le risorse dopo trenta giorni, riceverai un'BadRequest eccezione che spiega che la mesh è stata archiviata.

## Quali strumenti posso usare per lavorare con il canale di anteprima?

È possibile utilizzare ilAWS CLI file del modello del servizio Preview Channel e i file di input dei comandi. Per ulteriori informazioni su come utilizzare le funzionalità, consultare [Come posso usare le funzionalità del canale di anteprima?](#). Non è possibile utilizzare le opzioni diAWS CLI comandoAWS Management Console, gli SDK oAWS CloudFormation utilizzare le funzionalità del canale di anteprima. Tuttavia, è possibile utilizzare tutti gli strumenti, una volta rilasciata una funzionalità nel servizio di produzione.

## Ci sarà una compatibilità futura delle API di Preview Channel?

No. Le API possono cambiare in base al feedback.

## Le funzionalità del canale di anteprima sono complete?

No. I nuovi oggetti API potrebbero non essere completamente integrati inAWS Management Console,AWS CloudFormation, oAWS CloudTrail. Man mano che le funzionalità si consolidano nel canale di anteprima e si avvicina alla disponibilità generale, le integrazioni alla fine diventeranno disponibili.

## La documentazione è disponibile per le funzionalità di Preview Channel?

Sì. La documentazione per le funzionalità di Preview Channel è inclusa nella documentazione di produzione. Ad esempio, se le funzionalità per la risorsa del percorso vengono rilasciate nel canale di anteprima, le informazioni su come utilizzare le funzionalità si trovano nel documento esistente sulle risorse del [percorso](#). Le funzionalità del canale di anteprima sono etichettate come disponibili solo nel canale di anteprima.

## Come faccio a sapere quando saranno disponibili nuove funzionalità nel canale di anteprima?

Quando vengono introdotte nuove funzionalità nel canale di anteprima, viene aggiunta una voce alla [cronologia dei documenti dell'App Mesh](#). Puoi rivedere la pagina regolarmente o iscriverti al [feed RSS della cronologia dei documenti di App Mesh](#). Inoltre, puoi esaminare i [problemi](#) del GitHub repository dellaAWS App Mesh roadmap. Un link per il download del file JSON del modello di servizio Preview Channel viene aggiunto a un problema quando viene rilasciato nel canale di anteprima. Per ulteriori informazioni su come utilizzare il modello e la funzionalità, consultare [Come posso usare le funzionalità del canale di anteprima?](#).

## Come faccio a sapere quando una funzionalità è passata al servizio di produzione?

Il testo nella documentazione di App Mesh che indica che la funzionalità è disponibile solo nel canale di anteprima viene rimosso e viene aggiunta una voce alla [cronologia dei documenti dell'App Mesh](#). Puoi rivedere la pagina regolarmente o iscriverti al [feed RSS della cronologia dei documenti di App Mesh](#).

# Quotazioni di App Mesh Controller

AWS App Mesh è integrato con Service Quotas, un AWS servizio che ti consente di visualizzare e gestire le quote da una posizione centrale. Le quote di servizio sono indicate anche come limiti. Per ulteriori informazioni, consultare [Cos'è Service Quotas?](#) nella Guida per l'utente di Service Quotas.

## Service Quotas Controller

Per visualizzare le quote di esh ControllerAWS Management Console

1. Apri la console Service Quotas all'indirizzo <https://console.aws.amazon.com/servicequotas/>
2. Nel pannello di navigazione, scegli Servizi AWS.
3. Dall'elenco di servizi AWS, cerca e seleziona AWS App Mesh.

Nell'elenco Service quotas (Quote di servizio), è possibile visualizzare il nome della quota di servizio, il valore applicato (se è disponibile), la quota predefinita AWS e se il valore della quota è regolabile.

4. Per visualizzare ulteriori informazioni su una quota di servizio, ad esempio la descrizione, scegliere il nome della quota.

Per richiedere un aumento delle quote, consultare [Richiesta di aumento delle quote](#) nella Guida per l'utente di Service Quotas.

Per visualizzare le quote di esh ControllerAWS CLI

Esegui il seguente comando.

```
aws service-quotas list-aws-default-service-quotas \
 --query 'Quotas[*]'.
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
 --service-code appmesh \
 --output table
```

Per maggiori informazioni sulle quote di servizio utilizzando la AWS CLI, consultare [AWS CLI di riferimento sui comandi Service Quotas](#).

