



Benutzerhandbuch

AWS App Mesh



AWS App Mesh: Benutzerhandbuch

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

Die Handelsmarken und Handelsaufmachung von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, durch die Kunden irregeführt werden könnten oder Amazon in schlechtem Licht dargestellt oder diskreditiert werden könnte. Alle anderen Marken, die nicht im Besitz von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

Was ist AWS App Mesh?	1
App Mesh zu einer Beispielanwendung hinzufügen	1
Komponenten von App Mesh	2
Erste Schritte	4
Zugriff auf App Mesh	4
Erste Schritte	6
App Mesh und Amazon ECS	6
Szenario	6
Voraussetzungen	7
Schritt 1: Erstellen von einem Mesh und einem virtuellen Service	8
Schritt 2: Erstellen von einem virtuellen Knoten	9
Schritt 3: Erstellen von einem virtuellen Router und einer Route	10
Schritt 4: Überprüfen und Erstellen	12
Schritt 5: Erstellen zusätzlicher Ressourcen	13
Schritt 6: Aktualisieren der Services	19
Fortschrittliche Themen	40
App Mesh und Kubernetes	40
Voraussetzungen	41
Schritt 1: Installieren der Integrationskomponenten	42
Schritt 2: Bereitstellen von App-Mesh-Ressourcen	48
Schritt 3: Erstellen oder Aktualisieren von Services	62
Schritt 4: Bereinigen	69
App Mesh und Amazon EC2	70
Szenario	6
Voraussetzungen	7
Schritt 1: Erstellen von einem Mesh und einem virtuellen Service	71
Schritt 2: Erstellen von einem virtuellen Knoten	72
Schritt 3: Erstellen von einem virtuellen Router und einer Route	10
Schritt 4: Überprüfen und Erstellen	12
Schritt 5: Erstellen zusätzlicher Ressourcen	13
Schritt 6: Aktualisieren der Services	19
App Mesh	94
App Mesh	94
Konzepte	95

Gitter	95
Servicegitter erstellen	95
Löschen eines Netzes	98
Virtuelle Services	99
Erstellen eines virtuellen Services	100
Löschen eines virtuellen Dienstes	102
Virtuelle Gateways	104
Virtuelles Gateway erstellen	105
Virtuelles Gateway bereitstellen	110
Löschen eines virtuellen Netzes ein virtuelles Gateway	111
Gateway-Routen	113
Virtuelle Knoten	119
Einen virtuellen Knoten erstellen	120
Löschen eines virtuellen Knotens	131
Virtuelle Router	133
Erstellen eines virtuellen Routers	134
Löschen eines virtuellen Routers	136
Routen	138
Envoy	150
Envoy-Bildvarianten	150
Envoy-Konfigurationsvariablen	155
Erforderliche Variablen	155
Optionale Variablen	156
Von App Mesh festgelegte Envoy-StandardEinstellungen	163
Standardrichtlinie für die Wiederholung von Routen	164
Standardmäßiger Schutzschalter	165
Aktualisierung/Migration auf Envoy 1.17	166
Secret Discovery Service mit SPIRE	166
Änderungen regulärer Ausdrücke	166
Rückverweise	169
Beauftragter	169
Beobachtbarkeit	171
Protokollierung	171
Firelens und Cloudwatch	173
Metriken für Gesandte	174
Beispiele für Anwendungsmetriken	177

Exportieren von Kennzahlen	181
Nachverfolgung	190
X-Ray	190
Jäger	192
Datadog zur Nachverfolgung	159
Werkzeugbau	195
AWS CloudFormation	195
AWS CDK	195
App Mesh Controller für Kubernetes	195
Terraform	196
Mit gemeinsam genutzten Meshes arbeiten	197
Erteilen von Berechtigungen zum Teilen von Meshes	197
Erteilen Sie die Erlaubnis, ein Mesh gemeinsam zu nutzen	197
Erteilen von Berechtigungen für ein Mesh	198
Voraussetzungen für das Teilen von Meshes	199
Zugehörige Services	200
Ein Mesh teilen	200
Die gemeinsame Nutzung eines gemeinsam genutzten Meshs aufheben	201
Identifizieren eines gemeinsam genutzten Netzes	202
Fakturierung und Messung	203
Kontingente für Instanzen	203
Arbeiten mit anderen Services	204
App Mesh Mesh-Ressourcen erstellen mitAWS CloudFormation	204
App Mesh undAWS CloudFormation Vorlagen	204
Weitere Informationen zu AWS CloudFormation	205
App Mesh aufAWS Outposts	205
Voraussetzungen	205
Einschränkungen	205
Überlegungen zur Netzwerkkonnektivität	206
Einen App Mesh Envoy-Proxy auf einem Outpost erstellen	206
Bewährte Methoden	208
Instrumentiere alle Routen mit Wiederholungsversuchen	208
Bereitstellungsgeschwindigkeit anpassen	209
Vor Skalierung eskalieren	210
Implementieren Sie Zustandsprüfungen für Container	210
Sicherung von Anwendungen	211

Transport Layer Security (TLS)	212
Zertifikatanforderungen	213
TLS-Authentifizierungszertifikate	213
So konfiguriert App Mesh Envoy für die Aushandlung von TLS	216
Überprüfen Sie die Verschlüsselung	217
Zertifikatserneuerung	218
Konfigurieren Sie Amazon ECS-Workloads für die Verwendung der TLS-Authentifizierung mit AWS App Mesh	219
Konfigurieren Sie Kubernetes-Workloads für die Verwendung der TLS-Authentifizierung mit AWS App Mesh	219
Gegenseitige TLS-Authentifizierung	220
Gegenseitige TLS-Authentifizierungszertifikate	221
Mesh-Endpunkte konfigurieren	221
Migrieren Sie Dienste zur gegenseitigen TLS-Authentifizierung	222
Überprüfung der gegenseitigen TLS-Authentifizierung	223
Exemplarische Vorgehensweisen zur gegenseitigen TLS-Authentifizierung von App Mesh ..	224
Identity and Access Management	224
Zielgruppe	225
Authentifizierung mit Identitäten	225
Verwalten des Zugriffs mit Richtlinien	229
Wie AWS App Mesh funktioniert mit IAM	232
Beispiele für identitätsbasierte Richtlinien	236
Von AWS verwaltete Richtlinien	241
Verwenden von serviceverknüpften Rollen	244
En) Proxy-Autorisierung	247
Fehlerbehebung	253
CloudTrail protokolliert	255
App Mesh Mesh-Verwaltungsereignisse in CloudTrail	257
Beispiele für App Mesh Mesh-Ereignisse	257
Datenschutz	258
Datenverschlüsselung	259
Compliance-Validierung	260
Sicherheit der Infrastruktur	261
Schnittstellen-VPC-Endpunkte (AWS PrivateLink)	262
Ausfallsicherheit	263
Notfallwiederherstellung in AWS App Mesh	264

Konfigurations- und Schwachstellenanalyse	264
Fehlerbehebung	265
Bewährte Methoden	265
Aktivieren der Envoy-Proxy-Verwaltungsschnittstelle	265
Aktivieren der Envoy DogStats-D-Integration für Metrik-Offload	266
Aktivieren der Zugriffsprotokolle	266
Aktivieren der Envoy-Debug-Protokollierung in Vorproduktionsumgebungen	266
Überwachen der Envoy-Proxy-Konnektivität mit der App-Mesh-Steuerebene	267
Aufstellen	267
Envoy-Container-Image kann nicht abgerufen werden	267
Es kann keine Verbindung zum App Mesh Envoy Management Service hergestellt werden .	268
Envoy wurde mit Fehlertext vom App Mesh Envoy Management Service getrennt	269
Fehler bei der Zustandsprüfung des Envoy-Containers, der Bereitschafts-Sonde oder der Liveliness-Sonde	273
Zustandsprüfung vom Load Balancer zum Mesh-Endpunkt schlägt fehl	273
Das virtuelle Gateway akzeptiert keinen Datenverkehr über Ports 1024 oder weniger	274
Konnektivität	275
DNS-Name für einen virtuellen Service kann nicht aufgelöst werden	275
Es konnte keine Verbindung zu einem virtuellen Service-Backend hergestellt werden	276
Es konnte keine Verbindung zu einem externen Service hergestellt werden	278
Verbindung zu einem MySQL- oder SMTP-Server nicht möglich	279
Es konnte keine Verbindung zu einem Service hergestellt werden, der als virtueller TCP- Knoten oder virtueller Router in App Mesh modelliert wurde.	280
Die Verbindung zu einem Service, der nicht als virtuelles Service-Backend für einen virtuellen Knoten aufgeführt ist, ist erfolgreich	281
Einige Anfragen schlagen mit HTTP-Statuscode fehl503, wenn ein virtueller Service über einen virtuellen Knotenanbieter verfügt	281
Es konnte keine Verbindung zu einem Amazon-EFS-Dateisystem hergestellt werden	282
Die Verbindung zum -Service ist erfolgreich, aber die eingehende Anforderung wird nicht in Zugriffsprotokollen, Ablaufverfolgungen oder Metriken für Envoy angezeigt	283
Das Festlegen der Umgebungsvariablen HTTP_PROXY/HTTPS_PROXY auf Containerebene funktioniert nicht wie erwartet.	283
Upstream-Anforderungs-Timeouts auch nach Einstellung des Timeouts für Routen.	284
Envoy antwortet mit einer HTTP-Bad-Anforderung.	285
Das Timeout kann nicht richtig konfiguriert werden.	286
Skalierung	286

Konnektivität schlägt fehl und Container-Zustandsprüfungen schlagen fehl, wenn die Skalierung über 50 Replikate für einen virtuellen Knoten/ein virtuelles Gateway hinaus skaliert wird	286
Anforderungen schlagen mit fehl503, wenn ein virtuelles Service-Backend horizontal auf- oder abskaliert wird	287
Envoy-Container stürzt bei erhöhter Last mit segfault ab	287
Die Erhöhung der Standardressourcen spiegelt sich nicht in den Service Limits wider	288
Die Anwendung stürzt aufgrund einer großen Anzahl von Aufrufen für Zustandsprüfungen ab.	288
Beobachtbarkeit	289
AWS X-Ray Ablaufverfolgungen für meine Anwendungen konnten nicht angezeigt werden .	289
Envoy-Metriken für meine Anwendungen können in Amazon- CloudWatch Metriken nicht angezeigt werden	290
Benutzerdefinierte Samplingregeln für AWS X-Ray Ablaufverfolgungen können nicht konfiguriert werden	291
Sicherheit	292
Herstellen einer Verbindung zu einem virtuellen Backend-Service mit einer TLS-Client-Richtlinie nicht möglich	292
Verbindung zu einem virtuellen Backend-Service nicht möglich, wenn die Anwendung TLS erzeugt	294
Es konnte nicht bestätigt werden, dass die Konnektivität zwischen Envoy-Proxys TLS verwendet	294
Fehlerbehebung bei TLS mit Elastic Load Balancing	296
Kubernetes	297
In Kubernetes erstellte App-Mesh-Ressourcen können in App Mesh nicht gefunden werden	297
Pods bestehen nach dem Einfügen des Envoy-Sidecars keine Bereitschafts- und Liveliness-Prüfungen	298
Pods, die sich nicht als Instances registrieren oder deren Registrierung aufheben AWS Cloud Map	299
Es kann nicht festgestellt werden, wo ein Pod für eine App-Mesh-Ressource ausgeführt wird	300
Es kann nicht festgestellt werden, als welche App-Mesh-Ressource ein Pod ausgeführt wird	300
Client-Envoys können nicht mit dem App Mesh Envoy Management Service kommunizieren, wenn IMDSv1 deaktiviert ist	301

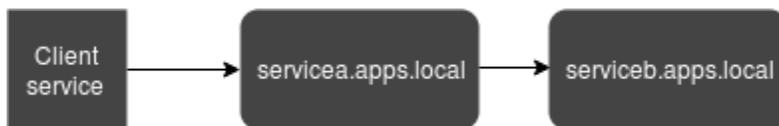
IRSA funktioniert nicht auf dem Anwendungscontainer, wenn App Mesh aktiviert ist und Envoy injiziert wird	301
Kanal vorschauen	303
Servicekontingente	308
Dokumentverlauf	309
.....	cccxvi

Was ist AWS App Mesh?

AWS App Mesh ist ein Service-Mesh, mit dem Services einfach überwacht und gesteuert werden können. Ein Service Mesh ist eine Infrastrukturebene, die der service-to-service Kommunikation gewidmet ist, in der Regel über eine Reihe leichter Netzwerk-Proxys, die zusammen mit dem Anwendungscode bereitgestellt werden. App Mesh standardisiert, wie Ihre Services kommunizieren und sorgt somit für end-to-end Transparenz und hohe Verfügbarkeit für Ihre Anwendungen. App Mesh bietet Ihnen konsistente Transparenz und Kontrolle über den Netzwerkdatenverkehr eines jeden Service in einer Anwendung.

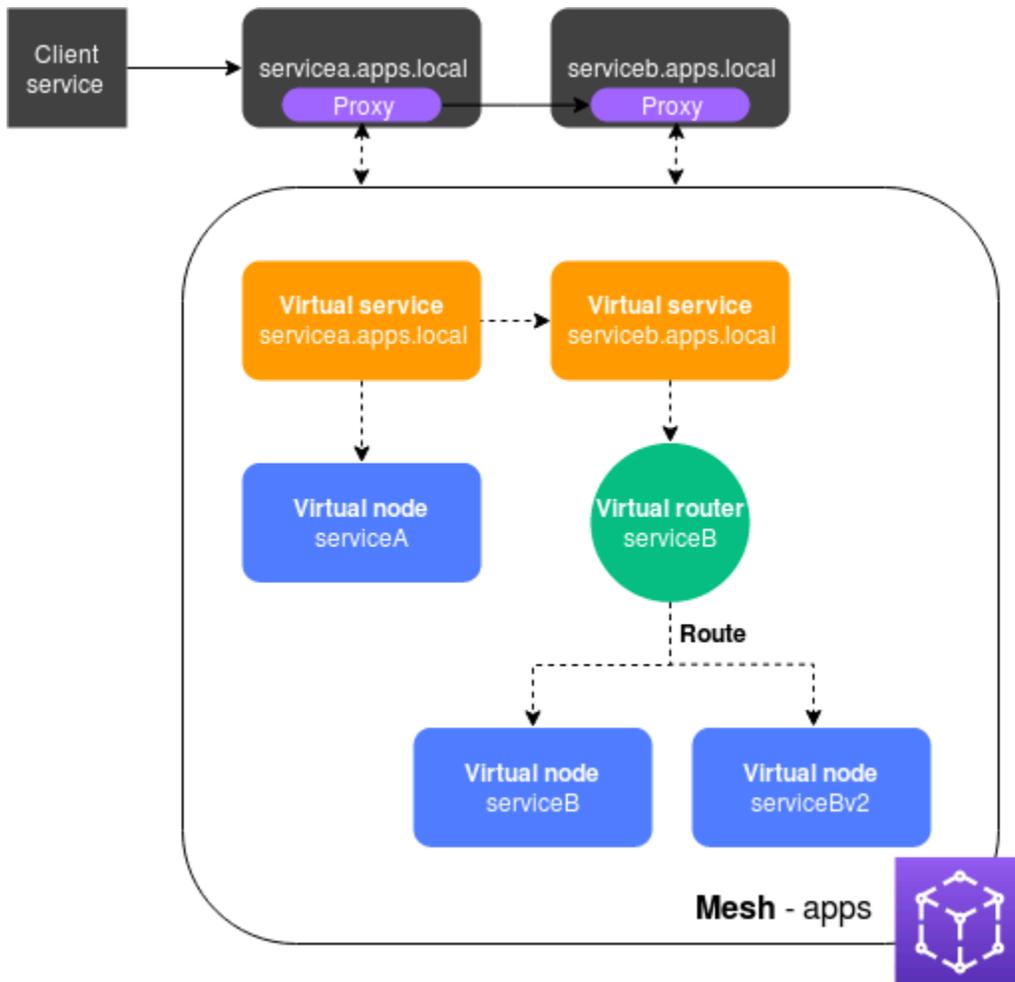
App Mesh zu einer Beispielanwendung hinzufügen

Betrachten Sie die folgende einfache Beispielanwendung, die App Mesh nicht verwendet. Die beiden Services können Amazon Elastic Kubernetes Service auf Elastic Compute Cloud (Amazon EC2) EC2-Instances mit Docker ausgeführt werden. AWS Fargate



In dieser Abbildung sind `serviceA` und `serviceB` über den `apps.local` Namespace auffindbar. Nehmen wir an, Sie entscheiden sich beispielsweise für die Bereitstellung einer neuen Version von `serviceb.apps.local` namens `servicebv2.apps.local`. Als Nächstes möchten Sie einen Prozentsatz des Datenverkehrs von `servicea.apps.local` an `serviceb.apps.local` und einen Prozentsatz an weiterleiten `servicebv2.apps.local`. Wenn Sie sicher sind, `servicebv2` dass das gut funktioniert, möchten Sie 100 Prozent des Traffics dorthin senden.

App Mesh kann Ihnen dabei helfen, ohne den Anwendungscode oder die registrierten Dienstnamen zu ändern. Wenn Sie App Mesh mit dieser Beispielanwendung verwenden, sieht Ihr Mesh möglicherweise wie in der folgenden Abbildung aus.



In dieser Konfiguration kommunizieren die Dienste nicht mehr direkt miteinander. Stattdessen kommunizieren sie über einen Proxy miteinander. Der mit dem `servicea.apps.local` Dienst bereitgestellte Proxy liest die App Mesh Mesh-Konfiguration und sendet Datenverkehr an `serviceb.apps.local` oder `servicebv2.apps.local` basierend auf der Konfiguration.

Komponenten von App Mesh

App Mesh besteht aus den folgenden Komponenten, wie im vorherigen Beispiel dargestellt:

- **Service-Mesh** — Ein Service-Mesh ist eine logische Begrenzung für den Netzwerkverkehr zwischen den darin vorhandenen Services. Im Beispiel wird das Netz benannt `apps` und es enthält alle anderen Ressourcen für das Netz. Weitere Informationen finden Sie unter [Servicegitter](#).
- **Virtuelle Services** — Ein virtueller Service ist eine Abstraktion eines tatsächlichen Services, der direkt oder indirekt über einen virtuellen Router bereitgestellt wird. In der Abbildung stellen zwei virtuelle Dienste die beiden tatsächlichen Dienste dar. Die Namen der virtuellen Dienste sind die

auffindbaren Namen der tatsächlichen Dienste. Wenn ein virtueller Dienst und ein tatsächlicher Dienst denselben Namen haben, können mehrere Dienste unter denselben Namen miteinander kommunizieren, die sie vor der Implementierung von App Mesh verwendet haben. Weitere Informationen finden Sie unter [Virtuelle Services](#).

- **Virtuelle Knoten** — Ein virtueller Knoten fungiert als logischer Verweis auf einen auffindbaren Service, z. B. einen Amazon ECS- oder Kubernetes-Dienst. Für jeden virtuellen Dienst verfügen Sie über mindestens einen virtuellen Knoten. In der Abbildung erhält `derservicea.apps.local` virtuelle Dienst Konfigurationsinformationen für den benannten virtuellen Knoten `serviceA`. `derserviceA` virtuelle Knoten ist mit dem `derservicea.apps.local` Namen für Service Discovery konfiguriert. `derserviceb.apps.local` virtuelle Dienst ist so konfiguriert, dass er den Datenverkehr über einen virtuellen Router namens `serviceBv2` an die virtuellen Knoten weiterleitet `serviceB`. Weitere Informationen finden Sie unter [Virtuelle Knoten](#).
- **Virtuelle Router und Routen** — Virtuelle Router verarbeiten den Datenverkehr für einen oder mehrere virtuelle Services innerhalb Ihres Netzes. Eine Route ist einem virtuellen Router zugeordnet. Die Route wird verwendet, um Anfragen für den virtuellen Router abzugleichen und den Datenverkehr an die zugehörigen virtuellen Knoten zu verteilen. In der vorherigen Abbildung verfügt `derserviceB` virtuelle Router über eine Route, die einen Prozentsatz des Datenverkehrs an `derserviceB` virtuellen Knoten und einen Prozentsatz des Datenverkehrs an `derserviceBv2` virtuellen Knoten weiterleitet. Sie können den Prozentsatz des Datenverkehrs festlegen, der an einen bestimmten virtuellen Knoten weitergeleitet wird, und ihn im Laufe der Zeit ändern. Sie können den Datenverkehr anhand von Kriterien wie HTTP-Headern, URL-Pfaden oder gRPC-Dienst- und Methodennamen weiterleiten. Sie können Wiederholungsrichtlinien so konfigurieren, dass eine Verbindung erneut versucht wird, wenn in der Antwort ein Fehler auftritt. In der Abbildung kann die Wiederholungsrichtlinie für die Route beispielsweise angeben, dass eine Verbindung zu `serviceb.apps.local` fünfmal wiederholt wird, wobei zwischen den Wiederholungsversuchen zehn Sekunden liegen, wenn bestimmte Fehlertypen `serviceb.apps.local` zurückgegeben werden. Weitere Informationen erhalten Sie unter [Virtuelle Router](#) und [Routen](#).
- **Proxy** — Sie konfigurieren Ihre Dienste so, dass sie den Proxy verwenden, nachdem Sie Ihr Mesh und seine Ressourcen erstellt haben. Der Proxy liest die App Mesh Mesh-Konfiguration und leitet den Datenverkehr entsprechend weiter. In der Abbildung `serviceb.apps.local` läuft die gesamte Kommunikation von `servicea.apps.local` bis über den Proxy, der für jeden Dienst bereitgestellt wird. Die Dienste kommunizieren miteinander unter Verwendung derselben Service Discovery-Namen, die sie vor der Einführung von App Mesh verwendet haben. Da der Proxy die App Mesh Mesh-Konfiguration liest, können Sie steuern, wie die beiden Dienste miteinander kommunizieren. Wenn Sie die App Mesh Mesh-Konfiguration ändern möchten, müssen Sie die

Dienste selbst oder die Proxys nicht ändern oder erneut bereitstellen. Weitere Informationen finden Sie unter [Bild des Gesandten](#).

Erste Schritte

Um App Mesh verwenden zu können AWS Fargate, muss ein vorhandener Service auf Amazon ECS, Amazon EKS, Kubernetes auf Amazon EC2 oder Amazon EC2 mit Docker ausgeführt werden.

Informationen zu den ersten Schritten mit App Mesh finden Sie in einer der folgenden Anleitungen:

- [Erste Schritte mit App Mesh und Amazon ECS](#)
- [Erste Schritte mit App Mesh und Kubernetes](#)
- [Erste Schritte mit App Mesh und Amazon EC2](#)

Zugriff auf App Mesh

Sie können App Mesh wie folgt nutzen:

AWS Management Console

Die Konsole ist eine browserbasierte Schnittstelle, mit der Sie App Mesh -Ressourcen verwalten können. Sie können die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/> öffnen.

AWS CLI

Bietet Befehle für zahlreiche AWS-Produkte und wird unter Windows-, Mac- und Linux-Betriebssystemen unterstützt. Informationen zu den ersten Schritten finden Sie im [AWS Command Line Interface-Benutzerhandbuch](#). Weitere Informationen zu den Befehlen für App Mesh finden Sie unter [appmesh](#) in der [AWS CLIBefehlsreferenz](#).

AWS Tools for Windows PowerShell

Bietet Befehle für zahlreiche AWS -Produkte für Benutzer, die PowerShell Umgebung erstellen. Informationen zu den ersten Schritten finden Sie im [AWS Tools for Windows PowerShell-Benutzerhandbuch](#). Weitere Informationen zu den Cmdlets für App Mesh finden Sie unter [App Mesh](#) in der [AWSTools for PowerShell Cmdlet-Referenz](#).

AWS CloudFormation

Ermöglicht es Ihnen, eine Vorlage zu erstellen, die gewünschten AWS Ressourcen beschreibt. Mithilfe der Vorlage werden die Ressourcen für Sie bereitgestellt und konfiguriert. AWS CloudFormation Informationen zu den ersten Schritten finden Sie im [AWS CloudFormation-Benutzerhandbuch](#). Weitere Informationen zu den App Mesh Mesh-Ressourcentypen finden Sie unter [App Mesh Mesh-Ressourcentypreferenz](#) in der [AWS CloudFormation Vorlagenreferenz](#).

AWS-SDKs

Wir bieten auch SDKs, mit denen Sie aus einer Vielzahl von Programmiersprachen auf App Mesh zugreifen können. Die SDKs kümmern sich automatisch um Aufgaben wie:

- Kryptographisches Signieren Ihrer Serviceanfragen
- Wiederholen von Anfragen
- Umgang mit Fehlerreaktionen

Weitere Informationen über verfügbare SDKs finden Sie unter [Tools für Amazon Web Services](#).

Weitere Informationen über die App Mesh APIs finden Sie in der [AWS App Mesh API-Referenz](#).

Erste Schritte mit App Mesh

Sie können App Mesh mit Anwendungen verwenden, die Sie auf Amazon ECS, Kubernetes (die Sie auf Ihren eigenen Amazon EC2-Instances bereitstellen oder auf Amazon EKS ausführen) und Amazon EC2 bereitstellen. Um mit App Mesh zu beginnen, wählen Sie einen der Dienste aus, für die Sie Anwendungen bereitgestellt haben und die Sie mit App Mesh verwenden möchten. Sie können Anwendungen in den anderen Diensten jederzeit aktivieren, damit sie auch mit App Mesh funktionieren, nachdem Sie eine der Anleitungen für die ersten Schritte abgeschlossen haben.

Themen

- [Erste Schritte mit AWS App Mesh Amazon ECS](#)
- [Erste Schritte mit AWS App Mesh und Kubernetes](#)
- [Erste Schritte mit AWS App Mesh und Amazon EC2](#)
- [App Mesh](#)
- [App Mesh](#)

Erste Schritte mit AWS App Mesh Amazon ECS

Dieses Thema hilft Ihnen bei der Verwendung AWS App Mesh mit einem tatsächlichen Service, der auf Amazon ECS ausgeführt wird. Dieses Tutorial behandelt die grundlegenden Funktionen verschiedener App Mesh Mesh-Ressourcentypen.

Szenario

Gehen Sie zur Veranschaulichung der Verwendung von App Mesh davon aus, dass Sie über eine Anwendung mit den folgenden Eigenschaften verfügen:

- Besteht aus zwei Diensten mit dem Namen `serviceA` und `serviceB`.
- Beide Services sind in einem Namespace namens `apps.local` registriert.
- `ServiceA` kommuniziert mit `serviceB` über HTTP/2, Port 80.
- Sie haben bereits Version 2 von `serviceB` bereitgestellt und mit dem Namen `serviceBv2` im `apps.local`-Namespace registriert.

Es gelten die folgenden Anforderungen:

- Sie möchten 75 Prozent des Datenverkehrs von `serviceB` und 25 Prozent des Datenverkehrs `serviceA` nach senden, um `serviceBv2` zu überprüfen, ob der Datenverkehr fehlerfrei `serviceBv2` ist, bevor Sie 100 Prozent des Datenverkehrs von dorthin `serviceA` weiterleiten.
- Sie möchten in der Lage sein, die Datenverkehrsgewichtung so anzupassen, dass 100 Prozent des Datenverkehrs an `serviceBv2` gehen, sobald sich die Zuverlässigkeit davon erwiesen hat. Sobald der gesamte Verkehr an `serviceBv2` gesendet wurde, möchten Sie den Vorgang beenden `serviceB`.
- Sie möchten für Ihre eigentlichen Dienste keinen bestehenden Anwendungscode oder eine Registrierung zur Serviceerkennung ändern müssen, um die vorherigen Anforderungen zu erfüllen.

Um Ihren Anforderungen gerecht zu werden, entscheiden Sie sich dafür, ein App Mesh Service Mesh mit virtuellen Diensten, virtuellen Knoten, einem virtuellen Router und einer Route zu erstellen. Nach der Implementierung Ihres Meshs aktualisieren Sie Ihre Dienste so, dass sie den Envoy-Proxy verwenden. Nach der Aktualisierung kommunizieren Ihre Services miteinander über den Envoy-Proxy und nicht direkt miteinander.

Voraussetzungen

- Ein vorhandenes Verständnis der App Mesh Mesh-Konzepte. Weitere Informationen finden Sie unter [Was ist AWS App Mesh?](#).
- Ein vorhandenes Verständnis der Konzepte von Amazon ECS. Weitere Informationen finden Sie unter [Was ist Amazon ECS](#) im Amazon Elastic Container Service Developer Guide.
- App Mesh unterstützt Linux-Dienste, die mit DNS oder beidem registriert sind. AWS Cloud Map Um dieses Handbuch für Erste Schritte zu verwenden, empfehlen wir, dass Sie über drei vorhandene Services verfügen, die bei DNS registriert sind. Bei den Verfahren in diesem Thema wird davon ausgegangen `serviceA` `serviceB`, dass die vorhandenen Dienste benannt sind `serviceBv2` und dass alle Dienste über einen Namespace namens auffindbar sind. `apps.local`

Sie können ein Service-Mesh und seine Ressourcen erstellen, auch wenn die Services nicht vorhanden sind. Sie können das Mesh jedoch erst verwenden, wenn Sie tatsächliche Services bereitgestellt haben. Weitere Informationen zur Serviceerkennung auf Amazon ECS finden Sie unter [Service Discovery](#). Informationen zum Erstellen eines Amazon ECS-Service mit Service Discovery finden Sie unter [Tutorial: Einen Service mithilfe von Service Discovery erstellen](#). Wenn Sie noch keine Dienste ausführen, können Sie [einen Amazon ECS-Service mit Service Discovery erstellen](#).

Schritt 1: Erstellen von einem Mesh und einem virtuellen Service

Ein Service Mesh ist eine logische Begrenzung für den Netzwerkdatenverkehr zwischen den darin vorhandenen Services. Weitere Informationen finden Sie unter [Servicegitter](#). Ein virtueller Service ist eine Abstraktion eines tatsächlichen Services. Weitere Informationen finden Sie unter [Virtuelle Services](#).

Erstellen Sie die folgenden Ressourcen:

- Ein Mesh mit dem Namen `apps`, da alle Services im Szenario im `apps.local`-Namespace registriert sind.
- Einen virtuellen Service mit dem Namen `serviceb.apps.local`, da der virtuelle Service einen Service darstellt, der mit diesem Namen gefunden werden kann, und Sie Ihren Code nicht ändern möchten, um auf einen anderen Namen zu verweisen. Ein virtueller Service mit dem Namen `servicea.apps.local` wird in einem späteren Schritt hinzugefügt.

Sie können die AWS Management Console oder die AWS CLI Version 1.18.116 oder höher oder 2.0.38 oder höher verwenden, um die folgenden Schritte auszuführen. Wenn Sie die verwenden AWS CLI, verwenden Sie den `aws --version` Befehl, um Ihre installierte Version zu überprüfen. AWS CLI Wenn Sie Version 1.18.116 oder höher oder 2.0.38 oder höher nicht installiert haben, müssen Sie die [installieren oder aktualisieren](#). AWS CLI Wählen Sie die Registerkarte für das Werkzeug aus, das Sie verwenden möchten.

AWS Management Console

1. Öffnen Sie den Assistenten für die erste Ausführung der App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/get-started>.
2. Geben Sie unter Mesh name (Mesh-Name) **apps** ein.
3. Geben Sie unter Virtual service name (Name des virtuellen Services) **serviceb.apps.local** ein.
4. Wählen Sie Next, um fortzufahren.

AWS CLI

1. Erstellen Sie mit dem [create-mesh](#)-Befehl ein Mesh.

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

- Erstellen Sie einen virtuellen Service mit dem `create-virtual-service`-Befehl.

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

Schritt 2: Erstellen von einem virtuellen Knoten

Ein virtueller Knoten fungiert als ein logischer Verweis auf einen tatsächlichen Service. Weitere Informationen finden Sie unter [Virtuelle Knoten](#).

Erstellen Sie einen virtuellen Knoten mit dem Namen `serviceB`, da einer der virtuellen Knoten den tatsächlichen Service mit dem Namen `serviceB` darstellt. Der tatsächliche Service, den der virtuelle Knoten darstellt, kann über DNS mit dem Hostnamen von `serviceb.apps.local` gefunden werden. Alternativ können Sie aktuelle Dienste mithilfe von `aws cloudmap` ermitteln. Der virtuelle Knoten überwacht den Datenverkehr mit dem HTTP/2-Protokoll auf Port 80. Auch andere Protokolle werden unterstützt, ebenso wie Zustandsprüfungen. Sie erstellen virtuelle Knoten für `serviceA` und `serviceBv2` in einem späteren Schritt.

AWS Management Console

- Geben Sie unter Virtual node name (Name des virtuellen Knotens) **serviceB** ein.
- Wählen Sie unter Service discovery method (Serviceerkennungungsverfahren) die Option DNS aus, und geben Sie **serviceb.apps.local** als DNS hostname (DNS-Hostname) ein.
- Wählen Sie unter Listener-Konfiguration `http2` als Protokoll und geben Sie **80** als Port ein.
- Wählen Sie Next, um fortzufahren.

AWS CLI

- Erstellen Sie eine Datei `create-virtual-node-serviceb.json` mit dem folgenden Inhalt:

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

```
        "protocol": "http2"
      }
    },
    "serviceDiscovery": {
      "dns": {
        "hostname": "serviceB.apps.local"
      }
    }
  },
  "virtualNodeName": "serviceB"
}
```

- Erstellen Sie den virtuellen Knoten mit dem [create-virtual-node](#) Befehl und verwenden Sie dabei die JSON-Datei als Eingabe.

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

Schritt 3: Erstellen von einem virtuellen Router und einer Route

Virtuelle Router routen den Datenverkehr für einen oder mehrere virtuelle Services innerhalb Ihres Meshes. Weitere Informationen finden Sie unter [Virtuelle Router](#) und [Routen](#).

Erstellen Sie die folgenden Ressourcen:

- Einen virtuellen Router mit dem Namen `serviceB`, da der virtuelle `serviceB.apps.local`-Service keine ausgehende Kommunikation mit einem anderen Service initiiert. Denken Sie daran, dass der virtuelle Service, den Sie zuvor erstellt haben, eine Abstraktion Ihres tatsächlichen `serviceb.apps.local`-Services ist. Der virtuelle Service sendet Datenverkehr an den virtuellen Router. Der virtuelle Router überwacht den Datenverkehr mithilfe des HTTP/2-Protokolls an Port 80. Andere Protokolle werden ebenfalls unterstützt.
- Eine Route namens `serviceB`. Er leitet 100 Prozent seines Datenverkehrs an den `serviceB` virtuellen Knoten weiter. Die Gewichtung erfolgt in einem späteren Schritt, sobald Sie den `serviceBv2` virtuellen Knoten hinzugefügt haben. Obwohl in diesem Handbuch nicht behandelt, können Sie zusätzliche Filterkriterien für die Route hinzufügen und eine Wiederholungsrichtlinie hinzufügen, damit der Envoy-Proxy mehrere Versuche unternimmt, Datenverkehr an einen virtuellen Knoten zu senden, wenn ein Kommunikationsproblem auftritt.

AWS Management Console

1. Geben Sie für Virtual router name (Name des virtuellen Routers) **serviceB** ein.
2. Wählen Sie unter Listener-Konfiguration http2 als Protokoll und geben Sie als Port **80** an.
3. Geben Sie unter Route name (Name der Route) **serviceB** ein.
4. Wählen Sie für Route type (Routentyp) die Option http2.
5. Wählen Sie für den Namen des virtuellen Knotens unter Zielkonfiguration die Option Gewicht aus **serviceB** und geben **100** Sie den Wert ein.
6. Wählen Sie unter Konfiguration anpassen eine Methode aus.
7. Wählen Sie Next, um fortzufahren.

AWS CLI

1. Erstellen Sie einen virtuellen Router.
 - a. Erstellen Sie eine Datei `create-virtual-router.json` mit dem folgenden Inhalt:

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

- b. Erstellen Sie den virtuellen Router mit dem [create-virtual-router](#) Befehl und verwenden Sie dabei die JSON-Datei als Eingabe.

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

2. Erstellen Sie eine Route.

- a. Erstellen Sie eine Datei `create-route.json` mit dem folgenden Inhalt:

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

- b. Erstellen Sie die Route mit dem Befehl [create-route](#) unter Verwendung der JSON-Datei als Eingabe.

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

Schritt 4: Überprüfen und Erstellen

Überprüfen Sie die Einstellungen anhand der vorherigen Anweisungen.

AWS Management Console

Wählen Sie Edit (Bearbeiten) aus, wenn Sie Änderungen in einem Abschnitt vornehmen müssen. Sobald Sie mit den Einstellungen zufrieden sind, wählen Sie Create mesh (Netz erstellen) aus.

Im Fenster Status werden alle erstellten Netzressourcen angezeigt. Sie können die erstellten Ressourcen in der Konsole anzeigen, indem Sie View mesh (Netz anzeigen) auswählen.

AWS CLI

Überprüfen Sie die Einstellungen des Meshes, das Sie erstellt haben, mit dem Befehl [describe-mesh](#).

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

Überprüfen Sie die Einstellungen des virtuellen Dienstes, den Sie mit dem [describe-virtual-service](#) Befehl erstellt haben.

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

Überprüfen Sie die Einstellungen des virtuellen Knotens, den Sie mit dem [describe-virtual-node](#) Befehl erstellt haben.

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

Überprüfen Sie die Einstellungen des virtuellen Routers, den Sie mit dem [describe-virtual-router](#) Befehl erstellt haben.

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

Überprüfen Sie die Einstellungen der Route, die Sie erstellt haben, mit dem Befehl [describe-route](#).

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

Schritt 5: Erstellen zusätzlicher Ressourcen

Um das Szenario abzuschließen, müssen Sie:

- Einen virtuellen Knoten mit dem Namen `serviceBv2` und einen anderen mit dem Namen `serviceA` erstellen. Beide virtuellen Knoten warten auf Anfragen über HTTP/2-Port 80. Konfigurieren Sie für den `serviceA` virtuellen Knoten ein Backend von `serviceb.apps.local`. Der gesamte ausgehende Datenverkehr vom `serviceA` virtuellen Knoten wird an den genannten virtuellen Dienst gesendet. `serviceb.apps.local` Obwohl dies in diesem Handbuch nicht

behandelt wird, können Sie auch einen Dateipfad angeben, in den Zugriffsprotokolle für einen virtuellen Knoten geschrieben werden sollen.

- Erstellen Sie einen zusätzlichen virtuellen Dienst mit dem Namensservice `servicea.apps.local`, der den gesamten Datenverkehr direkt an den `serviceA` virtuellen Knoten sendet.
- Aktualisieren Sie die `serviceB`-Route, die Sie in einem vorherigen Schritt erstellt haben, so, dass 75 Prozent des Datenverkehrs an den virtuellen `serviceB`-Knoten und 25 Prozent des Datenverkehrs an den virtuellen `serviceBv2`-Knoten gesendet werden. Im Laufe der Zeit können Sie die Gewichtungen weiter ändern, bis `serviceBv2` 100 Prozent des Datenverkehrs erhält. Sobald der gesamte Datenverkehr an `serviceBv2` gesendet wurde, können Sie den `serviceB` virtuellen Knoten und den eigentlichen Dienst herunterfahren und beenden. Wenn Sie Gewichtungen ändern, erfordert Ihr Code keine Änderung, da sich die Namen des virtuellen `serviceb.apps.local`- und des tatsächlichen Services nicht ändern. Denken Sie daran, dass der virtuelle `serviceb.apps.local`-Service Datenverkehr an den virtuellen Router sendet, der den Datenverkehr an die virtuellen Knoten weiterleitet. Die Serviceerkennungsnamen für die virtuellen Knoten können jederzeit geändert werden.

AWS Management Console

1. Wählen Sie im linken Navigationsbereich Meshes.
2. Wählen Sie das apps-Mesh aus, das Sie in einem vorherigen Schritt erstellt haben.
3. Wählen Sie im linken Navigationsbereich Virtual nodes (Virtuelle Knoten) aus.
4. Wählen Sie Create virtual node (Virtuellen Knoten erstellen).
5. Geben Sie unter Virtual node name (Name des virtuellen Knotens) **serviceBv2** ein, wählen Sie für Service discovery method (Serviceerkennungungsverfahren) die Option DNS aus, und geben Sie für den DNS hostname (DNS-Hostname) **servicebv2.apps.local** ein.
6. Wählen Sie für die Listener-Konfiguration http2 als Protokoll und geben Sie **80** für Port ein.
7. Wählen Sie Create virtual node (Virtuellen Knoten erstellen).
8. Wählen Sie erneut Create virtual node (Virtuellen Knoten erstellen) aus. Geben Sie „**serviceA**“ als Virtual node name (Name des virtuellen Knotens) ein. Wählen Sie als Service discovery method (Diensterkennungsmethode) die Option DNS ein, und geben Sie als DNS hostname (DNS-Hostname) den Wert „**servicea.apps.local**“ ein.
9. Geben Sie bei Enter a virtual service name (Neuen virtuellen Service-Name eingeben) unter New backend (Neues Backend) den Wert „**serviceb.apps.local**“ ein.

10. Wählen Sie unter Listener configuration (Listener-Konfiguration) die Option `http2` für Protocol (Protokoll) aus, geben Sie „**80**“ bei Port ein, und wählen Sie dann `Create virtual node` (Virtuellen Knoten erstellen) aus.
11. Wählen Sie im linken Navigationsbereich `Virtual routers` (Virtuelle Router) aus, und wählen Sie dann den virtuellen Router `serviceB` aus der Liste aus.
12. Wählen Sie unter `Routes` (Routen) die Route `ServiceB` aus, die Sie in einem vorherigen Schritt erstellt haben, und wählen Sie dann `Edit` (Bearbeiten) aus.
13. Ändern Sie unter `Targets` (Ziele), `Virtual node name` (Name des virtuellen Knotens) den Wert von `Weight` (Gewicht) für `serviceB` zu **75**.
14. Wählen Sie `Ziel hinzufügen`, wählen Sie **`serviceBv2`** aus der Dropdownliste und legen Sie den Wert für `Weight` auf fest. **25**
15. Wählen Sie `Speichern`.
16. Wählen Sie im linken Navigationsbereich `Virtual services` (Virtuelle Services) aus, und wählen Sie dann `Create virtual service` (Virtuellen Dienst erstellen) aus.
17. Geben Sie „**`servicea.apps.local`**“ bei `Virtual service name` (Virtueller Dienstname) ein, wählen Sie `Virtual node` (Virtueller Knoten) als `Provider` (Anbieter), wählen Sie „`serviceA`“ bei `Virtual node` (Virtueller Knoten) und dann `Create virtual service` (Virtuellen Dienst erstellen) aus.

AWS CLI

1. Erstellen Sie den virtuellen Knoten `serviceBv2`.
 - a. Erstellen Sie eine Datei `create-virtual-node-servicebv2.json` mit dem folgenden Inhalt:

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

```
    "serviceDiscovery": {
      "dns": {
        "hostname": "serviceBv2.apps.local"
      }
    },
    "virtualNodeName": "serviceBv2"
  }
}
```

- b. Erstellen Sie den virtuellen Knoten.

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

2. Erstellen Sie den virtuellen Knoten serviceA.

- a. Erstellen Sie eine Datei `create-virtual-node-servicea.json` mit dem folgenden Inhalt:

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

```
"virtualNodeName" : "serviceA"
}
```

- b. Erstellen Sie den virtuellen Knoten.

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

3. Aktualisieren Sie den virtuellen Service `serviceb.apps.local`, den Sie in einem vorherigen Schritt erstellt haben, um seinen Datenverkehr an den virtuellen Router `serviceB` zu senden. Als der virtuelle Service ursprünglich erstellt wurde, sendete er keinen Datenverkehr, da der virtuelle Router `serviceB` noch nicht erstellt war.

- a. Erstellen Sie eine Datei `update-virtual-service.json` mit dem folgenden Inhalt:

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

- b. Aktualisieren Sie den virtuellen Dienst mit dem [update-virtual-service](#) Befehl.

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

4. Aktualisieren Sie die Route `serviceB`, die Sie in einem vorherigen Schritt erstellt haben.

- a. Erstellen Sie eine Datei `update-route.json` mit dem folgenden Inhalt:

```
{
  "meshName" : "apps",
  "routeName" : "serviceB",
  "spec" : {
    "http2Route" : {
      "action" : {
```

```
        "weightedTargets" : [
          {
            "virtualNode" : "serviceB",
            "weight" : 75
          },
          {
            "virtualNode" : "serviceBv2",
            "weight" : 25
          }
        ],
        "match" : {
          "prefix" : "/"
        }
      }
    },
    "virtualRouterName" : "serviceB"
  }
}
```

- b. Aktualisieren Sie die Route mit dem Befehl [update-route](#) .

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

5. Erstellen Sie den virtuellen Service serviceA.

- a. Erstellen Sie eine Datei `create-virtual-servicea.json` mit dem folgenden Inhalt:

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

- b. Erstellen Sie den virtuellen Service.

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

Mesh-Zusammenfassung

Bevor Sie das Service-Mesh erstellt haben, hatten Sie drei aktuelle Services mit den Namen `servicea.apps.local`, `serviceb.apps.local` und `servicebv2.apps.local`. Zusätzlich zu den tatsächlichen Services verfügen Sie jetzt über ein Service-Mesh, das die folgenden Ressourcen enthält, die die tatsächlichen Services darstellen:

- Zwei virtuelle Services. Der Proxy sendet den gesamten Datenverkehr vom virtuellen Service `servicea.apps.local` über einen virtuellen Router an den virtuellen Service `serviceb.apps.local`.
- Drei virtuelle Knoten mit den Namen `serviceA`, `serviceB` und `serviceBv2`. Der Envoy-Proxy verwendet die für die virtuellen Knoten konfigurierten Service-Erkennungsinformationen, um die IP-Adressen der tatsächlichen Services zu suchen.
- Einen virtuellen Router mit einer Route, die den Envoy-Proxy anweist, 75 Prozent des eingehenden Datenverkehrs an den virtuellen Knoten `serviceB` und 25 Prozent des Datenverkehrs an den virtuellen Knoten `serviceBv2` zu leiten.

Schritt 6: Aktualisieren der Services

Nachdem Sie Ihr Mesh erstellt haben, müssen Sie die folgenden Aufgaben ausführen:

- Autorisieren Sie den Envoy-Proxy, den Sie mit jeder Amazon ECS-Aufgabe bereitstellen, zum Lesen der Konfiguration eines oder mehrerer virtueller Knoten. Weitere Informationen zum Autorisieren des Proxys finden Sie unter [Proxy-Autorisierung](#).
- Aktualisieren Sie jede Ihrer vorhandenen Amazon ECS-Aufgabendefinitionen, um den Envoy-Proxy zu verwenden.

Anmeldeinformationen

Der Envoy-Container benötigt AWS Identity and Access Management Anmeldeinformationen zum Signieren von Anfragen, die an den App Mesh Mesh-Dienst gesendet werden. Für Amazon ECS-Aufgaben, die mit dem Amazon EC2 EC2-Starttyp bereitgestellt werden, können die Anmeldeinformationen aus der [Instance-Rolle](#) oder aus einer [Task-IAM-Rolle](#) stammen. Amazon ECS-Aufgaben, die mit Fargate auf Linux-Containern bereitgestellt werden, haben keinen Zugriff auf den Amazon EC2-Metadatenserver, der die Anmeldeinformationen für das Instance-IAM-Profil bereitstellt. Um die Anmeldeinformationen bereitzustellen, müssen Sie allen Aufgaben, die mit dem Containertyp Fargate on Linux bereitgestellt werden, eine IAM-Aufgabenrolle zuordnen.

Wenn eine Aufgabe mit dem Amazon EC2-Starttyp bereitgestellt wird und der Zugriff auf den Amazon EC2-Metadatenserver blockiert ist, wie in der Anmerkung Wichtig in [IAM-Rolle für Aufgaben](#) beschrieben, muss der Aufgabe auch eine Aufgaben-IAM-Rolle zugewiesen werden. [Der Rolle, die Sie der Instance oder Aufgabe zuweisen, muss eine IAM-Richtlinie zugeordnet sein, wie unter Proxy-Autorisierung beschrieben.](#)

Um Ihre Aufgabendefinitionen mit dem zu aktualisieren AWS Management Console

In den folgenden Schritten wird nur die Aktualisierung der taskB-Aufgabe für das Szenario berücksichtigt. Sie müssen auch die Aufgaben taskBv2 und taskA aktualisieren, indem Sie die Werte entsprechend ändern.

1. Öffnen Sie die Konsole unter <https://console.aws.amazon.com/ecs/v2>.
2. Um auf die klassische Amazon ECS-Konsole zuzugreifen, klicken Sie oben links auf New ECS Experience.

 Important

Die App Mesh Mesh-Integration ist nur in der klassischen Amazon ECS-Konsole verfügbar.

3. Wählen Sie auf der Navigationsleiste die Region aus, in der Ihre Aufgabendefinition enthalten ist.
4. Wählen Sie im Navigationsbereich Task Definitions aus.
5. Aktivieren Sie auf der Seite Task Definitions (Aufgabendefinitionen) das Kontrollkästchen links neben der Aufgabendefinition, um diese zu ändern. Aus den Voraussetzungen und vorherigen Schritten haben Sie möglicherweise Aufgabendefinitionen mit den Namen taskA, taskB und taskBv2. Wählen Sie taskB, und wählen Sie Create new revision (Neue Version erstellen).
6. Nehmen Sie auf der Seite Neue Version der Aufgabendefinition erstellen die folgenden Änderungen vor, um die App Mesh Mesh-Integration zu aktivieren.
 - a. Um die Parameter für die App Mesh Mesh-Integration zu konfigurieren, wählen Sie App Mesh Mesh-Integration aktivieren und gehen Sie dann wie folgt vor:
 - i. Wählen Sie unter Name des Anwendungscontainers den Namen des Containers aus, der für die App Mesh Mesh-Anwendung verwendet werden soll. Dieser Container muss innerhalb der Aufgabendefinition bereits definiert sein.
 - ii. Führen Sie für Envoy image die folgende Aufgabe aus und geben Sie den zurückgegebenen Wert ein.

Alle [unterstützten](#) Regionen außer `me-south-1`, `ap-east-1`, `ap-southeast-3`, `eu-south-1`, `il-central-1`, und `af-south-1`. Sie können den *Regionalcode* durch eine beliebige Region `me-south-1`, `ap-east-1`, `ap-southeast-3`, `eu-south-1`, `il-central-1`, und ersetzen. `af-south-1`

Standard

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod
```

FIPS-konform

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod-fips
```

me-south-1

Standard

```
772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod
```

FIPS-konform

```
772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod-fips
```

ap-east-1

Standard

```
856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod
```

FIPS-konform

```
856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod-fips
```

ap-southeast-3

Standard

```
909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod
```

FIPS-konform

```
909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod-fips
```

eu-south-1

Standard

```
422531588944.dkr.ecr.eu-south-1.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod
```

FIPS-konform

```
422531588944.dkr.ecr.eu-south-1.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod-fips
```

il-central-1

Standard

```
564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod
```

FIPS-konform

```
564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod-fips
```

af-south-1

Standard

```
924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod
```

FIPS-konform

```
924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-  
envoy:v1.27.3.0-prod-fips
```

Public repository

Standard

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

Important

Nur Version v1.9.0.0-prod oder höher wird für die Verwendung mit App Mesh unterstützt.

- iii. Wählen Sie als Mesh-Name das zu verwendende App Mesh Service Mesh aus. In diesem Thema ist der Name des erstellten Meshes apps .
- iv. Wählen Sie unter Virtual node name (Name des virtuellen Knotens) den zu verwendenden virtuellen App Mesh-Knoten aus. Beispielsweise wählen Sie für die Aufgabe taskB den virtuellen Knoten serviceB aus, den Sie in einem vorherigen Schritt erstellt haben.
- v. Der Wert für den Virtual Node Port (Port des virtuellen Knotens) wird mit dem Listener-Port ausgefüllt, den Sie beim Erstellen des virtuellen Knotens angegeben haben.
- vi. Wählen Sie Apply (Anwenden) und dann Close (Schließen). Ein neuer Envoy-Proxy-Container wird erstellt und der Aufgabendefinition hinzugefügt, und die Einstellungen zur Unterstützung des Containers werden ebenfalls erstellt. Der Envoy-Proxy-Container

füllt dann die App Mesh Mesh-Proxy-Konfigurationseinstellungen für den nächsten Schritt vorab aus.

- b. Überprüfen Sie unter Proxy Configuration (Proxykonfiguration) alle vorausgefüllten Werte.
 - c. Stellen Sie sicher, dass der Netzwerkmodus ausgewählt ist `awsvpc`. Weitere Informationen zum `awsvpc`-Netzwerkmodus finden Sie unter [Task Networking mit dem the `awsvpc`-Netzwerkmodus](#).
7. Wählen Sie Erstellen.
 8. Aktualisieren Sie Ihren Service mit der aktualisierten Aufgabendefinition. Weitere Informationen finden Sie unter [Aktualisieren eines Service](#).

Die Konsole erstellt die JSON-Spezifikation der Aufgabendefinition. Sie können einige Einstellungen ändern, andere jedoch nicht. Weitere Informationen finden Sie im folgenden Abschnitt.

Aufgabendefinition (json)

Proxykonfiguration

Um Ihren Amazon ECS-Service für die Verwendung von App Mesh zu konfigurieren, muss die Aufgabendefinition Ihres Services den folgenden Abschnitt zur Proxykonfiguration enthalten. Stellen Sie die Proxy-Konfiguration `type` auf `APPMESH` und `containerName` auf `envoy` ein. Legen Sie die folgenden Eigenschaftswerte entsprechend fest.

IgnoredUID

Der Envoy-Proxy leitet keinen Datenverkehr von Prozessen weiter, die diese Benutzer-ID verwenden. Sie können eine beliebige Benutzer-ID für diesen Eigenschaftswert auswählen, diese ID muss jedoch mit der `user`-ID für den Envoy-Container in der Aufgabendefinition übereinstimmen. Diese Übereinstimmung ermöglicht Envoy, eigenen Datenverkehr zu ignorieren, ohne den Proxy zu verwenden. Unsere Beispiele verwenden `1337` für historische Zwecke.

ProxyIngressPort

Dies ist der eingehende Port für den Envoy-Proxycontainer. Legen Sie diesen Wert auf `15000` fest.

ProxyEgressPort

Dies ist der ausgehende Port für den Envoy-Proxycontainer. Legen Sie diesen Wert auf `15001` fest.

AppPorts

Geben Sie alle eingehenden Ports an, auf die Ihre Anwendungscontainer warten. In diesem Beispiel wird vom Anwendungs-Container Port **9080** überwacht. Der angegebene Port muss mit dem Port übereinstimmen, der auf dem virtuellen Knoten-Listener konfiguriert ist.

EgressIgnoredIPs

Envoy führt keine Proxy-Weiterleitung an diese IP-Adressen durch. Setzen Sie diesen Wert auf `169.254.170.2,169.254.169.254`, wodurch der Amazon EC2-Metadatenserver und der Amazon ECS-Aufgabenmetadaten-Endpunkt ignoriert werden. Der Metadaten-Endpunkt stellt IAM-Rollen für Aufgabenanmeldedaten bereit. Sie können zusätzliche Adressen hinzufügen.

EgressIgnoredPorts

Sie können eine Liste der Ports (mit Kommas als Trennzeichen) hinzufügen. Envoy führt keine Proxy-Weiterleitung an diese Ports durch. Auch wenn Sie keine Ports auflisten, wird Port 22 ignoriert.

Note

Die maximale Anzahl von ausgehenden Ports, die ignoriert werden können, ist 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"
  }
]
}
```

Envoy-Abhängigkeit des Anwendungs-Containers

Die Anwendungs-Container in Ihren Aufgabendefinitionen müssen mit dem Starten warten, bis der Envoy-Proxy den Bootstrap-Vorgang durchgeführt hat und gestartet wurde. Um sicherzustellen, dass dies geschieht, legen Sie in jeder Anwendungscontainer-Definition einen `dependsOn` Abschnitt fest, der darauf wartet, dass sich der Envoy-Container als `HEALTHY` meldet. Der folgende Codeblock zeigt ein Beispiel für eine Anwendungs-Containerdefinition mit dieser Abhängigkeit. Alle Eigenschaften im folgenden Beispiel sind erforderlich. Einige der Eigenschaftswerte sind ebenfalls erforderlich, aber einige sind *austauschbar*.

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

Envoy-Containerdefinition

Ihre Amazon ECS-Aufgabendefinitionen müssen ein App Mesh Envoy-Container-Image enthalten.

Alle [unterstützten](#) Regionen außer `me-south-1`, `ap-east-1`, `ap-southeast-3`, `eu-south-1`, `il-central-1`, und `af-south-1`. Sie können den *Regionalcode* durch eine beliebige Region `me-south-1`, `ap-east-1`, `ap-southeast-3`, `eu-south-1`, `il-central-1`, und ersetzen. `af-south-1`

Standard

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

me-south-1

Standard

```
772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

ap-east-1

Standard

```
856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

ap-southeast-3

Standard

```
909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

eu-south-1

Standard

```
422531588944.dkr.ecr.eu-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
422531588944.dkr.ecr.eu-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

il-central-1

Standard

```
564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

af-south-1

Standard

```
924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

Public repository

Standard

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

Important

Nur Version v1.9.0.0-prod oder höher wird für die Verwendung mit App Mesh unterstützt.

Sie müssen das App Mesh Envoy-Container-Image verwenden, bis das Envoy-Projektteam Änderungen zusammenführt, die App Mesh unterstützen. [Weitere Informationen finden Sie in der GitHub Roadmap-Ausgabe.](#)

Alle Eigenschaften im folgenden Beispiel sind erforderlich. Einige der Eigenschaftswerte sind ebenfalls erforderlich, aber einige sind *austauschbar*.

Note

- Die Envoy-Containerdefinition muss als `essential` gekennzeichnet sein.
- Wir empfehlen, dem Envoy-Container 512 CPU-Einheiten und mindestens 64 MiB Arbeitsspeicher zuzuweisen. Bei Fargate ist der niedrigste Wert, den Sie einstellen können, 1024 MiB Arbeitsspeicher.
- Der Name des virtuellen Knotens für den Amazon ECS-Service muss auf den Wert der `APPMESH_RESOURCE_ARN` Eigenschaft gesetzt werden. Für diese Eigenschaft ist eine Version `1.15.0` oder eine neuere Version des Envoy-Images erforderlich. Weitere Informationen finden Sie unter [Envoy](#).
- Der Wert für die `user`-Einstellung muss mit dem `IgnoredUID`-Wert aus der Proxykonfiguration der Aufgabendefinition übereinstimmen. In diesem Beispiel verwenden wir `1337`.
- Die hier gezeigte Integritätsprüfung wartet, bis der Envoy-Container ordnungsgemäß gestartet wurde, bevor Amazon ECS gemeldet wird, dass der Envoy-Container fehlerfrei und bereit für den Start der Anwendungscontainer ist.
- Standardmäßig verwendet App Mesh den Namen der Ressource, die Sie in `APPMESH_RESOURCE_ARN` angegeben haben, wenn sich Envoy in Metriken und Ablaufverfolgungen auf sich selbst bezieht. Sie können dieses Verhalten übergehen, indem Sie die `APPMESH_RESOURCE_CLUSTER`-Umgebungsvariable mit Ihrem eigenen Namen

festlegen. Für diese Eigenschaft ist eine Version 1.15.0 oder eine neuere Version des Envoy-Images erforderlich. Weitere Informationen finden Sie unter [Envoy](#).

Der folgende Codeblock zeigt ein Beispiel einer Envoy-Containerdefinition.

```
{
  "name": "envoy",
  "image": "840364872350.dkr.ecr.us-west-2.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-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"
}
```

Beispiel für Aufgabendefinitionen

Die folgenden Amazon ECS-Beispielaufgabendefinitionen zeigen, wie Sie die obigen Beispiele zu einer Aufgabendefinition für zusammenführen können `taskB`. Es werden Beispiele für die Erstellung von Aufgaben für beide Amazon ECS-Starttypen mit oder ohne Verwendung bereitgestellt AWS X-Ray. Ändern Sie ggf. die *ersetzbaren* Werte, um Aufgabendefinitionen für die benannten Aufgaben `taskBv2` und `taskA` aus dem Szenario zu erstellen. Geben Sie in Ihrer Proxykonfiguration als Wert für `APPMESH_RESOURCE_ARN` den Namen Ihres Mesh und den Namen des virtuellen Knotens an. Geben Sie zudem als Wert für `AppPorts` eine Liste der Ports an, die von Ihrer Anwendung überwacht werden. Standardmäßig verwendet App Mesh den Namen der Ressource, die Sie in `APPMESH_RESOURCE_ARN` angegeben haben, wenn sich Envoy in Metriken und Ablaufverfolgungen auf sich selbst bezieht. Sie können dieses Verhalten übergehen, indem Sie die `APPMESH_RESOURCE_CLUSTER`-Umgebungsvariable mit Ihrem eigenen Namen festlegen. Alle

Eigenschaften in den folgenden Beispielen sind erforderlich. Einige der Eigenschaftswerte sind ebenfalls erforderlich, aber einige sind *austauschbar*.

Wenn Sie eine Amazon ECS-Aufgabe ausführen, wie im Abschnitt Anmeldeinformationen beschrieben, müssen Sie den Beispielen eine bestehende [Task-IAM-Rolle](#) hinzufügen.

⚠ Important

Fargate muss einen Portwert verwenden, der größer als 1024 ist.

Example JSON für Amazon ECS-Aufgabendefinition — Fargate auf Linux-Containern

```
{
  "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:v1.27.3.0-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" : "awsipc"  
}
```

Example JSON für Amazon ECS-Aufgabendefinition mit AWS X-Ray — Fargate auf Linux-Containern

X-Ray ermöglicht es Ihnen, Daten über Anfragen zu sammeln, die eine Anwendung bedient, und stellt Tools bereit, mit denen Sie den Verkehrsfluss visualisieren können. Durch die Verwendung des X-Ray-Treibers für Envoy kann Envoy Tracing-Informationen an X-Ray melden. Sie können X-Ray Tracing mit der [Envoy-Konfiguration](#) aktivieren. Je nach Konfiguration sendet Envoy Tracing-Daten an den X-Ray-Daemon, der als [Sidecar-Container](#) läuft, und der Daemon leitet die Traces an den X-Ray-Dienst weiter. Sobald die Traces auf X-Ray veröffentlicht wurden, können Sie die X-Ray-Konsole verwenden, um das Diagramm der Serviceanrufe zu visualisieren und Trace-Details anzufordern. Das folgende JSON stellt eine Aufgabendefinition zur Aktivierung der X-Ray-Integration dar.

```
{  
  
  "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:v1.27.3.0-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 für Amazon ECS-Aufgabendefinition — EC2-Starttyp

```

{
  "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:v1.27.3.0-prod",
  "essential": true,
  "environment": [
    {
      "name": "APPMESH_VIRTUAL_NODE_NAME",
      "value": "mesh/apps/virtualNode/serviceB"
    }
  ],
  "healthCheck": {
    "command": [
      "CMD-SHELL",
      "curl -s http://localhost:9901/server_info | grep state | grep -q LIVE"
    ],
    "startPeriod": 10,
    "interval": 5,
    "timeout": 2,
    "retries": 3
  },
  "user": "1337"
}
],
"requiresCompatibilities" : [ "EC2" ],
"taskRoleArn" : "arn:aws:iam::123456789012:role/ecsTaskRole",
"executionRoleArn" : "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
"networkMode": "awsvpc"
}

```

Example JSON für Amazon ECS-Aufgabendefinition mit AWS X-Ray EC2-Starttyp

```

{
  "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:v1.27.3.0-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"  
}
```

Fortschrittliche Themen

Kanarische Bereitstellungen mit App Mesh

Mithilfe von Bereitstellungen und Releases auf Canary können Sie den Datenverkehr zwischen einer alten Version einer Anwendung und einer neu bereitgestellten Version umschalten. Außerdem wird der Zustand der neu bereitgestellten Version überwacht. Falls es Probleme mit der neuen Version gibt, kann das Canary-Deployment den Traffic automatisch wieder auf die alte Version umschalten. Mit Bereitstellungen auf Canary hast du die Möglichkeit, den Datenverkehr zwischen den Anwendungsversionen mit mehr Kontrolle umzuschalten.

Weitere Informationen zur Implementierung von Canary-Bereitstellungen für Amazon ECS mithilfe von App Mesh finden [Sie unter Erstellen einer Pipeline mit Canary-Bereitstellungen für Amazon ECS mithilfe von App Mesh](#)

Note

Weitere Beispiele und Komplettlösungen für App Mesh finden Sie im App Mesh [Mesh-Beispiel-Repository](#).

Erste Schritte mit AWS App Mesh und Kubernetes

Wenn Sie mit dem App-Mesh-Controller für Kubernetes AWS App Mesh in Kubernetes integrieren, verwalten Sie App-Mesh-Ressourcen wie Gitter, virtuelle Services, virtuelle Knoten, virtuelle Router und Routen über Kubernetes. Sie fügen die App Mesh-Sidecar-Container-Images auch automatisch zu den Kubernetes-Pod-Spezifikationen hinzu. Dieses Tutorial führt Sie durch die Installation des App-Mesh-Controllers für Kubernetes, um diese Integration zu ermöglichen.

Der Controller geht mit der Bereitstellung der folgenden benutzerdefinierten Kubernetes-Ressourcendefinitionen einher: `meshes`, `virtual services`, `virtual nodes` und `virtual`

routers. Der Controller überwacht die Erstellung, Änderung und Löschung der benutzerdefinierten Ressourcen und nimmt Änderungen an den entsprechenden App-Mesh-[the section called “Gitter”](#), [the section called “Virtuelle Services”](#)-, [the section called “Virtuelle Knoten”](#)-, [the section called “Virtuelle Gateways”](#)-, [the section called “Gateway-Routen”](#), [the section called “Virtuelle Router”](#) (einschließlich [the section called “Router”](#)) Ressourcen über die App-Mesh-API vor. Weitere Informationen oder einen Beitrag zum Controller finden Sie im [GitHub Projekt](#) .

Der Controller installiert auch einen Webhook, der die folgenden Container in Kubernetes-Pods injiziert, die mit einem von Ihnen angegebenen Namen beschriftet sind.

- App-Mesh-Envoy-Proxy – Envoy verwendet die in der App-Mesh-Steuerebene definierte Konfiguration, um zu bestimmen, wohin Ihr Anwendungsdatenverkehr gesendet werden soll.
- App-Mesh-Proxy-Routenmanager – Aktualisiert iptables Regeln im Netzwerk-Namespace eines Pods, die ein- und ausgehenden Datenverkehr über Envoy weiterleiten. Dieser Container wird als Kubernetes-Init-Container innerhalb des Pods ausgeführt.

Voraussetzungen

- Ein vorhandenes Verständnis von App Mesh-Konzepten. Weitere Informationen finden Sie unter [Was ist AWS App Mesh?](#).
- Vorhandene Grundlagen zu Kubernetes-Konzepten. Weitere Informationen finden Sie unter [Was ist Kubernetes?](#) in der Kubernetes-Dokumentation.
- Ein vorhandener Kubernetes-Cluster. Wenn Sie noch keinen Cluster haben, finden Sie weitere Informationen unter [Erste Schritte mit Amazon EKS](#) im Amazon-EKS-Benutzerhandbuch. Wenn Sie Ihren eigenen Kubernetes-Cluster auf Amazon EC2 ausführen, stellen Sie sicher, dass Docker bei dem Amazon-ECR-Repository authentifiziert ist, in dem sich das Envoy-Image befindet. Weitere Informationen finden Sie unter [Envoy image](#) , [Registry authentication](#) im Amazon Elastic Container Registry-Benutzerhandbuch und [Pull an Image from a Private Registry](#) in der Kubernetes-Dokumentation.
- App Mesh unterstützt Linux-Services, die bei DNS AWS Cloud Map oder bei dem registriert sind. Um dieses Handbuch für Erste Schritte zu verwenden, empfehlen wir, dass Sie über drei vorhandene Services verfügen, die bei DNS registriert sind. Bei den Verfahren in diesem Thema wird davon ausgegangen, dass die vorhandenen Services den Namen `serviceA`, `serviceB` und `serviceC` haben und dass alle Services über einen Namespace `namespace` auffindbar sind `namespace.local`.

Sie können ein Service-Mesh und seine Ressourcen erstellen, auch wenn die Services nicht vorhanden sind. Sie können das Mesh jedoch erst verwenden, wenn Sie tatsächliche Services bereitgestellt haben.

- Die installierte AWS CLI Version 1.18.116 oder höher oder 2.0.38 oder höher. Informationen zum Installieren oder Aktualisieren der AWS CLI finden Sie unter [Installieren der AWS CLI](#).
- Einen `kubectl`-Client, der für die Kommunikation mit Ihrem Kubernetes-Cluster konfiguriert ist. Wenn Sie Amazon Elastic Kubernetes Service verwenden, können Sie die Anweisungen zum Installieren [kubectl](#) und Konfigurieren einer [kubeconfig](#) Datei verwenden.
- Helm Version 3.0 oder höher ist installiert. Wenn Sie Helm nicht installiert haben, finden Sie weitere Informationen unter [Verwenden von Helm mit Amazon EKS](#) im Amazon-EKS-Benutzerhandbuch.
- Amazon EKS unterstützt derzeit nur IPv4_ONLY und IPv6_ONLY nur IP-Einstellungen, da Amazon EKS derzeit nur Pods unterstützt, die entweder nur IPv4 Datenverkehr oder nur IPv6 Datenverkehr bedienen können.

Die verbleibenden Schritte gehen davon aus, dass die tatsächlichen Services als `serviceA`, `serviceB` und `serviceBv2` benannt sind, und dass alle Services über einen Namespace mit dem Namen `apps.local` gefunden werden können.

Schritt 1: Installieren der Integrationskomponenten

Installieren Sie die Integrationskomponenten einmal auf jedem Cluster, der Pods hostet, die Sie mit App Mesh verwenden möchten.

So installieren Sie die Integrationskomponenten

1. Für die verbleibenden Schritte dieses Verfahrens ist ein Cluster erforderlich, ohne dass eine Vorabversion des Controllers installiert ist. Wenn Sie eine Vorabversion installiert haben oder nicht sicher sind, ob Sie diese haben, können Sie ein Skript herunterladen und ausführen, das prüft, ob eine Vorabversion auf Ihrem Cluster installiert ist.

```
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
```

Wenn das Skript `Your cluster is ready for upgrade. Please proceed to the installation instructions` zurückgibt, können Sie mit dem nächsten Schritt fortfahren.

Wenn eine andere Meldung zurückgegeben wird, müssen Sie die Upgrade-Schritte ausführen, bevor Sie fortfahren. Weitere Informationen zum Aktualisieren einer Vorabversion finden Sie unter [Upgrade](#) auf GitHub.

2. Fügen Sie das eks-charts-Repository Helm hinzu.

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

3. Installieren Sie die benutzerdefinierte Ressourcendefinitionen (CRD) von App Mesh Kubernetes.

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

4. Erstellen Sie einen Kubernetes-Namespace für den Controller.

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

5. Legen Sie die folgenden Variablen für die Verwendung in späteren Schritten fest. Ersetzen Sie *cluster-name* und *Region-code* durch die Werte für Ihren vorhandenen Cluster.

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

6. (Optional) Wenn Sie den Controller auf Fargate ausführen möchten, müssen Sie ein Fargate-Profil erstellen. Wenn Sie nicht eksctl installiert haben, finden Sie weitere Informationen unter [Installieren oder Aktualisieren eksctl](#) im Amazon-EKS-Benutzerhandbuch. Wenn Sie das Profil lieber mit der Konsole erstellen möchten, finden Sie weitere Informationen unter [Erstellen eines Fargate-Profiles](#) im Amazon-EKS-Benutzerhandbuch.

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

7. Erstellen Sie einen OpenID Connect (OIDC)-Identitätsanbieter für Ihren Cluster. Wenn Sie nicht eksctl installiert haben, können Sie es mit den Anweisungen unter [Installieren oder Aktualisieren eksctl](#) im Amazon-EKS-Benutzerhandbuch installieren. Wenn Sie den Anbieter lieber über die Konsole erstellen möchten, finden Sie weitere Informationen unter [Aktivieren von IAM-Rollen für Servicekonten in Ihrem Cluster](#) im Amazon-EKS-Benutzerhandbuch.

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

```
--approve
```

- Erstellen Sie eine IAM-Rolle, fügen Sie ihr die [AWSCloudMapFullAccess](#) AWS verwalteten Richtlinien [AWSAppMeshFullAccess](#) und an und binden Sie sie an das `appmesh-controller` Kubernetes-Servicekonto an. Die Rolle ermöglicht es dem Controller, App Mesh-Ressourcen hinzuzufügen, zu entfernen und zu ändern.

 Note

Der Befehl erstellt eine AWS IAM-Rolle mit einem automatisch generierten Namen. Sie können den erstellten IAM-Rollennamen nicht angeben.

```
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
```

Wenn Sie das Servicekonto lieber mit der AWS Management Console oder erstellen möchten AWS CLI, finden Sie weitere Informationen unter [Erstellen einer IAM-Rolle und -Richtlinie für Ihr Servicekonto](#) im Amazon-EKS-Benutzerhandbuch. Wenn Sie die AWS Management Console oder verwenden AWS CLI , um das Konto zu erstellen, müssen Sie die Rolle auch einem Kubernetes-Servicekonto zuordnen. Weitere Informationen finden Sie unter [Angeben einer IAM-Rolle für Ihr Servicekonto](#) im Amazon-EKS-Benutzerhandbuch.

- Stellen Sie den App-Mesh-Controller bereit. Eine Liste aller Konfigurationsoptionen finden Sie unter [Konfiguration](#) auf GitHub.
 - Um den App-Mesh-Controller für einen privaten Cluster bereitzustellen, müssen Sie zunächst App Mesh und Amazon-VPC-Endpunkte für die Serviceerkennung im verknüpften privaten Subnetz aktivieren. Sie müssen auch festlegen `accountId`.

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

Um die X-Ray-Ablaufverfolgung in einem privaten Cluster zu aktivieren, aktivieren Sie die Amazon-VPC-Endpunkte von X-Ray und Amazon ECR. Der Controller verwendet `public.ecr.aws/xray/aws-xray-daemon:latest` standardmäßig. Rufen Sie dieses Image daher auf lokal ab und [übertragen Sie es in Ihr persönliches ECR-Repository](#).

 Note

[Amazon-VPC-Endpunkte](#) unterstützen derzeit keine öffentlichen Amazon-ECR-Repositories.

Das folgende Beispiel zeigt die Bereitstellung des Controllers mit Konfigurationen für 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
```

Überprüfen Sie, ob der X-Ray-Daemon erfolgreich injiziert wurde, wenn Sie die Anwendungsbereitstellung mit Ihrem virtuellen Knoten oder Gateway binden.

Weitere Informationen finden Sie unter [Private Cluster](#) im Amazon-EKS-Benutzerhandbuch.

2. Stellen Sie den App-Mesh-Controller für andere Cluster bereit. Eine Liste aller Konfigurationsoptionen finden Sie unter [Konfiguration](#) auf 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

Wenn Ihre Amazon-EKS-Clusterfamilie IPv6 ist, legen Sie bei der Bereitstellung des App-Mesh-Controllers den Clusternamen fest, indem Sie dem vorherigen Befehl die folgende Option hinzufügen `--set clusterName=$CLUSTER_NAME`.

Wichtig

Wenn sich Ihr Cluster in den `af-south-1` Regionen `me-south-1`, `ap-east-1`, `ap-southeast-3`, `eu-south-1` oder `il-central-1` befindet, müssen Sie dem vorherigen Befehl die folgende Option hinzufügen:

Ersetzen Sie *account-id* und *Region-code* durch einen der entsprechenden Wertesätze.

- Für das Sidecar-Image:

- ```
--set image.repository=account-id.dkr.ecr.Region-code.amazonaws.com/
amazon/appmesh-controller
```
- `772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod`
- `856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod`
- `909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod`
- `422531588944.dkr.ecr.eu-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod`
- `564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod`
- `924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod`
- Die älteren Image-URIs finden Sie im [Änderungsprotokoll](#) auf GitHub. Die AWS Konten, auf denen die Images vorhanden sind, haben sich in Version geändert `v1.5.0`. Ältere Versionen der Images werden auf AWS Konten gehostet, die

sich in den Amazon Elastic Kubernetes Service [Amazon Container Image-Registries](#) befinden.

- Für das Controller-Image:

- ```
--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:v1.12.3
- 856666278305.dkr.ecr.ap-east-1.amazonaws.com/amazon/appmesh-controller:v1.12.3
- 909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/amazon/appmesh-controller:v1.12.3
- 422531588944.dkr.ecr.eu-south-1.amazonaws.com/amazon/appmesh-controller:v1.12.3
- 564877687649.dkr.ecr.il-central-1.amazonaws.com/amazon/appmesh-controller:v1.12.3
- 924023996002.dkr.ecr.af-south-1.amazonaws.com/amazon/appmesh-controller:v1.12.3

- Für das Sidecar-Init-Image:

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

- 924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-proxy-route-manager:v7-prod

### Important

Nur Version v1.9.0.0-prod oder höher wird für die Verwendung mit App Mesh unterstützt.

10. Vergewissern Sie sich, dass die Controller-Version v1.4.0 oder höher ist. Sie können das [Änderungsprotokoll](#) auf überprüfen GitHub.

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

### Note

Wenn Sie das Protokoll für den laufenden Container anzeigen, wird möglicherweise eine Zeile angezeigt, die den folgenden Text enthält. Dies kann gefahrlos ignoriert werden kann.

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

## Schritt 2: Bereitstellen von App-Mesh-Ressourcen

Wenn Sie eine Anwendung in Kubernetes bereitstellen, erstellen Sie auch die benutzerdefinierten Kubernetes-Ressourcen, sodass der Controller die entsprechenden App Mesh-Ressourcen erstellen kann. Das folgende Verfahren hilft Ihnen bei der Bereitstellung von App Mesh-Ressourcen mit einigen ihrer Funktionen. Beispielmanifeste für die Bereitstellung anderer App-Mesh-Ressourcenfunktionen finden Sie in den v1beta2 Unterordnern vieler Feature-Ordner, die unter [App-Mesh-Walkthroughs](#) aufgeführt sind GitHub.

**⚠ Important**

Sobald der Controller eine App-Mesh-Ressource erstellt hat, empfehlen wir Ihnen, nur Änderungen an der App-Mesh-Ressource mit dem Controller vorzunehmen oder diese zu löschen. Wenn Sie Änderungen an der Ressource mit App Mesh vornehmen oder löschen, ändert oder erstellt der Controller die geänderte oder gelöschte App-Mesh-Ressource standardmäßig nicht für zehn Stunden. Sie können diese Dauer so konfigurieren, dass sie geringer ist. Weitere Informationen finden Sie unter [Konfiguration](#) auf GitHub.

So stellen Sie App-Mesh-Ressourcen bereit

1. Erstellen Sie einen Kubernetes-Namespace, in dem App Mesh-Ressourcen bereitgestellt werden sollen.
  - a. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `namespace.yaml` auf Ihrem Computer.

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

- b. Erstellen Sie den Namespace.

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

2. Erstellen Sie ein App Mesh-Servicegitter.
  - a. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `mesh.yaml` auf Ihrem Computer. Die Datei wird verwendet, um eine Mesh-Ressource mit dem Namen zu erstellen *my-mesh*. Ein Service Mesh ist eine logische Begrenzung für den Netzwerkdatenverkehr zwischen den darin vorhandenen Services.

```
apiVersion: appmesh.k8s.aws/v1beta2
kind: Mesh
metadata:
```

```

name: my-mesh
spec:
 namespaceSelector:
 matchLabels:
 mesh: my-mesh

```

- b. Erzeugen Sie das Mesh.

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

- c. Zeigen Sie die Details der erstellten Kubernetes-Mesh-Ressource an.

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

## Output

```

Name: my-mesh
Namespace:
Labels: <none>
Annotations: kubect1.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. Zeigen Sie die Details zum App Mesh-Servicegitter an, das der Controller erstellt hat.

```
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. Erstellen Sie einen virtuellen App-Mesh-Knoten. Ein virtueller Knoten fungiert als logischer Zeiger auf eine Kubernetes-Bereitstellung.
- a. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `virtual-node.yaml` auf Ihrem Computer. Die `-Datei` wird verwendet, um einen virtuellen App-Mesh-Knoten mit dem Namen `my-service-a` im `my-apps` Namespace zu erstellen. Der virtuelle Knoten stellt einen Kubernetes-Service dar, der in einem späteren Schritt erstellt wird. Der Wert für `hostname` ist der vollqualifizierte DNS-Hostname des eigentlichen Services, den dieser virtuelle Knoten darstellt.

```
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
```

Virtuelle Knoten verfügen über Funktionen wie end-to-end Verschlüsselung und Zustandsprüfungen, die in diesem Tutorial nicht behandelt werden. Weitere Informationen finden Sie unter [the section called “Virtuelle Knoten”](#). Führen Sie den folgenden Befehl aus, um alle verfügbaren Einstellungen für einen virtuellen Knoten anzuzeigen, die Sie in der vorherigen Spezifikation festlegen können.

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

- b. Stellen Sie den virtuellen Knoten bereit.

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

- c. Zeigen Sie die Details der erstellten virtuellen Knotenressource von Kubernetes an.

```
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. Zeigen Sie die Details des virtuellen Knotens an, den der Controller in App Mesh erstellt hat.

 Note

Obwohl der Name des in Kubernetes erstellten virtuellen Knotens *my-service-a* lautet, hat der in App Mesh erstellte virtuelle Knoten den Namen *my-service-a\_my-apps*. Der Controller hängt den Kubernetes-Namespace-Namen an den

virtuellen App-Mesh-Knotennamen an, wenn er die App-Mesh-Ressource erstellt. Der Namespace-Name wird hinzugefügt, da Sie in Kubernetes virtuelle Knoten mit demselben Namen in verschiedenen Namespaces erstellen können, aber in App Mesh muss ein virtueller Knotenname innerhalb eines Gitters eindeutig sein.

```
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. Erstellen Sie einen virtuellen App Mesh-Router. Virtuelle Router verarbeiten den Datenverkehr für einen oder mehrere virtuelle Services innerhalb Ihres Gitters.
  - a. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `virtual-router.yaml` auf Ihrem Computer. Die Datei wird verwendet, um einen virtuellen Router zu erstellen, der den Datenverkehr an den virtuellen Knoten namens `weiterleitemy-service-a`, der im vorherigen Schritt erstellt wurde. Der Controller erstellt den virtuellen App Mesh-Router und leitet Ressourcen weiter. Sie können viele weitere Funktionen für Ihre Routen angeben und andere Protokolle als `http` verwenden. Weitere Informationen finden Sie unter [the section called "Virtuelle Router"](#) und [the section called "Routen"](#). Beachten Sie, dass der Name des virtuellen Knotens, auf den verwiesen wird, der Name des virtuellen Kubernetes-Knotens ist, nicht der Name des virtuellen App-Mesh-Knotens, der vom Controller in App Mesh erstellt wurde.

```
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
```

(Optional) Um alle verfügbaren Einstellungen für einen virtuellen Router anzuzeigen, die Sie in der vorherigen Spezifikation festlegen können, führen Sie den folgenden Befehl aus.

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

Führen Sie den folgenden Befehl aus, um alle verfügbaren Einstellungen für eine Route anzuzeigen, die Sie in der vorherigen Spezifikation festlegen können.

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

- b. Stellen Sie den virtuellen Router bereit.

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

- c. Zeigen Sie die virtuelle Routerressource von Kubernetes an, die erstellt wurde.

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

### Gekürzte Ausgabe

```
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 ARNs:
 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. Zeigen Sie die virtuelle Routerressource an, die der Controller in App Mesh erstellt hat. Sie geben `my-service-a-virtual-router_my-apps` für `aname`, da der Controller beim Erstellen des virtuellen Routers in App Mesh den Kubernetes-namespace-Namen an den Namen des virtuellen Routers angehängt hat.

```
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. Zeigen Sie die Routenressource an, die der Controller in App Mesh erstellt hat. Eine Routenressource wurde in Kubernetes nicht erstellt, da die Route Teil der Konfiguration des virtuellen Routers in Kubernetes ist. Die Routeninformationen wurden in den Kubernetes-Ressourcendetails im Teilschritt c angezeigt. Der Controller hat den Kubernetes-Namespace-Namen nicht an den App-Mesh-Routennamen angehängt, als er die Route in App Mesh erstellt hat, da Routennamen für einen virtuellen Router eindeutig sind.

```

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. Erstellen Sie einen virtuellen App-Mesh-Service. Bei einem virtuellen Service handelt es sich um eine Abstraktion eines echten Service, der von einem virtuellen Knoten direkt oder indirekt mittels eines virtuellen Routers bereitgestellt wird. Abhängige Services rufen Ihren virtuellen Service über den Namen auf. Obwohl der Name für App Mesh keine Rolle spielt, empfehlen wir, dem virtuellen Service den vollqualifizierten Domännennamen des tatsächlichen Services zu benennen, den der virtuelle Service darstellt. Wenn Sie Ihre virtuellen Services auf diese Weise benennen, müssen Sie Ihren Anwendungscode nicht ändern, um auf einen anderen Namen zu verweisen. Die Anforderungen werden an den virtuellen Knoten oder virtuellen Router weitergeleitet, der als Anbieter für den virtuellen Service angegeben ist.
  - a. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `virtual-service.yaml` auf Ihrem Computer. Die Datei wird verwendet, um einen virtuellen Service zu erstellen, der einen virtuellen Routeranbieter verwendet, um Datenverkehr an den virtuellen Knoten namens `weiterzuleitenmy-service-a`, der in einem vorherigen Schritt

erstellt wurde. Der Wert für `awsName` in der `spec` ist der vollqualifizierte Domänenname (FQDN) des tatsächlichen Kubernetes-Services, den dieser virtuelle Service abstrahiert. Der Kubernetes-Service wird in [the section called “Schritt 3: Erstellen oder Aktualisieren von Services”](#) erstellt. Weitere Informationen finden Sie unter [the section called “Virtuelle Services”](#).

```
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
```

Führen Sie den folgenden Befehl aus, um alle verfügbaren Einstellungen für einen virtuellen Service anzuzeigen, die Sie in der vorherigen Spezifikation festlegen können.

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

- b. Erstellen Sie den virtuellen Service.

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

- c. Zeigen Sie die Details der virtuellen Serviceresource von Kubernetes an, die erstellt wurde.

```
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. Zeigen Sie die Details der virtuellen Service-Ressource an, die der Controller in App Mesh erstellt hat. Der Kubernetes-Controller hat den Kubernetes-Namespace-Namen nicht an den virtuellen App-Mesh-Servicenamen angehängt, als er den virtuellen Service in App Mesh erstellt hat, da der Name des virtuellen Services ein eindeutiger FQDN ist.

```

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"
 }
}
```

Obwohl in diesem Tutorial nicht behandelt, kann der Controller auch App Mesh [the section called “Virtuelle Gateways”](#) und bereitstellen [the section called “Gateway-Routen”](#). Eine exemplarische Vorgehensweise zur Bereitstellung dieser Ressourcen mit dem Controller finden Sie unter [Konfigurieren von eingehendem Gateway](#) oder ein [Beispielmanifest](#), das die Ressourcen auf enthält GitHub.

## Schritt 3: Erstellen oder Aktualisieren von Services

Allen Pods, die Sie mit App Mesh verwenden möchten, müssen die App-Mesh-Sidecar-Container hinzugefügt werden. Der Injektor fügt die Sidecar-Container automatisch jedem Pod hinzu, der mit einer von Ihnen angegebenen Beschriftung bereitgestellt wird.

1. Proxy-Autorisierung aktivieren. Wir empfehlen, dass Sie für jede Kubernetes-Bereitstellung aktivieren, um nur die Konfiguration für ihren eigenen virtuellen App-Mesh-Knoten zu streamen.

- a. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `proxy-auth.json` auf Ihrem Computer. Stellen Sie sicher, dass Sie die *alternativfarbigen Werte* durch Ihre eigenen ersetzen.

```
{
 "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. Erstellen Sie die Richtlinie.

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

- c. Erstellen Sie eine IAM-Rolle, fügen Sie ihr die im vorherigen Schritt erstellte Richtlinie an, erstellen Sie ein Kubernetes-Servicekonto und binden Sie die Richtlinie an das Kubernetes-Servicekonto an. Die Rolle ermöglicht es dem Controller, App Mesh-Ressourcen hinzuzufügen, zu entfernen und zu ändern.

```
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
```

Wenn Sie das Servicekonto lieber mit der AWS Management Console erstellen möchten, finden Sie weitere Informationen unter [Erstellen einer IAM-Rolle und -Richtlinie für Ihr Servicekonto](#) im Amazon-EKS-Benutzerhandbuch. Wenn Sie die AWS Management Console oder verwenden AWS CLI, um das Konto zu erstellen, müssen Sie

die Rolle auch einem Kubernetes-Servicekonto zuordnen. Weitere Informationen finden Sie unter [Angeben einer IAM-Rolle für Ihr Servicekonto](#) im Amazon-EKS-Benutzerhandbuch.

- (Optional) Wenn Sie Ihre Bereitstellung auf Fargate-Pods bereitstellen möchten, müssen Sie ein Fargate-Profil erstellen. Wenn Sie nicht `eksctl` installiert haben, können Sie es mit den Anweisungen unter [Installieren oder Aktualisieren eksctl](#) im Amazon-EKS-Benutzerhandbuch installieren. Wenn Sie das Profil lieber mit der Konsole erstellen möchten, finden Sie weitere Informationen unter [Erstellen eines Fargate-Profiles](#) im Amazon-EKS-Benutzerhandbuch.

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

- Erstellen Sie einen Kubernetes-Service und eine entsprechende Bereitstellung. Wenn Sie über eine vorhandene Bereitstellung verfügen, die Sie mit App Mesh verwenden möchten, müssen Sie einen virtuellen Knoten bereitstellen, wie im Unterschritt 3 von [the section called "Schritt 2: Bereitstellen von App-Mesh-Ressourcen"](#). Aktualisieren Sie Ihre Bereitstellung, um sicherzustellen, dass ihre Bezeichnung mit der Bezeichnung übereinstimmt, die Sie auf dem virtuellen Knoten festgelegt haben, sodass die Sidecar-Container automatisch zu den Pods hinzugefügt und die Pods erneut bereitgestellt werden.
  - Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `example-service.yaml` auf Ihrem Computer. Wenn Sie den Namespace-Namen ändern und Fargate-Pods verwenden, stellen Sie sicher, dass der Namespace-Name mit dem Namespace-Namen übereinstimmt, den Sie in Ihrem Fargate-Profil definiert haben.

```
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

Der Wert für die `app` `matchLabels` `selector` in der Spezifikation muss mit dem Wert übereinstimmen, den Sie beim Erstellen des virtuellen Knotens im Teilschritt 3 von [the section called “Schritt 2: Bereitstellen von App-Mesh-Ressourcen”](#) angegeben haben, anderenfalls werden die Sidecar-Container nicht in den Pod injiziert. Im vorherigen Beispiel lautet der Wert für die Beschriftung `my-app-1`. Wenn Sie anstelle eines virtuellen Knotens ein virtuelles Gateway bereitstellen, sollte das Deployment Manifest nur den Envoy-Container enthalten. Weitere Informationen zum zu verwendenden Image finden Sie unter [Envoy](#). Ein Beispiel-Manifest finden Sie im [Bereitstellungsbeispiel](#) auf GitHub.

- b. Bereitstellen des Services.

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

- c. Zeigen Sie den Service und die Bereitstellung an.

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

## Output

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

- d. Zeigen Sie die Details zu einem der bereitgestellten Pods an.

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

## Gekürzte Ausgabe

```
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

```

In der vorherigen Ausgabe können Sie sehen, dass die Container `proxyinit` und `envoy` vom Controller dem Pod hinzugefügt wurden. Wenn Sie den Beispielservice in Fargate bereitgestellt haben, wurde der `envoy` Container vom Controller zum Pod hinzugefügt, der `proxyinit` Container jedoch nicht.

- (Optional) Installieren Sie Add-Ons wie Prometheus, Grafana AWS X-Ray, Jaeger und Datadog. Weitere Informationen finden Sie unter [App-Mesh-Add-ons](#) auf GitHub und im Abschnitt [Beobachtbarkeit](#) des App-Mesh-Benutzerhandbuchs.

#### Note

Weitere Beispiele und exemplarische Vorgehensweisen für App Mesh finden Sie im [App Mesh-Beispiel-Repository](#).

## Schritt 4: Bereinigen

Entfernen Sie alle Beispielressourcen, die in diesem Tutorial erstellt wurden. Der Controller entfernt auch die Ressourcen, die im `my-mesh` App Mesh-Servicegitter erstellt wurden.

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

Wenn Sie ein Fargate-Profil für den Beispielservice erstellt haben, entfernen Sie es.

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

Löschen Sie das Mesh.

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

(Optional) Sie können die Kubernetes-Integrationskomponenten entfernen.

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

(Optional) Wenn Sie die Kubernetes-Integrationskomponenten in Fargate bereitgestellt haben, löschen Sie das Fargate-Profil.

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

## Erste Schritte mit AWS App Mesh und Amazon EC2

Dieses Thema hilft Ihnen bei der Verwendung von AWS App Mesh mit einem tatsächlichen -Service, der auf Amazon EC2 ausgeführt wird. In diesem Tutorial werden die grundlegenden Funktionen verschiedener App Mesh-Ressourcentypen behandelt.

### Szenario

Um die Verwendung von App Mesh zu veranschaulichen, gehen Sie davon aus, dass Sie über eine Anwendung mit den folgenden Merkmalen verfügen:

- Besteht aus zwei Services mit dem Namen `serviceA` und `serviceB`.
- Beide Services sind in einem Namespace namens `apps.local` registriert.
- `ServiceA` kommuniziert mit `serviceB` über HTTP/2, Port 80.
- Sie haben bereits Version 2 von `serviceB` bereitgestellt und mit dem Namen `serviceBv2` im `apps.local`-Namespace registriert.

Es gelten die folgenden Anforderungen:

- Sie möchten 75 Prozent des Datenverkehrs von `serviceA` an `serviceB` und 25 Prozent des Datenverkehrs an `serviceBv2` senden, um zu überprüfen, ob fehlerfrei `serviceBv2` ist, bevor Sie 100 Prozent des Datenverkehrs von `serviceA` an ihn senden.

- Sie möchten in der Lage sein, die Datenverkehrsgewichtung so anzupassen, dass 100 Prozent des Datenverkehrs an `serviceBv2` gehen, sobald sich die Zuverlässigkeit davon erwiesen hat. Sobald der gesamte Datenverkehr an `serviceBv2` gesendet wurde, möchten Sie beenden `serviceB`.
- Sie möchten keinen vorhandenen Anwendungscode oder Service Discovery-Registrierung für Ihre tatsächlichen Services ändern müssen, um die vorherigen Anforderungen zu erfüllen.

Um Ihre Anforderungen zu erfüllen, entscheiden Sie sich, ein App-Mesh-Servicegitter mit virtuellen Services, virtuellen Knoten, einem virtuellen Router und einer Route zu erstellen. Nach der Implementierung Ihres Gitters aktualisieren Sie Ihre Services, um den Envoy-Proxy zu verwenden. Nach der Aktualisierung kommunizieren Ihre Services miteinander über den Envoy-Proxy und nicht direkt miteinander.

## Voraussetzungen

App Mesh unterstützt Linux-Services, die bei DNS AWS Cloud Map oder bei dem registriert sind. Um dieses Handbuch für Erste Schritte zu verwenden, empfehlen wir, dass Sie über drei vorhandene Services verfügen, die bei DNS registriert sind. Sie können ein Service-Mesh und seine Ressourcen erstellen, auch wenn die Services nicht vorhanden sind. Sie können das Mesh jedoch erst verwenden, wenn Sie tatsächliche Services bereitgestellt haben.

Wenn noch keine Services ausgeführt werden, können Sie Amazon EC2-Instances starten und Anwendungen darauf bereitstellen. Weitere Informationen finden Sie unter [Tutorial: Erste Schritte mit Amazon EC2-Linux-Instances](#) im Amazon EC2-Benutzerhandbuch für Linux-Instances. Die verbleibenden Schritte gehen davon aus, dass die tatsächlichen Services als `serviceA`, `serviceB` und `serviceBv2` benannt sind, und dass alle Services über einen Namespace mit dem Namen `apps.local` gefunden werden können.