## Cronologia dei documenti per App Mesh

La tabella riportata di seguito illustra i principali aggiornamenti e le nuove caratteristiche della Guida per l'utente di AWS App Mesh . Inoltre, aggiorniamo frequentemente la documentazione tenendo conto dei feedback ricevuti.

| Modifica                                                              | Descrizione                                                                                                                                                                         | Data            |
|-----------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| <a href="#">AWSAppMeshFullAccess Politica aggiornata</a>              | Aggiornato <code>AWSAppMeshFullAccess</code> per consentire l'accesso alle <code>UntagResource</code> API <code>TagResource</code> e.                                               | 24 aprile 2024  |
| <a href="#">CloudTrail documentazione di integrazione aggiornata</a>  | La documentazione che descrive l'integrazione di App Mesh con CloudTrail to log dell'attività dell'API è stata aggiornata.                                                          | 28 marzo 2024   |
| <a href="#">Politiche aggiornate</a>                                  | Aggiornato <code>AWSServiceRoleForAppMesh</code> e <code>AWSAppMeshServiceRolePolicy</code> per consentire l'accesso all' <code>AWS Cloud Map DiscoverInstancesRevision</code> API. | 12 ottobre 2023 |
| <a href="#">Supporto delle policy degli endpoint VPC per App Mesh</a> | App Mesh ora supporta le policy degli endpoint VPC.                                                                                                                                 | 11 maggio 2023  |
| <a href="#">Listener multipli per App Mesh</a>                        | App Mesh ora supporta più listener.                                                                                                                                                 | 18 agosto 2022  |
| <a href="#">IPv6 per App Mesh</a>                                     | App Mesh ora supporta IPv6.                                                                                                                                                         | 18 maggio 2022  |

|                                                                                              |                                                                                                                 |                  |
|----------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|------------------|
| <a href="#">CloudTrail supporto alla registrazione per App Mesh Envoy Management Service</a> | App Mesh ora supporta il supporto CloudTrail di registrazione per App Mesh Envoy Management Service.            | 18 marzo 2022    |
| <a href="#">App Mesh Agent per Envoy</a>                                                     | App Mesh ora supporta Agent for Envoy.                                                                          | 25 febbraio 2022 |
| <a href="#">Listener multipli per App Mesh</a>                                               | (Solo <a href="#">App Mesh Preview Channel</a> ) È possibile implementare più listener per App Mesh.            | 23 novembre 2021 |
| <a href="#">Supporto ARM64 per App Mesh</a>                                                  | App Mesh ora supporta ARM64.                                                                                    | 19 novembre 2021 |
| <a href="#">Estensione delle metriche per App Mesh</a>                                       | Puoi implementare estensioni di metriche per App Mesh.                                                          | 29 ottobre 2021  |
| <a href="#">Implementa miglioramenti del traffico in entrata</a>                             | È possibile implementare la corrispondenza tra nome host e header e riscrivere il nome e il percorso dell'host. | 14 giugno 2021   |
| <a href="#">Implementa l'autenticazione TLS reciproca</a>                                    | È possibile implementare l'autenticazione TLS reciproca.                                                        | 4 febbraio 2021  |
| <a href="#">Lancio della regione in af-south-1</a>                                           | App Mesh è ora disponibile nella regione af-south-1.                                                            | 22 gennaio 2021  |
| <a href="#">Implementa l'autenticazione TLS reciproca</a>                                    | (Solo <a href="#">App Mesh Preview Channel</a> ) È possibile implementare l'autenticazione TLS reciproca.       | 23 novembre 2020 |
| <a href="#">Implementa il pool di connessioni per un listener di gateway virtuale</a>        | È possibile implementare il pool di connessioni per un listener di gateway virtuale.                            | 5 novembre 2020  |

|                                                                                                                        |                                                                                                                                                                        |                   |
|------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| <a href="#">Implementa il pool di connessioni e il rilevamento degli outlier per un listener di nodi virtuali</a>      | È possibile implementare il pool di connessioni e il rilevamento dei valori anomali per un listener di nodi virtuali.                                                  | 5 novembre 2020   |
| <a href="#">Lancio della regione in eu-south-1</a>                                                                     | App Mesh è ora disponibile nella regione eu-south-1.                                                                                                                   | 21 ottobre 2020   |
| <a href="#">Implementa il pooling delle connessioni per un listener di gateway virtuale</a>                            | (Solo <a href="#">App Mesh Preview Channel</a> ) È possibile implementare il pool di connessioni per un listener gateway virtuale.                                     | 28 settembre 2020 |
| <a href="#">Implementa il pool di connessioni e il rilevamento dei valori anomali per un listener di nodi virtuali</a> | (Solo <a href="#">App Mesh Preview Channel</a> ) È possibile implementare il pool di connessioni e il rilevamento dei valori anomali per un listener di nodi virtuali. | 28 settembre 2020 |
| <a href="#">Crea un gateway virtuale e un gateway route per la rete mesh in entrata</a>                                | Abilita le risorse esterne a una mesh di comunicare con le risorse che si trovano all'interno di una mesh.                                                             | 10 luglio 2020    |
| <a href="#">Creazione e gestione delle risorse App Mesh da Kubernetes con il controller App Mesh per Kubernetes</a>    | Le risorse App Mesh possono essere create e gestite da Kubernetes. Il controller inserisce automaticamente il proxy Envoy e i container init nei pod implementati.     | 18 giugno 2020    |
| <a href="#">Aggiungi un valore di timeout al listener e al routing di un nodo virtuale</a>                             | È possibile aggiungere un valore di timeout a un listener e a un routing del nodo virtuale.                                                                            | 18 giugno 2020    |