## Schritt 1: Erstellen von einem Mesh und einem virtuellen Service

Ein Service Mesh ist eine logische Begrenzung für den Netzwerkdatenverkehr zwischen den darin vorhandenen Services. Weitere Informationen finden Sie unter [Servicegitter](#). Ein virtueller Service ist eine Abstraktion eines tatsächlichen Services. Weitere Informationen finden Sie unter [Virtuelle Services](#).

Erstellen Sie die folgenden Ressourcen:

- Ein Mesh mit dem Namen `apps`, da alle Services im Szenario im `apps.local`-Namespace registriert sind.

- Einen virtuellen Service mit dem Namen `serviceb.apps.local`, da der virtuelle Service einen Service darstellt, der mit diesem Namen gefunden werden kann, und Sie Ihren Code nicht ändern möchten, um auf einen anderen Namen zu verweisen. Ein virtueller Service mit dem Namen `servicea.apps.local` wird in einem späteren Schritt hinzugefügt.

Sie können die AWS Management Console oder die AWS CLI Version 1.18.116 oder höher oder 2.0.38 oder höher verwenden, um die folgenden Schritte auszuführen. Wenn Sie die verwenden AWS CLI, verwenden Sie den `aws --version` Befehl, um Ihre installierte AWS CLI Version zu überprüfen. Wenn Sie Version 1.18.116 oder höher oder 2.0.38 oder höher nicht installiert haben, müssen Sie [installieren oder aktualisieren AWS CLI](#). Wählen Sie die Registerkarte für das Werkzeug aus, das Sie verwenden möchten.

### AWS Management Console

1. Öffnen Sie den App-Mesh-Konsolenassistenten für die Erstauführung unter <https://console.aws.amazon.com/appmesh/get-started>.
2. Geben Sie unter Mesh name (Mesh-Name) **apps** ein.
3. Geben Sie unter Virtual service name (Name des virtuellen Services) **serviceb.apps.local** ein.
4. Wählen Sie Next, um fortzufahren.

### AWS CLI

1. Erstellen Sie mit dem [create-mesh](#)-Befehl ein Mesh.

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

2. Erstellen Sie einen virtuellen Service mit dem [create-virtual-service](#)-Befehl.

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

## Schritt 2: Erstellen von einem virtuellen Knoten

Ein virtueller Knoten fungiert als ein logischer Verweis auf einen tatsächlichen Service. Weitere Informationen finden Sie unter [Virtuelle Knoten](#).

Erstellen Sie einen virtuellen Knoten mit dem Namen `serviceB`, da einer der virtuellen Knoten den tatsächlichen Service mit dem Namen `serviceB` darstellt. Der tatsächliche Service, den der virtuelle Knoten darstellt, kann über DNS mit dem Hostnamen von `serviceb.apps.local` gefunden werden. Alternativ können Sie tatsächliche Services mit AWS Cloud Map finden. Der virtuelle Knoten überwacht den Datenverkehr mithilfe des HTTP/2-Protokolls auf Port 80. Auch andere Protokolle werden unterstützt, ebenso wie Zustandsprüfungen. Sie erstellen virtuelle Knoten für `serviceA` und `serviceBv2` in einem späteren Schritt.

## AWS Management Console

1. Geben Sie unter Virtual node name (Name des virtuellen Knotens) **serviceB** ein.
2. Wählen Sie unter Service discovery method (Serviceerkennungsverfahren) die Option DNS aus, und geben Sie **serviceb.apps.local** als DNS hostname (DNS-Hostname) ein.
3. Wählen Sie unter Listener-Konfiguration die Option http2 für Protocol und geben Sie **80** für Port ein.
4. Wählen Sie Next, um fortzufahren.

## AWS CLI

1. Erstellen Sie eine Datei `create-virtual-node-serviceb.json` mit dem folgenden Inhalt:

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

```
"virtualNodeName": "serviceB"
}
```

- Erstellen Sie den virtuellen Knoten mit dem [create-virtual-node](#) Befehl unter Verwendung der JSON-Datei als Eingabe.

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

## Schritt 3: Erstellen von einem virtuellen Router und einer Route

Virtuelle Router routen den Datenverkehr für einen oder mehrere virtuelle Services innerhalb Ihres Meshes. Weitere Informationen finden Sie unter [Virtuelle Router](#) und [Routen](#).

Erstellen Sie die folgenden Ressourcen:

- Einen virtuellen Router mit dem Namen `serviceB`, da der virtuelle `serviceB.apps.local`-Service keine ausgehende Kommunikation mit einem anderen Service initiiert. Denken Sie daran, dass der virtuelle Service, den Sie zuvor erstellt haben, eine Abstraktion Ihres tatsächlichen `serviceb.apps.local`-Services ist. Der virtuelle Service sendet Datenverkehr an den virtuellen Router. Der virtuelle Router überwacht den Datenverkehr mithilfe des HTTP/2-Protokolls auf Port 80. Andere Protokolle werden ebenfalls unterstützt.
- Eine Route namens `serviceB`. Es leitet 100 Prozent seines Datenverkehrs an den `serviceB` virtuellen Knoten weiter. Die Gewichtung erfolgt in einem späteren Schritt, sobald Sie den `serviceBv2` virtuellen Knoten hinzugefügt haben. Obwohl in diesem Handbuch nicht behandelt, können Sie zusätzliche Filterkriterien für die Route hinzufügen und eine Wiederholungsrichtlinie hinzufügen, damit der Envoy-Proxy mehrere Versuche unternimmt, Datenverkehr an einen virtuellen Knoten zu senden, wenn ein Kommunikationsproblem auftritt.

### AWS Management Console

- Geben Sie für Virtual router name (Name des virtuellen Routers) **serviceB** ein.
- Wählen Sie unter Listener-Konfiguration die Option `http2` für Protocol und geben Sie **80** für Port an.
- Geben Sie unter Route name (Name der Route) **serviceB** ein.
- Wählen Sie für Route type (Routentyp) die Option `http2`.

5. Wählen Sie für Name des virtuellen Knotens unter Zielkonfiguration aus `serviceB` und geben Sie **100** für Gewicht ein.
6. Wählen Sie unter Konfiguration abgleichen eine Methode aus.
7. Wählen Sie Next, um fortzufahren.

## AWS CLI

1. Erstellen Sie einen virtuellen Router.
  - a. Erstellen Sie eine Datei `create-virtual-router.json` mit dem folgenden Inhalt:

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

- b. Erstellen Sie den virtuellen Router mit dem [create-virtual-router](#) Befehl unter Verwendung der JSON-Datei als Eingabe.

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

2. Erstellen Sie eine Route.
  - a. Erstellen Sie eine Datei `create-route.json` mit dem folgenden Inhalt:

```
{
 "meshName" : "apps",
 "routeName" : "serviceB",
 "spec" : {
 "httpRoute" : {
```

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

- b. Erstellen Sie die Route mit dem Befehl [create-route](#) unter Verwendung der JSON-Datei als Eingabe.

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

## Schritt 4: Überprüfen und Erstellen

Überprüfen Sie die Einstellungen anhand der vorherigen Anweisungen.

### AWS Management Console

Wählen Sie Edit (Bearbeiten) aus, wenn Sie Änderungen in einem Abschnitt vornehmen müssen. Sobald Sie mit den Einstellungen zufrieden sind, wählen Sie Create mesh (Netz erstellen) aus.

Im Fenster Status werden alle erstellten Netzressourcen angezeigt. Sie können die erstellten Ressourcen in der Konsole anzeigen, indem Sie View mesh (Netz anzeigen) auswählen.

### AWS CLI

Überprüfen Sie die Einstellungen des Meshes, das Sie erstellt haben, mit dem Befehl [describe-mesh](#).

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

Überprüfen Sie die Einstellungen des virtuellen Services, den Sie mit dem [describe-virtual-service](#) Befehl erstellt haben.

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

Überprüfen Sie die Einstellungen des virtuellen Knotens, den Sie mit dem [describe-virtual-node](#) Befehl erstellt haben.

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

Überprüfen Sie die Einstellungen des virtuellen Routers, den Sie mit dem [describe-virtual-router](#) Befehl erstellt haben.

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

Überprüfen Sie die Einstellungen der Route, die Sie erstellt haben, mit dem Befehl [describe-route](#).

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

## Schritt 5: Erstellen zusätzlicher Ressourcen

Um das Szenario abzuschließen, müssen Sie:

- Einen virtuellen Knoten mit dem Namen `serviceBv2` und einen anderen mit dem Namen `serviceA` erstellen. Beide virtuellen Knoten warten auf Anfragen über HTTP/2-Port 80. Konfigurieren Sie für den `serviceA` virtuellen Knoten ein Backend von `serviceb.apps.local`. Der gesamte ausgehende Datenverkehr vom `serviceA` virtuellen Knoten wird an den virtuellen Service namens `gesendetserviceb.apps.local`. Obwohl dies in diesem Handbuch nicht behandelt wird, können Sie auch einen Dateipfad angeben, in den Zugriffsprotokolle für einen virtuellen Knoten geschrieben werden sollen.
- Erstellen Sie einen zusätzlichen virtuellen Service mit dem Namen `servicea.apps.local`, der den gesamten Datenverkehr direkt an den `serviceA` virtuellen Knoten sendet.
- Aktualisieren Sie die `serviceB`-Route, die Sie in einem vorherigen Schritt erstellt haben, so, dass 75 Prozent des Datenverkehrs an den virtuellen `serviceB`-Knoten und 25 Prozent des Datenverkehrs an den virtuellen `serviceBv2`-Knoten gesendet werden. Im Laufe der Zeit können Sie die Gewichtungen weiter ändern, bis `serviceBv2` 100 Prozent des Datenverkehrs erhält. Sobald der gesamte Datenverkehr an `gesendet serviceBv2` gesendet wurde, können Sie den

`serviceB` virtuellen Knoten und den tatsächlichen Service herunterfahren und beenden. Wenn Sie Gewichtungen ändern, erfordert Ihr Code keine Änderung, da sich die Namen des virtuellen `serviceb.apps.local`- und des tatsächlichen Services nicht ändern. Denken Sie daran, dass der virtuelle `serviceb.apps.local`-Service Datenverkehr an den virtuellen Router sendet, der den Datenverkehr an die virtuellen Knoten weiterleitet. Die Serviceerkennungsnamen für die virtuellen Knoten können jederzeit geändert werden.

## AWS Management Console

1. Wählen Sie im linken Navigationsbereich Meshes.
2. Wählen Sie das `apps`-Mesh aus, das Sie in einem vorherigen Schritt erstellt haben.
3. Wählen Sie im linken Navigationsbereich Virtual nodes (Virtuelle Knoten) aus.
4. Wählen Sie Create virtual node (Virtuellen Knoten erstellen).
5. Geben Sie unter Virtual node name (Name des virtuellen Knotens) **serviceBv2** ein, wählen Sie für Service discovery method (Serviceerkennungsverfahren) die Option DNS aus, und geben Sie für den DNS hostname (DNS-Hostname) **servicebv2.apps.local** ein.
6. Wählen Sie für Listener-Konfiguration `http2` für Protocol aus und geben Sie **80** für Port ein.
7. Wählen Sie Create virtual node (Virtuellen Knoten erstellen).
8. Wählen Sie erneut Create virtual node (Virtuellen Knoten erstellen) aus. Geben Sie „**serviceA**“ als Virtual node name (Name des virtuellen Knotens) ein. Wählen Sie als Service discovery method (Diensterkennungsmethode) die Option DNS ein, und geben Sie als DNS hostname (DNS-Hostname) den Wert „**servicea.apps.local**“ ein.
9. Geben Sie bei Enter a virtual service name (Neuen virtuellen Service-Name eingeben) unter New backend (Neues Backend) den Wert „**serviceb.apps.local**“ ein.
10. Wählen Sie unter Listener configuration (Listener-Konfiguration) die Option `http2` für Protocol (Protokoll) aus, geben Sie „**80**“ bei Port ein, und wählen Sie dann Create virtual node (Virtuellen Knoten erstellen) aus.
11. Wählen Sie im linken Navigationsbereich Virtual routers (Virtuelle Router) aus, und wählen Sie dann den virtuellen Router `serviceB` aus der Liste aus.
12. Wählen Sie unter Routes (Routen) die Route `ServiceB` aus, die Sie in einem vorherigen Schritt erstellt haben, und wählen Sie dann Edit (Bearbeiten) aus.
13. Ändern Sie unter Targets (Ziele), Virtual node name (Name des virtuellen Knotens) den Wert von Weight (Gewicht) für `serviceB` zu **75**.

14. Wählen Sie Ziel hinzufügen, wählen Sie `serviceBv2` aus der Dropdown-Liste aus und legen Sie den Wert von Gewicht auf `fest25`.
15. Wählen Sie Speichern.
16. Wählen Sie im linken Navigationsbereich Virtual services (Virtuelle Services) aus, und wählen Sie dann Create virtual service (Virtuellen Dienst erstellen) aus.
17. Geben Sie „**servicea.apps.local**“ bei Virtual service name (Virtueller Dienstname) ein, wählen Sie Virtual node (Virtueller Knoten) als Provider (Anbieter), wählen Sie „`serviceA`“ bei Virtual node (Virtueller Knoten) und dann Create virtual service (Virtuellen Dienst erstellen) aus.

## AWS CLI

1. Erstellen Sie den virtuellen Knoten `serviceBv2`.
  - a. Erstellen Sie eine Datei `create-virtual-node-servicebv2.json` mit dem folgenden Inhalt:

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

- b. Erstellen Sie den virtuellen Knoten.

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

2. Erstellen Sie den virtuellen Knoten serviceA.
  - a. Erstellen Sie eine Datei `create-virtual-node-servicea.json` mit dem folgenden Inhalt:

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

- b. Erstellen Sie den virtuellen Knoten.

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

3. Aktualisieren Sie den virtuellen Service `serviceb.apps.local`, den Sie in einem vorherigen Schritt erstellt haben, um seinen Datenverkehr an den virtuellen Router

serviceB zu senden. Als der virtuelle Service ursprünglich erstellt wurde, sendete er keinen Datenverkehr, da der virtuelle Router serviceB noch nicht erstellt war.

- a. Erstellen Sie eine Datei `update-virtual-service.json` mit dem folgenden Inhalt:

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

- b. Aktualisieren Sie den virtuellen Service mit dem [update-virtual-service](#) Befehl .

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

4. Aktualisieren Sie die Route serviceB, die Sie in einem vorherigen Schritt erstellt haben.

- a. Erstellen Sie eine Datei `update-route.json` mit dem folgenden Inhalt:

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

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

- b. Aktualisieren Sie die Route mit dem Befehl [update-route](#) .

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

5. Erstellen Sie den virtuellen Service serviceA.

- a. Erstellen Sie eine Datei `create-virtual-servicea.json` mit dem folgenden Inhalt:

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

- b. Erstellen Sie den virtuellen Service.

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

## Mesh-Zusammenfassung

Bevor Sie das Service-Mesh erstellt haben, hatten Sie drei aktuelle Services mit den Namen `servicea.apps.local`, `serviceb.apps.local` und `servicebv2.apps.local`. Zusätzlich zu den tatsächlichen Services verfügen Sie jetzt über ein Service-Mesh, das die folgenden Ressourcen enthält, die die tatsächlichen Services darstellen:

- Zwei virtuelle Services. Der Proxy sendet den gesamten Datenverkehr vom virtuellen Service `servicea.apps.local` über einen virtuellen Router an den virtuellen Service `serviceb.apps.local`.
- Drei virtuelle Knoten mit den Namen `serviceA`, `serviceB` und `serviceBv2`. Der Envoy-Proxy verwendet die für die virtuellen Knoten konfigurierten Service-Erkennungsinformationen, um die IP-Adressen der tatsächlichen Services zu suchen.
- Einen virtuellen Router mit einer Route, die den Envoy-Proxy anweist, 75 Prozent des eingehenden Datenverkehrs an den virtuellen Knoten `serviceB` und 25 Prozent des Datenverkehrs an den virtuellen Knoten `serviceBv2` zu leiten.

## Schritt 6: Aktualisieren der Services

Nachdem Sie Ihr Mesh erstellt haben, müssen Sie die folgenden Aufgaben ausführen:

- Autorisieren Sie den Envoy-Proxy, den Sie mit jedem Service bereitstellen, um die Konfiguration eines oder mehrerer virtueller Knoten zu lesen. Weitere Informationen zur Autorisierung des Proxys finden Sie unter [En\) Proxy-Autorisierung](#).
- Führen Sie die folgenden Schritte aus, um Ihren vorhandenen Service zu aktualisieren.

So konfigurieren Sie eine Amazon EC2-Instance als Mitglied eines virtuellen Knotens

1. Erstellen Sie eine IAM-Rolle.
  - a. Erstellen Sie eine Datei mit dem Namen `ec2-trust-relationship.json` und dem folgenden Inhalt.

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

- b. Erstellen Sie eine IAM-Rolle mit dem folgenden Befehl.

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

2. Fügen Sie der Rolle IAM-Richtlinien hinzu, die es ihr ermöglichen, aus Amazon ECR zu lesen, und nur die Konfiguration eines bestimmten virtuellen App-Mesh-Knotens.

- a. Erstellen Sie eine Datei namens „virtual-node-policy.json“ mit dem folgenden Inhalt. apps ist der Name des Netzes, das Sie in [the section called “Schritt 1: Erstellen von einem Mesh und einem virtuellen Service”](#) erstellt haben, und serviceB ist der Name des virtuellen Knotens, den Sie in [the section called “Schritt 2: Erstellen von einem virtuellen Knoten”](#) erstellt haben. Ersetzen Sie *111122223333* durch Ihre Konto-ID und *us-west-2* durch die Region, in der Sie Ihr Gitter erstellt haben.

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

- b. Erstellen Sie die Richtlinie mit dem folgenden Befehl.

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

- c. Hängen Sie die Richtlinie, die Sie im vorherigen Schritt erstellt haben, an die Rolle an, damit die Rolle die Konfiguration nur für den serviceB virtuellen Knoten aus App Mesh lesen kann.

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

- d. Hängen Sie die AmazonEC2ContainerRegistryReadOnly verwaltete Richtlinie an die Rolle an, damit sie das Envoy-Container-Image von Amazon ECR abrufen kann.

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

3. [Starten Sie eine Amazon EC2-Instance mit der von Ihnen erstellten IAM-Rolle.](#)
4. Stellen Sie eine Verbindung zur Instance per SSH her.
5. Installieren Sie Docker und die gemäß der Betriebssystemdokumentation AWS CLI auf Ihrer Instance.
6. Authentifizieren Sie sich beim Envoy Amazon ECR-Repository in der Region, aus der Ihr Docker-Client das Image abrufen soll.
  - Alle Regionen außer me-south-1, ap-east-1, ap-southeast-3, eu-south-1, il-central-1, und af-south-1. Sie können *us-west-2* durch jede [unterstützte Region](#) ersetzen me-south-1, außer ap-east-1, eu-south-1, ap-southeast-3, il-central-1, und af-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
```

- me-south-1-Region

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

- ap-east-1-Region

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

7. Führen Sie einen der folgenden Befehle aus, um den App Mesh Envoy-Container auf Ihrer Instance zu starten, je nachdem, aus welcher Region Sie das Image abrufen möchten. Die *App*- und *ServiceB*-Werte sind die im Szenario definierten Namen des Meshes und der virtuellen Knoten. Diese Informationen teilen dem Proxy mit, welche Konfiguration des virtuellen Knotens aus App Mesh gelesen werden soll. Um das Szenario abzuschließen, müssen Sie diese Schritte auch für die Amazon EC2-Instances ausführen, die die durch die *serviceA* virtuellen Knoten *serviceBv2* und repräsentierten Services hosten. Ersetzen Sie diese Werte für Ihre eigene Anwendung durch Ihre eigenen.
  - Alle Regionen außer *me-south-1*, *ap-east-1*, *ap-southeast-3*, *eu-south-1*, *il-central-1*, und *af-south-1*. Sie können *Regionscode* durch jede [unterstützte Region](#) ersetzen *me-south-1*, mit Ausnahme der *af-south-1* Regionen *ap-east-1*, *ap-southeast-3*, *il-central-1*, *eu-south-1* und . Sie können *1337* durch einen beliebigen Wert zwischen 0 und 2147483647 ersetzen.

```
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:v1.27.3.0-prod
```

- Region *me-south-1*: Sie können *1337* durch einen beliebigen Wert zwischen 0 und 2147483647 ersetzen.

```
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:v1.27.3.0-prod
```

- Region *ap-east-1*: Sie können *1337* durch einen beliebigen Wert zwischen 0 und 2147483647 ersetzen.

```
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:v1.27.3.0-prod
```

**Note**

Die `-APPMESH_RESOURCE_ARNE`Eigenschaft erfordert Version `1.15.0` oder höher des Envoy-Images. Weitere Informationen finden Sie unter [Envoy](#).

**⚠ Important**

Nur Version `v1.9.0.0-prod` oder höher wird für die Verwendung mit App Mesh unterstützt.

- Wählen Sie im Folgenden `Show more` aus. Erstellen Sie eine Datei namens `envoy-networking.sh` mit dem folgenden Inhalt auf Ihrer Instance. Ersetzen Sie `8000` durch den Port, den Ihr Anwendungscode für eingehenden Datenverkehr verwendet. Sie können den Wert für `APPMESH_IGNORE_UID` ändern, aber der Wert muss mit dem Wert übereinstimmen, den Sie im vorherigen Schritt angegeben haben, z. B. `1337`. Sie können bei Bedarf `APPMESH_EGRESS_IGNORED_IP` zusätzliche Adressen hinzufügen. Ändern Sie keine anderen Linien.

```
#!/bin/bash -e

#
Start of configurable options
#

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

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

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

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

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

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

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

#
End of configurable options
#

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

- Um iptables-Regeln für die Weiterleitung des Anwendungsdatenverkehrs an den Envoy-Proxy zu konfigurieren, führen Sie das Skript aus, das Sie im vorherigen Schritt erstellt haben.

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

- Starten Sie den Anwendungscode Ihres virtuellen Knotens.

### Note

Weitere Beispiele und exemplarische Vorgehensweisen für App Mesh finden Sie im [App Mesh-Beispiel-Repository](#).

## App Mesh

Dies ist die experimentelle öffentliche Roadmap für AWS App Mesh. Die Roadmap ermöglicht es Kunden, sich über unsere kommenden Produkte und Prioritäten zu informieren, sodass Kunden planen können, wie sie App Mesh in future verwenden können. Dieses Repository enthält Informationen darüber, woran wir arbeiten, und ermöglicht es AWS -Kunden, direktes Feedback zu geben.

[App Mesh Mesh-Roadmap](#)

## App Mesh

Im folgenden Repository finden Sie end-to-end Walkthroughs, die AWS App Mesh in Aktion zeigen, sowie Codebeispiele für die Integration mit verschiedenen AWS Diensten:

[Beispiele für App Mesh](#)

# App Mesh

App Mesh besteht aus den folgenden Konzepten.

- [Servicegitter](#)
- [Virtuelle Services](#)
- [Virtuelle Gateways](#)
- [Virtuelle Knoten](#)
- [Virtuelle Router](#)

## Servicegitter

Ein Service-Mesh ist eine logische Begrenzung für den Netzwerkdatenverkehr zwischen den darin vorhandenen Services. Nach dem Erstellen Ihres Service-Mesh können Sie virtuelle Services, virtuelle Knoten, virtuelle Router und Routen zur Verteilung des Datenverkehrs zwischen den Anwendungen in Ihrem Mesh erstellen.

## Servicegitter erstellen

### Note

Wenn Sie ein Mesh erstellen, müssen Sie einen Namespace-Selektor hinzufügen. Wenn der Namespace-Selektor leer ist, werden alle Namespaces ausgewählt. Um die Namespaces einzuschränken, verwenden Sie ein Label, um App Mesh Mesh-Ressourcen mit dem erstellten Mesh zu verknüpfen.

## AWS Management Console

So erstellen Sie ein Servicegitter für die AWS Management Console

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie Create mesh (Mesh erstellen).
3. Geben Sie für Mesh name (Name des Mesh) einen Namen für das Service-Mesh an.
4. (Optional) Wählen Sie „Externen Datenverkehr zulassen“. Standardmäßig leiten Proxys im Mesh nur Datenverkehr untereinander weiter. Wenn Sie externen Datenverkehr zulassen,

leiten die Proxys im Mesh auch TCP-Verkehr direkt an Dienste weiter, die nicht mit einem im Mesh definierten Proxy bereitgestellt werden.

 Note

Wenn Sie Backends auf einem virtuellen Knoten angeben, wenn Sie `ALLOW_ALL` verwenden, müssen Sie alle Ausgänge für diesen virtuellen Knoten als Backends angeben. Andernfalls funktioniert `ALLOW_ALL` nicht mehr für diesen virtuellen Knoten.

## 5. Bevorzugte IP-Version

Steuern Sie, welche IP-Version für den Datenverkehr innerhalb des Meshs verwendet werden soll, indem Sie die Option Standard-IP-Versionsverhalten überschreiben aktivieren. Standardmäßig verwendet App Mesh eine Vielzahl von IP-Versionen.

 Note

Das Mesh wendet die IP-Präferenz auf alle virtuellen Knoten und virtuellen Gateways innerhalb eines Meshs an. Dieses Verhalten kann für einen einzelnen virtuellen Knoten außer Kraft gesetzt werden, indem Sie die IP-Präferenz festlegen, wenn Sie den Knoten erstellen oder bearbeiten. Die IP-Präferenz kann auf einem virtuellen Gateway nicht überschrieben werden, da die Konfiguration für virtuelle Gateways, die es ihnen ermöglicht, sowohl IPv4- als auch IPv6-Verkehr abzuhearschen, dieselbe ist, unabhängig davon, welche Präferenz für das Mesh festgelegt ist.

- Standard
  - Der DNS-Resolver von Envoy bevorzugt IPv6 und greift auf ihn zurück IPv4.
  - Wir verwenden die von zurückgegebene IPv4 Adresse, AWS Cloud Map sofern verfügbar, und greifen auf die IPv6 Adresse zurück.
  - Der für die lokale App erstellte Endpunkt verwendet eine IPv4 Adresse.
  - Die Envoy-Listener binden sich an alle IPv4 Adressen.
- IPv6 bevorzugt
  - Der DNS-Resolver von Envoy bevorzugt IPv6 und greift auf ihn zurück IPv4.
  - Die von zurückgegebene IPv6 Adresse AWS Cloud Map wird verwendet, sofern verfügbar, und verwendet wieder die IPv4 Adresse

- Der für die lokale App erstellte Endpunkt verwendet eine IPv6 Adresse.
  - Die Envoy-Zuhörer binden sich an alle IPv4 und IPv6 Adressen.
  - IPv4 bevorzugt
    - Der DNS-Resolver von Envoy bevorzugt IPv4 und greift auf ihn zurück IPv6.
    - Wir verwenden die von zurückgegebene IPv4 Adresse, AWS Cloud Map sofern verfügbar, und greifen auf die IPv6 Adresse zurück.
    - Der für die lokale App erstellte Endpunkt verwendet eine IPv4 Adresse.
    - Die Envoy-Zuhörer binden sich an alle IPv4 und IPv6 Adressen.
  - Nur für IPv6
    - Der DNS-Resolver von Envoy verwendet nur IPv6.
    - Es wird nur die von AWS Cloud Map zurückgegebene IPv6 Adresse verwendet. Wenn eine IPv4 Adresse AWS Cloud Map zurückgegeben wird, werden keine IP-Adressen verwendet und leere Ergebnisse werden an den Envoy zurückgegeben.
    - Der für die lokale App erstellte Endpunkt verwendet eine IPv6 Adresse.
    - Die Envoy-Zuhörer binden sich an alle IPv4 und IPv6 Adressen.
  - Nur IPv4
    - Der DNS-Resolver von Envoy verwendet nur IPv4.
    - Es wird nur die von AWS Cloud Map zurückgegebene IPv4 Adresse verwendet. Wenn eine IPv6 Adresse AWS Cloud Map zurückgegeben wird, werden keine IP-Adressen verwendet und leere Ergebnisse werden an den Envoy zurückgegeben.
    - Der für die lokale App erstellte Endpunkt verwendet eine IPv4 Adresse.
    - Die Envoy-Zuhörer binden sich an alle IPv4 und IPv6 Adressen.
6. Wählen Sie `Create mesh` (Mesh erstellen), um den Vorgang abzuschließen.
  7. (Optional) Geben Sie das Mesh für andere Konten frei. Ein gemeinsam genutztes Mesh ermöglicht es Ressourcen, die von verschiedenen Konten erstellt wurden, im selben Mesh miteinander zu kommunizieren. Weitere Informationen finden Sie unter [Mit gemeinsam genutzten Meshes arbeiten](#).

## AWS CLI

Um ein Netz mit dem zu erstellen AWS CLI.

Erstellen Sie ein Service-Mesh mit dem folgenden Befehl (ersetzen Sie die *roten* Werte durch Ihre eigenen):

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

2. Beispielausgabe:

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

Weitere Informationen zum Erstellen eines Meshs mit dem AWS CLI for App Mesh finden Sie unter dem Befehl [create-mesh](#) in der AWS CLI Referenz.

## Löschen eines Netzes

### AWS Management Console

So löschen Sie ein virtuelles Gateway mit der AWS Management Console

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie das Mesh aus, das Sie löschen möchten. Alle Netze, die Sie besitzen und die mit Ihnen [geteilt](#) wurden, werden aufgelistet.
3. Geben Sie in das Bestätigungsfeld **eindelete** und klicken Sie dann auf Löschen.

## AWS CLI

So löschen Sie ein Netz mit dem AWS CLI

1. Verwenden Sie den folgenden Befehl, um Ihr Netz zu löschen (ersetzen Sie die *roten* Werte durch Ihre eigenen):

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

2. Beispielausgabe:

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

Weitere Informationen zum Löschen eines Meshs mit dem AWS CLI für App Mesh finden Sie unter dem Befehl [delete-mesh](#) in der AWS CLI Referenz.

## Virtuelle Services

Bei einem virtuellen Service handelt es sich um eine Abstraktion eines echten Service, der von einem virtuellen Knoten direkt oder indirekt mittels eines virtuellen Routers bereitgestellt wird. Abhängige Services rufen Ihren virtuellen Service unter dessen `virtualServiceName` auf. Diese

Anforderungen werden dann zu dem virtuellen Knoten oder virtuellen Router weitergeleitet, der als Anbieter des virtuellen Service angegeben wird.

## Erstellen eines virtuellen Services

### AWS Management Console

Um einen virtuellen Dienst mit dem zu erstellen AWS Management Console

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie das Mesh aus, in dem Sie den virtuellen Service erstellen möchten. Alle Netze, die Sie besitzen und die mit Ihnen geteilt wurden, werden aufgelistet.
3. Wählen Sie im linken Navigationsbereich die Option Virtual services (Virtuelle Services) aus.
4. Klicken Sie auf Create virtual service (Virtuellen Service erstellen).
5. Wählen Sie unter Virtual service name (Name des virtuellen Service) einen Namen für Ihren virtuellen Service aus. Sie können einen beliebigen Namen wählen, aber der Service Discovery-Name des realen Dienstes, auf den Sie abzielen `my-service.default.svc.cluster.local`, wie z. B., wird empfohlen, um die Korrelation Ihrer virtuellen Dienste mit realen Diensten zu erleichtern. Auf diese Weise müssen Sie Ihren Code nicht ändern, um auf einen anderen Namen zu verweisen, auf den Ihr Code derzeit verweist. Der von Ihnen angegebene Name muss in eine Nicht-Loopback-IP-Adresse aufgelöst werden, da der App-Container in der Lage sein muss, den Namen erfolgreich aufzulösen, bevor die Anfrage an den Envoy-Proxy gesendet wird. Sie können jede Nicht-Loopback-IP-Adresse verwenden, da weder die App noch die Proxy-Container mit dieser IP-Adresse kommunizieren. Der Proxy kommuniziert mit anderen virtuellen Diensten über die Namen, die Sie in App Mesh für sie konfiguriert haben, nicht über IP-Adressen, auf die sich die Namen auflösen.
6. Wählen Sie unter Provider (Anbieter) die Art von Anbieter für Ihren virtuellen Service aus:
  - Wenn Sie möchten, dass der virtuelle Service Datenverkehr auf mehrere virtuelle Knoten verteilt wird, wählen Sie Virtual router (Virtueller Router) und danach den virtuellen Router aus dem Dropdown-Menü aus.
  - Wenn Sie möchten, dass der virtuelle Dienst einen virtuellen Knoten direkt ohne virtuellen Router erreicht, wählen Sie Virtueller Knoten und dann den zu verwendenden virtuellen Knoten aus dem Dropdownmenü aus.

**Note**

App Mesh erstellt möglicherweise automatisch eine standardmäßige Envoy-Routenwiederholungsrichtlinie für jeden virtuellen Knotenanbieter, den Sie am oder nach dem 29. Juli 2020 definieren, obwohl Sie eine solche Richtlinie nicht über die App Mesh Mesh-API definieren können. Weitere Informationen finden Sie unter [Standardrichtlinie für die Wiederholung von Routen](#).

- Wenn Sie zu diesem Zeitpunkt nicht möchten, dass der virtuelle Service Datenverkehr weiterleitet (z. B. noch keine virtuellen Knoten oder virtuellen Router vorhanden sind), wählen Sie None (Keine) aus. Sie können den Anbieter dieses virtuellen Service zu einem späteren Zeitpunkt aktualisieren.
7. Wählen Sie Create virtual service (Virtuellen Service erstellen), um den Vorgang abzuschließen.

## AWS CLI

Um einen virtuellen Dienst mit dem zu erstellen AWS CLI.

Erstellen Sie einen virtuellen Dienst mit einem Anbieter für virtuelle Knoten mithilfe des folgenden Befehls und einer JSON-Eingabedatei (ersetzen Sie die *roten* Werte durch Ihre eigenen):

1. 

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

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

### 3. Beispielausgabe:

```
{
 "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"
 }
}
```

Weitere Informationen zum Erstellen eines virtuellen Dienstes mit dem AWS CLI for App Mesh finden Sie im [create-virtual-service](#) Befehl in der AWS CLI Referenz.

## Löschen eines virtuellen Dienstes

### Note

Sie können einen virtuellen Service, auf den eine Gateway-Route verweist, nicht löschen. Sie müssen zuerst die Gateway-Route löschen.

## AWS Management Console

So löschen Sie einen virtuellen Service mithilfe der AWS Management Console

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie das Mesh aus, aus dem Sie einen virtuellen Service löschen möchten. Alle Netze, die Sie besitzen und die mit Ihnen [geteilt](#) wurden, werden aufgelistet.
3. Wählen Sie im linken Navigationsbereich die Option Virtual services (Virtuelle Services) aus.
4. Wählen Sie den virtuellen Dienst aus, den Sie löschen möchten, und klicken Sie oben rechts auf Löschen. Sie können nur ein virtuelles Gateway löschen, bei dem Ihr Konto als Ressourcenbesitzer aufgeführt ist.
5. Geben Sie in das Bestätigungsfeld ein **delete** und klicken Sie dann auf Löschen.

## AWS CLI

So löschen Sie einen virtuellen Service mithilfe der AWS CLI

1. Verwenden Sie den folgenden Befehl, um Ihren virtuellen Dienst zu löschen (ersetzen Sie die *roten* Werte durch Ihre eigenen):

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

2. Beispielausgabe:

```
{
 "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": {
```



umgeschrieben/1. Informationen zum Konfigurieren von Rewrites finden Sie im Abschnitt [Erstellen einer Gateway-Route](#) unter Gateway-Routen.

Beim Erstellen eines virtuellen Gateways `proxyConfiguration` und `user` sollte nicht konfiguriert werden.

Eine end-to-end Komplettlösung finden Sie unter [Configuring Inbound Gateway](#).

## Virtuelles Gateway erstellen

### Note

Wenn Sie ein virtuelles Gateway erstellen, müssen Sie einen Namespace-Selektor mit einer Bezeichnung hinzufügen, um die Liste der Namespaces zu identifizieren, denen Gateway-Routen dem erstellten virtuellen Gateway zugeordnet werden sollen.

## AWS Management Console

So erstellen Sie ein virtuelles Gateway mit der AWS Management Console

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie das Netz aus, in dem Sie das virtuelle Gateway erstellen möchten. Alle Netze, die Sie besitzen und die mit Ihnen [geteilt](#) wurden, werden aufgelistet.
3. Wählen Sie in der linken Navigationsleiste Virtuelle Gateways aus.
4. Wählen Sie Virtuelles Gateway erstellen.
5. Geben Sie für Virtual Gateway Name einen Namen für Ihr virtuelles Gateway ein.
6. (Optional, aber empfohlen) Konfigurieren Sie die Standardeinstellungen der Client-Richtlinien.
  - a. (Optional) Wählen Sie TLS erzwingen aus, wenn das Gateway mithilfe von Transport Layer Security (TLS) nur mit virtuellen Diensten kommunizieren soll.
  - b. (Optional) Geben Sie für Ports einen oder mehrere Ports an, auf denen Sie die TLS-Kommunikation mit virtuellen Diensten erzwingen möchten.
  - c. Wählen Sie für Validierungsmethode eine der folgenden Optionen. Das von Ihnen angegebene Zertifikat muss bereits vorhanden sein und bestimmte Anforderungen erfüllen. Weitere Informationen finden Sie unter [Zertifikatanforderungen](#).

- AWS Private Certificate Authority Hosting — Wählen Sie ein oder mehrere vorhandene Zertifikate aus.
  - Hosting des Envoy Secret Discovery Service (SDS) — Geben Sie den Namen des Geheimnisses ein, das der Envoy mithilfe des Secret Discovery Service abrufen.
  - Lokales Dateihosting — Geben Sie den Pfad zur Zertifikatskettendatei auf dem Dateisystem an, in dem Envoy bereitgestellt wird.
- d. (Optional) Geben Sie einen alternativen Betreff ein. Um weitere SANs hinzuzufügen, wählen Sie SAN hinzufügen aus. SANs müssen mit FQDN oder URI formatiert sein.
- e. (Optional) Wählen Sie Client-Zertifikat bereitstellen und eine der folgenden Optionen aus, um ein Client-Zertifikat bereitzustellen, wenn ein Server es anfordert, und um die gegenseitige TLS-Authentifizierung zu aktivieren. Weitere Informationen zu gegenseitigem TLS finden Sie in den Dokumenten zur [gegenseitigen TLS-Authentifizierung](#) von App Mesh.
- Hosting des Envoy Secret Discovery Service (SDS) — Geben Sie den Namen des Geheimnisses ein, das der Envoy mithilfe des Secret Discovery Service abrufen.
  - Lokales Dateihosting — Geben Sie den Pfad zur Zertifikatskettendatei sowie den privaten Schlüssel auf dem Dateisystem an, in dem Envoy bereitgestellt wird. Eine vollständige Anleitung zur Bereitstellung end-to-end eines Meshs mit einer Beispielanwendung, die Verschlüsselung mit lokalen Dateien verwendet, finden Sie unter [Konfiguration von TLS mit von Dateien bereitgestellten TLS-Zertifikaten](#) auf GitHub.
7. (Optional) Wählen Sie Protokollierung aus, um die Protokollierung zu konfigurieren. Geben Sie den Pfad der HTTP-Zugriffsprotokolle ein, den Envoy verwenden soll. Wir empfehlen den `/dev/stdout` Pfad, damit Sie Docker-Protokolltreiber verwenden können, um Ihre Envoy-Protokolle an einen Dienst wie Amazon CloudWatch Logs zu exportieren.

 Note

Die Protokolle müssen noch von einem Agenten in Ihrer Anwendung übernommen und an ein Ziel gesendet werden. Dieser Dateipfad teilt Envoy nur mit, wohin die Protokolle gesendet werden sollen.

8. Konfigurieren Sie den Listener.

- a. Wählen Sie ein Protokoll aus und geben Sie den Port an, auf dem Envoy den Datenverkehr überwacht. Der HTTP-Listener ermöglicht den Verbindungsübergang zu Websockets. Sie können auf Listener hinzufügen klicken, um mehrere Listener hinzuzufügen. Mit der Schaltfläche Entfernen wird dieser Listener entfernt.
- b. (Optional) Verbindungspool aktivieren

Das Verbindungspool begrenzt die Anzahl der Verbindungen, die der Virtual Gateway Envoy gleichzeitig herstellen kann. Es soll Ihre Envoy-Instanz davor schützen, mit Verbindungen überfordert zu werden, und ermöglicht es Ihnen, das Traffic-Shaping an die Bedürfnisse Ihrer Anwendungen anzupassen.

Sie können die Einstellungen für den zweiseitigen Verbindungspool für einen virtuellen Gateway-Listener konfigurieren. App Mesh setzt die clientseitigen Verbindungspool-Einstellungen standardmäßig auf unendlich, was die Mesh-Konfiguration vereinfacht.

 Note

Das Protokoll `connectionPool` und das `connectionPool` `PortMapping`-Protokoll müssen identisch sein. Wenn Ihr Listener-Protokoll `grpc` oder `isthttp2`, geben Sie `esmaxRequests` nur an. Wenn Ihr Listener-Protokoll `isthttp`, können Sie `maxConnections` sowohl als auch `maxPendingRequests` angeben.

- Geben Sie für Maximale Verbindungen die maximale Anzahl ausgehender Verbindungen an.
  - Geben Sie für Maximale Anforderungen die maximale Anzahl von parallel Anforderungen an, die mit dem Virtual Gateway Envoy eingerichtet werden können.
  - (Optional) Geben Sie unter Maximale Anzahl ausstehender Anfragen hinter Maximale Anzahl ausstehender Anfragen, die ein Abgesandter in die Warteschlange stellt, die Anzahl der überlaufenden Anfragen an. Der Standardwert ist 2147483647.
- c. (Optional) Wenn Sie einen Gesundheitscheck für Ihren Listener konfigurieren möchten, wählen Sie Health Check aktivieren aus.

Eine Zustandsprüfungsrichtlinie ist optional. Wenn Sie jedoch Werte für eine Integritätsrichtlinie angeben, müssen Sie Werte für den Schwellenwert für den Zustand,

das Intervall für die Integritätsprüfung, das Protokoll für die Integritätsprüfung, den Zeitraum für die Zeitüberschreitung und den Schwellenwert für fehlerhaft angeben.

- Wählen Sie für das Health Check-Protokoll ein Protokoll aus. Wenn Sie `grpc` auswählen, muss Ihr Service mit dem [GRPC Health Checking Protocol](#) konform sein.
  - Geben Sie unter Health check port (Zustandsprüfungsport) den Port an, auf dem die Zustandsprüfung ausgeführt werden soll.
  - Geben Sie für Healthy threshold (Schwellenwert für fehlerfreien Zustand) die Anzahl der fortlaufenden erfolgreichen Zustandsprüfungen an, die auftreten müssen, damit der Listener als fehlerfrei deklariert wird.
  - Geben Sie für Health check interval (Zustandsprüfungsintervall) den Zeitraum in Millisekunden zwischen den einzelnen Zustandsprüfungen an.
  - Geben Sie unter Path (Pfad) den Zielpfad für die Zustandsprüfungsanforderung an. Dieser Wert wird nur verwendet, wenn das Health Check-Protokoll `http` oder `http2` ist. Für andere Protokolle wird der Wert ignoriert.
  - Geben Sie für Timeout-Zeitraum die Wartezeit in Millisekunden ein, bis eine Antwort von der Zustandsprüfung eingeht.
  - Geben Sie unter Unhealthy threshold (Schwellenwert für fehlerhaften Zustand) die Anzahl der fortlaufenden fehlgeschlagenen Zustandsprüfungen an, die auftreten müssen, damit der Listener als fehlerhaft deklariert wird.
- d. (Optional) Wenn Sie angeben möchten, ob Clients über TLS mit diesem virtuellen Gateway kommunizieren, wählen Sie TLS-Terminierung aktivieren aus.
- Wählen Sie unter Modus den Modus aus, für den TLS auf dem Listener konfiguriert werden soll.
  - Wählen Sie für Zertifikatsmethode eine der folgenden Optionen. Das Zertifikat muss bestimmte Anforderungen erfüllen. Weitere Informationen finden Sie unter [Zertifikatanforderungen](#).
    - AWS Certificate Manager Hosting — Wählen Sie ein vorhandenes Zertifikat aus.
    - Hosting des Envoy Secret Discovery Service (SDS) — Geben Sie den Namen des Geheimnisses ein, das der Envoy mithilfe des Secret Discovery Service abrufen kann.
    - Lokales Dateihosting — Geben Sie den Pfad zur Zertifikatskette und den privaten Schlüsseldateien auf dem Dateisystem an, in dem Envoy bereitgestellt wird.
  - (Optional) Wählen Sie Clientzertifikat erforderlich und eine der folgenden Optionen aus, um die gegenseitige TLS-Authentifizierung zu aktivieren, wenn der Client ein

Zertifikat bereitstellt. Weitere Informationen zu gegenseitigem TLS finden Sie in den Dokumenten zur [gegenseitigen TLS-Authentifizierung](#) von App Mesh.

- Hosting des Envoy Secret Discovery Service (SDS) — Geben Sie den Namen des Geheimnisses ein, das der Envoy mithilfe des Secret Discovery Service abrufen.
- Lokales Dateihosting — Geben Sie den Pfad zur Zertifikatskettendatei auf dem Dateisystem an, in dem Envoy bereitgestellt wird.
- (Optional) Geben Sie einen alternativen Betreff ein. Um weitere SANs hinzuzufügen, wählen Sie SAN hinzufügen aus. SANs müssen mit FQDN oder URI formatiert sein.

9. Wählen Sie Virtuelles Gateway erstellen, um den Vorgang abzuschließen.

## AWS CLI

Um ein virtuelles Gateway mit dem zu erstellen AWS CLI.

Erstellen Sie ein virtuelles Gateway mit dem folgenden Befehl und geben Sie JSON ein (ersetzen Sie die *roten* Werte durch Ihre eigenen):

1.

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

2. Inhalt des Beispiels create-virtual-gateway .json:

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

3. Beispielausgabe:

```
{
```

```
"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"
}
```

Weitere Informationen zum Erstellen eines virtuellen Gateways mit dem AWS CLI for App Mesh finden Sie im [create-virtual-gateway](#) Befehl in der AWS CLI Referenz.

## Virtuelles Gateway bereitstellen

Stellen Sie einen Amazon ECS- oder Kubernetes-Service bereit, der nur den [Envoy-Container](#) enthält. Sie können den Envoy-Container auch auf einer Amazon EC2 EC2-Instance bereitstellen. Weitere Informationen finden Sie unter [Erste Schritte mit App Mesh und Amazon EC2](#). Weitere Informationen zur Bereitstellung auf Amazon ECS finden Sie unter [Erste Schritte mit App Mesh und Amazon ECS](#) oder [Erste Schritte mit AWS App Mesh und Kubernetes](#) für die Bereitstellung auf Kubernetes. Sie müssen die `APPMESH_RESOURCE_ARN` Umgebungsvariable auf `set mesh/mesh-`

`name/virtualGateway/virtual-gateway-name` und Sie dürfen die Proxykonfiguration nicht angeben, damit der Verkehr des Proxys nicht zu sich selbst umgeleitet wird. Standardmäßig verwendet App Mesh den Namen der Ressource, die Sie in `APPMESH_RESOURCE_ARN` angegeben haben, wenn sich Envoy in Metriken und Ablaufverfolgungen auf sich selbst bezieht. Sie können dieses Verhalten übergehen, indem Sie die `APPMESH_RESOURCE_CLUSTER` -Umgebungsvariable mit Ihrem eigenen Namen festlegen.

Wir empfehlen, mehrere Instances des Containers bereitzustellen und einen Network Load Balancer einzurichten, um den Traffic zu den Instances zu verteilen. Der Service Discovery-Name des Load Balancers ist der Name, den externe Dienste verwenden sollen, um auf Ressourcen zuzugreifen, die sich im Mesh befinden, z. B. `myapp.example.com`. Weitere Informationen finden Sie unter [Erstellen eines Network Load Balancer](#) (Amazon ECS), [Erstellen eines externen Load Balancers](#) (Kubernetes) oder [Tutorial: Erhöhen Sie die Verfügbarkeit Ihrer Anwendung auf Amazon EC2](#). Weitere Beispiele und Komplettlösungen finden Sie auch in unseren [App Mesh Beispielen](#).

Aktivieren Sie die Proxy-Autorisierung für Envoy. Weitere Informationen finden Sie unter [En\) Proxy-Autorisierung](#).

## Löschen eines virtuellen Netzes ein virtuelles Gateway

### AWS Management Console

So löschen Sie ein virtuelles Gateway mit der AWS Management Console

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie das Netz aus, aus dem Sie ein virtuelles Gateway löschen möchten. Alle Netze, die Sie besitzen und die mit Ihnen [geteilt](#) wurden, werden aufgelistet.
3. Wählen Sie in der linken Navigationsleiste Virtuelle Gateways aus.
4. Wählen Sie das virtuelle Gateway aus, das Sie löschen möchten, und klicken Sie auf Löschen. Sie können ein virtuelles Gateway nicht löschen, wenn es über zugeordnete Gateway-Routen verfügt. Sie müssen zuerst alle zugehörigen Gateway-Routen löschen. Sie können nur ein virtuelles Gateway löschen, bei dem Ihr Konto als Ressourcenbesitzer aufgeführt ist.
5. Geben Sie in das Bestätigungsfeld den Text **indelete** und wählen Sie dann Löschen aus.

## AWS CLI

So löschen Sie ein virtuelles Gateway mit der AWS CLI

1. Verwenden Sie den folgenden Befehl, um Ihr virtuelles Gateway zu löschen (ersetzen Sie die *roten* Werte durch Ihre eigenen):

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

2. Beispielausgabe:

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

Weitere Informationen zum Löschen eines virtuellen Gateways mit dem AWS CLI for App Mesh finden Sie im [delete-virtual-gateway](#) Befehl in der AWS CLI Referenz.

## Gateway-Routen

Eine Gateway-Route ist an ein virtuelles Gateway angefügt und leitet den Datenverkehr an einen vorhandenen virtuellen Service weiter. Wenn eine Route mit einer Anforderung übereinstimmt, kann sie den Datenverkehr an einen virtuellen Zieldienst verteilen. Dieses Thema hilft Ihnen keine Gateway-Routen angezeigt werden.

### Eine Gateway-Route erstellen

#### AWS Management Console

So erstellen Sie eine Gateway-Route mit der AWS Management Console

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie das Mesh aus, in dem Sie die Gateway-Route erstellen möchten. Alle Netze, die Sie besitzen und die mit Ihnen [geteilt](#) wurden, werden aufgelistet.
3. Wählen Sie in der linken Navigationsleiste Virtuelle Gateways aus.
4. Wählen Sie das virtuelle Gateway aus, dem Sie eine neue Gateway-Route zuordnen möchten. Wenn keine aufgelistet sind, müssen Sie zuerst [eine erstellen](#). Sie können nur eine Gateway-Route für ein virtuelles Gateway erstellen, für das Ihr Konto als Ressourcenbesitzer aufgeführt ist.
5. Wählen Sie in der Tabelle Gateway-Routen die Option Gateway-Route erstellen aus.
6. Geben Sie unter Gateway-Routenname den Namen an, der für Ihre Gateway-Route verwendet werden soll.
7. Wählen Sie als Gateway-Routentyp entweder http, http2 oder grpc.
8. Wählen Sie einen vorhandenen Namen für den virtuellen Dienst aus. Wenn keine aufgelistet sind, müssen Sie zuerst [eine erstellen](#).
9. Wählen Sie den Port, der dem Ziel für den Virtual Service Provider-Port entspricht. Der Port des virtuellen Diensteanbieters ist erforderlich, wenn der Anbieter (Router oder Knoten) des ausgewählten virtuellen Dienstes mehrere Listener hat.
10. (Optional) Geben Sie unter Priorität die Priorität für diese Gateway-Route an.
11. Geben Sie für die Match-Konfiguration Folgendes an:

- Wenn http/http2 der gewählte Typ ist:
  - (Optionale) Methode - Gibt den Methodenheader an, der in den eingehenden http/ http2-Anfragen abgeglichen werden soll.
  - (Optional) Portübereinstimmung - Passen Sie den Port für eingehenden Verkehr an. Eine Portübereinstimmung ist erforderlich, wenn dieses virtuelle Gateway mehrere Listener hat.
  - (Optional) Exact/Suffix-Hostname - Gibt den Hostnamen an, der in der eingehenden Anfrage zur Weiterleitung an den virtuellen Zieldienst abgeglichen werden soll.
  - (Optional) Präfix/Genauer/Regex-Pfad - Die Methode, mit der der Pfad der URL abgeglichen wird.
  - Präfixübereinstimmung - Eine übereinstimmende Anfrage von einer Gateway-Route wird in den Namen des virtuellen Zieldienstes umgeschrieben/, und das passende Präfix wird standardmäßig in umgeschrieben. Je nachdem, wie Sie Ihren virtuellen Dienst konfigurieren, kann er einen virtuellen Router verwenden, um die Anfrage auf der Grundlage bestimmter Präfixe oder Header an verschiedene virtuelle Knoten weiterzuleiten.

 Important

- Sie können weder `/aws-appmesh*` oder `/aws-app-mesh*` als Präfix Match angeben. Diese Präfixe sind für die future interne Verwendung von App Mesh reserviert.
- Wenn mehrere Gateway-Routen definiert sind, wird eine Anfrage der Route mit dem längsten Präfix zugeordnet. Wenn beispielsweise zwei Gateway-Routen existieren, von denen eine das Präfix von `/chapter` und eine das Präfix von `hat/`, dann `www.example.com/chapter/` würde eine Anfrage für der Gateway-Route mit dem `/chapter` Präfix zugeordnet.

 Note

Wenn Sie den auf Pfad und Präfix basierenden Abgleich aktivieren, aktiviert App Mesh die Pfadnormalisierung ([normalize\\_path](#) und [merge\\_slashes](#)), um die Wahrscheinlichkeit von Sicherheitslücken durch Pfadverwirrung zu minimieren.

Sicherheitslücken durch Pfadverwirrung treten auf, wenn an der Anfrage beteiligte Parteien unterschiedliche Pfaddarstellungen verwenden.

- Exakte Übereinstimmung — Der exakte Parameter deaktiviert den teilweisen Abgleich für eine Route und stellt sicher, dass die Route nur zurückgegeben wird, wenn der Pfad EXAKT mit der aktuellen URL übereinstimmt.
- Regex-Match — Wird verwendet, um Muster zu beschreiben, bei denen mehrere URLs tatsächlich eine einzelne Seite auf der Website identifizieren können.
- (Optional) Abfrageparameter - In diesem Feld können Sie die Abfrageparameter abgleichen.
- (Optional) Header - Gibt die Header für http und http2 an. Es sollte mit der eingehenden Anfrage übereinstimmen, um an den virtuellen Zieldienst weitergeleitet zu werden..
- Wenn grpc der gewählte Typ ist:
  - Hostnamen-Übereinstimmungstyp und (optional) Genaue/Suffix-Übereinstimmung - Gibt den Hostnamen an, der bei der eingehenden Anfrage abgeglichen werden soll, um an den virtuellen Zieldienst weitergeleitet zu werden.
  - grpc-Servicename - Der grpc-Service fungiert als API für Ihre Anwendung und ist mit definiert ProtoBuf.

 Important

Sie können nicht/aws . app-mesh\* oderaws . appmesh für den Dienstenamen angeben. Diese Dienstenamen sind für die future interne Verwendung von App Mesh reserviert.

- (Optional) Metadaten - Gibt die Metadaten für grpc an. Es sollte mit der eingehenden Anfrage übereinstimmen, um an den virtuellen Zieldienst weitergeleitet zu werden.

## 12. (Optional) Für die Rewrite-Konfiguration:

- Wenn http/http2 der gewählte Typ ist:
  - Wenn Präfix der gewählte Match-Typ ist:
    - Automatisches Umschreiben des Hostnamens überschreiben - Standardmäßig wird der Hostname in den Namen des virtuellen Zieldienstes umgeschrieben.
    - Automatisches Umschreiben von Präfix überschreiben - Wenn diese Option aktiviert ist, gibt Prefix rewrite den Wert des umgeschriebenen Präfixes an.

- Wenn Exact Path der gewählte Match-Typ ist:
  - Überschreibt das automatische Umschreiben des Hostnamens — standardmäßig wird der Hostname in den Namen des virtuellen Zieldienstes umgeschrieben.
  - Path rewrite - Gibt den Wert des umgeschriebenen Pfads an. Kein Standardpfad.
- Wenn Regex Path der gewählte Match-Typ ist:
  - Überschreibt das automatische Umschreiben des Hostnamens — standardmäßig wird der Hostname in den Namen des virtuellen Zieldienstes umgeschrieben.
  - Path rewrite - Gibt den Wert des umgeschriebenen Pfads an. Kein Standardpfad.
- Wenn grpc der gewählte Typ ist:
  - Automatisches Umschreiben des Hostnamens überschreiben - Standardmäßig wird der Hostname in den Namen des virtuellen Zieldienstes umgeschrieben.

13. Wählen Sie zum Abschluss Gateway-Route erstellen.

## AWS CLI

Um eine Gateway-Route mit dem zu erstellen AWS CLI.

Erstellen Sie eine Gateway-Route mit dem folgenden Befehl und geben Sie JSON ein (ersetzen Sie die *roten* Werte durch Ihre eigenen):

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. Inhalt des Beispiels create-gateway-route .json:

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



Weitere Informationen zum Erstellen einer Gateway-Route mit dem AWS CLI for App Mesh finden Sie im [create-gateway-route](#) Befehl in der AWS CLI Referenz.

## Eine Gateway-Route löschen

### AWS Management Console

So löschen Sie eine Gateway-Route mit der AWS Management Console

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie das Mesh aus, aus dem Sie eine Gateway-Route löschen möchten. Alle Netze, die Sie besitzen und die mit Ihnen [geteilt](#) wurden, werden aufgelistet.
3. Wählen Sie in der linken Navigationsleiste Virtuelle Gateways aus.
4. Wählen Sie das virtuelle Gateway aus, von dem Sie eine Gateway-Route löschen möchten.
5. Wählen Sie in der Tabelle Gateway-Routen die Gateway-Route aus, die Sie löschen möchten, und klicken Sie auf Löschen. Sie können eine Gateway-Route nur löschen, wenn Ihr Konto als Ressourcenbesitzer aufgeführt ist.
6. Geben Sie in das Bestätigungsfeld ein **delete** und klicken Sie dann auf Löschen.

### AWS CLI

So löschen Sie eine Gateway-Route mit der AWS CLI

1. Verwenden Sie den folgenden Befehl, um Ihre Gateway-Route zu löschen (ersetzen Sie die *roten* Werte durch Ihre eigenen):

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

2. Beispielausgabe:

```
{
 "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"
}
}

```

Weitere Informationen zum Löschen einer Gateway-Route mit dem AWS CLI for App Mesh finden Sie im [delete-gateway-route](#) Befehl in der AWS CLI Referenz.

## Virtuelle Knoten

Ein virtueller Knoten fungiert als logischer Verweis auf eine bestimmte Aufgabengruppe, z. B. einen Amazon ECS-Service oder eine Kubernetes-Bereitstellung. Wenn Sie einen virtuellen Knoten erstellen, müssen Sie eine Methode zur Diensterkennung für Ihre Aufgabengruppe angeben. Jeglicher eingehender Datenverkehr, den Ihr virtueller Knoten erwartet, wird als Listener angegeben.

Jeder virtuelle Dienst, an den ein virtueller Knoten ausgehenden Datenverkehr sendet, wird als Backend angegeben.

Die Antwortmetadaten für Ihren neuen virtuellen Knoten enthalten den Amazon-Ressourcennamen (ARN), der dem virtuellen Knoten zugeordnet ist. Legen Sie diesen Wert als `APPMESH_RESOURCE_ARN` Umgebungsvariable für den Envoy-Proxy-Container Ihrer Aufgabengruppe in Ihrer Amazon ECS-Aufgabendefinition oder Kubernetes-Pod-Spezifikation fest. Der Wert könnte beispielsweise sein. `arn:aws:appmesh:us-west-2:111122223333:mesh/myMesh/virtualNode/myVirtualNode` Dieser wird dann den Envoy-Parametern `node.id` und `node.cluster` zugeordnet. Sie müssen das Envoy-Image `1.15.0` oder eine neuere Version verwenden, wenn Sie diese Variable setzen. Weitere Informationen zu App Mesh Envoy-Variablen finden Sie unter [Envoy](#).

### Note

Standardmäßig verwendet App Mesh den Namen der Ressource, die Sie in `APPMESH_RESOURCE_ARN` angegeben haben, wenn sich Envoy in Metriken und Ablaufverfolgungen auf sich selbst bezieht. Sie können dieses Verhalten übergehen, indem Sie die `APPMESH_RESOURCE_CLUSTER`-Umgebungsvariable mit Ihrem eigenen Namen festlegen.

## Einen virtuellen Knoten erstellen

### AWS Management Console

Um einen virtuellen Knoten mit dem zu erstellen AWS Management Console

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie das Mesh aus, in dem Sie den virtuellen Knoten erstellen möchten. Alle Meshes, die Sie besitzen und die mit Ihnen [geteilt](#) wurden, werden aufgelistet.
3. Wählen Sie im linken Navigationsbereich Virtual nodes (Virtuellen Knoten).
4. Wählen Sie Virtuellen Knoten erstellen und geben Sie dann die Einstellungen für Ihren virtuellen Knoten an.
5. Geben Sie unter Name des virtuellen Knotens einen Namen für Ihren virtuellen Knoten ein.
6. Wählen Sie für Service Discovery Method eine der folgenden Optionen aus:

- **DNS** — Geben Sie den DNS-Hostnamen des eigentlichen Dienstes an, für den der virtuelle Knoten steht. Der Envoy-Proxy wird in einer Amazon VPC bereitgestellt. Der Proxy sendet Anfragen zur Namensauflösung an den DNS-Server, der für die VPC konfiguriert ist. Wenn der Hostname aufgelöst wird, gibt der DNS-Server eine oder mehrere IP-Adressen zurück. Weitere Informationen zu VPC-DNS-Einstellungen finden Sie unter [DNS mit Ihrer VPC verwenden](#). Geben Sie für den DNS-Antworttyp (optional) die Typen von Endpunkten an, die vom DNS-Resolver zurückgegeben werden. Load Balancer bedeutet, dass der DNS-Resolver eine Gruppe von Endpunkten mit Lastenausgleich zurückgibt. Endpunkte bedeutet, dass der DNS-Resolver alle Endpunkte zurückgibt. Standardmäßig wird davon ausgegangen, dass der Antworttyp Load Balancer ist.

 Note

Wenn Sie Route53 verwenden, müssen Sie Load Balancer verwenden.

- **AWS Cloud Map**— Geben Sie einen vorhandenen Dienstenamen und einen HTTP-Namespace an. Optional können Sie auch Attribute angeben, die App Mesh abfragen AWS Cloud Map kann, indem Sie Zeile hinzufügen auswählen und einen Schlüssel und einen Wert angeben. Nur Instances, die mit allen der angegebenen Schlüssel-Wert-Paare übereinstimmen, werden zurückgegeben. Zur Verwendung AWS Cloud Map muss Ihr Konto über die `AWSServiceRoleForAppMesh` [dienstverknüpfte Rolle](#) verfügen. Weitere Informationen über AWS Cloud Map finden Sie im [AWS Cloud Map-Entwicklerleitfaden](#).
- **Keine** — Wählen Sie diese Option aus, wenn Ihr virtueller Knoten keinen eingehenden Datenverkehr erwartet.

## 7. Bevorzugte IP-Version

Steuern Sie, welche IP-Version für den Datenverkehr innerhalb des Meshs verwendet werden soll, indem Sie die Option Verhalten der Standard-IP-Version außer Kraft setzen aktivieren. Standardmäßig verwendet App Mesh eine Vielzahl von IP-Versionen.

 Note

Das Festlegen der IP-Einstellung auf dem virtuellen Knoten überschreibt nur die IP-Voreinstellung, die für das Mesh auf diesem speziellen Knoten festgelegt wurde.

- **Standard**

- Der DNS-Resolver von Envoy bevorzugt IPv6 und greift auf ihn zurück. IPv4
- Wir verwenden die von zurückgegebene IPv4 Adresse, AWS Cloud Map sofern verfügbar, und greifen auf die Verwendung der IPv6 Adresse zurück.
- Der für die lokale App erstellte Endpunkt verwendet eine IPv4 Adresse.
- Die Envoy-Listener binden sich an alle IPv4 Adressen.
- IPv6 bevorzugt
  - Der DNS-Resolver von Envoy bevorzugt IPv6 und greift auf ihn zurück. IPv4
  - Die von zurückgegebene IPv6 Adresse AWS Cloud Map wird verwendet, sofern verfügbar, und es wird auf die Verwendung der Adresse zurückgegriffen IPv4
  - Der für die lokale App erstellte Endpunkt verwendet eine IPv6 Adresse.
  - Die Envoy-Listener binden sich an alle End-AdressenIPv4. IPv6
- IPv4 bevorzugt
  - Der DNS-Resolver von Envoy bevorzugt IPv4 und greift auf ihn zurück. IPv6
  - Wir verwenden die von zurückgegebene IPv4 Adresse, AWS Cloud Map sofern verfügbar, und greifen auf die Verwendung der IPv6 Adresse zurück.
  - Der für die lokale App erstellte Endpunkt verwendet eine IPv4 Adresse.
  - Die Envoy-Listener binden sich an alle End-AdressenIPv4. IPv6
- Nur IPv6
  - Der DNS-Resolver von Envoy verwendet nur. IPv6
  - Es wird nur die von zurückgegebene IPv6 Adresse AWS Cloud Map verwendet. Wenn eine IPv4 Adresse AWS Cloud Map zurückgegeben wird, werden keine IP-Adressen verwendet und leere Ergebnisse werden an den Envoy zurückgegeben.
  - Der für die lokale App erstellte Endpunkt verwendet eine IPv6 Adresse.
  - Die Envoy-Listener binden sich an alle End-AdressenIPv4. IPv6
- Nur IPv4
  - Der DNS-Resolver von Envoy verwendet nur. IPv4
  - Es wird nur die von zurückgegebene IPv4 Adresse AWS Cloud Map verwendet. Wenn eine IPv6 Adresse AWS Cloud Map zurückgegeben wird, werden keine IP-Adressen verwendet und leere Ergebnisse werden an den Envoy zurückgegeben.
  - Der für die lokale App erstellte Endpunkt verwendet eine IPv4 Adresse.

8. (Optional) Standardeinstellungen für Client-Richtlinien — Konfigurieren Sie Standardanforderungen für die Kommunikation mit virtuellen Back-End-Diensten.

 Note

- Wenn Sie Transport Layer Security (TLS) für einen vorhandenen virtuellen Knoten aktivieren möchten, empfehlen wir Ihnen, einen neuen virtuellen Knoten zu erstellen, der denselben Dienst wie der vorhandene virtuelle Knoten darstellt, auf dem TLS aktiviert werden soll. Dann verlagern Sie den Verkehr mithilfe eines virtuellen Routers und einer Route schrittweise auf den neuen virtuellen Knoten. Weitere Informationen zum Erstellen einer Route und zum Anpassen der Gewichtungen für den Übergang finden Sie unter [Routen](#). Wenn Sie einen vorhandenen virtuellen Knoten, der den Datenverkehr bedient, mit TLS aktualisieren, besteht die Möglichkeit, dass die Envoy-Proxys des Downstream-Clients den TLS-Validierungskontext erhalten, bevor der Envoy-Proxy für den virtuellen Knoten, den Sie aktualisiert haben, das Zertifikat erhält. Dies kann zu TLS-Verhandlungsfehlern auf den nachgeschalteten Envoy-Proxys führen.
- Die [Proxyautorisierung](#) muss für den Envoy-Proxy aktiviert sein, der mit der Anwendung bereitgestellt wird, die durch die virtuellen Knoten des Back-End-Dienstes repräsentiert wird. Wir empfehlen, bei der Aktivierung der Proxyautorisierung den Zugriff nur auf die virtuellen Knoten zu beschränken, mit denen dieser virtuelle Knoten kommuniziert.

- (Optional) Wählen Sie TLS erzwingen aus, wenn Sie möchten, dass der virtuelle Knoten mithilfe von Transport Layer Security (TLS) mit allen Back-Ends kommuniziert.
- (Optional) Wenn Sie die Verwendung von TLS nur für einen oder mehrere bestimmte Ports vorschreiben möchten, geben Sie im Feld Ports eine Zahl ein. Um weitere Ports hinzuzufügen, wählen Sie Port hinzufügen aus. Wenn Sie keine Ports angeben, wird TLS für alle Ports erzwungen.
- Wählen Sie als Validierungsmethode eine der folgenden Optionen aus. Das von Ihnen angegebene Zertifikat muss bereits vorhanden sein und bestimmte Anforderungen erfüllen. Weitere Informationen finden Sie unter [Zertifikatanforderungen](#).
  - AWS Private Certificate Authority Hosting — Wählen Sie ein oder mehrere vorhandene Zertifikate aus. Eine vollständige Anleitung zur Bereitstellung eines Meshs mit einer Beispielanwendung, end-to-end die Verschlüsselung mit einem ACM-Zertifikat

verwendet, finden Sie unter [Konfiguration von TLS mit aktiviertem AWS Certificate Manager](#). GitHub

- Hosting durch den Envoy Secret Discovery Service (SDS) — Geben Sie den Namen des Geheimnisses ein, das der Envoy mithilfe des Secret Discovery Service abrufen wird.
  - Lokales Datei-Hosting — Geben Sie den Pfad zur Zertifikatskettendatei auf dem Dateisystem an, auf dem Envoy bereitgestellt wird. Eine vollständige Anleitung zur Bereitstellung eines Meshs mit einer Beispielanwendung, end-to-end die Verschlüsselung mit lokalen Dateien verwendet, finden Sie unter [Konfiguration von TLS mit in der Datei bereitgestellten TLS-Zertifikaten](#). GitHub
  - (Optional) Geben Sie einen alternativen Namen für den Betreff ein. Um weitere SANs hinzuzufügen, wählen Sie SAN hinzufügen. SANs müssen FQDN- oder URI-formatiert sein.
  - (Optional) Wählen Sie Clientzertifikat bereitstellen und eine der folgenden Optionen aus, um ein Client-Zertifikat bereitzustellen, wenn ein Server es anfordert, und um die gegenseitige TLS-Authentifizierung zu aktivieren. Weitere Informationen zu Mutual TLS finden Sie in den Dokumenten zur App Mesh [Mutual TLS Authentication](#).
    - Hosting durch den Envoy Secret Discovery Service (SDS) — Geben Sie den Namen des Geheimdienstes ein, den der Absender mithilfe des Secret Discovery Service abrufen wird.
    - Lokales Datei-Hosting — Geben Sie den Pfad zur Zertifikatskettendatei sowie den privaten Schlüssel auf dem Dateisystem an, auf dem Envoy bereitgestellt wird.
9. (Optional) Service-Backends — Geben Sie den virtuellen App Mesh Mesh-Dienst an, mit dem der virtuelle Knoten kommunizieren soll.
- Geben Sie einen virtuellen App Mesh Mesh-Dienstnamen oder den vollständigen Amazon Resource Name (ARN) für den virtuellen Service ein, mit dem Ihr virtueller Knoten kommuniziert.
  - (Optional) Wenn Sie eindeutige TLS-Einstellungen für ein Backend festlegen möchten, wählen Sie TLS-Einstellungen und dann Standardwerte überschreiben aus.
    - (Optional) Wählen Sie TLS erzwingen aus, wenn Sie möchten, dass der virtuelle Knoten über TLS mit allen Back-Ends kommuniziert.
    - (Optional) Wenn Sie die Verwendung von TLS nur für einen oder mehrere bestimmte Ports vorschreiben möchten, geben Sie im Feld Ports eine Zahl ein. Um weitere Ports hinzuzufügen, wählen Sie Port hinzufügen aus. Wenn Sie keine Ports angeben, wird TLS für alle Ports erzwungen.

- Wählen Sie als Validierungsmethode eine der folgenden Optionen aus. Das von Ihnen angegebene Zertifikat muss bereits vorhanden sein und bestimmte Anforderungen erfüllen. Weitere Informationen finden Sie unter [Zertifikatanforderungen](#).
- AWS Private Certificate Authority Hosting — Wählen Sie ein oder mehrere vorhandene Zertifikate aus.
- Hosting durch den Envoy Secret Discovery Service (SDS) — Geben Sie den Namen des Geheimdienstes ein, den der Gesandte mithilfe des Secret Discovery Service abrufen wird.
- Lokales Datei-Hosting — Geben Sie den Pfad zur Zertifikatskettendatei auf dem Dateisystem an, auf dem Envoy bereitgestellt wird.
- (Optional) Geben Sie einen alternativen Namen für den Betreff ein. Um weitere SANs hinzuzufügen, wählen Sie SAN hinzufügen. SANs müssen FQDN- oder URI-formatiert sein.
- (Optional) Wählen Sie Clientzertifikat bereitstellen und eine der folgenden Optionen aus, um ein Client-Zertifikat bereitzustellen, wenn ein Server es anfordert, und um die gegenseitige TLS-Authentifizierung zu aktivieren. Weitere Informationen zu Mutual TLS finden Sie in den Dokumenten zur App Mesh [Mutual TLS Authentication](#).
- Hosting durch den Envoy Secret Discovery Service (SDS) — Geben Sie den Namen des Geheimdienstes ein, den der Absender mithilfe des Secret Discovery Service abrufen wird.
- Lokales Datei-Hosting — Geben Sie den Pfad zur Zertifikatskettendatei sowie den privaten Schlüssel auf dem Dateisystem an, auf dem Envoy bereitgestellt wird.
- Um weitere Backends hinzuzufügen, wählen Sie Backend hinzufügen.

## 10. (Optional) Protokollierung

Um die Protokollierung zu konfigurieren, geben Sie den Pfad zu den HTTP-Zugriffsprotokollen ein, die Envoy verwenden soll. Wir empfehlen den `/dev/stdout` Pfad, damit Sie Docker-Protokolltreiber verwenden können, um Ihre Envoy-Protokolle in einen Service wie Amazon CloudWatch Logs zu exportieren.

**Note**

Die Protokolle müssen noch von einem Agenten in Ihrer Anwendung übernommen und an ein Ziel gesendet werden. Dieser Dateipfad teilt Envoy nur mit, wohin die Protokolle gesendet werden sollen.

## 11. Listener-Konfiguration

Listener unterstützen HTTP, HTTP/2/GRPC, und TCP Protokolle. HTTP wird nicht unterstützt.

- a. Wenn Ihr virtueller Knoten eingehenden Datenverkehr erwartet, geben Sie einen Port und ein Protokoll für den Listener an. Der HTTP-Listener ermöglicht den Verbindungsübergang zu Websockets. Sie können auf Listener hinzufügen klicken, um mehrere Listener hinzuzufügen. Mit der Schaltfläche Entfernen wird dieser Listener entfernt.
- b. (Optional) Aktivieren Sie den Verbindungspool

Das Verbindungspooling begrenzt die Anzahl der Verbindungen, die ein Envoy gleichzeitig mit dem lokalen Anwendungscluster herstellen kann. Es soll Ihre lokale Anwendung vor einer Überlastung durch Verbindungen schützen und ermöglicht es Ihnen, das Traffic Shaping an die Bedürfnisse Ihrer Anwendungen anzupassen.

Sie können zweiseitige Verbindungspool-Einstellungen für einen virtuellen Node-Listener konfigurieren. App Mesh setzt die clientseitigen Verbindungspool-Einstellungen standardmäßig auf unendlich, was die Mesh-Konfiguration vereinfacht.

**Note**

Die Protokolle ConnectionPool und PortMapping müssen identisch sein. Wenn Ihr Listener-Protokoll tcp ist, geben Sie nur MaxConnections an. Wenn Ihr Listener-Protokoll grpc oder http2 ist, geben Sie nur maxRequests an. Wenn Ihr Listener-Protokoll http ist, können Sie sowohl maxConnections als auch maxPendingRequests angeben.

- Geben Sie für Maximale Anzahl an Verbindungen die maximale Anzahl ausgehender Verbindungen an.

- (Optional) Geben Sie für Maximale Anzahl ausstehender Anfragen die Anzahl der überlaufenden Anfragen nach Maximum an Verbindungen an, die ein Envoy in die Warteschlange stellen soll. Der Standardwert ist 2147483647.
- c. (Optional) Aktivieren Sie die Erkennung von Ausreißern

Die Ausreißerererkennung wird auf dem Client angewendet. Envoy ermöglicht es Kunden, bei Verbindungen, bei denen bekannte fehlerhafte Fehler beobachtet wurden, nahezu sofort Maßnahmen zu ergreifen. Dabei handelt es sich um eine Art Schutzbrecherimplementierung, die den Integritätsstatus einzelner Hosts im Upstream-Service verfolgt.

Bei der Erkennung von Ausreißern wird dynamisch ermittelt, ob Endpoints in einem Upstream-Cluster eine andere Leistung erbringen als die anderen, und sie werden aus dem fehlerfreien Load-Balancing-Set entfernt.

 Note

Um die Ausreißerererkennung für einen virtuellen Serverknoten effektiv zu konfigurieren, kann die Diensterkennungsmethode für diesen virtuellen Knoten AWS Cloud Map entweder DNS sein, wobei das Antworttyp-Feld auf gesetzt ist. ENDPOINTS Wenn Sie die Methode zur Erkennung von DNS-Diensten mit dem Antworttyp als verwendenLOADBALANCER, wählt der Envoy-Proxy nur eine einzige IP-Adresse für das Routing zum Upstream-Dienst aus. Dadurch wird das Verhalten bei der Erkennung von Ausreißern, bei dem ein fehlerhafter Host aus einer Gruppe von Hosts ausgeworfen wird, zunichte gemacht. Weitere Informationen zum Verhalten des Envoy-Proxys in Bezug auf den Service Discovery-Typ finden Sie im Abschnitt Methode zur Diensterkennung.

- Geben Sie für Serverfehler die Anzahl der aufeinanderfolgenden 5xx-Fehler an, die für den Auswurf erforderlich sind.
- Geben Sie für das Intervall zur Erkennung von Ausreißern das Zeitintervall und die Einheit zwischen der Analyse des Auswurfvorgangs an.
- Geben Sie für Basisauswurfdauer den Basiszeitraum und die Einheit an, für die ein Host ausgeworfen wird.

- Geben Sie unter Auswurfprozentsatz den maximalen Prozentsatz der Hosts im Load Balancing-Pool an, die ausgeworfen werden können.
- d. (Optional) Integritätsprüfung aktivieren — Konfigurieren Sie Einstellungen für eine Integritätsprüfungsrichtlinie.

Eine Integritätsprüfungsrichtlinie ist optional. Wenn Sie jedoch Werte für eine Integritätsrichtlinie angeben, müssen Sie Werte für den Schwellenwert „Fehlerfrei“, das Intervall für die Integritätsprüfung, das Protokoll für die Integritätsprüfung, den Zeitraum für die Zeitüberschreitung und den Schwellenwert für fehlerhafte Daten angeben.

- Wählen Sie für Health Check Protocol ein Protokoll aus. Wenn Sie grpc wählen, muss Ihr Service dem [GRPC Health](#) Checking Protocol entsprechen.
  - Geben Sie unter Health check port (Zustandsprüfungsport) den Port an, auf dem die Zustandsprüfung ausgeführt werden soll.
  - Geben Sie für Healthy threshold (Schwellenwert für fehlerfreien Zustand) die Anzahl der fortlaufenden erfolgreichen Zustandsprüfungen an, die auftreten müssen, damit der Listener als fehlerfrei deklariert wird.
  - Geben Sie für Health check interval (Zustandsprüfungsintervall) den Zeitraum in Millisekunden zwischen den einzelnen Zustandsprüfungen an.
  - Geben Sie unter Path (Pfad) den Zielpfad für die Zustandsprüfungsanforderung an. Dieser Wert wird nur verwendet, wenn das Health Check-Protokoll ht tp oder istht tp2. Der Wert wird für andere Protokolle ignoriert.
  - Geben Sie unter Timeout period (Timeout-Zeit) an, wie lange in Millisekunden bei Empfang einer Antwort von der Zustandsprüfung gewartet werden soll.
  - Geben Sie unter Unhealthy threshold (Schwellenwert für fehlerhaften Zustand) die Anzahl der fortlaufenden fehlgeschlagenen Zustandsprüfungen an, die auftreten müssen, damit der Listener als fehlerhaft deklariert wird.
- e. (Optional) TLS-Terminierung aktivieren — Konfigurieren Sie, wie andere virtuelle Knoten mithilfe von TLS mit diesem virtuellen Knoten kommunizieren.
- Wählen Sie unter Modus den Modus aus, für den TLS auf dem Listener konfiguriert werden soll.
  - Wählen Sie für Certificate method eine der folgenden Optionen aus. Das Zertifikat muss bestimmte Anforderungen erfüllen. Weitere Informationen finden Sie unter [Zertifikatanforderungen](#).

- AWS Certificate Manager Hosting — Wählen Sie ein vorhandenes Zertifikat aus.
  - Hosting durch den Envoy Secret Discovery Service (SDS) — Geben Sie den Namen des Geheimdienstes ein, den der Gesandte mithilfe des Secret Discovery Service abrufen wird.
  - Lokales Datei-Hosting — Geben Sie den Pfad zur Zertifikatskettendatei sowie den privaten Schlüssel auf dem Dateisystem an, auf dem der Envoy-Proxy bereitgestellt wird.
  - (Optional) Wählen Sie Client-Zertifikate erforderlich und eine der folgenden Optionen aus, um die gegenseitige TLS-Authentifizierung zu aktivieren, wenn ein Client ein Zertifikat bereitstellt. Weitere Informationen zu Mutual TLS finden Sie in den Dokumenten zur App Mesh [Mutual TLS Authentication](#).
  - Hosting durch den Envoy Secret Discovery Service (SDS) — Geben Sie den Namen des Geheimdienstes ein, den der Absender mithilfe des Secret Discovery Service abrufen wird.
  - Lokales Datei-Hosting — Geben Sie den Pfad zur Zertifikatskettendatei auf dem Dateisystem an, auf dem Envoy bereitgestellt wird.
  - (Optional) Geben Sie einen alternativen Namen für den Betreff ein. Um weitere SANs hinzuzufügen, wählen Sie SAN hinzufügen. SANs müssen FQDN- oder URI-formatiert sein.
- f. (Optional) Timeouts

 Note

Wenn Sie ein Timeout angeben, das über dem Standard liegt, stellen Sie sicher, dass Sie einen virtuellen Router und eine Route mit einem Timeout einrichten, das über dem Standard liegt. Wenn Sie das Timeout jedoch auf einen Wert reduzieren, der unter dem Standardwert liegt, ist es optional, das Timeout unter Route zu aktualisieren. [Weitere Informationen finden Sie unter Routen.](#)

- Anforderungs-Timeout — Sie können ein Anforderungs-Timeout angeben, wenn Sie grpc, http oder http2 für das Protokoll des Listeners ausgewählt haben. Die Standardeinstellung ist 15 Sekunden. Der Wert 0 deaktiviert das Timeout.
- Dauer des Leerlaufs — Sie können für jedes Listener-Protokoll eine Dauer im Leerlauf angeben. Standardmäßig ist ein Zeitraum von 300 Sekunden festgelegt.

## 12. Wählen Sie zum Abschluss „Virtuellen Knoten erstellen“.

### AWS CLI

Um einen virtuellen Knoten mit dem zu erstellen AWS CLI.

Erstellen Sie mit dem folgenden Befehl und einer JSON-Eingabedatei einen virtuellen Knoten, der DNS für die Diensterkennung verwendet (ersetzen Sie die *roten* Werte durch Ihre eigenen):

1. 

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

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

3. Beispielausgabe:

```
{
 "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"
}
```

Weitere Informationen zum Erstellen eines virtuellen Knotens mit dem AWS CLI for App Mesh finden Sie im [create-virtual-node](#) Befehl in der AWS CLI Referenz.

## Löschen eines virtuellen Knotens

### Note

Sie können einen virtuellen Knoten nicht löschen, wenn er als Ziel in einer [Route](#) oder als Anbieter in einem [virtuellen Dienst](#) angegeben ist.

## AWS Management Console

Um einen virtuellen Knoten mit dem zu löschen AWS Management Console

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie das Mesh aus, aus dem Sie einen virtuellen Knoten löschen möchten. Alle Meshes, die Sie besitzen und die mit Ihnen [geteilt](#) wurden, werden aufgelistet.
3. Wählen Sie im linken Navigationsbereich Virtual nodes (Virtuellen Knoten).
4. Wählen Sie in der Tabelle Virtuelle Knoten den virtuellen Knoten aus, den Sie löschen möchten, und wählen Sie Löschen aus. Um einen virtuellen Knoten zu löschen, muss Ihre Konto-ID entweder in den Spalten Mesh-Besitzer oder Ressourcenbesitzer des virtuellen Knotens aufgeführt sein.
5. Geben Sie in das Bestätigungsfeld den Text ein **delete** und wählen Sie dann Löschen aus.

## AWS CLI

Um einen virtuellen Knoten mit dem zu löschen AWS CLI

1. Verwenden Sie den folgenden Befehl, um Ihren virtuellen Knoten zu löschen (ersetzen Sie die *roten* Werte durch Ihre eigenen):

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

2. Beispielausgabe:

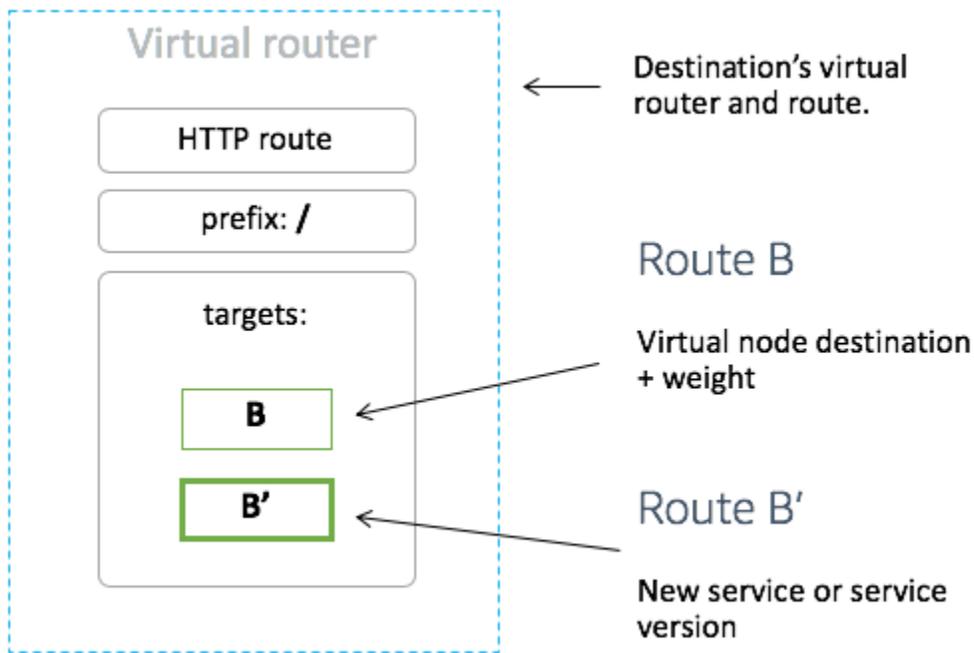
```
{
 "virtualNode": {
 "meshName": "meshName",
 "metadata": {
 "arn": "arn:aws:appmesh:us-west-2:210987654321:mesh/meshName/
virtualNode/nodeName",
 "createdAt": "2022-04-06T09:12:24.348000-05:00",
 "lastUpdatedAt": "2022-04-07T11:03:48.120000-05:00",
 "meshOwner": "123456789012",
 "resourceOwner": "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"
}
```

Weitere Informationen zum Löschen eines virtuellen Knotens mit dem AWS CLI for App Mesh finden Sie im [delete-virtual-node](#) Befehl in der AWS CLI Referenz.

## Virtuelle Router

Virtuelle Router verarbeiten den Datenverkehr für einen oder mehrere virtuelle Services innerhalb Ihres Gitters. Nachdem Sie einen virtuellen Router erstellt haben, können Sie Routen erstellen und Ihrem virtuellen Router zuordnen, über die eingehende Anforderungen an verschiedene virtuelle Knoten weitergeleitet werden.



Jeder eingehende Verkehr, den Ihr virtueller Router erwartet, sollte als Listener angegeben werden.

## Erstellen eines virtuellen Routers

### AWS Management Console

Um einen virtuellen Router mit dem zu erstellen AWS Management Console

#### Note

Wenn Sie einen virtuellen Router erstellen, müssen Sie einen Namespace-Selektor mit einer Bezeichnung hinzufügen, um die Liste der Namespaces zu identifizieren, die dem erstellten virtuellen Router zugewiesen werden sollen.

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie das Mesh aus, in dem Sie den virtuellen Router erstellen möchten. Alle Netze, die Sie besitzen und die mit Ihnen [geteilt](#) wurden, werden aufgelistet.
3. Wählen Sie im linken Navigationsbereich die Option Virtual routers (Virtuelle Router) aus.
4. Wählen Sie Create virtual router (Virtuellen Router erstellen).

5. Geben Sie für Virtual router name (Name des virtuellen Routers) einen Namen für Ihren virtuellen Router ein. Zulässig sind bis zu 255 Buchstaben, Ziffern, Bindestriche und Unterstriche.
6. (Optional) Geben Sie für die Listener-Konfiguration einen Port und ein Protokoll für Ihren virtuellen Router an. Der http Listener ermöglicht den Verbindungsübergang zu Websockets. Sie können auf Listener hinzufügen klicken, um mehrere Listener hinzuzufügen. Mit der Schaltfläche Entfernen wird dieser Listener entfernt.
7. Wählen Sie Create virtual router (Virtuellen Router erstellen), um den Vorgang abzuschließen.

## AWS CLI

Um einen virtuellen Router mit dem zu erstellen AWS CLI.

Erstellen Sie einen virtuellen Router mit dem folgenden Befehl und geben Sie JSON ein (ersetzen Sie die *roten* Werte durch Ihre eigenen):

1. 

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

2. Inhalt des Beispiels create-virtual-router .json

3. 

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

4. Beispielausgabe:

```
{
```

```
"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"
}
```

Weitere Informationen zum Erstellen eines virtuellen Routers mit dem AWS CLI for App Mesh finden Sie im [create-virtual-router](#) Befehl in der AWS CLI Referenz.

## Löschen eines virtuellen Routers

### Note

Sie können einen virtuellen Router nicht löschen, wenn er über [Routen](#) verfügt oder wenn er als Anbieter für einen [virtuellen Dienst](#) angegeben ist.

## AWS Management Console

Um einen virtuellen Router mit dem zu löschenAWS Management Console

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie das Mesh aus, aus dem Sie einen virtuellen Router löschen möchten. Alle Netze, die Sie besitzen und die mit Ihnen [geteilt](#) wurden, werden aufgelistet.
3. Wählen Sie im linken Navigationsbereich die Option Virtual routers (Virtuelle Router) aus.
4. Wählen Sie in der Tabelle Virtuelle Router den virtuellen Router aus, den Sie löschen möchten, und wählen Sie in der oberen rechten Ecke Löschen aus. Um einen virtuellen Router zu löschen, muss Ihre Konto-ID entweder in den Spalten Mesh-Besitzer oder Ressourcenbesitzer des virtuellen Routers aufgeführt sein.
5. Geben Sie in das Bestätigungsfeld ein**delete** und klicken Sie dann auf Löschen.

## AWS CLI

Um einen virtuellen Router mit dem zu löschenAWS CLI

1. Verwenden Sie den folgenden Befehl, um Ihren virtuellen Router zu löschen (ersetzen Sie die *roten* Werte durch Ihre eigenen):

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

2. Beispielausgabe:

```
{
 "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"
 }
}
```

Weitere Informationen zum Löschen eines virtuellen Routers mit dem AWS CLI for App Mesh finden Sie im [delete-virtual-router](#) Befehl in der AWS CLI Referenz.

## Routen

Eine Route ist mit einem virtuellen Router verknüpft. Die Route wird verwendet, um Anfragen für den virtuellen Router abzugleichen und den Datenverkehr an die zugehörigen virtuellen Knoten zu verteilen. Wenn eine Route mit einer Anforderung übereinstimmt, kann sie den Datenverkehr an einen oder mehrere virtuelle Zielknoten verteilen. Sie können die relative Gewichtung für jeden virtuellen Knoten angeben. Dieses Thema hilft Ihnen bei der Arbeit mit Routen in einem Servicegitter.

### Erstellen einer Route

#### AWS Management Console

Erstellen einer Route mithilfe der AWS Management Console

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie das Mesh aus, in dem Sie die Route erstellen möchten. Alle Netze, die Sie besitzen und die mit Ihnen [geteilt](#) wurden, werden aufgelistet.
3. Wählen Sie im linken Navigationsbereich die Option Virtual routers (Virtuelle Router) aus.

4. Wählen Sie den virtuellen Router aus, dem Sie eine neue Route zuordnen möchten. Wenn keine aufgelistet sind, müssen Sie zuerst [einen virtuellen Router erstellen](#).
5. Wählen Sie in der Tabelle Routes (Routen) die Option Create route (Route erstellen). Um eine Route zu erstellen, muss Ihre Konto-ID als Ressourcenbesitzer der Route aufgeführt sein.
6. Geben Sie unter Route name (Routen-Name) den Namen ein, der für Ihre Route verwendet werden soll.
7. Wählen Sie unter Routentyp das Protokoll aus, das Sie routen möchten. Das von Ihnen ausgewählte Protokoll muss mit dem Listener-Protokoll übereinstimmen, das Sie für Ihren virtuellen Router ausgewählt haben, und dem virtuellen Knoten, an den Sie den Datenverkehr weiterleiten.
8. (Optional) Geben Sie für Routenpriorität eine Priorität zwischen 0 und 1000 an, die für Ihre Route verwendet werden soll. Routen werden auf Grundlage des angegebenen Werts zugewiesen, wobei 0 die höchste Priorität darstellt.
9. (Optional) Wählen Sie Zusätzliche Konfiguration. Wählen Sie aus den unten aufgeführten Protokollen das Protokoll aus, das Sie für den Routentyp ausgewählt haben, und geben Sie die gewünschten Einstellungen in der Konsole an.
10. Wählen Sie für die Zielkonfiguration den vorhandenen virtuellen App Mesh-Knoten aus, an den der Datenverkehr weitergeleitet werden soll, und geben Sie eine Gewichtung an. Sie können Ziel hinzufügen wählen, um weitere Ziele hinzuzufügen. Der Prozentsatz für alle Ziele muss 100 ergeben. Wenn keine virtuellen Knoten aufgelistet sind, müssen Sie zuerst einen [erstellen](#). Wenn der ausgewählte virtuelle Knoten mehrere Listener hat, ist der Zielport erforderlich.
11. Geben Sie für die Match-Konfiguration Folgendes an:

Die Match-Konfiguration ist nicht verfügbar für `tcp`

- Wenn `http/http2` der gewählte Typ ist:
  - (Optionale) Methode - gibt den Methodenheader an, der in den eingehenden `http/ http2`-Anfragen abgeglichen werden soll.
  - (Optional) Portübereinstimmung - Passen Sie den Port für eingehenden Verkehr an. Eine Portübereinstimmung ist erforderlich, wenn dieser virtuelle Router mehrere Listener hat.
  - (Optional) Präfix/Genauer/Regex-Pfad - Methode zum Abgleich mit dem Pfad der URL.
    - Präfixübereinstimmung - Eine übereinstimmende Anfrage von einer Gateway-Route wird in den Namen des virtuellen Zieldienstes umgeschrieben und das passende

Präfix wird standardmäßig in umgeschrieben./ Je nachdem, wie Sie Ihren virtuellen Dienst konfigurieren, kann er einen virtuellen Router verwenden, um die Anfrage auf der Grundlage bestimmter Präfixe oder Header an verschiedene virtuelle Knoten weiterzuleiten.

 Note

Wenn Sie den auf Pfad und Präfix basierenden Abgleich aktivieren, aktiviert App Mesh die Pfadnormalisierung ([normalize\\_path](#) und [merge\\_slashes](#)), um die Wahrscheinlichkeit von Sicherheitslücken durch Pfadverwirrung zu minimieren.