---

|                                                                              |                                                                                                                                                                                                                           |                 |
|------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| <a href="#">Aggiungi un valore di timeout a un listener di nodi virtuali</a> | (Solo <a href="#">App Mesh Preview Channel</a> ) È possibile aggiungere un valore di timeout a un listener di nodi virtuali.                                                                                              | 29 maggio 2020  |
| <a href="#">Crea un gateway virtuale per la rete mesh in entrata</a>         | (Solo <a href="#">App Mesh Preview Channel</a> ) Abilita le risorse esterne a una mesh per comunicare con le risorse all'interno di una mesh.                                                                             | 8 aprile 2020   |
| <a href="#">Crittografia TLS</a>                                             | (Solo <a href="#">App Mesh Preview Channel</a> ) Utilizza i certificati di una AWS Private Certificate Authority o della tua autorità di certificazione per crittografare le comunicazioni tra nodi virtuali tramite TLS. | 17 gennaio 2020 |
| <a href="#">Condividi una mesh con un altro account</a>                      | (Solo <a href="#">App Mesh Preview Channel</a> ) Puoi condividere una mesh con un altro account. Le risorse create da qualsiasi account possono comunicare con altre risorse della mesh.                                  | 17 gennaio 2020 |
| <a href="#">Aggiungi un valore di timeout a un percorso</a>                  | (Solo <a href="#">App Mesh Preview Channel</a> ) È possibile aggiungere un valore di timeout a un percorso.                                                                                                               | 17 gennaio 2020 |
| <a href="#">Crea un proxy App Mesh su un AWS Outpost</a>                     | Puoi creare un proxy App Mesh Envoy su un AWS Outpost.                                                                                                                                                                    | 3 dicembre 2019 |

|                                                                                   |                                                                                                                                                                                                                                                                                                                               |                   |
|-----------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| <a href="#">Supporto HTTP/2 e gRPC per rotte, router virtuali e nodi virtuali</a> | È possibile indirizzare il traffico che utilizza i protocolli HTTP/2 e gRPC. <a href="#">È inoltre possibile aggiungere un listener per questi protocolli a nodi e router virtuali.</a>                                                                                                                                       | 25 ottobre 2019   |
| <a href="#">Politica di riprova</a>                                               | Una policy per i nuovi tentativi consente ai client di proteggersi da guasti di rete intermittenti o da guasti lato server intermittenti. È possibile aggiungere la logica di ripetizione a un percorso.                                                                                                                      | 10 settembre 2019 |
| <a href="#">Crittografia TLS</a>                                                  | (Solo <a href="#">App Mesh Preview Channel</a> ) Crittografia la comunicazione tra nodi virtuali utilizzando TLS.                                                                                                                                                                                                             | 6 settembre 2019  |
| <a href="#">Routing basato su intestazioni HTTP</a>                               | Indirizza il traffico in base alla presenza e ai valori delle intestazioni HTTP in una richiesta.                                                                                                                                                                                                                             | 15 agosto 2019    |
| <a href="#">Disponibilità dell'App Mesh Preview Channel</a>                       | L'App Mesh Preview Channel è una variante distinta del servizio App Mesh. Il canale di anteprima presenta le funzionalità imminenti da provare man mano che vengono sviluppate. Quando utilizzate le funzionalità del Canale di anteprima, potete fornire feedback GitHub per modificare il comportamento delle funzionalità. | 19 luglio 2019    |

[Endpoint VPC con interfaccia App Mesh \(AWS PrivateLink\)](#)

Migliora il livello di sicurezza del tuo VPC configurando App Mesh per utilizzare un endpoint VPC di interfaccia. Gli endpoint di interfaccia sono alimentati da AWS PrivateLink, una tecnologia che consente di accedere in modo privato alle API App Mesh utilizzando indirizzi IP privati. PrivateLink limita tutto il traffico di rete tra il tuo VPC e App Mesh alla rete Amazon.

14 giugno 2019

[Aggiunto AWS Cloud Map come metodo di individuazione dei servizi dei nodi virtuali](#)

È possibile specificare il DNS o AWS Cloud Map come metodo di rilevamento dei servizi dei nodi virtuali. AWS Cloud Map Per utilizzare l'individuazione dei servizi, l'account deve avere il ruolo [collegato al servizio](#) App Mesh.

13 giugno 2019

[Crea automaticamente risorse App Mesh in Kubernetes](#)

Crea risorse App Mesh e aggiungi automaticamente le immagini del contenitore collaterale App Mesh alle tue implementazioni Kubernetes quando crei risorse in Kubernetes.

11 giugno 2019

[Disponibilità generale di App Mesh](#)

Il servizio App Mesh è ora generalmente disponibile per l'uso in produzione.

27 marzo 2019

[Aggiornamento dell'API App Mesh](#)

Le API App Mesh sono state aggiornate per migliorare e l'usabilità. Per ulteriori informazioni, consulta [\[FEATURE\] Aggiungi listener ai router virtuali](#) e [\[BUG\] Routes to Target Virtual Nodes with Mismatched Ports Blackhole](#).

7 marzo 2019

[Versione iniziale di App Mesh](#)

Documentazione iniziale per l'anteprima pubblica del servizio

28 novembre 2018

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.