Sicherheitslücken durch Pfadverwirrung treten auf, wenn an der Anfrage beteiligte Parteien unterschiedliche Pfaddarstellungen verwenden.

- Exakte Übereinstimmung — der exakte Parameter deaktiviert den teilweisen Abgleich für eine Route und stellt sicher, dass die Route nur zurückgegeben wird, wenn der Pfad EXAKT mit der aktuellen URL übereinstimmt.
- Regex-Match — Wird verwendet, um Muster zu beschreiben, bei denen mehrere URLs tatsächlich eine einzelne Seite auf der Website identifizieren können.
- (Optional) Abfrageparameter - In diesem Feld können Sie die Abfrageparameter abgleichen.
- (Optional) Headers - gibt die Header für http und http2 an. Es sollte mit der eingehenden Anfrage übereinstimmen, um an den virtuellen Zieldienst weitergeleitet zu werden..
- Wenn grpc der gewählte Typ ist:
  - Dienstname - der Zieldienst, für den die Anfrage erfüllt werden soll.
  - Methodenname - die Zielmethode, für die die Anfrage ermittelt werden soll.
  - (Optional) Metadaten - spezifiziert das,Mat ch was auf dem Vorhandensein von Metadaten basiert. Alle müssen übereinstimmen, damit die Anfrage bearbeitet werden kann.

12. Wählen Sie Route erstellen aus.

## AWS CLI

Um eine Route mit dem zu erstellenAWS CLI.

Erstellen Sie eine gRPC-Route mit dem folgenden Befehl und geben Sie JSON ein (ersetzen Sie die *roten* Werte durch Ihre eigenen):

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

- Inhalt des Beispiels 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. Beispielausgabe:

```

{
  "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"
}
}

```

Weitere Informationen zum Erstellen einer Route mit dem AWS CLI for App Mesh finden Sie unter dem Befehl [create-route](#) in der AWS CLI Referenz.

gRPC

(Optionales) Spiel

- (Optional) Geben Sie den Dienstnamen des Zieldienstes ein, der der Anfrage entspricht. Wenn Sie keinen Namen angeben, werden Anforderungen an einen beliebigen Service zugeordnet.
- (Optional) Geben Sie den Methodennamen der Zielmethode ein, die der Anfrage entspricht. Wenn Sie keinen Namen angeben, werden Anforderungen an eine beliebige Methode abgeglichen. Wenn Sie einen Methodennamen angeben, müssen Sie einen Servicegitter angeben.

(Optionale) Metadaten

Wählen Sie **Add metadata** (Metadaten hinzufügen) aus.

- (Optional) Geben Sie den Metadatennamen ein, auf dessen Grundlage Sie eine Route erstellen möchten, wählen Sie einen Übereinstimmungstyp und geben Sie einen Übereinstimmungswert ein. Wenn Sie **Umkehren** wählen, wird das Gegenteil erreicht. Wenn Sie beispielsweise den Metadatennamen `vonmyMetadata`, den Vergleichstyp **Exakt**, einen Übereinstimmungswert `von123` angeben und **Umkehren** auswählen, wird die Route für jede Anfrage abgeglichen, deren Metadatenname mit einem anderen als `beginnt123`.
- (Optional) Wählen Sie **Metadaten hinzufügen** aus, um bis zu zehn Metadatenelemente hinzuzufügen.

(Optionale) Richtlinie für Wiederholungsversuche

Eine Wiederholungsrichtlinie ermöglicht Clients, sich vor zeitweiligen Netzwerkausfällen oder zeitweiligen serverseitigen Ausfällen zu schützen. Eine Wiederholungsrichtlinie ist zwar optional, wird aber empfohlen. Die Timeout-Werte für den Wiederholungsversuch definieren das Timeout pro Wiederholungsversuch (einschließlich des ersten Versuchs). Wenn Sie keine Wiederholungsrichtlinie definieren, erstellt App Mesh möglicherweise automatisch eine Standardrichtlinie für jede Ihrer Routen. Weitere Informationen finden Sie unter [Standardrichtlinie für die Wiederholung von Routen](#).

- Geben Sie für Wiederholungs-Timeout die Anzahl der Einheiten für die Timeout-Dauer ein. Ein Wert ist erforderlich, wenn Sie ein beliebiges Protokollwiederholungsereignis auswählen.
- Wählen Sie als Einheit für Wiederholungs-Timeout eine Einheit aus. Ein Wert ist erforderlich, wenn Sie ein beliebiges Protokollwiederholungsereignis auswählen.
- Geben Sie unter **Max. Wiederholungsversuche** die maximale Anzahl der Wiederholungsversuche ein, die bei einem Fehlschlagen der Anforderung durchgeführt werden müssen. Ein Wert ist erforderlich, wenn Sie ein beliebiges Protokollwiederholungsereignis auswählen. Wir empfehlen einen Wert von mindestens zwei.
- Wählen Sie ein oder mehrere HTTP-Wiederholungsereignisse aus. Wir empfehlen, mindestens **Stream-Error** und **Gateway-Error** auszuwählen.
- Wählen Sie ein TCP-Wiederholungsereignis aus.
- Wählen Sie ein oder mehrere gRPC-Wiederholungsereignisse aus. Wir empfehlen, mindestens **storniert** und **nicht verfügbar** auszuwählen.

(Optionale) Timeouts

- Standardmäßig ist ein Zeitraum von 15 Sekunden festgelegt. Wenn Sie eine Wiederholungsrichtlinie angegeben haben, sollte die hier angegebene Dauer immer größer oder gleich der Wiederholungsdauer multipliziert mit der maximalen Anzahl von Wiederholungen sein, die Sie in der Wiederholungsrichtlinie definiert haben, damit Ihre Wiederholungsrichtlinie abgeschlossen werden kann. Wenn Sie eine Dauer von mehr als 15 Sekunden angeben, stellen Sie sicher, dass das für den Listener eines beliebigen virtuellen Knotenziels angegebene Timeout ebenfalls länger als 15 Sekunden ist. Weitere Informationen finden Sie unter [Virtuelle Knoten](#).
- Ein Wert von 0 deaktiviert das Timeout.
- Die maximale Zeitspanne, die die Route im Leerlauf sein kann.

HTTP und HTTP/2

(Optionales) Spiel

- Geben Sie das Präfix an, dem die Route entsprechen soll. Beispiel: Wenn der Name Ihres virtuellen Service `service-b.local` lautet und die Route Anforderungen `service-b.local/metrics` zuordnen soll, sollte das Präfix `/metrics` lauten. Die Angabe `/` leitet den gesamten Verkehr weiter.
- (Optional) Wählen Sie eine Methode aus.
- (Optional) Wählen Sie ein Schema aus. Gilt nur für HTTP2-Routen.

(Optionale) Kopfzeilen

- (Optional) Wählen Sie Header hinzufügen aus. Geben Sie den Header-Namen ein, auf dessen Grundlage Sie weiterleiten möchten, wählen Sie einen Match-Typ aus und geben Sie einen Übereinstimmungswert ein. Wenn Sie Umkehren wählen, wird das Gegenteil erreicht. Wenn Sie beispielsweise einen Header `ClientRequestId` mit dem Präfix von `123` angeben und Umkehren auswählen, gilt die Route für jede Anfrage, deren Header mit einem anderen als `123` beginnt.
- (Optional) Wählen Sie Header hinzufügen aus. Sie können bis zu zehn Header hinzufügen.

(Optionale) Richtlinie für Wiederholungsversuche

Eine Wiederholungsrichtlinie ermöglicht Clients, sich vor zeitweiligen Netzwerkausfällen oder zeitweiligen serverseitigen Ausfällen zu schützen. Eine Wiederholungsrichtlinie ist zwar optional,

wird aber empfohlen. Die Timeout-Werte für den Wiederholungsversuch definieren das Timeout pro Wiederholungsversuch (einschließlich des ersten Versuchs). Wenn Sie keine Wiederholungsrichtlinie definieren, erstellt App Mesh möglicherweise automatisch eine Standardrichtlinie für jede Ihrer Routen. Weitere Informationen finden Sie unter [Standardrichtlinie für die Wiederholung von Routen](#).

- Geben Sie für Wiederholungs-Timeout die Anzahl der Einheiten für die Timeout-Dauer ein. Ein Wert ist erforderlich, wenn Sie ein beliebiges Protokollwiederholungsereignis auswählen.
- Wählen Sie als Einheit für Wiederholungs-Timeout eine Einheit aus. Ein Wert ist erforderlich, wenn Sie ein beliebiges Protokollwiederholungsereignis auswählen.
- Geben Sie unter Max. Wiederholungsversuche die maximale Anzahl der Wiederholungsversuche ein, die bei einem Fehlschlagen der Anforderung durchgeführt werden müssen. Ein Wert ist erforderlich, wenn Sie ein beliebiges Protokollwiederholungsereignis auswählen. Wir empfehlen einen Wert von mindestens zwei.
- Wählen Sie ein oder mehrere HTTP-Wiederholungsereignisse aus. Wir empfehlen, mindestens Stream-Error und Gateway-Error auszuwählen.
- Wählen Sie ein TCP-Wiederholungsereignis aus.

(Optionale) Timeouts

- Anforderungs-Timeout — Die Standardeinstellung ist 15 Sekunden. Wenn Sie eine Wiederholungsrichtlinie angegeben haben, sollte die hier angegebene Dauer immer größer oder gleich der Wiederholungsdauer multipliziert mit der maximalen Anzahl von Wiederholungen sein, die Sie in der Wiederholungsrichtlinie definiert haben, damit Ihre Wiederholungsrichtlinie abgeschlossen werden kann.
- Leerlaufdauer — Die Standardeinstellung ist 300 Sekunden.
- Ein Wert von 0 deaktiviert das Timeout.

Note

Wenn Sie ein höheres als das Standardzeitlimit angeben, stellen Sie sicher, dass das für den Listener für alle virtuellen Knotenteilnehmer angegebene Timeout ebenfalls größer als der Standardwert ist. Wenn Sie den Timeout jedoch auf einen Wert verringern, der unter dem Standardwert liegt, ist es optional, die Timeouts an virtuellen Knoten zu aktualisieren. Weitere Informationen finden Sie unter [Virtuelle Knoten](#).

TCP

(Optionale) Timeouts

- Leerlaufdauer — Die Standardeinstellung ist 300 Sekunden.
- Ein Wert von 0 deaktiviert das Timeout.

Löschen einer Route

AWS Management Console

So löschen Sie eine Route mit der AWS Management Console

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie das Mesh aus, aus dem Sie eine Route löschen möchten. Alle Netze, die Sie besitzen und die mit Ihnen [geteilt](#) wurden, werden aufgelistet.
3. Wählen Sie im linken Navigationsbereich die Option Virtual routers (Virtuelle Router) aus.
4. Wählen Sie den Router aus, von dem Sie eine Route löschen möchten.
5. Wählen Sie in der Tabelle Routen die Route aus, die Sie löschen möchten, und wählen Sie in der oberen rechten Ecke Löschen aus.
6. Geben Sie in das Bestätigungsfeld ein **delete** und klicken Sie dann auf Löschen.

AWS CLI

So löschen Sie eine Route mit der AWS CLI

1. Verwenden Sie den folgenden Befehl, um Ihre Route zu löschen (ersetzen Sie die *roten* Werte durch Ihre eigenen):

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

2. Beispielausgabe:

```
{  
  "route": {
```

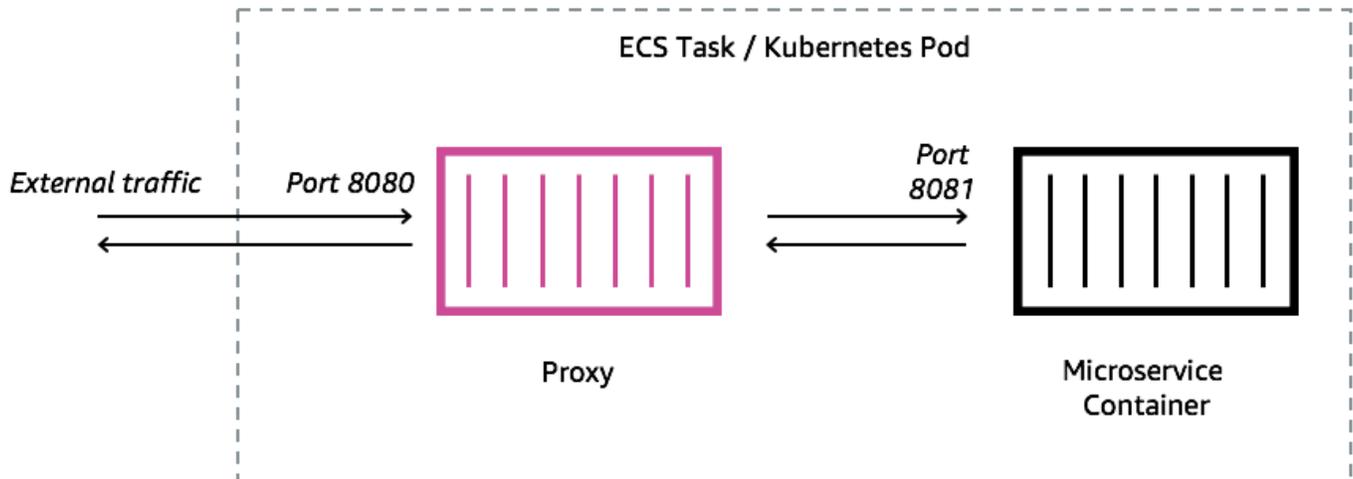
```
    "meshName": "meshName",
    "metadata": {
      "arn": "arn:aws:appmesh:us-west-2:210987654321:mesh/meshName/
virtualRouter/routerName/route/routeName",
      "createdAt": "2022-04-06T13:46:54.750000-05:00",
      "lastUpdatedAt": "2022-04-07T10:43:57.152000-05:00",
      "meshOwner": "123456789012",
      "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"
}
}
```

Weitere Informationen zum Löschen einer Route mit dem AWS CLI for App Mesh finden Sie im Befehl [delete-route](#) in der AWS CLI Referenz.

Bild des Gesandten

AWS App Mesh ist ein Service Mesh, das auf dem [Envoy-Proxy](#) basiert.



Sie müssen der Amazon ECS-Aufgabe, dem Kubernetes-Pod oder der Amazon EC2 EC2-Instance, die durch Ihren App Mesh Mesh-Endpunkt repräsentiert wird, einen Envoy-Proxy hinzufügen, z. B. einen virtuellen Knoten oder ein virtuelles Gateway. App Mesh verkauft ein Envoy-Proxy-Container-Image, das mit den neuesten Sicherheitslücken- und Leistungsupdates gepatcht ist. App Mesh testet jede neue Envoy-Proxy-Version anhand des App Mesh Mesh-Funktionsumfangs, bevor Ihnen ein neues Image zur Verfügung gestellt wird.

Envoy-Bildvarianten

App Mesh bietet zwei Varianten des Envoy-Proxy-Container-Images. Der Unterschied zwischen den beiden besteht darin, wie der Envoy-Proxy mit der App Mesh Mesh-Datenebene kommuniziert und wie die Envoy-Proxys miteinander kommunizieren. Eines ist ein Standard-Image, das mit den Standard-App Mesh-Dienstendpunkten kommuniziert. Die andere Variante ist FIPS-konform, sie kommuniziert mit den App Mesh Mesh-FIPS-Dienstendpunkten und erzwingt FIPS-Kryptografie bei der TLS-Kommunikation zwischen App Mesh Mesh-Diensten.

[Sie können entweder ein regionales Bild aus der folgenden Liste oder ein Bild aus unserem öffentlichen Repository mit dem Namen auswählen.](#) aws-appmesh-envoy

⚠ Important

- Ab dem 30. Juni 2023 ist nur Envoy Image v1.17.2.0-prod oder höher für die Verwendung mit App Mesh kompatibel. Für aktuelle Kunden, die bereits ein Envoy-Image verwendet haben v1.17.2.0, empfehlen wir dringend, auf die neueste Version zu migrieren, obwohl bestehende Envoys weiterhin kompatibel sein werden.
- Als bewährte Methode wird dringend empfohlen, die Envoy-Version regelmäßig auf die neueste Version zu aktualisieren. Nur die neueste Envoy-Version wird mit den neuesten Sicherheitspatches, Feature-Releases und Leistungsverbesserungen validiert.
- Die Version 1.17 war ein wichtiges Update für Envoy. Weitere Informationen finden Sie unter [Aktualisierung/Migration auf Envoy 1.17](#).
- Version oder höher 1.20.0.1 ist kompatibel. ARM64
- Für den IPv6 Support ist die Envoy-Version 1.20 oder höher erforderlich.

Alle [unterstützten](#) Regionen außer me-south-1, ap-east-1, ap-southeast-3, eu-south-1, il-central-1, und af-south-1. Sie können den *Regionalcode* durch eine beliebige Region außer me-south-1, ap-east-1, ap-southeast-3, eu-south-1, il-central-1, und ersetzen. af-south-1

Standard

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

me-south-1**Standard**

```
772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
772975370895.dkr.ecr.me-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

ap-east-1

Standard

```
856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
856666278305.dkr.ecr.ap-east-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

ap-southeast-3

Standard

```
909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
909464085924.dkr.ecr.ap-southeast-3.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

eu-south-1

Standard

```
422531588944.dkr.ecr.eu-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
422531588944.dkr.ecr.eu-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

il-central-1

Standard

```
564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
564877687649.dkr.ecr.il-central-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

af-south-1

Standard

```
924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
924023996002.dkr.ecr.af-south-1.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

Public repository

Standard

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.27.3.0-prod
```

FIPS-konform

```
public.ecr.aws/appmesh/aws-appmesh-envoy:v1.27.3.0-prod-fips
```

Note

Wir empfehlen, dem Envoy-Container 512 CPU-Einheiten und mindestens 64 MiB Arbeitsspeicher zuzuweisen. Auf Fargate ist die niedrigste Speichermenge, die Sie einstellen können, 1024 MiB Speicher. Die Ressourcenzuweisung für den Envoy-Container kann erhöht werden, wenn Erkenntnisse aus dem Container oder andere Messwerte darauf hindeuten, dass aufgrund der höheren Auslastung nicht genügend Ressourcen zur Verfügung stehen.

Note

Alle `aws-appmesh-envoy` Image-Release-Versionen, beginnend mit, `v1.22.0.0` werden als Docker-Image ohne Distribution erstellt. Wir haben diese Änderung vorgenommen, um

die Image-Größe zu reduzieren und unsere Sicherheitsanfälligkeit in ungenutzten Paketen, die im Image vorhanden sind, zu verringern. Wenn Sie auf einem `aws-appmesh-envoy` Image aufbauen und sich auf einige der AL2-Basispakete (z. B. `yum`) und Funktionen verlassen, empfehlen wir Ihnen, die Binärdateien aus einem Image zu kopieren, um ein neues `aws-appmesh-envoy` Docker-Image mit AL2-Basis zu erstellen.

Führen Sie dieses Skript aus, um ein benutzerdefiniertes Docker-Image mit dem Tag zu generieren `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 .
```

Der Zugriff auf dieses Container-Image in Amazon ECR wird von AWS Identity and Access Management (IAM) gesteuert. Daher müssen Sie IAM verwenden, um zu überprüfen, ob Sie Lesezugriff auf Amazon ECR haben. Wenn Sie beispielsweise Amazon ECS verwenden, können Sie einer Amazon ECS-Aufgabe eine entsprechende Aufgabenausführungsrolle zuweisen. Wenn Sie IAM-Richtlinien verwenden, die den Zugriff auf bestimmte Amazon ECR-Ressourcen einschränken, stellen Sie sicher, dass Sie den Zugriff auf den regionsspezifischen Amazon-Ressourcennamen (ARN) zulassen, der `aws-appmesh-envoy` das Repository identifiziert. In der `us-west-2` Region gewähren Sie beispielsweise den Zugriff auf die folgende Ressource: `arn:aws:ecr:us-west-2:840364872350:repository/aws-appmesh-envoy`. Weitere Informationen finden Sie unter [Amazon ECR-verwaltete Richtlinien](#). Wenn Sie Docker auf einer Amazon EC2 EC2-Instance verwenden, authentifizieren Sie Docker im Repository. Weitere Informationen finden Sie unter [Registrierungsauthentifizierung](#).

Wir veröffentlichen gelegentlich neue App Mesh Mesh-Funktionen, die von Envoy-Änderungen abhängen, die noch nicht mit den Upstream-Envoy-Images zusammengeführt wurden. Um diese neuen App Mesh Mesh-Funktionen zu verwenden, bevor die Envoy-Änderungen im Upstream zusammengeführt werden, müssen Sie das von App Mesh angebotene Envoy-Container-Image verwenden. Eine Liste der Änderungen finden Sie in der [App Mesh GitHub Mesh-Roadmap Probleme](#) mit dem Envoy Upstream Label. Wir empfehlen, das App Mesh Envoy-Container-Image als die am besten unterstützte Option zu verwenden.

Envoy-Konfigurationsvariablen

Verwenden Sie die folgenden Umgebungsvariablen, um die Envoy-Container für Ihre virtuellen App Mesh Mesh-Node-Aufgabengruppen zu konfigurieren.

Note

App Mesh Envoy 1.17 unterstützt die v2 xDS-API von Envoy nicht. Wenn Sie [Envoy-Konfigurationsvariablen verwenden, die Envoy-Konfigurationsdateien](#) akzeptieren, müssen diese auf die neueste v3-xDS-API aktualisiert werden.

Erforderliche Variablen

Die folgende Umgebungsvariable ist für alle App Mesh Envoy-Container erforderlich. Diese Variable kann nur mit der Version 1.15.0 oder einer späteren Version des Envoy-Images verwendet werden. Wenn Sie eine frühere Version des Images verwenden, müssen Sie stattdessen die APPMESH_VIRTUAL_NODE_NAME Variable festlegen.

APPMESH_RESOURCE_ARN

Wenn Sie den Envoy-Container zu einer Aufgabengruppe hinzufügen, setzen Sie diese Umgebungsvariable auf den ARN des virtuellen Knotens oder des virtuellen Gateways, das die Aufgabengruppe darstellt. Die folgende Liste enthält Beispiel-ARNs:

- Virtueller Knoten – *arn:aws:appmesh:Regionalcode:111122223333:mesh/MeshName/VirtualNode/virtualNodeName*
- Virtuelles Gateway – *arn:aws:appmesh:Regionalcode:111122223333:mesh/MeshName/VirtualGateway/virtualGatewayName*

Bei Verwendung des [App Mesh Mesh-Vorschaukanals](#) müssen ARNs die Region *us-west-2* verwenden und stattdessen verwenden `appmesh-preview`. Zum Beispiel ist der ARN eines virtuellen Knotens im App Mesh Preview Channel `arn:aws:appmesh-preview:us-west-2:111122223333:mesh/meshName/virtualNode/virtualNodeName`.

Optionale Variablen

Die folgende Umgebungsvariable ist für App Mesh Envoy-Container optional.

ENVOY_LOG_LEVEL

Gibt die Protokollebene für den Envoy-Container an.

Zulässige Werte: `trace`, `debug`, `info`, `warn`, `error`, `critical`, `off`

Standard: `info`

ENVOY_INITIAL_FETCH_TIMEOUT

Gibt an, wie lange Envoy während des Initialisierungsvorgangs auf die erste Konfigurationsantwort vom Management-Server wartet.

Weitere Informationen finden Sie unter [Konfigurationsquellen](#) in der Envoy-Dokumentation. Wenn auf `0` gesetzt, gibt es kein Timeout.

Standard: `0`

ENVOY_CONCURRENCY

Legt die `--concurrency` Befehlszeilenoption beim Starten des Envoy fest. Dies ist standardmäßig nicht festgelegt. Diese Option ist ab der Envoy-Version `v1.24.0.0-prod` oder höher verfügbar.

Weitere Informationen finden Sie unter [Befehlszeilenoptionen](#) in der Envoy-Dokumentation.

Admin-Variablen

Verwenden Sie diese Umgebungsvariablen, um die Administrationsoberfläche von Envoy zu konfigurieren.

ENVOY_ADMIN_ACCESS_PORT

Geben Sie einen benutzerdefinierten Admin-Port an, den Envoy abhören soll. Standard: `9901`.

Note

Der Envoy-Admin-Port sollte sich von jedem Listener-Port auf dem virtuellen Gateway oder virtuellen Knoten unterscheiden

ENVOY_ADMIN_ACCESS_LOG_FILE

Geben Sie einen benutzerdefinierten Pfad an, in den Envoy-Zugriffsprotokolle geschrieben werden sollen. Standard: `/tmp/envoy_admin_access.log`.

ENVOY_ADMIN_ACCESS_ENABLE_IPV6

Schaltet die Administrationsoberfläche von Envoy so um, dass sie IPv6 Datenverkehr akzeptiert, sodass diese Schnittstelle sowohl IPv4 Datenverkehr als auch Datenverkehr akzeptieren kann. IPv6 Standardmäßig ist dieses Flag auf False gesetzt und Envoy hört nur den Datenverkehr ab. IPv4 Diese Variable kann nur mit der Envoy-Image-Version 1.22.0 oder höher verwendet werden.

Agentenvariablen

Verwenden Sie diese Umgebungsvariablen, um den AWS App Mesh Agenten für Envoy zu konfigurieren. Weitere Informationen finden Sie unter App Mesh [Agent for Envoy](#).

APPNET_ENVOY_RESTART_COUNT

Gibt an, wie oft der Agent den Envoy-Proxyprozess innerhalb einer laufenden Aufgabe oder eines Pods neu startet, wenn dieser beendet wird. Der Agent protokolliert außerdem bei jedem Beenden von Envoy den Exit-Status, um die Fehlerbehebung zu erleichtern. Der Standardwert dieser Variablen ist `0`. Wenn der Standardwert festgelegt ist, versucht der Agent nicht, den Prozess neu zu starten.

Standard: `0`

Maximum: `10`

PID_POLL_INTERVAL_MS

Gibt das Intervall in Millisekunden an, in dem der Prozessstatus des Envoy-Proxys vom Agenten überprüft wird. Der Standardwert ist `100`.

Standard: `100`

Minimum: 100

Maximum: 1000

LISTENER_DRAIN_WAIT_TIME_S

Gibt die Zeitspanne in Sekunden an, für die der Envoy-Proxy auf das Schließen aktiver Verbindungen wartet, bevor der Prozess beendet wird.

Standard: 20

Minimum: 5

Maximum: 110

APPNET_AGENT_ADMIN_MODE

Startet den Management-Interface-Server des Agenten und bindet ihn entweder an eine TCP-Adresse oder einen Unix-Socket.

Zulässige Werte: tcp, uds

APPNET_AGENT_HTTP_PORT

Geben Sie einen Port an, der für die Bindung der Verwaltungsschnittstelle des Agenten im tcp Modus verwendet werden soll. Stellen Sie sicher, dass der Portwert > 1024 if istuid! =0. Stellen Sie sicher, dass der Port kleiner ist als 65535.

Standard: 9902

APPNET_AGENT_ADMIN_UDS_PATH

Geben Sie den Unix-Domain-Socket-Pfad für die Verwaltungsschnittstelle des Agenten im uds Modus an.

Standard: /var/run/ecs/appnet_admin.sock

Variablen verfolgen

Sie können keinen oder einen der folgenden Ablaufverfolgungstreiber konfigurieren.

AWS X-Ray Variablen

Verwenden Sie die folgenden Umgebungsvariablen, um App Mesh mit zu konfigurieren AWS X-Ray. Weitere Informationen finden Sie im [AWS X-Ray -Entwicklerhandbuch](#).

ENABLE_ENVOY_XRAY_TRACING

Aktiviert die Röntgenverfolgung `127.0.0.1:2000` als Standard-Daemon-Endpunkt. Um ihn zu aktivieren, setzen Sie den Wert auf `1`. Der Standardwert ist `0`.

XRAY_DAEMON_PORT

Geben Sie einen Portwert an, um den standardmäßigen X-Ray-Daemon-Port zu überschreiben: `2000`.

XRAY_SAMPLING_RATE

Geben Sie eine Abtastrate an, um die standardmäßige Abtastrate des Röntgen-Tracers von `0.05` (5%) zu überschreiben. Geben Sie den Wert als Dezimalzahl zwischen `0` und `1.00` (100%) an. Dieser Wert wird überschrieben, wenn er angegeben `XRAY_SAMPLING_RULE_MANIFEST` ist. Diese Variable wird mit Envoy-Images der Version `v1.19.1.1-prod` und höher unterstützt.

XRAY_SAMPLING_RULE_MANIFEST

Geben Sie einen Dateipfad im Envoy-Container-Dateisystem an, um die lokalisierten benutzerdefinierten Sampling-Regeln für den X-Ray Tracer zu konfigurieren. Weitere Informationen finden Sie unter [Sampling-Regeln](#) im AWS X-Ray Developer Guide. Diese Variable wird mit Envoy-Images der Version `v1.19.1.0-prod` und höher unterstützt.

XRAY_SEGMENT_NAME

Geben Sie einen Segmentnamen für Traces an, um den standardmäßigen X-Ray-Segmentnamen zu überschreiben. Standardmäßig wird dieser Wert als `gesetzmesh/resourceName`. Diese Variable wird mit der Envoy-Image-Version `v1.23.1.0-prod` oder höher unterstützt.

Datadog-Tracing-Variablen

Die folgenden Umgebungsvariablen helfen Ihnen bei der Konfiguration von App Mesh mit dem Datadog Agent Tracer. Weitere Informationen finden Sie unter [Agentenkonfiguration](#) in der Datadog-Dokumentation.

ENABLE_ENVOY_DATADOG_TRACING

Aktiviert die Datadog-Trace-Erfassung `127.0.0.1:8126` als Standard-Datadog-Agent-Endpunkt. Um ihn zu aktivieren, setzen Sie den Wert auf `1` (der Standardwert ist). `0`

DATADOG_TRACER_PORT

Geben Sie einen Portwert an, um den standardmäßigen Datadog-Agent-Port zu überschreiben:.
8126

DATADOG_TRACER_ADDRESS

Geben Sie eine IP-Adresse an, um die Standardadresse des Datadog-Agenten zu überschreiben:.
127.0.0.1

DD_SERVICE

Geben Sie einen Dienstnamen für Traces an, um den standardmäßigen Datadog-Dienstnamen zu überschreiben:/. envoy-meshName virtualNodeName Diese Variable wird mit Envoy-Images der Version v1.18.3.0-prod und höher unterstützt.

Jaeger-Tracing-Variablen

Verwenden Sie die folgenden Umgebungsvariablen, um App Mesh mit Jaeger-Tracing zu konfigurieren. Weitere Informationen finden Sie unter [Erste Schritte](#) in der Jaeger-Dokumentation. Diese Variablen werden von Envoy-Images der Version 1.16.1.0-prod und höher unterstützt.

ENABLE_ENVOY_JAEGER_TRACING

Aktiviert die Jaeger-Trace-Erfassung 127.0.0.1:9411 als Standard-Jaeger-Endpunkt. Um ihn zu aktivieren, setzen Sie den Wert auf 1 (der Standardwert ist 0).

JAEGER_TRACER_PORT

Geben Sie einen Portwert an, um den Standardport von Jaeger zu überschreiben:9411.

JAEGER_TRACER_ADDRESS

Geben Sie eine IP-Adresse an, um die Standardadresse von Jaeger zu überschreiben:.
127.0.0.1

JAEGER_TRACER_VERSION

Geben Sie an, ob der Collector Traces im JSON oder PROTO codierten Format benötigt. Standardmäßig ist dieser Wert auf gesetztPROTO. Diese Variable wird mit der Envoy-Image-Version v1.23.1.0-prod oder höher unterstützt.

Envoy-Tracing-Variable

Stellen Sie die folgende Umgebungsvariable ein, um Ihre eigene Ablaufverfolgungskonfiguration zu verwenden.

ENVOY_TRACING_CFG_FILE

Geben Sie einen Dateipfad im Envoy-Container-Dateisystem an. Weitere Informationen finden Sie [config.trace.v3.Tracing](#) in der Envoy-Dokumentation.

Note

Wenn die Tracing-Konfiguration die Angabe eines Tracing-Clusters erfordert, stellen Sie sicher, dass Sie die zugehörige Cluster-Konfiguration unter `static_resources` in derselben Tracing-Konfigurationsdatei konfigurieren. Zipkin hat beispielsweise ein [collector_cluster](#)-Feld für den Clusternamen, der die Trace-Collectors hostet, und dieser Cluster muss statisch definiert werden.

DogStatsD-Variablen

Verwenden Sie die folgenden Umgebungsvariablen, um App Mesh mit DogStats D zu konfigurieren. Weitere Informationen finden Sie in der [DogStatsD-Dokumentation](#).

ENABLE_ENVOY_DOG_STATSD

Aktiviert DogStats D Stats `127.0.0.1:8125` als Standard-Daemon-Endpunkt. Um zu aktivieren, setzen Sie den Wert auf `1`.

STATSD_PORT

Geben Sie einen Portwert an, um den DogStats Standard-D-Daemon-Port zu überschreiben.

STATSD_ADDRESS

Geben Sie einen IP-Adresswert an, um die DogStats Standard-D-Daemon-IP-Adresse zu überschreiben. Standard: `127.0.0.1`. Diese Variable kann nur mit der Version `1.15.0` oder einer späteren Version des Envoy-Images verwendet werden.

STATSD_SOCKET_PATH

Geben Sie einen Unix-Domain-Socket für den DogStats D-Daemon an. Wenn diese Variable nicht angegeben und DogStats D aktiviert ist, wird für diesen Wert standardmäßig

der IP-Adressport des DogStats D-Daemons von verwendet. `127.0.0.1:8125`
Wenn die `ENVOY_STATS_SINKS_CFG_FILE` Variable angegeben wird, die eine Statistikenkonfiguration enthält, überschreibt sie alle D-Variablen. DogStats Diese Variable wird mit der Envoy-Image-Version `v1.19.1.0-prod` oder höher unterstützt.

App Mesh Mesh-Variablen

Die folgenden Variablen helfen Ihnen bei der Konfiguration von App Mesh.

APPMESH_PREVIEW

Stellen Sie den Wert auf ein, `1` um eine Verbindung zum App Mesh Preview Channel-Endpunkt herzustellen. Weitere Informationen zur Verwendung des App Mesh Mesh-Vorschaukanals finden Sie unter [App Mesh Preview Channel](#).

APPMESH_RESOURCE_CLUSTER

Standardmäßig verwendet App Mesh den Namen der Ressource, die Sie angegeben haben, `APPMESH_RESOURCE_ARN` wenn Envoy in Metriken und Traces auf sich selbst verweist. Sie können dieses Verhalten übergehen, indem Sie die `APPMESH_RESOURCE_CLUSTER`-Umgebungsvariable mit Ihrem eigenen Namen festlegen. Diese Variable kann nur mit der Version `1.15.0` oder einer späteren Version des Envoy-Images verwendet werden.

APPMESH_METRIC_EXTENSION_VERSION

Legen Sie den Wert auf fest, `1` um die App Mesh-Metrikerweiterung zu aktivieren. Weitere Informationen zur Verwendung der App Mesh-Metrikerweiterung finden Sie unter [Metrikerweiterung für App Mesh](#).

APPMESH_DUALSTACK_ENDPOINT

Stellen Sie den Wert auf ein, `1` um eine Verbindung zum App Mesh Dual Stack-Endpunkt herzustellen. Wenn dieses Flag gesetzt ist, verwendet Envoy unsere Dual-Stack-fähige Domain. Standardmäßig ist dieses Flag auf `False` gesetzt und stellt nur eine Verbindung zu unserer IPv4 Domain her. Diese Variable kann nur mit der Envoy-Image-Version `1.22.0` oder höher verwendet werden.

Envoy-Statistikvariablen

Verwenden Sie die folgenden Umgebungsvariablen, um App Mesh mit Envoy Stats zu konfigurieren. Weitere Informationen finden Sie in der Dokumentation zu [Envoy Stats](#).

ENABLE_ENVOY_STATS_TAGS

Ermöglicht die Verwendung von App Mesh Mesh-definierten Tags `appmesh.mesh` und `appmesh.virtual_node`. Weitere Informationen finden Sie unter [config.metrics.v3.TagSpecifier](#) in der Envoy-Dokumentation. Um zu aktivieren, setzen Sie den Wert auf 1.

ENVOY_STATS_CONFIG_FILE

Geben Sie einen Dateipfad im Envoy-Container-Dateisystem an, um die standardmäßige Konfigurationsdatei für die Statistik-Tags durch Ihre eigene zu überschreiben. Weitere Informationen finden Sie unter [config.metrics.v3.StatsConfig](#).

Note

Das Einrichten einer benutzerdefinierten Statistikkonfiguration mit Statistikfiltern kann dazu führen, dass Envoy in einen Status wechselt, in dem es nicht mehr ordnungsgemäß mit dem App Mesh Mesh-Status der Welt synchronisiert wird. Das ist ein [Bug](#) in Envoy. Wir empfehlen, in Envoy keine Statistiken zu filtern. Wenn eine Filterung unbedingt erforderlich ist, haben wir in dieser [Ausgabe](#) auf unserer Roadmap einige Problemumgehungen aufgeführt.

ENVOY_STATS_SINKS_CFG_FILE

Geben Sie einen Dateipfad im Envoy-Container-Dateisystem an, um die Standardkonfiguration mit Ihrer eigenen zu überschreiben. Weitere Informationen finden Sie unter [config.metrics.v3.StatsSink](#) in der Envoy-Dokumentation.

Veraltete Variablen

Die Umgebungsvariablen `APPMESH_VIRTUAL_NODE_NAME` und `APPMESH_RESOURCE_NAME` werden in der Envoy-Version 1.15.0 oder höher nicht mehr unterstützt. Sie werden jedoch weiterhin für bestehende Meshes unterstützt. Anstatt diese Variablen mit der Envoy-Version 1.15.0 oder höher zu verwenden, verwenden Sie `APPMESH_RESOURCE_ARN` für alle App Mesh Mesh-Endpunkte.

Von App Mesh festgelegte Envoy-StandardEinstellungen

Die folgenden Abschnitte enthalten Informationen zu den Envoy-StandardEinstellungen für die Route-Wiederholungsrichtlinie und den Circuit Breaker, die von App Mesh festgelegt wurden.

Standardrichtlinie für die Wiederholung von Routen

Wenn Sie vor dem 29. Juli 2020 keine Meshes in Ihrem Konto hatten, erstellt App Mesh am oder nach dem 29. Juli 2020 automatisch eine standardmäßige Envoy-Route-Wiederholungsrichtlinie für alle HTTP-, HTTP/2- und gRPC-Anfragen in einem beliebigen Mesh in Ihrem Konto. Wenn Sie vor dem 29. Juli 2020 Meshes in Ihrem Konto hatten, wurde keine Standardrichtlinie für Envoy-Routen erstellt, die vor, am oder nach dem 29. Juli 2020 existierten. Es sei denn, du [eröffnest ein Ticket beim AWS Support](#). Nachdem der Support das Ticket bearbeitet hat, wird die Standardrichtlinie für alle future Envoy-Routen erstellt, die App Mesh an oder nach dem Datum erstellt, an dem das Ticket bearbeitet wurde. [Weitere Informationen zu den Richtlinien für die Wiederholung von Envoy-Routen finden Sie unter `config.route.v3.RetryPolicy` in der Envoy-Dokumentation.](#)

App Mesh erstellt eine Envoy-Route, wenn Sie entweder eine App Mesh [Mesh-Route](#) erstellen oder einen virtuellen Knotenanbieter für einen [virtuellen App Mesh Mesh-Dienst](#) definieren. Sie können zwar eine App Mesh Mesh-Route-Wiederholungsrichtlinie erstellen, aber Sie können keine App Mesh Mesh-Wiederholungsrichtlinie für einen Anbieter virtueller Knoten erstellen.

Die Standardrichtlinie ist über die App Mesh Mesh-API nicht sichtbar. Die Standardrichtlinie ist nur über Envoy sichtbar. Um die Konfiguration einzusehen, [aktivieren Sie die Administrationsoberfläche](#) und senden Sie eine Anfrage an Envoy für eine. `config_dump` Die Standardrichtlinie umfasst die folgenden Einstellungen:

- Max. Anzahl der Wiederholungen — 2
- gRPC-Wiederholungsereignisse — UNAVAILABLE
- HTTP-Wiederholungsereignisse — 503

Note

Es ist nicht möglich, eine App Mesh-Route-Wiederholungsrichtlinie zu erstellen, die nach einem bestimmten HTTP-Fehlercode sucht. Eine App Mesh Mesh-Route-Wiederholungsrichtlinie kann jedoch nach `server-error` oder `gateway-error` suchen. Beide beinhalten 503 Fehler. Weitere Informationen finden Sie unter [Routen](#).

- TCP-Wiederholungsereignis — `connect-failure` und `refused-stream`

Note

Es ist nicht möglich, eine App Mesh Mesh-Route-Wiederholungsrichtlinie zu erstellen, die nach einem dieser Ereignisse sucht. Eine App Mesh Mesh-Route-Wiederholungsrichtlinie kann jedoch `suchenconnection-error`, was gleichbedeutend ist mit `connect-failure`. Weitere Informationen finden Sie unter [Routen](#).

- Zurücksetzen — Envoy versucht einen erneuten Versuch, wenn der Upstream-Server überhaupt nicht reagiert (Disconnect/Reset/Read Timeout).

Standardmäßiger Schutzschalter

Wenn Sie einen Envoy in App Mesh bereitstellen, werden Envoy-Standardwerte für einige der Circuit Breaker-Einstellungen festgelegt. [Weitere Informationen finden Sie unter Cluster.CircuitBreakers.Schwellenwerte](#) in der Envoy-Dokumentation. Diese Einstellungen sind über die App Mesh Mesh-API nicht sichtbar. Die Einstellungen sind nur über Envoy sichtbar. Um die Konfiguration einzusehen, [aktivieren Sie die Administrationsoberfläche](#) und senden Sie eine Anfrage an Envoy für eine `config_dump`

Wenn Sie vor dem 29. Juli 2020 keine Meshes in Ihrem Konto hatten, deaktiviert App Mesh für jeden Envoy, den Sie in einem Mesh einsetzen, das am oder nach dem 29. Juli 2020 erstellt wurde, effektiv Schutzschalter, indem es die Envoy-Standardwerte für die folgenden Einstellungen ändert. Wenn Sie vor dem 29. Juli 2020 Meshes in Ihrem Konto hatten, werden die Envoy-Standardwerte für jeden Envoy festgelegt, den Sie am oder nach dem 29. Juli 2020 in App Mesh bereitstellen, sofern Sie [kein Ticket beim Support öffnen](#). AWS Sobald der Support das Ticket bearbeitet hat, werden die App Mesh-Standardwerte für die folgenden Envoy-Einstellungen von App Mesh auf allen Envoys festgelegt, die Sie nach dem Datum der Ticketverarbeitung bereitstellen:

- **max_requests** – 2147483647
- **max_pending_requests** – 2147483647
- **max_connections** – 2147483647
- **max_retries** – 2147483647

Note

Unabhängig davon, ob Ihre Envoyos die Standard-Leistungsschalterwerte für Envoy oder App Mesh haben, können Sie die Werte nicht ändern.

Aktualisierung/Migration auf Envoy 1.17

Secret Discovery Service mit SPIRE

Wenn Sie SPIRE (SPIFFE Runtime Environment) mit App Mesh verwenden, um Vertrauenszertifikate an Ihre Dienste zu verteilen, stellen Sie sicher, dass Sie mindestens die Version 0.12.0 des [SPIRE-Agenten](#) (veröffentlicht im Dezember 2020) verwenden. Dies ist die erste Version, die spätere Envoy-Versionen unterstützen kann. 1.16

Änderungen regulärer Ausdrücke

Ab Envoy konfiguriert App Mesh Envoy so 1.17, dass es standardmäßig die [RE2-Engine](#) für reguläre Ausdrücke verwendet. Diese Änderung ist für die meisten Benutzer offensichtlich, aber Treffer in Routes oder Gateway Routes erlauben keine Vorausschau- oder Rückverweise mehr in regulären Ausdrücken.

Positiver und negativer Ausblick

Positiv — Ein positiver Ausblick ist ein Ausdruck in Klammern, der wie folgt beginnt: `?=`

```
(?=example)
```

Diese sind bei der Ersetzung von Zeichenketten am nützlichsten, da sie den Abgleich einer Zeichenfolge ermöglichen, ohne dass die Zeichen als Teil des Abgleichs verwendet werden. Da App Mesh das Ersetzen von Regex-Zeichenketten nicht unterstützt, empfehlen wir, diese durch reguläre Treffer zu ersetzen.

```
(example)
```

Negativ — Ein negativer Look-Ahead-Ausdruck ist ein Ausdruck in Klammern, der mit `?!>` beginnt.

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

Die Ausdrücke in Klammern werden verwendet, um zu bestätigen, dass ein Teil des Ausdrucks nicht mit einer bestimmten Eingabe übereinstimmt. In den meisten Fällen können Sie diese durch einen Nullquantifizierer ersetzen.

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

Wenn der Ausdruck selbst eine Zeichenklasse ist, können Sie die gesamte Klasse negieren und sie damit als optional markieren. ?

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

Abhängig von Ihrem Anwendungsfall können Sie möglicherweise auch Ihre Routen ändern, um dies zu handhaben.

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



```
    ]
  }
}
}
```

In diesem Beispiel werden die Routen umgekehrt, um Headern, die mit „Suffix“ enden, eine höhere Priorität einzuräumen. Alle anderen Header, die mit „Präfix“ beginnen, werden in der Route mit niedrigerer Priorität abgeglichen.

Rückverweise

Ein Rückverweis ist eine Möglichkeit, kürzere Ausdrücke zu schreiben, indem sie zu einer vorherigen Gruppe in Klammern wiederholt werden. Sie haben dieses Formular.

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

Ein umgekehrter Schrägstrich, \ gefolgt von einer Zahl, dient als Platzhalter für die n-te Gruppe in Klammern im Ausdruck. In diesem Beispiel \1 wird es als alternative Methode verwendet, um ein zweites Mal zu schreiben. (group1)

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

Diese können entfernt werden, indem einfach der Rückverweis durch die Gruppe ersetzt wird, auf die verwiesen wird, wie im Beispiel.

Beauftragter

Der Agent ist ein Prozessmanager innerhalb des Envoy-Images, das für App Mesh verkauft wird. Der Agent stellt sicher, dass Envoy läuft, gesund bleibt und Ausfallzeiten reduziert. Es filtert Envoy-Statistiken und Zusatzdaten, um einen detaillierten Überblick über den Betrieb des Envoy-Proxys in App Mesh zu erhalten. Dies kann Ihnen helfen, verwandte Fehler schneller zu beheben.

Sie können den Agenten verwenden, um zu konfigurieren, wie oft Sie den Envoy-Proxy neu starten möchten, falls der Proxy fehlerhaft wird. Tritt ein Fehler auf, protokolliert der Agent den abschließenden Exit-Status, wenn Envoy beendet wird. Sie können dies bei der Behebung des Fehlers verwenden. Der Agent erleichtert auch die Entleerung von Envoy-Verbindungen, wodurch Ihre Anwendungen widerstandsfähiger gegen Ausfälle werden.

Konfigurieren Sie den Agenten für Envoy mithilfe dieser Variablen:

- `APPNET_ENVOY_RESTART_COUNT`— Wenn diese Variable auf einen Wert ungleich Null gesetzt ist, versucht der Agent, den Envoy-Proxy-Prozess bis zu der von Ihnen festgelegten Zahl neu zu starten, wenn er den Status des Proxy-Prozesses bei der Abfrage als fehlerhaft einstuft. Dies trägt zur Reduzierung der Ausfallzeiten bei, da im Vergleich zu einem Task- oder Pod-Austausch durch den Container-Orchestrator ein schnellerer Neustart ermöglicht wird, falls die Proxy-Integritätsprüfung fehlschlägt.
- `PID_POLL_INTERVAL_MS`— Bei der Konfiguration dieser Variablen wird die Standardeinstellung beibehalten `100`. Wenn Sie diesen Wert wählen, ermöglichen Sie eine schnellere Erkennung und einen schnelleren Neustart des Envoy-Prozesses, wenn er beendet wird, als wenn Aufgaben oder Pods durch die Integritätsprüfungen des Container Orchestrator ersetzt werden.
- `LISTENER_DRAIN_WAIT_TIME_S`— Berücksichtigen Sie bei der Konfiguration dieser Variablen das Container-Orchestrator-Timeout, das für das Stoppen der Aufgabe oder des Pods festgelegt ist. Wenn dieser Wert beispielsweise größer als das Orchestrator-Timeout ist, kann der Envoy-Proxy nur für die Dauer entladen, bis der Orchestrator die Aufgabe oder den Pod gewaltsam stoppt.
- `APPNET_AGENT_ADMIN_MODE`— Wenn diese Variable auf `cp` oder `uds` gesetzt ist, stellt der Agent eine lokale Verwaltungsschnittstelle bereit. Diese Verwaltungsschnittstelle dient als sicherer Endpunkt für die Interaktion mit dem Envoy-Proxy und bietet die folgenden APIs für Zustandsprüfungen, Telemetriedaten und fasst den Betriebszustand des Proxys zusammen.
 - `GET /status`— Fragt Envoy-Statistiken ab und gibt Serverinformationen zurück.
 - `POST /drain_listeners`— Entleert alle eingehenden Zuhörer.
 - `POST /enableLogging?level=<desired_level>`— Ändert die Protokollierungsstufe von Envoy für alle Logger.
 - `GET /stats/prometheus`— Zeigt die Statistiken des Gesandten im Prometheus-Format an.
 - `GET /stats/prometheus?usedonly`— Zeige nur Statistiken, die Envoy aktualisiert hat.

Weitere Informationen zu den Konfigurationsvariablen für den `-Agenten` finden Sie unter [Konfigurationsvariablen für den -Agenten](#).

Der neue AWS App Mesh Agent ist ab der Version in App Mesh-optimierten Envoy-Images enthalten `1.21.0.0` und erfordert keine zusätzliche Ressourcenzuweisung für Kundenaufgaben oder Pods.

App Mesh-Obblaufverfolgbarkeit

Einer der Vorteile der Arbeit mit App Mesh ist die bessere Sichtbarkeit Ihrer Microservice-Anwendungen. App Mesh kann mit vielen verschiedenen Logging-, Metrik- und Tracing-Lösungen arbeiten.

Der Envoy-Proxy und App Mesh bieten die folgenden Arten von Tools, mit denen Sie sich einen klareren Überblick über Ihre Anwendungen und Proxys verschaffen können:

- [Protokollierung](#)
- [Metriken](#)
- [Rückverfolgung](#)

Protokollierung

Wenn Sie Ihre virtuellen Knoten und virtuellen Gateways erstellen, haben Sie die Möglichkeit, Envoy-Zugriffsprotokolle zu konfigurieren. In der Konsole befindet sich dies im Abschnitt Logging des virtuellen Knotens und des virtuellen Gateways zum Erstellen oder Bearbeiten von Workflows.

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.

Das vorherige Bild zeigt einen Protokollierungspfad `/dev/stdout` für Envoy-Zugriffsprotokolle.

Geben Sie für `format` eines von zwei möglichen Formaten an, `json` oder `text`, und das Muster. `json` nimmt Schlüsselpaare und wandelt sie in eine JSON-Struktur um, bevor sie an Envoy übergeben werden.

Der folgende Codeblock zeigt die JSON-Darstellung, die Sie in der verwenden können 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

Stellen Sie sicher, dass Ihr Eingabemuster für Envoy gültig ist. Andernfalls lehnt Envoy das Update ab und speichert die neuesten Änderungen in der `error` state.

Wenn Sie Envoy-Zugriffsprotokolle an `senden/dev/stdout`, werden sie mit den Envoy-Container-Logs vermischt. Sie können sie in einen Protokollspeicher- und Verarbeitungsdienst wie CloudWatch Logs exportieren, indem Sie standardmäßige Docker-Protokolltreiber verwenden, `awslogs` z. Weitere Informationen finden Sie unter [Verwenden des awslogs-Protokolltreibers im Amazon ECS-Entwicklerhandbuch](#) für Amazon ECS-Entwickler finden Sie unter [Verwenden des awslogs-Protokolltreibers im Um nur die Envoy-Zugriffsprotokolle zu exportieren \(und die anderen Envoy-Container-Logs zu ignorieren\)](#), können Sie das `ENVOY_LOG_LEVEL` auf `off` setzen. Sie können Anfragen ohne Abfragezeichenfolge protokollieren, indem Sie die Formatzeichenfolge angeben `%REQ_WITHOUT_QUERY(X?Y):Z%`. Beispiele finden Sie unter [ReqWithoutQueryFormatter](#). Weitere Informationen finden Sie unter [Access-Protokollierung](#) in der Envoy-Dokumentation zu finden Sie unter [Zugriffsprotokollverfolgung](#)

Zugriffsprotokolle auf Kubernetes aktivieren

Wenn Sie den [App Mesh Controller für Kubernetes](#) verwenden, können Sie virtuelle Knoten mit Zugriffsprotokollierung konfigurieren, indem Sie die Protokollierungskonfiguration zur virtuellen Knotenspezifikation hinzufügen, wie im folgenden Beispiel gezeigt.

```
---
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"
```

Ihr Cluster muss über einen Log-Forwarder verfügen, um diese Protokolle zu sammeln, z. B. Fluentd. Weitere Informationen finden Sie unter [Fluentd einrichten, DaemonSet um Protokolle an Logs zu CloudWatch senden](#).

Envoy schreibt auch verschiedene Debugging-Logs aus seinen Filtern instdout. Diese Protokolle sind nützlich, um Einblicke sowohl in die Kommunikation von Envoy mit App Mesh als auch in den service-to-service Traffic zu gewinnen. Ihre spezifische Protokollierungsebene kann mithilfe der `ENVOY_LOG_LEVEL` Umgebungsvariablen konfiguriert werden. Der folgende Text stammt beispielsweise aus einem Beispiel für ein Debug-Protokoll, das den Cluster zeigt, den Envoy für eine bestimmte HTTP-Anfrage gefunden hat.

```
[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 und Cloudwatch

[Firelens](#) ist ein Container-Log-Router, mit dem Sie Protokolle für Amazon ECS und sammeln können AWS Fargate. Ein Beispiel für die Verwendung von Firelens finden Sie in unserem [AWSSamples-Repository](#).

Sie können CloudWatch es verwenden, um sowohl Protokollinformationen als auch Metriken zu sammeln. Weitere Informationen dazu finden Sie CloudWatch in unserem Abschnitt [Exportieren von Metriken](#) in den App Mesh-Dokumenten.

Überwachung Ihrer Anwendung mithilfe von Envoy-Metriken

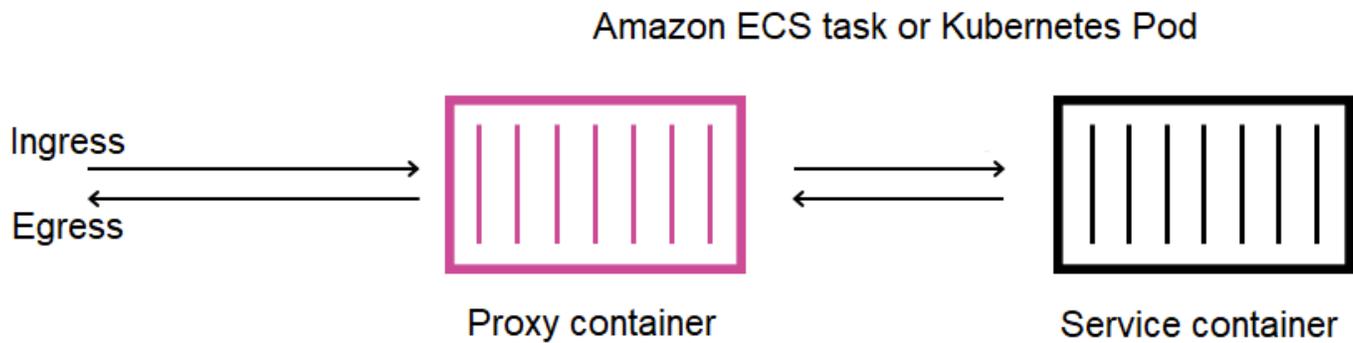
Envoy unterteilt seine Kennzahlen in die folgenden Hauptkategorien:

- **Downstream** — Metriken, die sich auf Verbindungen und Anfragen beziehen, die in den Proxy eingehen.
- **Upstream** — Metriken, die sich auf ausgehende Verbindungen und Anfragen des Proxys beziehen.
- **Server** — Metriken, die den internen Status von Envoy beschreiben. Dazu gehören Metriken wie Verfügbarkeit oder zugewiesener Speicher.

In App Mesh fängt der Proxy Upstream- und Downstream-Verkehr ab. Beispielsweise werden Anfragen, die Sie von Ihren Kunden erhalten, sowie Anfragen, die von Ihrem Service-Container gestellt werden, von Envoy als Downstream-Traffic eingestuft. Um zwischen diesen verschiedenen Arten von Upstream- und Downstream-Traffic zu unterscheiden, kategorisiert App Mesh die Envoy-Metriken in Abhängigkeit von der Verkehrsrichtung im Verhältnis zu Ihrem Dienst weiter:

- **Ingress** — Metriken und Ressourcen in Bezug auf Verbindungen und Anfragen, die an Ihren Service-Container fließen.
- **Egress** — Metriken und Ressourcen in Bezug auf Verbindungen und Anfragen, die aus Ihrem Service-Container und letztendlich aus Ihrer Amazon ECS-Aufgabe oder Ihrem Kubernetes-Pod fließen.

Das folgende Bild zeigt die Kommunikation zwischen dem Proxy und den Service-Containern.



Konventionen zur Benennung von Ressourcen

Es ist hilfreich zu verstehen, wie Envoy Ihr Mesh betrachtet und wie seine Ressourcen den Ressourcen zugeordnet werden, die Sie in App Mesh definieren. Dies sind die primären Envoy-Ressourcen, die App Mesh konfiguriert:

- **Listener** — Die Adressen und Ports, auf denen der Proxy auf Downstream-Verbindungen wartet. In der vorherigen Abbildung erstellt App Mesh einen Eingangs-Listener für Datenverkehr, der in Ihre Amazon ECS-Aufgabe oder Ihren Kubernetes-Pod eingeht, und einen Ausgangs-Listener für Datenverkehr, der Ihren Service-Container verlässt.
- **Cluster** — Eine benannte Gruppe von Upstream-Endpunkten, zu denen der Proxy eine Verbindung herstellt und an die der Datenverkehr weitergeleitet wird. In App Mesh wird Ihr Service-Container als Cluster dargestellt, ebenso wie alle anderen virtuellen Knoten, mit denen Ihr Service eine Verbindung herstellen kann.
- **Routen** — Diese entsprechen den Routen, die Sie in Ihrem Mesh definieren. Sie enthalten die Bedingungen, unter denen der Proxy einer Anfrage entspricht, sowie den Zielcluster, an den eine Anfrage gesendet wird.
- **Endpunkte und Cluster-Ladezuweisungen** — Die IP-Adressen der Upstream-Cluster. Wenn Sie App Mesh AWS Cloud Map als Service Discovery-Mechanismus für virtuelle Knoten verwenden, sendet App Mesh entdeckte Service-Instanzen als Endpunktrressourcen an Ihren Proxy.
- **Geheimnisse** — Dazu gehören, ohne darauf beschränkt zu sein, Ihre Verschlüsselungsschlüssel und TLS-Zertifikate. Bei Verwendung AWS Certificate Manager als Quelle für Client- und Serverzertifikate sendet App Mesh öffentliche und private Zertifikate als geheime Ressourcen an Ihren Proxy.

App Mesh verwendet ein konsistentes Schema für die Benennung von Envoy-Ressourcen, mit denen Sie eine Beziehung zu Ihrem Mesh herstellen können.

Das Benennungsschema für Listener und Cluster zu verstehen, ist wichtig, um die Metriken von Envoy in App Mesh zu verstehen.

Namen der Zuhörer

Listener werden mit folgendem Format benannt:

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

In der Regel werden die folgenden Listener in Envoy konfiguriert sehen:

- lds_ingress_0.0.0.0_15000
- lds_egress_0.0.0.0_15001

Mithilfe eines Kubernetes-CNI-Plug-ins oder IP-Tabellenregeln wird der Datenverkehr in Ihrer Amazon ECS-Aufgabe oder Ihrem Kubernetes-Pod an die Ports 15000 und geleitet 15001. App Mesh konfiguriert Envoy mit diesen beiden Listenern so, dass eingehender (eingehender) und ausgehender (ausgehender) Verkehr akzeptiert werden. Wenn Sie auf Ihrem virtuellen Knoten keinen Listener konfiguriert haben, sollten Sie keinen Ingress-Listener sehen.

Clusternamen

Die meisten Cluster verwenden das folgende Format:

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

Virtuelle Knoten, mit denen Ihre Dienste kommunizieren, haben jeweils ihren eigenen Cluster. Wie bereits erwähnt, erstellt App Mesh einen Cluster für den Dienst, der neben Envoy ausgeführt wird, sodass der Proxy eingehenden Datenverkehr an ihn senden kann.

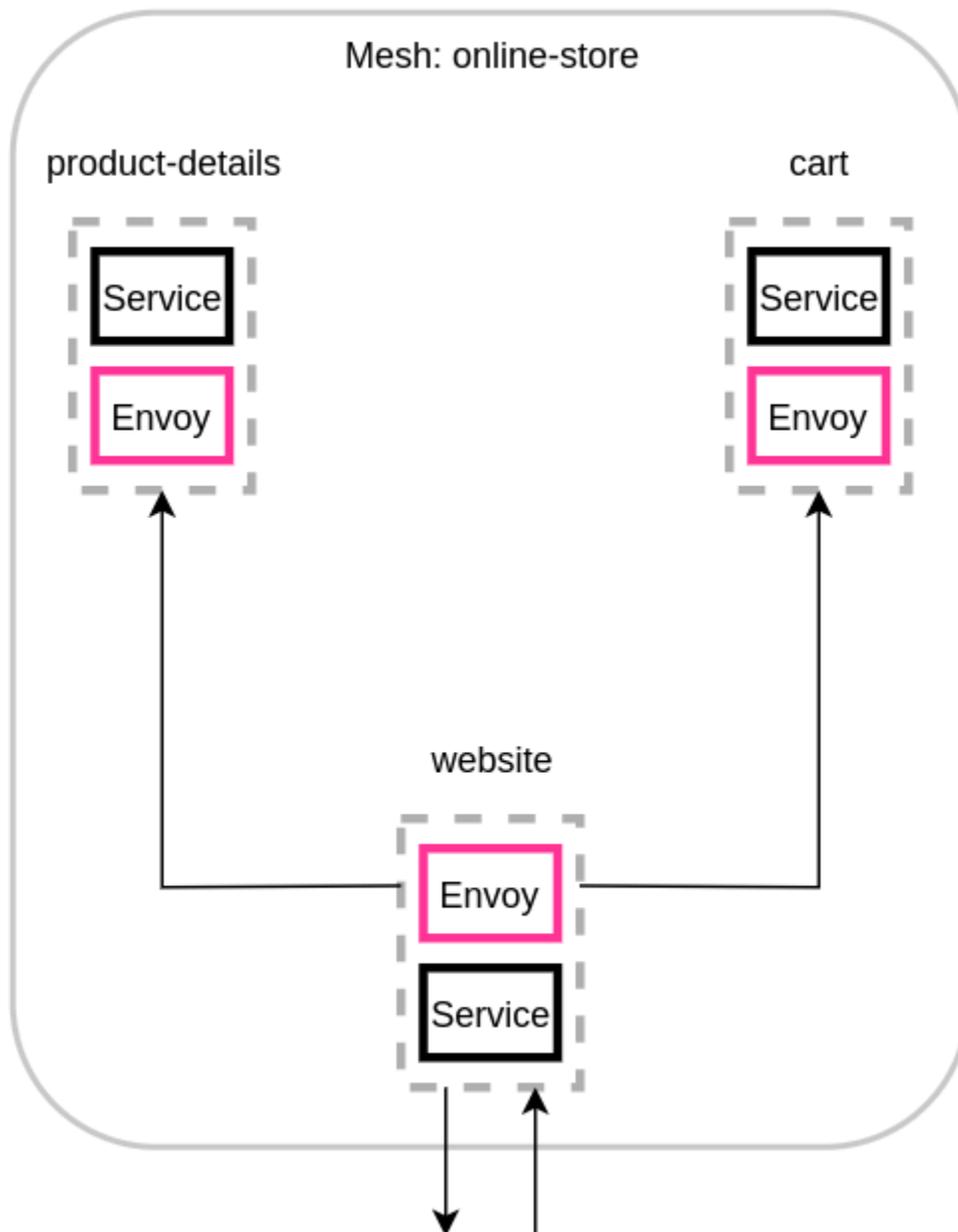
Wenn Sie beispielsweise einen virtuellen Knoten benannt `my-virtual-node` haben, der auf HTTP-Verkehr am Port wartet, `8080` und dieser virtuelle Knoten sich in einem Mesh mit dem Namen `my-mesh` befindet, erstellt App Mesh einen Cluster mit dem Namen `cds_ingress_my-mesh_my-virtual-node_http_8080`. Dieser Cluster dient als Ziel für den Datenverkehr in `my-virtual-node` den Service-Container.

App Mesh kann auch die folgenden Typen zusätzlicher Spezialcluster erstellen. Diese anderen Cluster entsprechen nicht unbedingt Ressourcen, die Sie explizit in Ihrem Mesh definieren.

- Cluster, die verwendet werden, um andere AWS Dienste zu erreichen. Mit diesem Typ kann Ihr Mesh standardmäßig die meisten AWS Dienste erreichen: `cds_egress_<mesh name>_amazonaws`.
- Cluster, der zum Routing für virtuelle Gateways verwendet wird. Dies kann im Allgemeinen getrost ignoriert werden:
 - Für einzelne Zuhörer: `cds_ingress_<mesh name>_<virtual gateway name>_self_redirect_<protocol>_<port>`
 - Für mehrere Zuhörer: `cds_ingress_<mesh name>_<virtual gateway name>_self_redirect_<ingress_listener_port>_<protocol>_<port>`
- Den Cluster, dessen Endpunkt Sie definieren können, z. B. TLS, wenn Sie Secrets mit dem Secret Discovery Service von Envoy abrufen: `static_cluster_sds_unix_socket`.

Beispiele für Anwendungsmetriken

Zur Veranschaulichung der in Envoy verfügbaren Metriken verfügt die folgende Beispielanwendung über drei virtuelle Knoten. Die virtuellen Dienste, virtuellen Router und Routen im Mesh können ignoriert werden, da sie sich nicht in den Metriken von Envoy widerspiegeln. In diesem Beispiel warten alle Dienste auf HTTP-Verkehr auf Port 8080.



Wir empfehlen, die Umgebungsvariable `ENABLE_ENVOY_STATS_TAGS=1` zu den Envoy-Proxy-Containern hinzuzufügen, die in Ihrem Mesh ausgeführt werden. Dadurch werden alle vom Proxy ausgegebenen Metriken um die folgenden Metrik-Dimensionen erweitert:

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

Diese Tags sind auf den Namen des Meshs, des virtuellen Knotens oder des virtuellen Gateways festgelegt, um das Filtern von Metriken anhand der Namen der Ressourcen in Ihrem Mesh zu ermöglichen.

Namen der Ressourcen

Der Proxy des virtuellen Knotens der Website verfügt über die folgenden Ressourcen:

- Zwei Listener für eingehenden und ausgehenden Traffic:
 - `lds_ingress_0.0.0.0_15000`
 - `lds_egress_0.0.0.0_15001`
- Zwei Ausgangscluster, die die beiden virtuellen Knoten-Backends darstellen:
 - `cds_egress_online-store-product-details_http_8080`
 - `cds_egress_online-store-cart_http_8080`
- Ein Ingress-Cluster für den Website-Service-Container:
 - `cds_ingress_online-store-website_http_8080`

Beispiel für Listener-Metriken

- `listener.0.0.0.0_15000.downstream_cx_active`— Anzahl der aktiven eingehenden Netzwerkverbindungen zu Envoy.
- `listener.0.0.0.0_15001.downstream_cx_active`— Anzahl der aktiven ausgehenden Netzwerkverbindungen zu Envoy. Verbindungen, die Ihre Anwendung zu externen Diensten herstellt, sind in dieser Zählung enthalten.
- `listener.0.0.0.0_15000.downstream_cx_total`— Gesamtzahl der eingehenden Netzwerkverbindungen zu Envoy.
- `listener.0.0.0.0_15001.downstream_cx_total`— Gesamtzahl der ausgehenden Netzwerkverbindungen zu Envoy.

Die vollständigen Listener-Metriken finden Sie unter [Statistiken](#) in der Envoy-Dokumentation.

Beispiel für Cluster-Metriken

- `cluster_manager.active_clusters`— Die Gesamtzahl der Cluster, zu denen Envoy mindestens eine Verbindung hergestellt hat.

- `cluster_manager.warming_clusters`— Die Gesamtzahl der Cluster, zu denen Envoy noch keine Verbindung hergestellt hat.

Die folgenden Cluster-Metriken verwenden das Format von `cluster.<cluster name>.<metric name>`. Diese Metrikenamen gelten nur für das Anwendungsbeispiel und werden vom Envoy-Container der Website ausgegeben:

- `cluster.cds_egress_online-store_product-details_http_8080.upstream_cx_total`—Gesamtzahl der Verbindungen zwischen Website und Produktdetails.
- `cluster.cds_egress_online-store_product-details_http_8080.upstream_cx_connect_fail`— Gesamtzahl der fehlgeschlagenen Verbindungen zwischen Website und Produktdetails.
- `cluster.cds_egress_online-store_product-details_http_8080.health_check.failure`—Gesamtzahl der fehlgeschlagenen Zustandsprüfungen zwischen Website und Produktdetails.
- `cluster.cds_egress_online-store_product-details_http_8080.upstream_rq_total`—Gesamtzahl der Anfragen, die zwischen Website und Produktdetails gestellt wurden.
- `cluster.cds_egress_online-store_product-details_http_8080.upstream_rq_time`—Zeit, die für Anfragen zwischen Website und Produktdetails benötigt wird.
- `cluster.cds_egress_online-store_product-details_http_8080.upstream_rq_2xx`—Anzahl der HTTP-2xx-Antworten, die die Website von product details erhalten hat.

Den vollständigen Satz der HTTP-Metriken finden Sie unter [Statistiken](#) in der Envoy-Dokumentation.

Metriken für Management-Server

Envoy gibt auch Metriken im Zusammenhang mit seiner Verbindung zur App Mesh-Steuerungsebene aus, die als Managementserver von Envoy fungiert. Wir empfehlen, einige dieser Metriken zu überwachen, um Sie zu benachrichtigen, wenn Ihre Proxys für längere Zeit von der Steuerungsebene desynchronisiert werden. Ein Verlust der Konnektivität zur Steuerungsebene oder fehlgeschlagene Updates verhindern, dass Ihre Proxys neue Konfigurationen von App Mesh erhalten, einschließlich Mesh-Änderungen, die über App Mesh Mesh-APIs vorgenommen wurden.

- `control_plane.connected_state`— Diese Metrik wird auf 1 gesetzt, wenn der Proxy mit App Mesh verbunden ist, andernfalls ist sie 0.
- `*.update_rejected`— Gesamtzahl der Konfigurationsupdates, die von Envoy abgelehnt wurden. Diese sind normalerweise auf eine Fehlkonfiguration des Benutzers zurückzuführen. Wenn Sie App Mesh beispielsweise so konfigurieren, dass ein TLS-Zertifikat aus einer Datei gelesen wird, die von Envoy nicht gelesen werden kann, wird das Update, das den Pfad zu diesem Zertifikat enthält, abgelehnt.
 - Wenn der Listener aktualisiert wurde, werden die Statistiken `abgelehntlistener_manager.lds.update_rejected`.
 - Für den abgelehnten Cluster-Update gelten die Statistiken `cluster_manager.cds.update_rejected`.
- `*.update_success`— Anzahl der erfolgreichen Konfigurationsupdates, die App Mesh an Ihrem Proxy vorgenommen hat. Dazu gehört die Nutzlast der Erstkonfiguration, die gesendet wird, wenn ein neuer Envoy-Container gestartet wird.
 - Wenn der Listener erfolgreich aktualisiert wurde, werden die Statistiken wie folgt aussehen `listener_manager.lds.update_success`.
 - Bei erfolgreicher Cluster-Aktualisierung werden die Statistiken wie folgt lauten `cluster_manager.cds.update_success`.

Eine Reihe von Management-Server-Metriken finden Sie unter [Management Server](#) in der Envoy-Dokumentation.

Exportieren von Kennzahlen

Envoy veröffentlicht viele Statistiken sowohl zu seinem eigenen Betrieb als auch zu verschiedenen Dimensionen des eingehenden und ausgehenden Datenverkehrs. Weitere Informationen zu den Statistiken von Envoy finden Sie unter [Statistiken](#) in der Envoy-Dokumentation. Diese Metriken sind über den `/stats` Endpunkt am Administrationsport des Proxys verfügbar, was normalerweise der Fall ist `9901`.

Das `stat` Präfix ist unterschiedlich, je nachdem, ob Sie einen oder mehrere Listener verwenden. Im Folgenden finden Sie einige Beispiele, um die Unterschiede zu veranschaulichen.

⚠ Warning

Wenn Sie Ihren einzelnen Listener auf die Funktion mit mehreren Listenern aktualisieren, kann es aufgrund des aktualisierten Statistikpräfixes, das in der folgenden Tabelle dargestellt ist, zu einer grundlegenden Änderung kommen.

Wir empfehlen Ihnen, das Envoy-Bild `1.22.2.1-prod` oder eine spätere Version zu verwenden. Auf diese Weise können Sie ähnliche Metrikenamen in Ihrem Prometheus-Endpoint sehen.

Einzelner Listener (SL) / Bestehende Statistiken mit dem Listener-Präfix „Ingress“	Mehrere Listener (ML) /Neue Statistiken mit „Ingress“. <protocol>. <port>"Listener-Präfix	
<pre>http.*ingress*.rds .rds_ingress_http_ 5555.version_text</pre>	<pre>http.*ingress.http .5555*.rds.rds_ing ress_http_5555.ver sion_text http.*ingress.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>	

Einzelner Listener (SL) / Bestehende Statistiken mit dem Listener-Präfix „Ingress“	Mehrere Listener (ML) /Neue Statistiken mit „Ingress“. <protocol>. <port>"Listener-Pr äfix	
http.*ingress*.downstream_cx_length_ms	http.*ingress.http.5555*.downstream_cx_length_ms http.*ingress.http.6666*.downstream_cx_length_ms	

Weitere Informationen zum Statistikendpunkt finden Sie unter [Statistikendpunkt](#) in der Envoy-Dokumentation. Weitere Informationen zur Administrationsoberfläche finden Sie unter [Aktivieren der Envoy-Proxy-Verwaltungsschnittstelle](#).

Prometheus für App Mesh mit Amazon EKS

Prometheus ist ein Open-Source-Toolkit zum Anzeigen und Analysieren der Daten. Eine seiner Funktionen besteht darin, ein Format für die Ausgabe von Metriken zu spezifizieren, die von anderen Systemen verwendet werden können. Weitere Informationen zu Prometheus finden Sie in der Prometheus-Dokumentation unter [Überblick](#). Envoy kann seine Metriken über seinen Statistik-Endpunkt ausgeben, indem er den Parameter übergibt/`stats?format=prometheus`.

Für Kunden, die den Envoy-Image-Build v1.22.2.1-prod verwenden, gibt es zwei zusätzliche Dimensionen, um spezifische Statistiken für den Ingress-Listener anzugeben:

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

Im Folgenden finden Sie einen Vergleich zwischen den vorhandenen Statistiken von Prometheus und den neuen Statistiken.

- Bestehende Statistiken mit dem Listener-Präfix „Ingress“

```
envoy_http_downstream_rq_xx{appmesh_mesh="multiple-listeners-
mesh",appmesh_virtual_node="foodteller-
vn",envoy_response_code_class="2",envoy_http_conn_manager_prefix="ingress"} 931433
```

- Neue Statistiken mit „Ingress“. <protocol>. <port>"+ Appmesh Envoy Image v1.22.2.1-prod oder später

```
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="5555
20
```

- Neue Statistiken mit „Ingress“. <protocol>. <port>"+ benutzerdefiniertes Envoy Imagebuild

```
envoy_http_http_5555_downstream_rq_xx{appmesh_mesh="multiple-listeners-
mesh",appmesh_virtual_node="foodteller-
vn",envoy_response_code_class="2",envoy_http_conn_manager_prefix="ingress"} 15983
```

Für mehrere Listener ist `dercds_ingress_<mesh name>_<virtual gateway name>_self_redirect_<ingress_listener_port>_<protocol>_<port>` spezielle Cluster hörerspezifisch.

- Bestehende Statistiken mit dem Listener-Präfix „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
```

- Neue Statistiken mit „Ingress“. <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
```

Installieren Sie Prometheus

1. Fügen Sie das EKS-Repository zu Helm hinzu:

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

2. Installieren Sie App Mesh Prometheus

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

Prometheus-Beispiel

Im Folgenden sehen Sie ein Beispiel für die Erstellung eines persistenten Speichers `PersistentVolumeClaim` für Prometheus.

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

Komplettlösung für die Verwendung von Prometheus

- [App Mesh mit EKS — Beobachtbarkeit: Prometheus](#)

Um mehr über Prometheus und Prometheus mit Amazon EKS zu erfahren

- [Prometheus-Dokumentation](#)
- EKS - [Steuerebene-Metriken mit Prometheus](#)

CloudWatch für App Mesh

Senden von Gesandtenstatistiken CloudWatch an Amazon EKS

Sie können den CloudWatch Agenten in Ihrem Cluster installieren und ihn so konfigurieren, dass er eine Teilmenge der Metriken von Ihren Proxys erfasst. Wenn Sie noch keinen Amazon EKS-Cluster haben, können Sie einen erstellen, indem Sie die Schritte unter [Walkthrough: App Mesh mit aktiviertem Amazon EKS](#) ausführen GitHub. Sie können eine Beispielanwendung auf dem Cluster installieren, indem Sie derselben Anleitung folgen.

Um die entsprechenden IAM-Berechtigungen für Ihren Cluster festzulegen und den Agenten zu installieren, folgen Sie den Schritten unter [Installieren des CloudWatch Agenten mit der Prometheus Metrics Collection](#). Die Standardinstallation enthält eine Prometheus-Scrape-Konfiguration, die eine nützliche Teilmenge der Envoy-Statistiken abrufen. Weitere Informationen finden Sie unter [Prometheus Metrics for App Mesh](#).

Um ein benutzerdefiniertes App CloudWatch Mesh-Dashboard zu erstellen, das so konfiguriert ist, dass die Metriken angezeigt werden, die der Agent sammelt, folgen Sie den Schritten im Tutorial [Viewing Your Prometheus Metrics](#). Ihre Grafiken werden mit den entsprechenden Metriken gefüllt, sobald Traffic in die App Mesh Mesh-Anwendung gelangt.

Filtern von Metriken für CloudWatch

Die App Mesh [Mesh-Metrikerweiterung](#) bietet eine Teilmenge nützlicher Metriken, die Ihnen Einblicke in das Verhalten der Ressourcen geben, die Sie in Ihrem Mesh definieren. Da der CloudWatch Agent das Scraping von Prometheus-Metriken unterstützt, können Sie eine Scrape-Konfiguration bereitstellen, um die Metriken auszuwählen, die Sie von Envoy abrufen und an sie senden möchten CloudWatch.

Ein Beispiel für das Scraping von Metriken mit Prometheus finden Sie in unserer Komplettlösung zur [Metrics Extension](#).

CloudWatch Beispiel

Eine Beispielkonfiguration von finden Sie CloudWatch in unserem [AWSSamples-Repository](#).

Walkthroughs für die Verwendung CloudWatch

- [Fügen Sie in unserem App Mesh Mesh-Workshop Überwachungs- und Protokollierungsfunktionen hinzu.](#)
- [App Mesh mit EKS — Beobachtbarkeit: CloudWatch](#)
- [Verwenden der Metrikerweiterung von App Mesh auf ECS](#)

Metrikerweiterung für App Mesh

Envoy generiert Hunderte von Kennzahlen, die in verschiedene Dimensionen unterteilt sind. Die Metriken sind in ihrer Beziehung zu App Mesh nicht einfach. Bei virtuellen Diensten gibt es keinen Mechanismus, um mit Sicherheit zu wissen, welcher virtuelle Dienst mit einem bestimmten virtuellen Knoten oder virtuellen Gateway kommuniziert.

Die App Mesh Mesh-Metrikerweiterung verbessert die in Ihrem Mesh laufenden Envoy-Proxys. Diese Verbesserung ermöglicht es den Proxys, zusätzliche Metriken auszugeben, die die von Ihnen definierten Ressourcen berücksichtigen. Diese kleine Teilmenge zusätzlicher Metriken hilft Ihnen dabei, einen besseren Einblick in das Verhalten der Ressourcen zu erhalten, die Sie in App Mesh definiert haben.

Um die App Mesh-Metrikerweiterung zu aktivieren, legen Sie die Umgebungsvariable `APPMESH_METRIC_EXTENSION_VERSION` auf `fest1`.

```
APPMESH_METRIC_EXTENSION_VERSION=1
```

Weitere Informationen zu Envoy-Konfigurationsvariablen finden Sie unter [Envoy-Konfigurationsvariablen](#).

Kennzahlen im Zusammenhang mit eingehendem Traffic

- **ActiveConnectionCount**

- `envoy.appmesh.ActiveConnectionCount`— Anzahl der aktiven TCP-Verbindungen.
- Abmessungen — Masche VirtualNode, VirtualGateway

- **NewConnectionCount**

- `envoy.appmesh.NewConnectionCount`— Gesamtzahl der TCP-Verbindungen.
- Abmessungen — Masche VirtualNode, VirtualGateway

- **ProcessedBytes**

- `envoy.appmesh.ProcessedBytes`— Gesamtzahl der an Downstream-Clients gesendeten und von Downstream-Clients empfangenen TCP-Bytes.
- Abmessungen — Masche VirtualNode, VirtualGateway

- **RequestCount**

- `envoy.appmesh.RequestCount`— Die Anzahl der verarbeiteten HTTP-Anfragen.
- Abmessungen — Masche VirtualNode, VirtualGateway

- **GrpcRequestCount**

- `envoy.appmesh.GrpcRequestCount`— Die Anzahl der verarbeiteten GPRC-Anfragen.
- Abmessungen — Masche VirtualNode, VirtualGateway

Kennzahlen im Zusammenhang mit ausgehendem Traffic

Ihre ausgehenden Metriken enthalten unterschiedliche Dimensionen, je nachdem, ob sie von einem virtuellen Knoten oder einem virtuellen Gateway stammen.

- **TargetProcessedBytes**

- `envoy.appmesh.TargetProcessedBytes`— Gesamtzahl der TCP-Bytes, die an Ziele vor Envoy gesendet und von diesen empfangen wurden.
- Abmessungen:
 - Dimensionen virtueller Knoten — Mesh, VirtualNode, TargetVirtualService, TargetVirtualNode
 - Abmessungen des virtuellen Gateways — Mesh VirtualGateway, TargetVirtualService, TargetVirtualNode

- **HTTPCode_Target_2XX_Count**

- `envoy.appmesh.HTTPCode_Target_2XX_Count`— Die Anzahl der HTTP-Anfragen an ein Ziel vor Envoy, die zu einer 2xx-HTTP-Antwort führten.
- Abmessungen:
 - Dimensionen virtueller Knoten — Mesh, VirtualNode, TargetVirtualService, TargetVirtualNode
 - Abmessungen des virtuellen Gateways — Mesh VirtualGateway, TargetVirtualService, TargetVirtualNode

- **HTTPCode_Target_3XX_Count**

- `envoy.appmesh.HTTPCode_Target_3XX_Count`— Die Anzahl der HTTP-Anfragen an ein Ziel vor Envoy, die zu einer 3xx-HTTP-Antwort führten.
- Abmessungen:
 - Dimensionen virtueller Knoten — Mesh, VirtualNode, TargetVirtualService, TargetVirtualNode
 - Abmessungen des virtuellen Gateways — Mesh VirtualGateway, TargetVirtualService, TargetVirtualNode

- **HTTPCode_Target_4XX_Count**

- `envoy.appmesh.HTTPCode_Target_4XX_Count`— Die Anzahl der HTTP-Anfragen an ein Ziel vor Envoy, die zu einer 4xx-HTTP-Antwort führten.

- **Abmessungen:**
 - Dimensionen virtueller Knoten — Mesh, VirtualNode, TargetVirtualService, TargetVirtualNode
 - Abmessungen des virtuellen Gateways — Mesh VirtualGateway, TargetVirtualService, TargetVirtualNode
- **HTTPCode_Target_5XX_Count**
 - `envoy.appmesh.HTTPCode_Target_5XX_Count`— Die Anzahl der HTTP-Anfragen an ein Ziel vor Envoy, die zu einer 5xx-HTTP-Antwort führten.
 - **Abmessungen:**
 - Dimensionen virtueller Knoten — Mesh, VirtualNode, TargetVirtualService, TargetVirtualNode
 - Abmessungen des virtuellen Gateways — Mesh VirtualGateway, TargetVirtualService, TargetVirtualNode
- **RequestCountPerTarget**
 - `envoy.appmesh.RequestCountPerTarget`— Die Anzahl der Anfragen, die an ein Ziel vor Envoy gesendet wurden.
 - **Abmessungen:**
 - Dimensionen virtueller Knoten — Mesh, VirtualNode, TargetVirtualService, TargetVirtualNode
 - Abmessungen des virtuellen Gateways — Mesh VirtualGateway, TargetVirtualService, TargetVirtualNode
- **TargetResponseTime**
 - `envoy.appmesh.TargetResponseTime`— Die Zeit, die verstrichen ist, als eine Anfrage an ein Ziel vor dem Envoy gestellt wurde, bis zum Eingang der vollständigen Antwort.
 - **Abmessungen:**
 - Dimensionen virtueller Knoten — Mesh, VirtualNode, TargetVirtualService, TargetVirtualNode
 - Abmessungen des virtuellen Gateways — Mesh VirtualGateway, TargetVirtualService, TargetVirtualNode

Datadog für App Mesh

Datadog ist eine Überwachungs- und Sicherheitsanwendung für die durchgängige Überwachung, Metrik und Protokollierung von Cloud-Anwendungen. Datadog macht Ihre Infrastruktur, Anwendungen und Anwendungen von Drittanbietern vollständig beobachtbar.

Installieren Sie Datadog

- EKS - Um Datadog mit EKS einzurichten, folgen Sie diesen Schritten in den [Datadog-Dokumenten](#).
- ECS EC2 — Um Datadog mit ECS EC2 einzurichten, folgen Sie diesen Schritten aus den [Datadog-Dokumenten](#).

Für weitere Informationen über Datadog

- [Datadog Dokumentation](#)

Nachverfolgung

Important

Um die Ablaufverfolgung vollständig zu implementieren, müssen Sie Ihre Anwendung aktualisieren.

Um alle verfügbaren Daten des von Ihnen ausgewählten Dienstes zu sehen, müssen Sie Ihre Anwendung mithilfe der entsprechenden Bibliotheken instrumentieren.

App Mesh mit AWS X-Ray überwachen

AWS X-Ray ist ein Service, der Tools zum Anzeigen, Filtern und Analysieren der Daten, die Ihre Anwendung verarbeitet, und bietet Tools zum Anzeigen, Filtern und Analysieren der Daten, die Ihre Anwendung verarbeitet, und Analysieren der Daten, die Ihre Anwendung verarbeitet, an die Daten. Diese Erkenntnisse helfen Ihnen, Probleme und Möglichkeiten zur Optimierung Ihrer App zu identifizieren. Sie können detaillierte Informationen zu Anfragen und Antworten sowie zu Downstream-Aufrufen sehen, die Ihre Anwendung an andere AWS Dienste tätigt.

X-Ray lässt sich in App Mesh integrieren, um Ihre Envoy-Microservices zu verwalten. Trace-Daten von Envoy werden an den X-Ray-Daemon gesendet, der in Ihrem Container läuft.

Implementieren Sie X-Ray in Ihren Anwendungscode, indem Sie den [SDK-Leitfaden](#) verwenden, der für Ihre Sprache spezifisch ist.

Aktivieren Sie X-Ray-Ablaufverfolgung über App Mesh

- Abhängig von der Art der Dienstleistung:
 - ECS — Stellen Sie in der Envoy-Proxy-Container-Definition die `ENABLE_ENVOY_XRAY_TRACING` Umgebungsvariable auf `1` und die `XRAY_DAEMON_PORT` Umgebungsvariable auf `2000`.
 - EKS — Schließen Sie in der App Mesh Controller-Konfiguration `--set tracing.enabled=true` und ein `--set tracing.provider=x-ray`.
- Stellen Sie in Ihrem X-Ray-Container den Port bereit `2000` und führen Sie ihn als Benutzer `aus1337`.

X-Beispiele

Eine Envoy-Container-Definition für 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
  }
}
```

Aktualisierung des App Mesh Mesh-Controllers für 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
```

Walk-Ablaufverfolgungen zur X-Ray-Verwendung

- [Monitor mit AWS X-Ray](#)
- [App Mesh mit Amazon EKS — Beobachtbarkeit: X-Ray](#)
- [Verteiltes Tracing mit X-Ray](#) in der AWS App Mesh [Werkstatt](#)

Um mehr über AWS X-Ray zu erfahren

- [AWS Röntgendokumentation](#)

Fehlerbehebung bei AWS X-Ray mit App Mesh

- [X-Ray-Traces AWS X-Ray meine Anwendungen können nicht angezeigt werden.](#)

Jäger für App Mesh mit Amazon EKS

Jaeger ist ein durchgängiges Open-Source-System für die verteilte Verfolgung. Es kann zur Profilierung von Netzwerken und zur Überwachung verwendet werden. Jaeger kann Ihnen auch bei der Problembehandlung komplexer Cloud-nativer Anwendungen behilflich sein.

Um Jaeger in Ihren Anwendungscode zu implementieren, finden Sie den für Ihre Sprache spezifischen Leitfaden in den [Jaeger-Dokumentationsverfolgungsbibliotheken](#).

Jaeger mit Helm installieren

1. Fügen Sie das EKS-Repository zu Helm hinzu:

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

2. Installieren Sie App Mesh Jaeger

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

Beispiel Jaeger

Im Folgenden finden Sie ein Beispiel für die Erstellung eines persistenten Speichers `PersistentVolumeClaim` für 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
```

Komplettlösung für die Verwendung des Jaeger

- [App Mesh mit EKS — Beobachtbarkeit: Jaeger](#)

Für weitere Informationen über Jaeger

- [Jaeger-Dokumentation](#)

Datadog zur Nachverfolgung

Datadog kann sowohl für die Nachverfolgung als auch für Metriken verwendet werden. Weitere Informationen und Installationsanweisungen finden Sie in der DataDog-Dokumentation in der [Dokumentation zu Datadog](#).

App Mesh Tools

App Mesh bietet Kunden die Möglichkeit, indirekt mit seinen APIs zu interagieren, indem sie Tools wie die folgenden verwenden:

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

App Mesh und AWS CloudFormation

AWS CloudFormation ist ein Dienst, mit dem Sie eine Vorlage mit allen Ressourcen erstellen können, die Sie für Ihre Anwendung benötigen. Anschließend werden die Ressourcen für Sie konfiguriert und bereitgestellt. Außerdem werden alle Abhängigkeiten konfiguriert, sodass Sie sich mehr auf Ihre Anwendung und weniger auf die Verwaltung von Ressourcen konzentrieren können.

Weitere Informationen und Beispiele zur Verwendung AWS CloudFormation mit App Mesh finden Sie in der [AWS CloudFormation Dokumentation](#).

App Mesh und AWS CDK

AWS CDK ist ein Entwicklungsframework, mit dem Sie Ihre Cloud-Infrastruktur mithilfe von Code definieren und bereitstellen können. AWS CDK unterstützt mehrere Programmiersprachen TypeScript, JavaScript, darunter Python, Java und C#.Net.

Weitere Informationen zur Verwendung AWS CDK mit App Mesh finden Sie in der [AWS CDK Dokumentation](#).

App Mesh Controller für Kubernetes

Der App Mesh Mesh-Controller für Kubernetes hilft Ihnen, Ihre App Mesh Mesh-Ressourcen für einen Kubernetes-Cluster zu verwalten und Sidecars in Pods einzufügen. Dieser Controller ist speziell für die Verwendung mit Amazon EKS vorgesehen und ermöglicht es Ihnen, Ihre Ressourcen auf eine Art und Weise zu verwalten, die für Kubernetes typisch ist.

Weitere Informationen zum App Mesh Controller finden Sie in der [App Mesh Controller-Dokumentation](#).

Eine Anleitung zur Implementierung von App Mesh mit Amazon EKS mithilfe des App Mesh Controllers für Kubernetes finden Sie im [Amazon EKS Workshop](#).

App Mesh und Terraform

[Terraform](#) ist eine Open-Source-Infrastruktur als Code-Software-Tool. Terraform kann Cloud-Dienste über ihre CLI verwalten und interagiert mit APIs mithilfe deklarativer Konfigurationsdateien.

Weitere Informationen zur Verwendung von App Mesh mit Terraform finden Sie in der [Terraform-Dokumentation](#).

Mit gemeinsam genutzten Meshes arbeiten

Mithilfe des AWS Resource Access Manager Dienstes können Sie Ihre App Mesh-Meshes AWS für mehrere Konten freigeben. Ein gemeinsam genutztes Mesh ermöglicht es Ressourcen, die von verschiedenen AWS Konten erstellt wurden, im selben Mesh miteinander zu kommunizieren.

Ein AWS Konto kann ein Eigentümer einer Mesh-Ressource, ein Mesh-Nutzer oder beides sein. Verbraucher können Ressourcen in einem Mesh erstellen, das mit ihrem Konto gemeinsam genutzt wird. Besitzer können Ressourcen in jedem Mesh erstellen, das dem Konto gehört. Ein Mesh-Besitzer kann ein Mesh mit den folgenden Arten von Mesh-Nutzern teilen.

- Spezifische AWS Konten innerhalb oder außerhalb seiner Organisation in AWS Organizations
- Eine Organisationseinheit innerhalb ihrer Organisation in AWS Organizations
- Ihre gesamte Organisation ist in AWS Organizations

Eine Anleitung end-to-end zur gemeinsamen Nutzung eines Meshs finden Sie unter [Accountübergreifendes Mesh](#) am GitHub.

Erteilen von Berechtigungen zum Teilen von Meshes

Bei der kontenübergreifenden Freigabe von Meshes sind Berechtigungen für den IAM-Prinzipal, der das Mesh gemeinsam nutzt, und Berechtigungen auf Ressourcenebene für das Mesh selbst erforderlich.

Erteilen Sie die Erlaubnis, ein Mesh gemeinsam zu nutzen

Damit ein IAM-Prinzipal ein Mesh gemeinsam nutzen kann, ist ein Mindestsatz an Berechtigungen erforderlich. Wir empfehlen, die IAM-Richtlinien `AWSAppMeshFullAccess` und die `AWSResourceAccessManagerFullAccess` verwalteten IAM-Richtlinien zu verwenden, um sicherzustellen, dass Ihre IAM-Prinzipale über die erforderlichen Berechtigungen verfügen, um gemeinsam genutzte Meshes zu teilen und zu verwenden.

Wenn Sie eine benutzerdefinierte IAM-Richtlinie verwenden, sind die Aktionen, `appmesh:PutMeshPolicy` und `appmesh:GetMeshPolicy` erforderlich. `appmesh>DeleteMeshPolicy` Dabei handelt es sich um IAM-Aktionen, für die nur Berechtigungen erforderlich sind. Wenn einem IAM-Prinzipal diese Berechtigungen nicht erteilt wurden, tritt beim Versuch, das Mesh über den Service gemeinsam zu nutzen, ein Fehler auf. AWS RAM

Weitere Informationen darüber, wie der AWS Resource Access Manager Dienst IAM verwendet, finden Sie unter [Wie AWS RAM verwendet IAM](#) im AWS Resource Access Manager Benutzerhandbuch.

Erteilen von Berechtigungen für ein Mesh

Ein gemeinsam genutztes Mesh hat die folgenden Berechtigungen.

- Verbraucher können alle Ressourcen in einem Mesh, das mit dem Konto gemeinsam genutzt wird, auflisten und beschreiben.
- Besitzer können alle Ressourcen in jedem Mesh, das dem Konto gehört, auflisten und beschreiben.
- Eigentümer und Verbraucher können Ressourcen in einem Mesh ändern, das das Konto erstellt hat, aber sie können keine Ressourcen ändern, die von einem anderen Konto erstellt wurden.
- Verbraucher können jede Ressource in einem Mesh löschen, die das Konto erstellt hat.
- Besitzer können jede Ressource in einem Mesh löschen, die von einem beliebigen Konto erstellt wurde.
- Die Ressourcen des Besitzers können nur auf andere Ressourcen im selben Konto verweisen. Beispielsweise kann ein virtueller Knoten nur auf ein AWS Certificate Manager Zertifikat AWS Cloud Map verweisen, das sich im selben Konto wie der Besitzer des virtuellen Knotens befindet.
- Besitzer und Verbraucher können einen Envoy-Proxy mit App Mesh als virtuellen Knoten verbinden, den das Konto besitzt.
- Besitzer können virtuelle Gateways und virtuelle Gateway-Routen erstellen.
- Eigentümer und Verbraucher können in einem Mesh, das das Konto erstellt hat, Tags auflisten und Ressourcen kennzeichnen/deren Markierung aufheben. Sie können in einem Mesh keine Tags auflisten und Ressourcen, die nicht vom Konto erstellt wurden, kennzeichnen/deren Markierung aufheben.

Gemeinsam genutzte Meshes verwenden eine richtlinienbasierte Autorisierung. Ein Mesh wird mit einem festen Satz von Berechtigungen gemeinsam genutzt. Diese Berechtigungen werden so ausgewählt, dass sie einer Ressourcenrichtlinie hinzugefügt werden. Je nach IAM-Benutzer/Rolle kann auch eine optionale IAM-Richtlinie ausgewählt werden. Der Zugriff eines Principals auf das Mesh hängt von der Schnittmenge der in diesen Richtlinien zulässigen Berechtigungen ab, abzüglich aller ausdrücklich verweigerten Berechtigungen.

Wenn Sie ein Mesh teilen, erstellt der AWS Resource Access Manager Service eine verwaltete Richtlinie mit dem Namen `AWSRAMDefaultPermissionAppMesh` und verknüpft sie mit Ihrem App Mesh, das die folgenden Berechtigungen bietet.

- `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`

Voraussetzungen für das Teilen von Meshes

Um ein Mesh gemeinsam nutzen zu können, müssen Sie die folgenden Voraussetzungen erfüllen.

- Sie müssen das Mesh in Ihrem AWS Konto besitzen. Sie können ein Mesh, das mit Ihnen geteilt wurde, nicht teilen.
- Um ein Mesh mit Ihrer Organisation oder einer Organisationseinheit in zu teilen AWS Organizations, müssen Sie das Teilen mit aktivieren AWS Organizations. Weitere Informationen finden Sie unter [Freigabe für AWS Organizations aktivieren](#) im AWS RAM -Benutzerhandbuch.
- Ihre Services müssen in einer Amazon-VPC bereitgestellt werden, die über eine gemeinsame Konnektivität für alle Konten verfügt, die die Mesh-Ressourcen enthalten, die Sie miteinander kommunizieren möchten. Eine Möglichkeit, Netzwerkkonnektivität gemeinsam zu nutzen, besteht darin, alle Dienste, die Sie in Ihrem Mesh verwenden möchten, in einem gemeinsam genutzten Subnetz bereitzustellen. Weitere Informationen und Einschränkungen finden Sie unter [Ein Subnetz gemeinsam nutzen](#).
- Dienste müssen über DNS oder auffindbar sein. AWS Cloud Map Weitere Informationen zur Diensterkennung finden Sie unter [Virtuelle Knoten](#).

Zugehörige Services

Mesh Sharing ist in AWS Resource Access Manager (AWS RAM) integriert. AWS RAM ist ein Dienst, mit dem Sie Ihre AWS Ressourcen mit einem beliebigen AWS Konto oder über dieses teilen können AWS Organizations. Mit können Sie Ressourcen AWS RAM, die Ihnen gehören, gemeinsam nutzen, indem Sie eine gemeinsame Nutzung erstellen. Eine Ressourcenfreigabe legt die freizugebenden Ressourcen und die Konsumenten fest, für die sie freigegeben werden sollen. Bei Verbrauchern kann es sich um einzelne AWS Konten, Organisationseinheiten oder eine gesamte Organisation handeln AWS Organizations.

Weitere Informationen zu AWS RAM finden Sie im [AWS RAM Benutzerhandbuch](#).

Ein Mesh teilen

Durch die gemeinsame Nutzung eines Meshs können Mesh-Ressourcen, die von verschiedenen Konten erstellt wurden, im selben Mesh miteinander kommunizieren. Sie können nur ein Mesh teilen, das Sie besitzen. Um ein Netz gemeinsam zu nutzen, müssen Sie es zu einer gemeinsam genutzten Ressource hinzufügen. Eine Ressourcenfreigabe ist eine AWS RAM Ressource, mit der Sie Ihre Ressourcen für mehrere AWS Konten gemeinsam nutzen können. Eine Ressourcenfreigabe gibt die Ressourcen an, die gemeinsam genutzt werden sollen, und die Verbraucher, mit denen sie geteilt werden. Wenn Sie ein Mesh über die Amazon Linux-Konsole teilen, fügen Sie es zu

einer vorhandenen Ressourcenfreigabe hinzu. Um das Mesh zu einer neuen Ressourcenfreigabe hinzuzufügen, erstellen Sie die Ressourcenfreigabe mithilfe der [AWS RAM Konsole](#).

Wenn Sie Teil einer Organisation in Ihrer Organisation sind AWS Organizations und die gemeinsame Nutzung innerhalb Ihrer Organisation aktiviert ist, kann Verbrauchern in Ihrer Organisation automatisch Zugriff auf das gemeinsam genutzte Mesh gewährt werden. Andernfalls erhalten Verbraucher eine Einladung, dem Resource Share beizutreten, und erhalten nach Annahme der Einladung Zugriff auf das gemeinsame Mesh.

Sie können ein Mesh, das Ihnen gehört, über die AWS RAM Konsole oder das teilen AWS CLI.

Um ein Mesh, das Ihnen gehört, über die AWS RAM Konsole zu teilen

Anweisungen finden Sie im AWS RAM Benutzerhandbuch unter [Creating a Resource Share](#).

Wenn Sie einen Ressourcentyp auswählen, wählen Sie Netze und dann das Netz aus, das Sie gemeinsam nutzen möchten. Wenn keine Netze aufgeführt sind, erstellen Sie zuerst ein Netz.

Weitere Informationen finden Sie unter [Servicegitter erstellen](#).

Um ein Netz, das Ihnen gehört, gemeinsam zu nutzen, verwenden Sie den AWS CLI

Verwenden Sie den [create-resource-share](#)-Befehl. Geben Sie für die `--resource-arns` Option den ARN des Meshs an, das Sie teilen möchten.

Die gemeinsame Nutzung eines gemeinsam genutzten Meshs aufheben

Wenn Sie die gemeinsame Nutzung eines Meshs rückgängig machen, deaktiviert App Mesh den weiteren Zugriff auf das Mesh durch ehemalige Nutzer des Meshs. App Mesh löscht jedoch nicht die von den Verbrauchern erstellten Ressourcen. Nachdem die gemeinsame Nutzung des Meshs aufgehoben wurde, kann nur der Mesh-Besitzer auf die Ressourcen zugreifen und sie löschen. App Mesh verhindert, dass das Konto, das Ressourcen im Mesh besaß, Konfigurationsinformationen empfängt, nachdem das Mesh nicht mehr gemeinsam genutzt wurde. App Mesh verhindert auch, dass andere Konten mit Ressourcen im Mesh Konfigurationsinformationen von einem nicht gemeinsam genutzten Mesh erhalten. Nur der Besitzer des Meshs kann die gemeinsame Nutzung rückgängig machen.

Um die gemeinsame Nutzung eines Meshs, dessen Eigentümer Sie sind, rückgängig zu machen, müssen Sie es aus der Ressourcenfreigabe entfernen. Sie können dies mit der AWS RAM Konsole oder dem AWS CLI tun.

Um die Freigabe eines gemeinsam genutzten Meshs, das Sie besitzen, mithilfe der AWS RAM Konsole rückgängig zu machen

Eine Anleitung dazu finden Sie im AWS RAM Benutzerhandbuch unter [Aktualisieren einer gemeinsam genutzten Ressource](#).

Um die gemeinsame Nutzung eines Netzes rückgängig zu machen, dessen Eigentümer Sie sind, verwenden Sie AWS CLI

Verwenden Sie den [disassociate-resource-share](#)-Befehl.

Identifizieren eines gemeinsam genutzten Netzes

Eigentümer und Verbraucher können gemeinsam genutzte Meshes und Mesh-Ressourcen mithilfe der Amazon Linux-Konsole identifizieren und AWS CLI

So identifizieren Sie ein gemeinsam genutztes Mesh mithilfe der Amazon Linux-Konsole

1. Öffnen Sie die App Mesh Mesh-Konsole unter <https://console.aws.amazon.com/appmesh/>.
2. Wählen Sie in der linken Navigationsleiste Meshes aus. Die Konto-ID des Mesh-Besitzers für jedes Mesh ist in der Spalte Mesh-Besitzer aufgeführt.
3. Wählen Sie in der linken Navigationsleiste Virtuelle Dienste, Virtuelle Router oder Virtuelle Knoten aus. Sie sehen die Konto-ID für den Mesh-Besitzer und den Ressourcenbesitzer für jede der Ressourcen.

Um ein gemeinsam benutztes Mesh zu identifizieren, verwenden Sie AWS CLI

Verwenden Sie den `aws appmesh list resource` Befehl, z. `aws appmesh list-meshes`. Der Befehl gibt die Netze zurück, die Sie besitzen, und die Netze, die mit Ihnen gemeinsam genutzt werden. Die `meshOwner` Eigenschaft zeigt die AWS Konto-ID von `meshOwner` und die `resourceOwner` Eigenschaft zeigt die AWS Konto-ID des Ressourcenbesitzers. Jeder Befehl, der für eine Mesh-Ressource ausgeführt wird, gibt diese Eigenschaften zurück.

Die benutzerdefinierten Tags, die Sie einem gemeinsam genutzten Netz zuordnen, sind nur für Sie verfügbar AWS-Konto. Sie sind für die anderen Konten, mit denen das Mesh geteilt wird, nicht verfügbar. Dem `aws appmesh list-tags-for-resource` Befehl für ein Mesh in einem anderen Konto wurde der Zugriff verweigert.

Fakturierung und Messung

Für die gemeinsame Nutzung eines Meshs fallen keine Gebühren an.

Kontingente für Instanzen

Alle Kontingente für ein Mesh gelten auch für gemeinsam genutzte Meshes, unabhängig davon, wer Ressourcen im Mesh erstellt hat. Nur ein Mesh-Besitzer kann eine Erhöhung der Quote beantragen. Weitere Informationen finden Sie unter [App Mesh service quotas](#). Der AWS Resource Access Manager Dienst hat auch Kontingente. Weitere Informationen finden Sie unter [Service Quotas](#).

AWS In App Mesh integrierte Dienste

App Mesh kann mit anderen AWS -Services verwendet werden, um zusätzliche Lösungen für Ihre geschäftlichen Herausforderungen bereitzustellen. In diesem Thema werden Services aufgeführt, die mithilfe von App Mesh ihre Funktionalität erweitern, sowie Services, die App Mesh zur Ausführung von Aufgaben

Inhalt

- [App Mesh Mesh-Ressourcen erstellen mit AWS CloudFormation](#)
- [App Mesh auf AWS Outposts](#)

App Mesh Mesh-Ressourcen erstellen mit AWS CloudFormation

App Mesh ist in integriert. Dies ist ein Service AWS CloudFormation, der Ihnen hilft, Ihre AWS -Ressourcen zu modellieren und einzurichten, so dass Sie weniger Zeit für die Erstellung und Verwaltung Ihrer Ressourcen und Infrastruktur mit Sie erstellen eine Vorlage, in der alle gewünschten AWS -Ressourcen, wie etwa ein App Mesh, beschrieben werden, und AWS CloudFormation übernimmt die Bereitstellung und Konfigurierung dieser Ressourcen für Sie.

Wenn Sie verwenden AWS CloudFormation, Sie verwenden Ihre Vorlage, um Ihre App Mesh Mesh-Ressourcen einheitlich und wiederholt einzurichten. Sie beschreiben Ihre Ressourcen dann einmal und können die gleichen Ressourcen dann in mehreren AWS-Konten und -Regionen immer wieder bereitstellen.

App Mesh und AWS CloudFormation Vorlagen

Um Ressourcen für App Mesh und zu konfigurieren, müssen Sie -Vorlagen -Vorlagen Sie -Vorlagen Sie -Vorlagen Sie -Vorlagen Sie -Vorlagen für die Verwendung von Sie [AWS CloudFormation-Vorlagen](#) Vorlagen sind formatierte Textdateien in JSON oder YAML. Diese Vorlagen beschreiben die Ressourcen, die Sie in Ihren AWS CloudFormation-Stacks bereitstellen möchten. Wenn Sie noch keine Erfahrungen mit JSON oder YAML haben, können Sie AWS CloudFormation Designer verwenden, der den Einstieg in die Arbeit mit AWS CloudFormation-Vorlagen erleichtert. Weitere Informationen finden Sie unter [Was ist AWS CloudFormation-Designer?](#) im AWS CloudFormation-Benutzerhandbuch.

App Mesh unterstützt das Erstellen von Meshes, Routen, virtuellen Knoten, virtuellen Routern und virtuellen Diensten in AWS CloudFormation. Weitere Informationen, einschließlich Beispielen für

JSON- und YAML-Vorlagen für Ihre App Mesh Mesh-Ressourcen, finden Sie in der [Referenz zum - App-Mesh-Ressourcentyp](#) im AWS CloudFormation-Benutzerhandbuch.

Weitere Informationen zu AWS CloudFormation

Weitere Informationen zu AWS CloudFormation finden Sie in den folgenden Ressourcen.

- [AWS CloudFormation](#)
- [AWS CloudFormation-Benutzerhandbuch](#)
- [AWS CloudFormation-Benutzerhandbuch für die Befehlszeilenschnittstelle](#)

App Mesh aufAWS Outposts

AWSOutposts ermöglicht nativeAWS -Services, Infrastrukturen und Betriebsmodelle in On-Premises-Einrichtungen. Sie könnenAWS Outposts Outposts-Umgebungen die gleichenAWS APIs und Tools sowie die gleiche Infrastruktur wie in derAWS Cloud verwenden. App Mesh aufAWS Outposts ist ideal für Workloads mit geringer Latenz, die in unmittelbarer Nähe zu On-Premises-Daten und -Anwendungen ausgeführt werden müssen. Weitere Informationen zuAWS Outposts finden Sie im [AWSOutposts-Benutzerhandbuch](#).

Voraussetzungen

Im Folgenden sind die Voraussetzungen für die Verwendung von App Mesh aufAWS Outposts aufgeführt:

- Sie müssen ein Outpost in Ihrem On-Premises-Rechenzentrum installiert und konfiguriert haben.
- Sie müssen über eine zuverlässige Netzwerkverbindung zwischen Ihrem Outpost und der entsprechenden AWS-Region verfügen.
- DieAWS -Region für den Outpost muss unterstützenAWS App Mesh. Eine Liste der unterstützten Regionen finden Sie unter [AWS App MeshEndpunkte und Kontingente](#) in der Allgemeine AWS-Referenz.

Einschränkungen

Im Folgenden sind die Einschränkungen der Verwendung von App Mesh aufAWS Outposts aufgeführt:

- AWS Identity and Access Management, Application Load Balancer, Network Load Balancer, Classic Load Balancer und Amazon Route 53 werden in der AWS -Region, nicht auf Outposts ausgeführt. Dies erhöht die Latenzen zwischen diesen Diensten und den Containern.

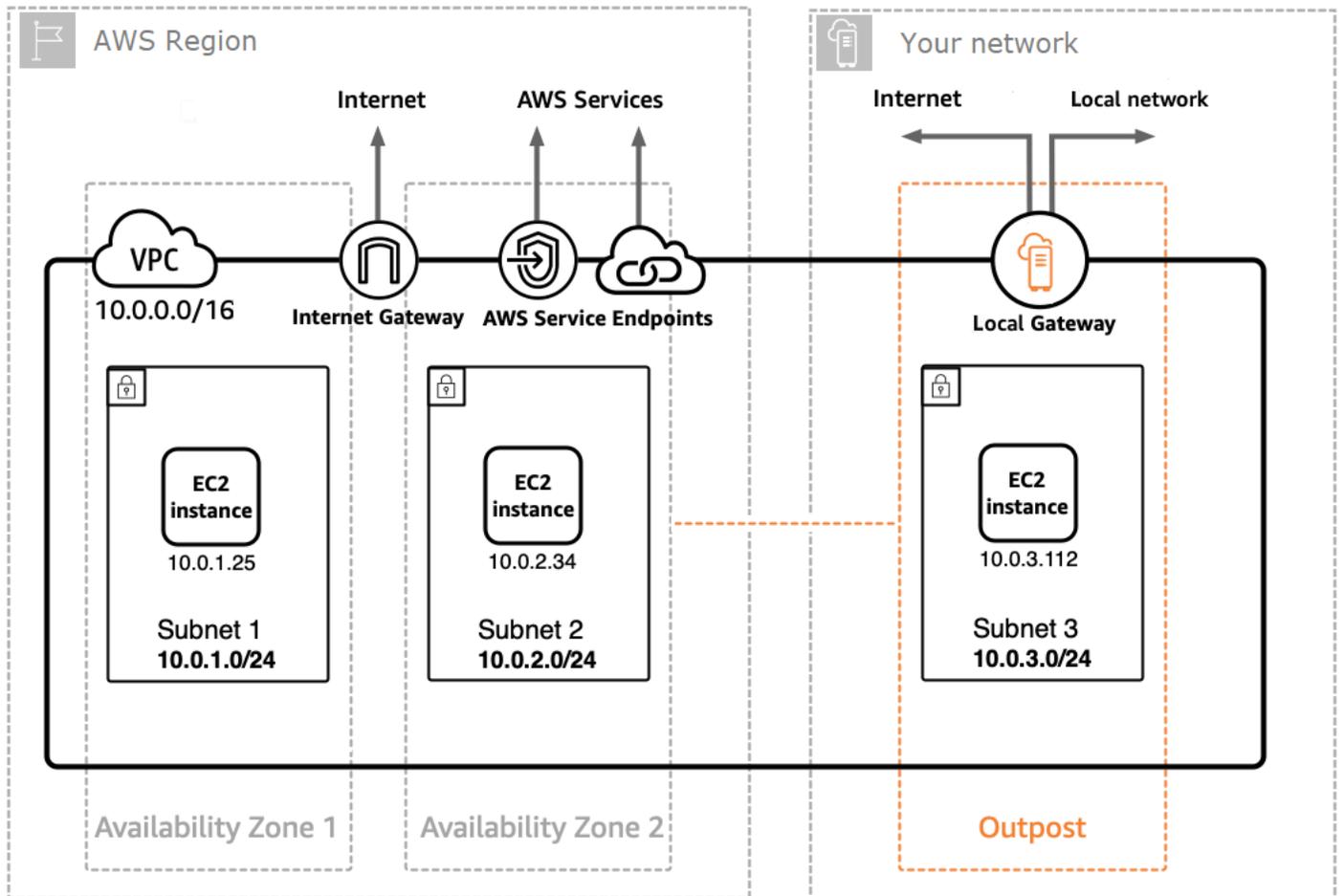
Überlegungen zur Netzwerkkonnektivität

Im Folgenden finden Sie Überlegungen zur Netzwerkkonnektivität für Amazon EKS AWS Outposts:

- Wenn die Netzwerkverbindung zwischen Ihrem Outpost und seiner AWS -Region verloren geht, werden die App Mesh Envoy-Proxys weiterhin ausgeführt. Sie können Ihr Service Mesh jedoch erst ändern, wenn die Konnektivität wiederhergestellt ist.
- Zwischen dem Outpost und seiner AWS -Region sollten Sie eine zuverlässige, hochverfügbare Verbindung mit niedriger Latenz bereitstellen.

Einen App Mesh Envoy-Proxy auf einem Outpost erstellen

Ein Outpost ist eine Erweiterung einer AWS -Region und Sie können eine Amazon VPC in einem Konto auf mehrere Availability Zones und alle zugehörigen erweitern. Wenn Sie den Outpost konfigurieren, ordnen Sie ihm ein Subnetz zu, um Ihre regionale VPC-Umgebung auf Ihre On-Premise-Einrichtung zu erweitern. Instances in einem Outpost werden als Teil Ihrer regionalen VPC angezeigt, ähnlich einer Availability Zone mit zugeordneten Subnetzen.



Um einen App Mesh Envoy-Proxy auf einem Outpost zu erstellen, fügen Sie das App Mesh Envoy-Container-Image zur Amazon ECS-Aufgabe oder zum Amazon EKS-Pod hinzu, der auf einem Outpost ausgeführt wird. Weitere Informationen finden Sie unter [Amazon Elastic Container Service onAWS Outposts](#) im Amazon Elastic Container Service Developer Guide und [Amazon Elastic Kubernetes Service onAWS Outposts](#) im Amazon EKS-Benutzerhandbuch.

Bewährte Methoden für App Mesh

Um das Ziel zu erreichen, dass bei geplanten Bereitstellungen und beim ungeplanten Verlust einiger Hosts keine fehlgeschlagenen Anfragen auftreten, wird mit den bewährten Methoden in diesem Thema die folgende Strategie umgesetzt:

- Erhöhen Sie die Wahrscheinlichkeit, dass eine Anfrage aus Sicht der Anwendung erfolgreich ist, indem Sie eine sichere Standardstrategie für Wiederholungsversuche verwenden. Weitere Informationen finden Sie unter [Instrumentiere alle Routen mit Wiederholungsversuchen](#).
- Erhöhen Sie die Wahrscheinlichkeit, dass eine wiederholte Anfrage erfolgreich ist, indem Sie die Wahrscheinlichkeit maximieren, dass die erneut versuchte Anfrage an ein echtes Ziel gesendet wird. Weitere Informationen finden Sie unter [Bereitstellungsgeschwindigkeit anpassen](#), [Vor Skalierung einskalieren](#) und [Implementieren Sie Zustandsprüfungen für Container](#).

Um Ausfälle deutlich zu reduzieren oder zu vermeiden, empfehlen wir Ihnen, die Empfehlungen in allen folgenden Vorgehensweisen umzusetzen.

Instrumentiere alle Routen mit Wiederholungsversuchen

Konfigurieren Sie alle virtuellen Dienste für die Verwendung eines virtuellen Routers und legen Sie eine Standardrichtlinie für Wiederholungsversuche für alle Routen fest. Dadurch werden fehlgeschlagene Anfragen vermieden, indem ein Host erneut ausgewählt und eine neue Anfrage gesendet wird. Für Wiederholungsrichtlinien empfehlen wir einen Wert von mindestens zwei für `maxRetries` und die Angabe der folgenden Optionen für jeden Wiederholungsereignistyp in jedem Routentyp, der den Wiederholungstyp unterstützt:

- TCP —`connection-error`
- HTTP und HTTP/2 —`stream-error` und `gateway-error`
- gRPC —`cancelled` und `unavailable`

Andere Wiederholungsereignisse müssen entsprechend berücksichtigt werden, case-by-case da sie möglicherweise nicht sicher sind, z. B. wenn die Anfrage nicht idempotent ist. Sie müssen Werte für berücksichtigen und `testen` `perRetryTimeout`, die den angemessenen Kompromiss zwischen der maximalen Latenz einer Anfrage (`maxRetries*perRetryTimeout`) `maxRetries` und der erhöhten Erfolgsquote bei mehr Wiederholungen eingehen. Wenn Envoy versucht, eine Verbindung zu einem

Endpunkt herzustellen, der nicht mehr vorhanden ist, sollten Sie außerdem damit rechnen, dass diese Anfrage vollständig verbraucht wird `perRetryTimeout`. Informationen zum Konfigurieren einer Wiederholungsrichtlinie finden Sie unter [Erstellen einer Route](#) und wählen Sie dann das Protokoll aus, das Sie weiterleiten möchten.

Note

Wenn Sie am oder nach dem 29. Juli 2020 eine Route implementiert und keine Wiederholungsrichtlinie angegeben haben, hat App Mesh möglicherweise für jede Route, die Sie am oder nach dem 29. Juli 2020 erstellt haben, automatisch eine Standard-Wiederholungsrichtlinie erstellt, die der vorherigen Richtlinie ähnelt. Weitere Informationen finden Sie unter [Standardrichtlinie für die Wiederholung von Routen](#).

Bereitstellungsgeschwindigkeit anpassen

Wenn Sie fortlaufende Bereitstellungen verwenden, reduzieren Sie die allgemeine Bereitstellungsgeschwindigkeit. Standardmäßig konfiguriert Amazon ECS eine Bereitstellungsstrategie mit mindestens 100 Prozent fehlerfreien Aufgaben und 200 Prozent Gesamtaufgaben. Beim Einsatz führt dies zu zwei starken Driftpunkten:

- Die 100-prozentige Flottengröße der neuen Aufgaben kann [Implementieren Sie Zustandsprüfungen für Container](#) für die Abgesandten sichtbar sein, bevor sie bereit sind, Anfragen zu bearbeiten (Abhilfemaßnahmen finden Sie unter).
- Die 100-prozentige Flottengröße alter Aufgaben kann für die Gesandten sichtbar sein, während die Aufgaben beendet werden.

Bei der Konfiguration mit diesen Bereitstellungsbeschränkungen können Container-Orchestratoren in einen Zustand übergehen, in dem sie gleichzeitig alle alten Ziele verbergen und alle neuen Ziele sichtbar machen. Da Ihre Envoy-Datenebene letztendlich konsistent ist, kann dies zu Zeiträumen führen, in denen die in Ihrer Datenebene sichtbaren Ziele vom Standpunkt des Orchestrators abweichen. Um dem entgegenzuwirken, empfehlen wir, mindestens 100 Prozent fehlerfreie Aufgaben beizubehalten, die Gesamtzahl der Aufgaben jedoch auf 125 Prozent zu senken. Dadurch werden Abweichungen verringert und die Zuverlässigkeit von Wiederholungsversuchen verbessert. Wir empfehlen die folgende Einstellung:

Amazon ECS

Wenn Ihr Service eine gewünschte Anzahl von zwei oder drei hat, stellen `SiemaximumPercent` den Wert auf 150 Prozent ein. Andernfalls stellen `SiemaximumPercent` den Wert auf 125 Prozent ein.

Kubernetes

Konfigurieren Sie Ihre Bereitstellungsupdate `strategy`, indem `SiemaxUnavailable` sie auf 0 Prozent und `maxSurge` auf 25 Prozent einstellen. Weitere Informationen zu Bereitstellungen finden Sie in der Dokumentation zu [Kubernetes-Bereitstellungen](#).

Vor Skalierung einskalieren

Sowohl das Ausskalieren als auch das Einskalieren können zu einer gewissen Wahrscheinlichkeit führen, dass Anfragen bei Wiederholungsversuchen fehlschlagen. Es gibt zwar Aufgabenempfehlungen, die eine Skalierung verhindern, aber die einzige Empfehlung für die Skalierung besteht darin, den Prozentsatz der gleichzeitig skalierten Aufgaben zu minimieren. Wir empfehlen Ihnen, eine Bereitstellungsstrategie zu verwenden, bei der neue Amazon ECS-Aufgaben oder Kubernetes-Bereitstellungen vor der Skalierung in alten Aufgaben oder Bereitstellungen skaliert werden. Durch diese Skalierungsstrategie wird Ihr Prozentsatz an skalierten Aufgaben oder Bereitstellungen bei gleichbleibender Geschwindigkeit niedriger gehalten. Diese Vorgehensweise gilt sowohl für Amazon ECS-Aufgaben als auch für Kubernetes-Bereitstellungen.

Implementieren Sie Zustandsprüfungen für Container

Im Scale-up-Szenario werden Container in einer Amazon ECS-Aufgabe möglicherweise nicht in der richtigen Reihenfolge angezeigt und reagieren möglicherweise zunächst nicht. Wir empfehlen die folgenden Vorschläge für verschiedene Container-Laufzeiten:

Amazon ECS

Um dem entgegenzuwirken, empfehlen wir die Durchführung von Container-Zustandsprüfungen und der Reihenfolge der Container-Abhängigkeiten, um sicherzustellen, dass Envoy läuft und fehlerfrei ist, bevor Container gestartet werden, die ausgehende Netzwerkkonnektivität benötigen. Informationen zur korrekten Konfiguration eines Anwendungscontainers und eines Envoy-Containers in einer Aufgabendefinition finden Sie unter [Container-Abhängigkeit](#).

Kubernetes

Keine, da [Kubernetes-Liveness- und Readiness-Tests](#) bei der Registrierung und Abmeldung von AWS Cloud Map Instanzen im [App Mesh Mesh-Controller für Kubernetes](#) nicht berücksichtigt werden. Weitere Informationen finden Sie in GitHub Ausgabe [#132](#).

Sicherheit in AWS App Mesh

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das [Modell der geteilten Verantwortung](#) beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- **Sicherheit der Cloud** — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS Dienste in der AWS Cloud ausführt. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Auditoren von Drittanbietern testen und überprüfen die Effektivität unserer Sicherheitsmaßnahmen im Rahmen der [AWS -Compliance-Programme](#) regelmäßig. Weitere Informationen zu den Compliance-Programmen für AWS App Mesh finden Sie unter [Durch das Compliance-Programm abgedeckte AWS -Services](#). App Mesh ist für die sichere Bereitstellung von Konfigurationen an lokale Proxys verantwortlich, einschließlich geheimer Geheimnisse wie private Schlüssel für TLS-Zertifikate.
- **Sicherheit in der Cloud** — Ihre Verantwortung richtet sich nach dem AWS Dienst, den Sie nutzen. Sie sind auch für andere Faktoren verantwortlich, darunter:
 - Die Sensibilität Ihrer Daten, die Anforderungen Ihres Unternehmens und die geltenden Gesetze und Vorschriften.
 - Die Sicherheitskonfiguration der App Mesh Mesh-Datenebene, einschließlich der Konfiguration der Sicherheitsgruppen, die den Verkehr zwischen Diensten innerhalb Ihrer VPC ermöglichen.
 - Die Konfiguration Ihrer Rechenressourcen, die mit App Mesh verknüpft sind.
 - Die mit Ihren Rechenressourcen verknüpften IAM-Richtlinien und welche Konfiguration sie von der App Mesh-Steuerebene abrufen dürfen.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der gemeinsamen Verantwortung bei der Verwendung von App Mesh anwenden können. In den folgenden Themen erfahren Sie, wie Sie App Mesh konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere AWS Dienste verwenden, mit denen Sie Ihre App Mesh Mesh-Ressourcen überwachen und sichern können.

Sicherheitsgrundsatz von App Mesh

Kunden sollten in der Lage sein, die Sicherheit so einzustellen, wie sie es benötigen. Die Plattform sollte sie nicht daran hindern, sicherer zu sein. Die Funktionen der Plattform sind standardmäßig sicher.

Themen

- [Transport Layer Security \(TLS\)](#)
- [Gegenseitige TLS-Authentifizierung](#)
- [Wie AWS App Mesh funktioniert mit IAM](#)
- [Protokollieren von AWS App Mesh API-Aufrufen mit AWS CloudTrail](#)
- [Datenschutz in AWS App Mesh](#)
- [Compliance-Validierung für AWS App Mesh](#)
- [Infrastruktursicherheit in AWS App Mesh](#)
- [Ausfallsicherheit in AWS App Mesh](#)
- [Konfiguration und Schwachstellenanalyse in AWS App Mesh](#)

Transport Layer Security (TLS)

In App Mesh verschlüsselt Transport Layer Security (TLS) die Kommunikation zwischen den Envoy-Proxy, die auf Rechenressourcen bereitgestellt werden, die in App Mesh durch Mesh-Endpunkte wie und repräsentiert werden. [Virtuelle Knoten](#) [Virtuelle Gateways](#) Der Proxy handelt TLS aus und beendet es. Wenn der Proxy mit einer Anwendung bereitgestellt wird, ist Ihr Anwendungscode nicht für die Aushandlung einer TLS-Sitzung verantwortlich. Der Proxy handelt TLS im Namen Ihrer Anwendung aus.

App Mesh ermöglicht es Ihnen, das TLS-Zertifikat auf folgende Weise für den Proxy bereitzustellen:

- Ein privates Zertifikat von AWS Certificate Manager (ACM), das von einem AWS Private Certificate Authority (AWS Private CA) ausgestellt wird.
- Ein Zertifikat, das im lokalen Dateisystem eines virtuellen Knotens gespeichert ist und von Ihrer eigenen Zertifizierungsstelle (CA) ausgestellt wurde
- Ein Zertifikat, das von einem Secrets Discovery Service (SDS) -Endpunkt über einen lokalen Unix-Domain-Socket bereitgestellt wird.

[En\) Proxy-Autorisierung](#) muss für den bereitgestellten Envoy-Proxy aktiviert sein, der durch einen Mesh-Endpunkt repräsentiert wird. Wir empfehlen, bei der Aktivierung der Proxy-Autorisierung den Zugriff nur auf den Mesh-Endpunkt zu beschränken, für den Sie die Verschlüsselung aktivieren.

Zertifikatanforderungen

Einer der Subject Alternative Names (SANs) auf dem Zertifikat muss bestimmten Kriterien entsprechen, je nachdem, wie der tatsächliche Service, der durch einen Mesh-Endpunkt repräsentiert wird, erkannt wird.

- DNS — Eines der Zertifikats-SANs muss dem in den Einstellungen für die DNS-Diensterkennung angegebenen Wert entsprechen. Für eine Anwendung mit dem Namen der Diensterkennung können Sie ein Zertifikat erstellen `mesh-endpoint.apps.local`, das diesem Namen entspricht, oder ein Zertifikat mit dem Platzhalter* `.apps.local`.
- AWS Cloud Map— Eines der Zertifikats-SANs muss mit dem Wert übereinstimmen, der in den AWS Cloud Map Service Discovery-Einstellungen unter Verwendung des Formats angegeben wurde `service-name.namespace-name`. Für eine Anwendung mit den AWS Cloud Map Service Discovery-Einstellungen von ServiceName `mesh-endpoint` und NamespaceName können Sie ein Zertifikat erstellen `apps.local`, das dem Namen entspricht `mesh-endpoint.apps.local`, oder ein Zertifikat mit dem Platzhalter * `.apps.local`.

Für beide Erkennungsmechanismen gilt: Wenn keines der Zertifikats-SANs den Einstellungen für die DNS-Diensterkennung entspricht, schlägt die Verbindung zwischen Envoys fehl und es wird die folgende Fehlermeldung angezeigt, die vom Client Envoy angezeigt wird.

```
TLS error: 268435581:SSL routines:OPENSSL_internal:CERTIFICATE_VERIFY_FAILED
```

TLS-Authentifizierungszertifikate

App Mesh unterstützt mehrere Quellen für Zertifikate, wenn die TLS-Authentifizierung verwendet wird.

AWS Private CA

Das Zertifikat muss in ACM in derselben Region und demselben AWS Konto wie der Mesh-Endpunkt gespeichert werden, der das Zertifikat verwenden wird. Das Zertifikat der Zertifizierungsstelle muss sich nicht im selben AWS Konto befinden, aber es muss sich dennoch in derselben Region wie der Mesh-Endpunkt befinden. Wenn Sie noch kein Zertifikat haben AWS Private CA, müssen Sie [eines erstellen](#), bevor Sie ein Zertifikat von diesem anfordern können.

Weitere Informationen zum Anfordern eines Zertifikats von einem vorhandenen AWS Private CA mithilfe von ACM finden Sie unter [Anfordern eines privaten Zertifikats](#). Das Zertifikat kann kein öffentliches Zertifikat sein.

Bei den privaten Zertifizierungsstellen, die Sie für TLS-Client-Richtlinien verwenden, muss es sich um Root-Benutzer-CAs handeln.

Um einen virtuellen Knoten mit Zertifikaten und Zertifizierungsstellen von zu konfigurieren AWS Private CA, muss der Prinzipal (z. B. ein Benutzer oder eine Rolle), den Sie zum Aufrufen von App Mesh verwenden, über die folgenden IAM-Berechtigungen verfügen:

- Für alle Zertifikate, die Sie der TLS-Konfiguration eines Listeners hinzufügen, muss der Principal über die entsprechende Berechtigung verfügen. `acm:DescribeCertificate`
- Für alle Zertifizierungsstellen, die mit einer TLS-Client-Richtlinie konfiguriert sind, muss der Principal über die `acm-pca:DescribeCertificateAuthority` entsprechende Berechtigung verfügen.

Important

Die gemeinsame Nutzung von Zertifizierungsstellen mit anderen Konten kann dazu führen, dass diese Konten unbeabsichtigte Rechte für die Zertifizierungsstelle erhalten. Wir empfehlen, ressourcenbasierte Richtlinien zu verwenden, um den Zugriff nur `acm-pca:DescribeCertificateAuthority` auf Konten zu beschränken, `acm-pca:GetCertificateAuthorityCertificate` für die keine Zertifikate von der Zertifizierungsstelle ausgestellt werden müssen.

Sie können diese Berechtigungen zu einer vorhandenen IAM-Richtlinie hinzufügen, die einem Prinzipal zugeordnet ist, oder einen neuen Prinzipal und eine neue Richtlinie erstellen und die Richtlinie dem Prinzipal zuordnen. Weitere Informationen finden Sie unter [Bearbeiten von IAM-Richtlinien](#), [Erstellen von IAM-Richtlinien](#) und [Hinzufügen von IAM-Identitätsberechtigungen](#).

Note

Sie zahlen eine monatliche Gebühr für den Betrieb der einzelnen Geräte, AWS Private CA bis Sie sie löschen. Sie zahlen auch für die privaten Zertifikate, die Sie jeden Monat ausstellen, und für private Zertifikate, die Sie exportieren. Weitere Informationen finden Sie unter [AWS Certificate Manager -Preisgestaltung](#).

Wenn Sie die [Proxyautorisierung](#) für den Envoy-Proxy aktivieren, den ein Mesh-Endpunkt darstellt, müssen der von Ihnen verwendeten IAM-Rolle die folgenden IAM-Berechtigungen zugewiesen werden:

- Für alle Zertifikate, die auf dem Listener eines virtuellen Knotens konfiguriert sind, muss die Rolle über die entsprechende Berechtigung verfügen. `acm:ExportCertificate`
- Für alle Zertifizierungsstellen, die für eine TLS-Client-Richtlinie konfiguriert sind, muss die Rolle über die `acm-pca:GetCertificateAuthorityCertificate` entsprechende Berechtigung verfügen.

Dateisystem

Sie können Zertifikate mithilfe des Dateisystems an Envoy verteilen. Sie können dies tun, indem Sie die Zertifikatskette und den entsprechenden privaten Schlüssel im Dateipfad verfügbar machen. Auf diese Weise sind diese Ressourcen vom Envoy-Sidecar-Proxy aus erreichbar.

Der Secret Discovery Service (SDS) des Gesandten

Envoy ruft über das Secrets Discovery-Protokoll Geheimnisse wie TLS-Zertifikate von einem bestimmten Endpunkt ab. [Weitere Informationen zu diesem Protokoll finden Sie in der SDS-Dokumentation von Envoy.](#)

App Mesh konfiguriert den Envoy-Proxy so, dass er einen lokalen Unix-Domain-Socket verwendet, der als Secret Discovery Service (SDS) -Endpunkt dient, wenn SDS als Quelle für Ihre Zertifikate und Zertifikatsketten dient. Sie können den Pfad zu diesem Endpunkt mithilfe der Umgebungsvariablen konfigurieren. `APPMESH_SDS_SOCKET_PATH`

Important

Local Secrets Discovery Service, der Unix Domain Socket verwendet, wird auf App Mesh Envoy Proxy Version 1.15.1.0 und höher unterstützt.

App Mesh unterstützt das V2 SDS-Protokoll mit gRPC.

Integration mit SPIFFE Runtime Environment (SPIRE)

Sie können jede beliebige Sidecar-Implementierung der SDS-API verwenden, einschließlich vorhandener Toolchains wie [SPIFFE Runtime Environment \(SPIRE\)](#). SPIRE wurde entwickelt, um die Implementierung der gegenseitigen TLS-Authentifizierung zwischen mehreren Workloads in verteilten Systemen zu ermöglichen. Es bestätigt die Identität von Workloads zur Laufzeit. SPIRE

stellt außerdem für Workloads spezifische, kurzlebige und automatisch rotierende Schlüssel und Zertifikate direkt für Workloads bereit.

Sie sollten den SPIRE-Agenten als SDS-Anbieter für Envoy konfigurieren. Erlauben Sie ihm, Envoy direkt mit dem Schlüsselmaterial zu versorgen, das es für die gegenseitige TLS-Authentifizierung benötigt. Führen Sie SPIRE-Agenten in Sidecars neben Envoy-Proxys aus. Der Agent kümmert sich bei Bedarf um die Neugenerierung der kurzlebigen Schlüssel und Zertifikate. Der Agent bestätigt Envoy und bestimmt, welche Dienstidentitäten und CA-Zertifikate er Envoy zur Verfügung stellen soll, wenn Envoy eine Verbindung zu dem vom SPIRE-Agent offengelegten SDS-Server herstellt.

Während dieses Vorgangs werden Dienstidentitäten und CA-Zertifikate rotiert, und Updates werden zurück an Envoy gestreamt. Envoy wendet sie sofort auf neue Verbindungen an, ohne dass es zu Unterbrechungen oder Ausfallzeiten kommt und ohne dass die privaten Schlüssel jemals das Dateisystem berühren.

So konfiguriert App Mesh Envoys für die Aushandlung von TLS

App Mesh verwendet die Mesh-Endpunktconfiguration des Clients und des Servers, um zu bestimmen, wie die Kommunikation zwischen Envoys in einem Mesh konfiguriert werden soll.

Mit Client-Richtlinien

Wenn eine Client-Richtlinie die Verwendung von TLS erzwingt und einer der Ports in der Client-Richtlinie dem Port der Serverrichtlinie entspricht, wird die Client-Richtlinie verwendet, um den TLS-Validierungskontext des Clients zu konfigurieren. Wenn beispielsweise die Client-Richtlinie eines virtuellen Gateways mit der Serverrichtlinie eines virtuellen Knotens übereinstimmt, wird versucht, mithilfe der in der Client-Richtlinie des virtuellen Gateways definierten Einstellungen eine TLS-Verhandlung zwischen den Proxys durchzuführen. Wenn die Client-Richtlinie nicht mit dem Port der Serverrichtlinie übereinstimmt, kann je nach den TLS-Einstellungen der Serverrichtlinie TLS zwischen den Proxys ausgehandelt werden oder auch nicht.

Ohne Client-Richtlinien

Wenn der Client keine Client-Richtlinie konfiguriert hat oder die Client-Richtlinie nicht mit dem Port des Servers übereinstimmt, verwendet App Mesh den Server, um zu bestimmen, ob und wie TLS vom Client ausgehandelt werden soll oder nicht. Wenn beispielsweise ein virtuelles Gateway keine Client-Richtlinie angegeben hat und ein virtueller Knoten keine TLS-Terminierung konfiguriert hat, wird TLS nicht zwischen den Proxys ausgehandelt. Wenn ein Client keine

passende Client-Richtlinie angegeben hat und ein Server mit TLS-Modi STRICT oder konfiguriert wurde, werden die Proxys so konfiguriert PERMISSIVE, dass sie TLS aushandeln. Je nachdem, wie die Zertifikate für die TLS-Terminierung bereitgestellt wurden, gilt das folgende zusätzliche Verhalten.

- Von ACM verwaltete TLS-Zertifikate — Wenn ein Server die TLS-Terminierung mithilfe eines von ACM verwalteten Zertifikats konfiguriert hat, konfiguriert App Mesh die Clients automatisch so, dass sie TLS aushandeln und das Zertifikat anhand der Root-Benutzer-CA validieren, mit der das Zertifikat verknüpft ist.
- Dateibasierte TLS-Zertifikate — Wenn ein Server die TLS-Terminierung mithilfe eines Zertifikats aus dem lokalen Dateisystem des Proxys konfiguriert hat, konfiguriert App Mesh automatisch einen Client für die Aushandlung von TLS, aber das Zertifikat des Servers wird nicht validiert.

Alternative Namen für den Betreff

Sie können optional eine Liste mit alternativen Namen (Subject Alternative Names, SANs) angeben, denen Sie vertrauen möchten. SANs müssen im FQDN- oder URI-Format vorliegen. Wenn SANs bereitgestellt werden, überprüft Envoy, ob der alternative Subject Name des vorgelegten Zertifikats mit einem der Namen auf dieser Liste übereinstimmt.

Wenn Sie keine SANs auf dem abschließenden Mesh-Endpunkt angeben, verifiziert der Envoy-Proxy für diesen Knoten das SAN auf einem Peer-Client-Zertifikat nicht. Wenn Sie auf dem Ursprungs-Mesh-Endpunkt keine SANs angeben, muss das SAN auf dem vom Zielendpunkt bereitgestellten Zertifikat mit der Konfiguration der Mesh-Endpunkt-Serviceerkennung übereinstimmen.

Weitere Informationen finden Sie unter App Mesh [TLS: Zertifikatsanforderungen](#).

Important

Sie können Wildcard-SANs nur verwenden, wenn die Client-Richtlinie für TLS auf `not enforced` eingestellt ist. Wenn die Client-Richtlinie für den virtuellen Clientknoten oder das virtuelle Gateway so konfiguriert ist, dass TLS erzwungen wird, kann sie kein Platzhalter-SAN akzeptieren.

Überprüfen Sie die Verschlüsselung

Sobald Sie TLS aktiviert haben, können Sie den Envoy-Proxy abfragen, um zu bestätigen, dass die Kommunikation verschlüsselt ist. Der Envoy-Proxy gibt Statistiken über Ressourcen aus,

anhand derer Sie nachvollziehen können, ob Ihre TLS-Kommunikation ordnungsgemäß funktioniert. Beispielsweise zeichnet der Envoy-Proxy Statistiken über die Anzahl der erfolgreichen TLS-Handshakes auf, die er für einen bestimmten Mesh-Endpunkt ausgehandelt hat. Ermitteln Sie, wie viele erfolgreiche TLS-Handshakes es für einen Mesh-Endpunkt gab, der mit dem folgenden Befehl benannt wurde `my-mesh-endpoint`.

```
curl -s 'http://my-mesh-endpoint.apps.local:9901/stats' | grep ssl.handshake
```

In der folgenden Beispielausgabe wurden drei Handshakes für den Mesh-Endpunkt zurückgegeben, sodass die Kommunikation verschlüsselt ist.

```
listener.0.0.0.0_15000.ssl.handshake: 3
```

Der Envoy-Proxy gibt auch Statistiken aus, wenn die TLS-Verhandlung fehlschlägt. Stellen Sie fest, ob TLS-Fehler für den Mesh-Endpunkt aufgetreten sind.

```
curl -s 'http://my-mesh-endpoint.apps.local:9901/stats' | grep -e "ssl.*\((fail\|error\n)"
```

In der zurückgegebenen Beispielausgabe gab es für mehrere Statistiken keine Fehler, sodass die TLS-Aushandlung erfolgreich war.

```
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
```

Weitere Informationen zu Envoy TLS-Statistiken finden Sie unter [Envoy Listener Statistics](#).

Zertifikatserneuerung

AWS Private CA

Wenn Sie ein Zertifikat mit ACM erneuern, wird das erneuerte Zertifikat innerhalb von 35 Minuten nach Abschluss der Verlängerung automatisch an Ihre verbundenen Proxys verteilt. Wir empfehlen, die verwaltete Verlängerung zu verwenden, um Zertifikate, die sich dem Ende ihrer Gültigkeitsdauer nähern, automatisch zu verlängern. Weitere Informationen finden Sie im Benutzerhandbuch unter [Managed Renewal für die von Amazon ausgestellten Zertifikate von ACM](#). AWS Certificate Manager

Ihr eigenes Zertifikat

Wenn Sie ein Zertifikat aus dem lokalen Dateisystem verwenden, lädt Envoy das Zertifikat nicht automatisch neu, wenn es sich ändert. Sie können den Envoy-Prozess entweder neu starten oder erneut bereitstellen, um ein neues Zertifikat zu laden. Sie können ein neueres Zertifikat auch in einem anderen Dateipfad platzieren und die Konfiguration des virtuellen Knotens oder des Gateways mit diesem Dateipfad aktualisieren.

Konfigurieren Sie Amazon ECS-Workloads für die Verwendung der TLS-Authentifizierung mit AWS App Mesh

Sie können Ihr Mesh für die Verwendung der TLS-Authentifizierung konfigurieren. Stellen Sie sicher, dass die Zertifikate für Envoy-Proxy-Sidecars verfügbar sind, die Sie zu Ihren Workloads hinzufügen. Sie können ein EBS- oder EFS-Volume an Ihre Envoy-Sidecar anhängen oder Zertifikate von Secrets Manager speichern und abrufen. [AWS](#)

- Wenn Sie die dateibasierte Zertifikatsverteilung verwenden, fügen Sie Ihrem Envoy-Sidecar ein EBS- oder EFS-Volume hinzu. Stellen Sie sicher, dass der Pfad zum Zertifikat und zum privaten Schlüssel mit dem Pfad übereinstimmt, der in konfiguriert ist. [AWS App Mesh](#)
- Wenn Sie eine SDS-basierte Distribution verwenden, fügen Sie einen Sidecar hinzu, der die SDS-API von Envoy mit Zugriff auf das Zertifikat implementiert.

Note

SPIRE wird auf Amazon ECS nicht unterstützt.

Konfigurieren Sie Kubernetes-Workloads für die Verwendung der TLS-Authentifizierung mit AWS App Mesh

Sie können den AWS App Mesh Controller für Kubernetes so konfigurieren, dass er die TLS-Authentifizierung für Backends und Listener von virtuellen Knoten- und virtuellen Gateway-Services aktiviert. Stellen Sie sicher, dass die Zertifikate für die Envoy-Proxy-Sidecars verfügbar sind, die Sie zu Ihren Workloads hinzufügen. Ein Beispiel für jeden Verteilungstyp finden Sie in der [exemplarischen Vorgehensweise von Mutual TLS Authentication](#).

- Wenn Sie die dateibasierte Zertifikatsverteilung verwenden, fügen Sie Ihrem Envoy-Sidecar ein EBS- oder EFS-Volume hinzu. Stellen Sie sicher, dass der Pfad zum Zertifikat und zum privaten Schlüssel mit dem im Controller konfigurierten Pfad übereinstimmt. Alternativ können Sie ein Kubernetes-Secret verwenden, das im Dateisystem bereitgestellt wird.
- Wenn Sie eine SDS-basierte Distribution verwenden, sollten Sie einen lokalen SDS-Anbieter für Knoten einrichten, der die SDS-API von Envoy implementiert. Envoy wird es über UDS erreichen. Um die SDS-basierte mTLS-Unterstützung im AppMesh EKS-Controller zu aktivieren, setzen Sie das `enable-sds` Flag auf `true` und geben Sie den UDS-Pfad des lokalen SDS-Anbieters zum Controller über das Flag an. `sds-uds-path` Wenn Sie Helm verwenden, legen Sie Folgendes als Teil Ihrer Controller-Installation fest:

```
--set sds.enabled=true
```

Note

Sie können SPIRE nicht für die Verteilung Ihrer Zertifikate verwenden, wenn Sie Amazon Elastic Kubernetes Service (Amazon EKS) im Fargate-Modus verwenden.

Gegenseitige TLS-Authentifizierung

Die gegenseitige TLS-Authentifizierung (Transport Layer Security) ist eine optionale Komponente von TLS, die eine bidirektionale Peer-Authentifizierung bietet. Die gegenseitige TLS-Authentifizierung bietet eine zusätzliche Sicherheitsebene gegenüber TLS und ermöglicht es Ihren Diensten, den Client zu verifizieren, der die Verbindung herstellt.

Der Client in der Client-Server-Beziehung stellt während der Sitzungs-aushandlung auch ein X.509-Zertifikat bereit. Der Server verwendet dieses Zertifikat, um den Client zu identifizieren und zu authentifizieren. Dieser Prozess hilft zu überprüfen, ob das Zertifikat von einer vertrauenswürdigen Zertifizierungsstelle (CA) ausgestellt wurde und ob es sich bei dem Zertifikat um ein gültiges Zertifikat handelt. Außerdem wird der auf dem Zertifikat angegebene Subject Alternative Name (SAN) verwendet, um den Client zu identifizieren.

Sie können die gegenseitige TLS-Authentifizierung für alle Protokolle aktivieren, die von unterstützt werden AWS App Mesh. Sie sind TCP, HTTP/1.1, HTTP/2, gRPC.

Note

Mit App Mesh können Sie die gegenseitige TLS-Authentifizierung für die Kommunikation zwischen Envoy-Proxys von Ihren Diensten aus konfigurieren. Die Kommunikation zwischen Ihren Anwendungen und Envoy-Proxys ist jedoch unverschlüsselt.

Gegenseitige TLS-Authentifizierungszertifikate

AWS App Mesh unterstützt zwei mögliche Zertifikatsquellen für die gegenseitige TLS-Authentifizierung. Clientzertifikate in einer TLS-Client-Richtlinie und die Servervalidierung in einer Listener-TLS-Konfiguration können bezogen werden von:

- Dateisystem — Zertifikate aus dem lokalen Dateisystem des Envoy-Proxys, der gerade ausgeführt wird. Um Zertifikate an Envoy zu verteilen, müssen Sie Dateipfade für die Zertifikatskette und den privaten Schlüssel für die App Mesh Mesh-API angeben.
- Der Secret Discovery Service (SDS) von Envoy — ring-your-own B-Sidecars, die SDS implementieren und das Senden von Zertifikaten an Envoy ermöglichen. Dazu gehört das SPIFFE Runtime Environment (SPIRE).

Important

App Mesh speichert keine Zertifikate oder privaten Schlüssel, die für die gegenseitige TLS-Authentifizierung verwendet werden. Stattdessen speichert Envoy sie im Speicher.

Mesh-Endpunkte konfigurieren

Konfigurieren Sie die gegenseitige TLS-Authentifizierung für Ihre Mesh-Endpunkte, z. B. virtuelle Knoten oder Gateways. Diese Endpunkte stellen Zertifikate bereit und spezifizieren vertrauenswürdige Behörden.

Dazu müssen Sie X.509-Zertifikate sowohl für den Client als auch für den Server bereitstellen und im Validierungskontext sowohl für die TLS-Terminierung als auch für den TLS-Ursprung explizit Zertifikate für vertrauenswürdige Stellen definieren.

Vertrauen innerhalb eines Netzes

Serverseitige Zertifikate werden in Virtual Node-Listnern (TLS-Terminierung) konfiguriert, und clientseitige Zertifikate werden in Virtual Nodes Service-Backends (TLS-Origination) konfiguriert. Als Alternative zu dieser Konfiguration können Sie eine Standard-Client-Richtlinie für alle Dienst-Back-Ends eines virtuellen Knotens definieren und diese Richtlinie dann bei Bedarf für bestimmte Backends überschreiben. Virtuelle Gateways können nur mit einer Standard-Client-Richtlinie konfiguriert werden, die für alle Back-Ends gilt.

Sie können das Vertrauen zwischen verschiedenen Meshes konfigurieren, indem Sie die gegenseitige TLS-Authentifizierung für eingehenden Datenverkehr auf den virtuellen Gateways für beide Meshes aktivieren.

Vertrauen außerhalb eines Meshs

Geben Sie serverseitige Zertifikate im Virtual Gateway-Listener für die TLS-Terminierung an. Konfigurieren Sie den externen Dienst, der mit Ihrem Virtual Gateway kommuniziert, so, dass er clientseitige Zertifikate vorlegt. Die Zertifikate sollten von einer der gleichen Zertifizierungsstellen (CAs) abgeleitet werden, die die serverseitigen Zertifikate auf dem Virtual Gateway-Listener für die TLS-Erstellung verwenden.

Migrieren Sie Dienste zur gegenseitigen TLS-Authentifizierung

Folgen Sie diesen Richtlinien, um die Konnektivität aufrechtzuerhalten, wenn Sie Ihre bestehenden Dienste in App Mesh auf gegenseitige TLS-Authentifizierung migrieren.

Migration von Diensten, die über Klartext kommunizieren

1. Aktivieren Sie den PERMISSIVE Modus für die TLS-Konfiguration auf dem Serverendpunkt. In diesem Modus kann Klartext-Verkehr eine Verbindung zum Endpunkt herstellen.
2. Konfigurieren Sie die gegenseitige TLS-Authentifizierung auf Ihrem Server und geben Sie das Serverzertifikat, die Vertrauenskette und optional die vertrauenswürdigen SANs an.
3. Stellen Sie sicher, dass die Kommunikation über eine TLS-Verbindung erfolgt.
4. Konfigurieren Sie die gegenseitige TLS-Authentifizierung auf Ihren Clients und geben Sie das Client-Zertifikat, die Vertrauenskette und optional die vertrauenswürdigen SANs an.
5. Aktivieren Sie den STRICT Modus für die TLS-Konfiguration auf dem Server.

Dienste, die über TLS kommunizieren, werden migriert

1. Konfigurieren Sie die gegenseitigen TLS-Einstellungen auf Ihren Clients und geben Sie das Client-Zertifikat und optional die vertrauenswürdigen SANs an. Das Client-Zertifikat wird erst an das Back-End gesendet, nachdem der Backend-Server es angefordert hat.
2. Konfigurieren Sie die gegenseitigen TLS-Einstellungen auf Ihrem Server und geben Sie dabei die Vertrauenskette und optional die vertrauenswürdigen SANs an. Dazu fordert Ihr Server ein Client-Zertifikat an.

Überprüfung der gegenseitigen TLS-Authentifizierung

In der Dokumentation [Transport Layer Security: Verify encryption](#) können Sie nachlesen, wie genau Envoy TLS-bezogene Statistiken ausgibt. Für die gegenseitige TLS-Authentifizierung sollten Sie sich die folgenden Statistiken ansehen:

- `ssl.handshake`
- `ssl.no_certificate`
- `ssl.fail_verify_no_cert`
- `ssl.fail_verify_san`

Die beiden folgenden Statistikbeispiele zeigen zusammen, dass erfolgreiche TLS-Verbindungen, die mit dem virtuellen Knoten enden, alle von einem Client stammen, der ein Zertifikat bereitgestellt hat.

```
listener.0.0.0.0_15000.ssl.handshake: 3
```

```
listener.0.0.0.0_15000.ssl.no_certificate: 0
```

Das nächste Beispiel einer Statistik zeigt, dass die Verbindungen von einem virtuellen Clientknoten (oder Gateway) zu einem virtuellen Backend-Knoten fehlgeschlagen sind. Der alternative Subject Name (SAN), der im Serverzertifikat angegeben ist, stimmt mit keinem der SANs überein, denen der Client vertraut.

```
cluster.cds_egress_my-mesh_my-backend-node_http_9080.ssl.fail_verify_san: 5
```

Exemplarische Vorgehensweisen zur gegenseitigen TLS-Authentifizierung von App Mesh

- [Exemplarische Vorgehensweise zur gegenseitigen TLS-Authentifizierung](#): In dieser exemplarischen Vorgehensweise wird beschrieben, wie Sie mit der App Mesh-CLI eine Farb-App mit gegenseitiger TLS-Authentifizierung erstellen können.
- [Amazon EKS Mutual TLS SDS-basierte Komplettlösung: Diese exemplarische](#) Vorgehensweise zeigt, wie Sie die gegenseitige TLS-SDS-basierte Authentifizierung mit Amazon EKS und SPIFFE Runtime Environment (SPIRE) verwenden können.
- [Dateibasierte Vorgehensweise für Amazon EKS Mutual TLS: Diese exemplarische](#) Vorgehensweise zeigt, wie Sie die gegenseitige dateibasierte TLS-Authentifizierung mit Amazon EKS und SPIFFE Runtime Environment (SPIRE) verwenden können.

Wie AWS App Mesh funktioniert mit IAM

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf Ressourcen sicher zu AWS kontrollieren. IAM-Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um App Mesh Mesh-Ressourcen zu verwenden. IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Themen

- [Zielgruppe](#)
- [Authentifizierung mit Identitäten](#)
- [Verwalten des Zugriffs mit Richtlinien](#)
- [Wie AWS App Mesh funktioniert mit IAM](#)
- [AWS App Mesh Beispiele für identitätsbasierte Richtlinien](#)
- [AWSverwaltete Richtlinien für App Mesh](#)
- [Verwenden von dienstverknüpften Rollen für App Mesh](#)
- [En\) Proxy-Autorisierung](#)
- [Problembehandlung bei AWS App Mesh Identität und Zugriff](#)

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von der Arbeit ab, die Sie in App Mesh ausführen.

Dienstbenutzer — Wenn Sie den App Mesh Mesh-Dienst für Ihre Arbeit verwenden, stellt Ihnen Ihr Administrator die Anmeldeinformationen und Berechtigungen zur Verfügung, die Sie benötigen. Da Sie für Ihre Arbeit mehr App Mesh Mesh-Funktionen verwenden, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Funktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anzufordern müssen. Wenn Sie in App Mesh nicht auf eine Funktion zugreifen können, finden Sie weitere Informationen unter [Problembehandlung bei AWS App Mesh Identität und Zugriff](#).

Service Administrator — Wenn Sie in Ihrem Unternehmen für App Mesh-Ressourcen verantwortlich sind, haben Sie wahrscheinlich vollen Zugriff auf App Mesh. Es ist Ihre Aufgabe, zu bestimmen, auf welche App Mesh Mesh-Funktionen und -Ressourcen Ihre Servicebenutzer zugreifen sollen. Sie müssen dann Anträge an Ihren IAM-Administrator stellen, um die Berechtigungen Ihrer Servicenutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um die Grundkonzepte von IAM nachzuvollziehen. Weitere Informationen darüber, wie Ihr Unternehmen IAM mit App Mesh verwenden kann, finden Sie unter [Wie AWS App Mesh funktioniert mit IAM](#).

IAM-Administrator — Wenn Sie ein IAM-Administrator sind, möchten Sie vielleicht mehr darüber erfahren, wie Sie Richtlinien schreiben können, um den Zugriff auf App Mesh zu verwalten. Beispiele für identitätsbasierte App Mesh Mesh-Richtlinien, die Sie in IAM verwenden können, finden Sie unter [AWS App Mesh Beispiele für identitätsbasierte Richtlinien](#)

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen als IAM-Benutzer authentifiziert (angemeldet AWS) sein oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich AWS als föderierte Identität anmelden, indem Sie Anmeldeinformationen verwenden, die über eine Identitätsquelle bereitgestellt wurden. AWS IAM Identity Center (IAM Identity Center) -Benutzer, die Single Sign-On-Authentifizierung Ihres Unternehmens und Ihre Google- oder Facebook-Anmeldeinformationen sind Beispiele für föderierte Identitäten. Wenn Sie sich als Verbundidentität anmelden, hat der Administrator vorher mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet. Wenn Sie über den Verbund darauf zugreifen AWS, übernehmen Sie indirekt eine Rolle.

Je nachdem, welcher Benutzertyp Sie sind, können Sie sich beim AWS Management Console oder beim AWS Zugangsportal anmelden. Weitere Informationen zur Anmeldung finden Sie AWS unter [So melden Sie sich bei Ihrem an AWS-Konto](#) im AWS-Anmeldung Benutzerhandbuch.

Wenn Sie AWS programmgesteuert darauf zugreifen, AWS stellt es ein Software Development Kit (SDK) und eine Befehlszeilenschnittstelle (CLI) bereit, um Ihre Anfragen mithilfe Ihrer Anmeldeinformationen kryptografisch zu signieren. Wenn Sie keine AWS Tools verwenden, müssen Sie Anfragen selbst signieren. Weitere Informationen zur Verwendung der empfohlenen Methode, um Anfragen selbst zu [signieren, finden Sie im IAM-Benutzerhandbuch unter AWS API-Anfragen](#) signieren.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen angeben. AWS empfiehlt beispielsweise, die Multi-Faktor-Authentifizierung (MFA) zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. Weitere Informationen finden Sie unter [Multi-Faktor-Authentifizierung](#) im AWS IAM Identity Center - Benutzerhandbuch und [Verwenden der Multi-Faktor-Authentifizierung \(MFA\) in AWS](#) im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services Ressourcen im Konto hat. Diese Identität wird als AWS-Konto Root-Benutzer bezeichnet. Sie können darauf zugreifen, indem Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, mit denen Sie das Konto erstellt haben. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen und verwenden Sie diese, um die Aufgaben auszuführen, die nur der Root-Benutzer ausführen kann. Eine vollständige Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter [Aufgaben, die Root-Benutzer-Anmeldeinformationen erfordern](#) im IAM-Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein [IAM-Benutzer](#) ist eine Identität innerhalb von Ihnen AWS-Konto, die über spezifische Berechtigungen für eine einzelne Person oder Anwendung verfügt. Wenn möglich, empfehlen wir, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben. Bei speziellen Anwendungsfällen, die langfristige Anmeldeinformationen mit IAM-Benutzern erfordern, empfehlen wir jedoch, die Zugriffsschlüssel zu rotieren. Weitere Informationen finden Sie unter [Regelmäßiges](#)

[Rotieren von Zugriffsschlüsseln für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern](#) im IAM-Benutzerhandbuch.

Eine [IAM-Gruppe](#) ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht als Gruppe anmelden. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzer gleichzeitig angeben. Gruppen vereinfachen die Verwaltung von Berechtigungen, wenn es zahlreiche Benutzer gibt. Sie könnten beispielsweise einer Gruppe mit dem Namen IAMAdmins Berechtigungen zum Verwalten von IAM-Ressourcen erteilen.

Benutzer unterscheiden sich von Rollen. Ein Benutzer ist einer einzigen Person oder Anwendung eindeutig zugeordnet. Eine Rolle kann von allen Personen angenommen werden, die sie benötigen. Benutzer besitzen dauerhafte Anmeldeinformationen. Rollen stellen temporäre Anmeldeinformationen bereit. Weitere Informationen finden Sie unter [Erstellen eines IAM-Benutzers \(anstatt einer Rolle\)](#) im IAM-Benutzerhandbuch.

IAM-Rollen

Eine [IAM-Rolle](#) ist eine Identität innerhalb Ihres Unternehmens AWS-Konto , die über bestimmte Berechtigungen verfügt. Sie ist einem IAM-Benutzer vergleichbar, ist aber nicht mit einer bestimmten Person verknüpft. Sie können vorübergehend eine IAM-Rolle in der übernehmen, AWS Management Console indem Sie die Rollen [wechseln](#). Sie können eine Rolle übernehmen, indem Sie eine AWS CLI oder AWS API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter [Verwenden von IAM-Rollen](#) im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- **Verbundbenutzerzugriff** – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert, so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter [Erstellen von Rollen für externe Identitätsanbieter](#) im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter [Berechtigungssätze](#) im AWS IAM Identity Center -Benutzerhandbuch.
- **Temporäre IAM-Benutzerberechtigungen** – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.

- **Kontoübergreifender Zugriff** – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. Bei einigen können Sie AWS-Services jedoch eine Richtlinie direkt an eine Ressource anhängen (anstatt eine Rolle als Proxy zu verwenden). Informationen zu den Unterschieden zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter [So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien](#) im IAM-Benutzerhandbuch.
- **Serviceübergreifender Zugriff** — Einige AWS-Services verwenden Funktionen in anderen AWS-Services. Wenn Sie beispielsweise einen Aufruf in einem Service tätigen, führt dieser Service häufig Anwendungen in Amazon-EC2 aus oder speichert Objekte in Amazon-S3. Ein Dienst kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.
- **Forward Access Sessions (FAS)** — Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Diensten könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anfrage, Anfragen an AWS-Service nachgelagerte Dienste zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter [Zugriffssitzungen weiterleiten](#).
- **Servicerolle** – Eine Servicerolle ist eine [IAM-Rolle](#), die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter [Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service](#) im IAM-Benutzerhandbuch.
- **Dienstbezogene Rolle** — Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Servicebezogene Rollen erscheinen in Ihrem Dienst AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.
- **Auf Amazon EC2 ausgeführte Anwendungen** — Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2-Instance ausgeführt werden und API-Anfragen stellen AWS CLI . AWS Das ist eher zu empfehlen, als Zugriffsschlüssel innerhalb der EC2-Instance zu speichern. Um einer EC2-Instance eine AWS

Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie ein Instance-Profil, das an die Instance angehängt ist. Ein Instance-Profil enthält die Rolle und ermöglicht, dass Programme, die in der EC2-Instance ausgeführt werden, temporäre Anmeldeinformationen erhalten. Weitere Informationen finden Sie unter [Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon-EC2-Instances ausgeführt werden](#) im IAM-Benutzerhandbuch.

Informationen dazu, wann Sie IAM-Rollen oder IAM-Benutzer verwenden sollten, finden Sie unter [Erstellen einer IAM-Rolle \(anstatt eines Benutzers\)](#) im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie ist ein Objekt, AWS das, wenn es einer Identität oder Ressource zugeordnet ist, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Prinzipal (Benutzer, Root-Benutzer oder Rollensitzung) eine Anfrage stellt. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu Struktur und Inhalten von JSON-Richtliniendokumenten finden Sie unter [Übersicht über JSON-Richtlinien](#) im IAM-Benutzerhandbuch.

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das bedeutet, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

IAM-Richtlinien definieren Berechtigungen für eine Aktion unabhängig von der Methode, die Sie zur Ausführung der Aktion verwenden. Angenommen, es gibt eine Richtlinie, die Berechtigungen für die `iam:GetRole`-Aktion erteilt. Ein Benutzer mit dieser Richtlinie kann Rolleninformationen von der AWS Management Console AWS CLI, der oder der AWS API abrufen.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern,

welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter [Erstellen von IAM-Richtlinien](#) im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können weiter als Inline-Richtlinien oder verwaltete Richtlinien kategorisiert werden. Inline-Richtlinien sind direkt in einen einzelnen Benutzer, eine einzelne Gruppe oder eine einzelne Rolle eingebettet. Verwaltete Richtlinien sind eigenständige Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem System zuordnen können AWS-Konto. Zu den verwalteten Richtlinien gehören AWS verwaltete Richtlinien und vom Kunden verwaltete Richtlinien. Informationen dazu, wie Sie zwischen einer verwalteten Richtlinie und einer eingebundenen Richtlinie wählen, finden Sie unter [Auswahl zwischen verwalteten und eingebundenen Richtlinien](#) im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie [einen Prinzipal angeben](#). Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Zugriffssteuerungslisten (ACLs)

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3 und Amazon VPC sind Beispiele für Services, die ACLs unterstützen. AWS WAF Weitere Informationen“ zu ACLs finden Sie unter [Zugriffskontrollliste \(ACL\) – Übersicht](#) (Access Control List) im Amazon-Simple-Storage-Service-Entwicklerhandbuch.

Weitere Richtlinientypen

AWS unterstützt zusätzliche, weniger verbreitete Richtlinientypen. Diese Richtlinientypen können die maximalen Berechtigungen festlegen, die Ihnen von den häufiger verwendeten Richtlinientypen erteilt werden können.

- **Berechtigungsgrenzen** – Eine Berechtigungsgrenze ist ein erweitertes Feature, mit der Sie die maximalen Berechtigungen festlegen können, die eine identitätsbasierte Richtlinie einer IAM-Entität (IAM-Benutzer oder -Rolle) erteilen kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die daraus resultierenden Berechtigungen sind der Schnittpunkt der identitätsbasierten Richtlinien einer Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter [Berechtigungsgrenzen für IAM-Entitäten](#) im IAM-Benutzerhandbuch.
- **Service Control Policies (SCPs)** — SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in festlegen. AWS Organizations AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung mehrerer Objekte AWS-Konten , die Ihrem Unternehmen gehören. Wenn Sie innerhalb einer Organisation alle Features aktivieren, können Sie Service-Kontrollrichtlinien (SCPs) auf alle oder einzelne Ihrer Konten anwenden. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen Entitäten. Root-Benutzer des AWS-Kontos Weitere Informationen zu Organizations und SCPs finden Sie unter [Funktionsweise von SCPs](#) im AWS Organizations -Benutzerhandbuch.
- **Sitzungsrichtlinien** – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter [Sitzungsrichtlinien](#) im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn mehrere auf eine Anforderung mehrere Richtlinientypen angewendet werden können, sind die entsprechenden Berechtigungen komplizierter. Informationen darüber, wie AWS bestimmt wird,

ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie im IAM-Benutzerhandbuch unter [Bewertungslogik für Richtlinien](#).

Wie AWS App Mesh funktioniert mit IAM

Bevor Sie IAM verwenden, um den Zugriff auf App Mesh zu verwalten, sollten Sie wissen, welche IAM-Funktionen für die Verwendung mit App Mesh verfügbar sind. Einen allgemeinen Überblick darüber, wie App Mesh und andere AWS Dienste mit IAM funktionieren, finden Sie unter [AWS Services That Work with IAM](#) im IAM-Benutzerhandbuch.

Themen

- [Identitätsbasierte Richtlinien für App Mesh](#)
- [Ressourcenbasierte Richtlinien für App Mesh](#)
- [Autorisierung basierend auf App Mesh-Tags](#)
- [App Mesh IAM-Rollen](#)

Identitätsbasierte Richtlinien für App Mesh

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. App Mesh unterstützt bestimmte Aktionen, Ressourcen und Bedingungsschlüssel. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der [IAM-Referenz für JSON-Richtlinienelemente](#) im IAM-Benutzerhandbuch.

Aktionen

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Richtlinienaktionen haben normalerweise denselben Namen wie der zugehörige AWS API-Vorgang. Es gibt einige Ausnahmen, z. B. Aktionen, die nur mit Genehmigung durchgeführt werden können und für die es keine passende API-Operation gibt. Es gibt auch einige Operationen, die mehrere Aktionen in einer Richtlinie erfordern. Diese zusätzlichen Aktionen werden als abhängige Aktionen bezeichnet.

Schließen Sie Aktionen in eine Richtlinie ein, um Berechtigungen zur Durchführung der zugeordneten Operation zu erteilen.

Richtlinienaktionen in App Mesh verwenden vor der Aktion das folgende Präfix: `appmesh:`.

Um beispielsweise jemandem die Erlaubnis zu erteilen, Meshes in einem Konto mit der `appmesh:ListMeshes` API-Operation aufzulisten, nehmen Sie die `appmesh:ListMeshes` Aktion in seine Richtlinie auf. Richtlinienanweisungen müssen entweder ein `Action` oder ein `NotAction`-Element enthalten.

Um mehrere -Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie folgendermaßen durch Kommas.

```
"Action": [
  "appmesh:ListMeshes",
  "appmesh:ListVirtualNodes"
]
```

Sie können auch Platzhalter (*) verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort `Describe` beginnen, einschließlich der folgenden Aktion:

```
"Action": "appmesh:Describe*"
```

Eine Liste der App Mesh Mesh-Aktionen finden Sie unter [Definierte Aktionen von AWS App Mesh](#) im IAM-Benutzerhandbuch.

Ressourcen

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das bedeutet die Festlegung, welcher Prinzipal Aktionen für welche Ressourcen unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Anweisungen müssen entweder ein `Resource` oder ein `NotResource`-Element enthalten. Als bewährte Methode geben Sie eine Ressource mit dem zugehörigen [Amazon-Ressourcennamen \(ARN\)](#) an. Sie können dies für Aktionen tun, die einen bestimmten Ressourcentyp unterstützen, der als Berechtigungen auf Ressourcenebene bezeichnet wird.

Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, z. B. Auflistungsoperationen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Die App Mesh mesh Mesh-Ressource hat den folgenden ARN.

```
arn:${Partition}:appmesh:${Region}:${Account}:mesh/${MeshName}
```

Weitere Informationen zum Format von ARNs finden Sie unter [Amazon Resource Names \(ARNs\) und AWS Service Namespaces](#).

Verwenden Sie beispielsweise den folgenden ARN, um in Ihrer Anweisung die Mesh-benannten *Apps* in der Region mit dem *Regionalcode* anzugeben.

```
arn:aws:appmesh:Region-code:111122223333:mesh/apps
```

Um alle Instances anzugeben, die zu einem bestimmten Konto gehören, verwenden Sie den Platzhalter (*).

```
"Resource": "arn:aws:appmesh:Region-code:111122223333:mesh/*"
```

Einige App Mesh Mesh-Aktionen, z. B. zum Erstellen von Ressourcen, können nicht für eine bestimmte Ressource ausgeführt werden. In diesen Fällen müssen Sie den Platzhalter (*) verwenden.

```
"Resource": "*"
```

Viele App Mesh Mesh-API-Aktionen beinhalten mehrere Ressourcen. CreateRouteErstellt beispielsweise eine Route mit einem virtuellen Knotenziel, sodass ein IAM-Benutzer über Berechtigungen zur Verwendung der Route und des virtuellen Knotens verfügen muss. Um mehrere Ressourcen in einer einzigen Anweisung anzugeben, trennen Sie die ARNs durch Kommata voneinander.

```
"Resource": [  
  "arn:aws:appmesh:Region-code:111122223333:mesh/apps/virtualRouter/serviceB/route/  
  *",  
  "arn:aws:appmesh:Region-code:111122223333:mesh/apps/virtualNode/serviceB"
```

]

Eine Liste der App Mesh Mesh-Ressourcentypen und ihrer ARNs finden Sie unter [Resources Defined by AWS App Mesh](#) im IAM-Benutzerhandbuch. Informationen zu den Aktionen, mit denen Sie den ARN einzelner Ressourcen angeben können, finden Sie unter [Von AWS App Mesh definierte Aktionen](#).

Bedingungsschlüssel

App Mesh unterstützt die Verwendung einiger globaler Bedingungsschlüssel. Eine Liste aller globalen AWS -Bedingungsschlüssel finden Sie unter [Globale AWS -Bedingungskontextschlüssel](#) im IAM-Benutzerhandbuch. Eine Liste der globalen Bedingungsschlüssel, die App Mesh unterstützt, finden Sie unter [Bedingungsschlüssel für AWS App Mesh](#) im IAM-Benutzerhandbuch. Informationen zu den Aktionen und Ressourcen, die Sie mit einem Bedingungsschlüssel verwenden können, finden Sie unter [Aktionen definiert von AWS App Mesh](#).

Beispiele

Beispiele für identitätsbasierte App Mesh Mesh-Richtlinien finden Sie unter [AWS App Mesh Beispiele für identitätsbasierte Richtlinien](#)

Ressourcenbasierte Richtlinien für App Mesh

App Mesh unterstützt keine ressourcenbasierten Richtlinien. Wenn Sie jedoch den Dienst AWS Resource Access Manager (AWS RAM) verwenden, um ein Mesh für mehrere AWS Dienste gemeinsam zu nutzen, wird vom Dienst eine ressourcenbasierte Richtlinie auf Ihr Mesh angewendet. AWS RAM Weitere Informationen finden Sie unter [Erteilen von Berechtigungen für ein Mesh](#).

Autorisierung basierend auf App Mesh-Tags

Sie können Tags an App Mesh-Ressourcen anhängen oder Tags in einer Anfrage an App Mesh übergeben. Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungelement einer [Richtlinie Tag-Informationen](#) an, indem Sie die Schlüssel `appmesh:ResourceTag/key-name`, `aws:RequestTag/key-name`, oder Bedingung `aws:TagKeys` verwenden. Weitere Informationen zum Taggen von App Mesh Mesh-Ressourcen finden Sie unter [Tagging AWS Resources](#).

Ein Beispiel für eine identitätsbasierte Richtlinie zur Einschränkung des Zugriffs auf eine Ressource auf der Grundlage der Markierungen dieser Ressource finden Sie unter [App Mesh-Meshes mit eingeschränkten Tags erstellen](#).

App Mesh IAM-Rollen

Eine [IAM-Rolle](#) ist eine Entität in Ihrem AWS Konto, die über bestimmte Berechtigungen verfügt.

Temporäre Anmeldeinformationen mit App Mesh verwenden

Sie können temporäre Anmeldeinformationen verwenden, um sich über einen Verbund anzumelden, eine IAM-Rolle anzunehmen oder eine kontenübergreifende Rolle anzunehmen. Sie erhalten temporäre Sicherheitsanmeldedaten, indem Sie AWS STS API-Operationen wie [AssumeRole](#) oder aufrufen [GetFederationToken](#).

App Mesh unterstützt die Verwendung temporärer Anmeldeinformationen.

Service-verknüpfte Rollen

[Mit Diensten verknüpfte Rollen](#) ermöglichen es AWS Diensten, auf Ressourcen in anderen Diensten zuzugreifen, um eine Aktion in Ihrem Namen auszuführen. Serviceverknüpfte Rollen werden in Ihrem IAM-Konto angezeigt und gehören zum Service. Ein IAM-Administrator kann die Berechtigungen für serviceverknüpfte Rollen anzeigen, aber nicht bearbeiten.

App Mesh unterstützt dienstverknüpfte Rollen. Einzelheiten zum Erstellen oder Verwalten von dienstverknüpften App Mesh Mesh-Rollen finden Sie unter [Verwenden von dienstverknüpften Rollen für App Mesh](#).

Servicerollen

Dieses Feature ermöglicht einem Service das Annehmen einer [Servicerolle](#) in Ihrem Namen. Diese Rolle gewährt dem Service Zugriff auf Ressourcen in anderen Diensten, um eine Aktion in Ihrem Namen auszuführen. Servicerollen werden in Ihrem IAM-Konto angezeigt und gehören zum Konto. Dies bedeutet, dass ein IAM-Administrator die Berechtigungen für diese Rolle ändern kann. Dies kann jedoch die Funktionalität des Dienstes beeinträchtigen.

App Mesh unterstützt keine Servicerollen.

AWS App Mesh Beispiele für identitätsbasierte Richtlinien

Standardmäßig sind IAM-Benutzer und -Rollen nicht berechtigt, App Mesh Mesh-Ressourcen zu erstellen oder zu ändern. Sie können auch keine Aufgaben mit der AWS Management Console AWS CLI, oder AWS API ausführen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern

und Rollen die Berechtigung zum Ausführen bestimmter API-Operationen für die angegebenen Ressourcen gewähren, die diese benötigen. Der Administrator muss diese Richtlinien anschließend den IAM-Benutzern oder -Gruppen anfügen, die diese Berechtigungen benötigen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter [Erstellen von Richtlinien auf der JSON-Registrierkarte](#) im IAM-Benutzerhandbuch.

Themen

- [Bewährte Methoden für Richtlinien](#)
- [Die App Mesh Mesh-Konsole verwenden](#)
- [Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer](#)
- [Erstellen Sie ein Netz](#)
- [Listet alle Meshes auf und beschreibt sie](#)
- [App Mesh-Meshes mit eingeschränkten Tags erstellen](#)

Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand App Mesh-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder sie löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um Ihren Benutzern und Workloads zunächst Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter [AWS -verwaltete Richtlinien](#) oder [AWS -verwaltete Richtlinien für Auftrags-Funktionen](#) im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum

Anwenden von Berechtigungen finden Sie unter [Richtlinien und Berechtigungen in IAM](#) im IAM-Benutzerhandbuch.

- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. AWS CloudFormation B. Weitere Informationen finden Sie unter [IAM-JSON-Richtlinienelemente: Bedingung](#) im IAM-Benutzerhandbuch.
- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter [Richtlinienvvalidierung zum IAM Access Analyzer](#) im IAM-Benutzerhandbuch.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter [Konfigurieren eines MFA-geschützten API-Zugriffs](#) im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter [Bewährte Methoden für die Sicherheit in IAM](#) im IAM-Benutzerhandbuch.

Die App Mesh Mesh-Konsole verwenden

Um auf die AWS App Mesh Konsole zugreifen zu können, benötigen Sie ein Mindestmaß an Berechtigungen. Diese Berechtigungen müssen es Ihnen ermöglichen, Details zu den App Mesh Mesh-Ressourcen in Ihrem AWS Konto aufzulisten und anzuzeigen. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (IAM-Benutzer oder -Rollen) mit dieser Richtlinie. Sie können die [AWSAppMeshReadOnly](#) verwaltete Richtlinie an Benutzer anhängen. Weitere Informationen finden Sie unter [Hinzufügen von Berechtigungen zu einem Benutzer](#) im IAM-Benutzerhandbuch.

Sie müssen Benutzern, die nur die API AWS CLI oder die AWS API aufrufen, keine Mindestberechtigungen für die Konsole gewähren. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die den API-Operation entsprechen, die Sie ausführen möchten.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der API AWS CLI oder AWS .

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsWithUser",
        "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": "*"
    }
  ]
}
```

```
}
```

Erstellen Sie ein Netz

Dieses Beispiel zeigt, wie Sie eine Richtlinie erstellen können, die es einem Benutzer ermöglicht, ein Mesh für ein Konto in einer beliebigen Region zu erstellen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "appmesh:CreateMesh",
      "Resource": "arn:aws:appmesh:*:123456789012:CreateMesh"
    }
  ]
}
```

Listet alle Meshes auf und beschreibt sie

Dieses Beispiel zeigt, wie Sie eine Richtlinie erstellen können, die einem Benutzer nur Lesezugriff gewährt, um alle Meshes aufzulisten und zu beschreiben.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "appmesh:DescribeMesh",
        "appmesh:ListMeshes"
      ],
      "Resource": "*"
    }
  ]
}
```

App Mesh-Meshes mit eingeschränkten Tags erstellen

Sie können Tags in Ihren IAM-Richtlinien verwenden, um zu steuern, welche Tags in der IAM-Anfrage übergeben werden können. Sie können angeben, welche Tag-Schlüssel-Wert-Paare einem

IAM-Benutzer oder einer IAM-Rolle hinzugefügt, geändert oder daraus entfernt werden können.

Dieses Beispiel zeigt, wie Sie eine Richtlinie erstellen könnten, die das Erstellen eines Meshs ermöglicht, aber nur, wenn das Mesh mit einem Tag namens `TeamName` und dem Wert `BookSteam` erstellt wird.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "appmesh:CreateMesh",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/teamName": "booksTeam"
        }
      }
    }
  ]
}
```

Sie können diese Richtlinie den IAM-Benutzern in Ihrem Konto anfügen. Wenn ein Benutzer versucht, ein Netz zu erstellen, muss das Netz ein Tag mit dem Namen `teamName` und dem Wert `booksTeam` enthalten. Wenn das Netz dieses Tag und diesen Wert nicht enthält, schlägt der Versuch, das Netz zu erstellen, fehl. Weitere Informationen finden Sie unter [IAM-JSON-Richtlinienelemente: Bedingung](#) im IAM-Benutzerhandbuch.

AWSverwaltete Richtlinien für App Mesh

Eine von AWS verwaltete Richtlinie ist eine eigenständige Richtlinie, die von AWS erstellt und verwaltet wird. Von AWS verwaltete Richtlinien stellen Berechtigungen für viele häufige Anwendungsfälle bereit, damit Sie beginnen können, Benutzern, Gruppen und Rollen Berechtigungen zuzuweisen.

Beachten Sie, dass AWS-verwaltete Richtlinien möglicherweise nicht die geringsten Berechtigungen für Ihre spezifischen Anwendungsfälle gewähren, da sie für alle AWS-Kunden verfügbar sind. Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie [kundenverwaltete Richtlinien](#) definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind.

Die Berechtigungen, die in den von AWS verwalteten Richtlinien definiert sind, können nicht geändert werden. Wenn AWS Berechtigungen aktualisiert, die in einer von AWS verwalteten Richtlinie definiert

werden, wirkt sich das Update auf alle Prinzipalidentitäten (Benutzer, Gruppen und Rollen) aus, denen die Richtlinie zugeordnet ist. AWS aktualisiert am wahrscheinlichsten eine von AWS verwaltete Richtlinie, wenn ein neuer AWS-Service gestartet wird oder neue API-Operationen für bestehende Services verfügbar werden.

Weitere Informationen finden Sie unter [Von AWS verwaltete Richtlinien](#) im IAM-Benutzerhandbuch.

AWS verwaltete Richtlinie: AWSAppMeshServiceRolePolicy

Sie können `AWSAppMeshServiceRolePolicy` an Ihre IAM-Entitäten anhängen. Ermöglicht den Zugriff auf AWS Dienste und Ressourcen, die von verwendet oder verwaltet werden AWS App Mesh.

Die Berechtigungen für diese Richtlinie finden Sie [AWSAppMeshServiceRolePolicy](#) in der Referenz zu AWS verwalteten Richtlinien.

Informationen zu den Berechtigungsdetails für finden Sie unter [Dienstbezogene Rollenberechtigungen für App Mesh](#). `AWSAppMeshServiceRolePolicy`

AWS verwaltete Richtlinie: AWSAppMeshEnvoyAccess

Sie können `AWSAppMeshEnvoyAccess` an Ihre IAM-Entitäten anhängen. App Mesh Envoy-Richtlinie für den Zugriff auf die Konfiguration virtueller Knoten.

Die Berechtigungen für diese Richtlinie finden Sie [AWSAppMeshEnvoyAccess](#) in der Referenz zu AWS verwalteten Richtlinien.

AWS verwaltete Richtlinie: AWSAppMeshFullAccess

Sie können `AWSAppMeshFullAccess` an Ihre IAM-Entitäten anhängen. Bietet vollen Zugriff auf die AWS App Mesh APIs und AWS Management Console.

Die Berechtigungen für diese Richtlinie finden Sie [AWSAppMeshFullAccess](#) in der Referenz zu AWS verwalteten Richtlinien.

AWS verwaltete Richtlinie: AWSAppMeshPreviewEnvoyAccess

Sie können `AWSAppMeshPreviewEnvoyAccess` an Ihre IAM-Entitäten anhängen. App Mesh Preview Envoy-Richtlinie für den Zugriff auf die Konfiguration virtueller Knoten.

Die Berechtigungen für diese Richtlinie finden Sie [AWSAppMeshPreviewEnvoyAccess](#) in der Referenz zu AWS verwalteten Richtlinien.

AWS verwaltete Richtlinie: AWSAppMeshPreviewServiceRolePolicy

Sie können `AWSAppMeshPreviewServiceRolePolicy` an Ihre IAM-Entitäten anhängen. Ermöglicht den Zugriff auf AWS Dienste und Ressourcen, die von verwendet oder verwaltet werden AWS App Mesh.

Die Berechtigungen für diese Richtlinie finden Sie [AWSAppMeshPreviewServiceRolePolicy](#) in der Referenz zu AWS verwalteten Richtlinien.

AWS verwaltete Richtlinie: AWSAppMeshReadOnly

Sie können `AWSAppMeshReadOnly` an Ihre IAM-Entitäten anhängen. Bietet schreibgeschützten Zugriff auf die AWS App Mesh APIs und. AWS Management Console

Die Berechtigungen für diese Richtlinie finden Sie [AWSAppMeshReadOnly](#) in der Referenz zu AWS verwalteten Richtlinien.

AWS App Mesh-Aktualisierungen für AWS verwaltete Richtlinien

Anzeigen von Details zu Aktualisierungen für AWS-verwaltete Richtlinien für AWS App Mesh, seit dieser Dienst mit der Verfolgung dieser Änderungen begonnen hat. Um automatische Benachrichtigungen über Änderungen an dieser Seite zu erhalten, abonnieren Sie den RSS-Feed auf der Seite AWS App Mesh-Dokumentverlauf.

Änderung	Beschreibung	Datum
AWSAppMeshServiceRolePolicy , AWSServiceRoleForAppMesh — Aktualisierte Richtlinie.	Aktualisiert <code>AWSServiceRoleForAppMesh</code> und <code>AWSAppMeshServiceRolePolicy</code> ermöglicht den Zugriff auf die AWS Cloud Map <code>DiscoverInstancesRevision</code> API.	12. Oktober 2023

Um Zugriff zu gewähren, fügen Sie Ihren Benutzern, Gruppen oder Rollen Berechtigungen hinzu:

- Benutzer und Gruppen in AWS IAM Identity Center:

Erstellen Sie einen Berechtigungssatz. Befolgen Sie die Anweisungen unter [Erstellen eines Berechtigungssatzes](#) im AWS IAM Identity Center-Benutzerhandbuch.

- Benutzer, die in IAM über einen Identitätsanbieter verwaltet werden:

Erstellen Sie eine Rolle für den Identitätsverbund. Befolgen Sie die Anweisungen unter [Erstellen einer Rolle für einen externen Identitätsanbieter \(Verbund\)](#) im IAM-Benutzerhandbuch.

- IAM-Benutzer:

- Erstellen Sie eine Rolle, die Ihr Benutzer annehmen kann. Folgen Sie den Anweisungen unter [Erstellen einer Rolle für einen IAM-Benutzer](#) im IAM-Benutzerhandbuch.
- (Nicht empfohlen) Weisen Sie einem Benutzer eine Richtlinie direkt zu oder fügen Sie einen Benutzer zu einer Benutzergruppe hinzu. Befolgen Sie die Anweisungen unter [Hinzufügen von Berechtigungen zu einem Benutzer \(Konsole\)](#) im IAM-Benutzerhandbuch.

Verwenden von dienstverknüpften Rollen für App Mesh

AWS App Mesh verwendet [serviceverknüpfte Rollen](#) von AWS Identity and Access Management (IAM). Eine serviceverknüpfte Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit App Mesh verknüpft ist. Dienstbezogene Rollen sind von App Mesh vordefiniert und enthalten alle Berechtigungen, die der Dienst benötigt, um andere AWS Dienste in Ihrem Namen aufzurufen.

Eine dienstverknüpfte Rolle erleichtert die Einrichtung von App Mesh, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. App Mesh definiert die Berechtigungen seiner dienstbezogenen Rollen, und sofern nicht anders definiert, kann nur App Mesh seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauensrichtlinie und die Berechtigungsrichtlinie, und diese Berechtigungsrichtlinie kann keiner anderen juristischen Stelle von IAM zugeordnet werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem die zugehörigen Ressourcen gelöscht wurden. Dies schützt Ihre App Mesh Mesh-Ressourcen, da Sie die Zugriffsberechtigung für die Ressourcen nicht versehentlich entfernen können.

Informationen zu anderen Services, die servicegebundene Rollen unterstützen, finden Sie unter [AWS-Services, die mit IAM funktionieren](#). Suchen Sie nach den Services, für die Ja in der Spalte Servicegebundene Rolle angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer servicegebundenen Rolle für diesen Service anzuzeigen.

Dienstbezogene Rollenberechtigungen für App Mesh

App Mesh verwendet die dienstverknüpfte Rolle mit dem Namen `AWSServiceRoleForAppMesh`— Die Rolle ermöglicht es App Mesh, AWS Dienste in Ihrem Namen aufzurufen.

Die `AWSServiceRoleForAppMesh` dienstverknüpfte Rolle vertraut darauf, dass der `appmesh.amazonaws.com` Dienst die Rolle übernimmt.

Einzelheiten zur Genehmigung

- `servicediscovery:DiscoverInstances`- Ermöglicht App Mesh, Aktionen für alle AWS Ressourcen abzuschließen.
- `servicediscovery:DiscoverInstancesRevision`- Ermöglicht App Mesh, Aktionen für alle AWS Ressourcen abzuschließen.

AWSServiceRoleForAppMesh

Diese Richtlinie umfasst die folgenden Berechtigungen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CloudMapServiceDiscovery",
      "Effect": "Allow",
      "Action": [
        "servicediscovery:DiscoverInstances",
        "servicediscovery:DiscoverInstancesRevision"
      ],
      "Resource": "*"
    },
    {
      "Sid": "ACMCertificateVerification",
      "Effect": "Allow",
      "Action": [
        "acm:DescribeCertificate"
      ],
      "Resource": "*"
    }
  ]
}
```

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. Benutzer, Gruppe oder Rolle) eine servicegebundene Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter [serviceverknüpfte Rollenberechtigungen](#) im IAM-Benutzerhandbuch.

Eine serviceverknüpfte Rolle für App Mesh erstellen

Wenn Sie nach dem 5. Juni 2019 in der, der oder der AWS Management Console AWS API ein Mesh erstellt haben AWS CLI, hat App Mesh die serviceverknüpfte Rolle für Sie erstellt. Damit die serviceverknüpfte Rolle für Sie erstellt wurde, muss dem IAM-Konto, mit dem Sie das Mesh erstellt haben, die [AWSAppMeshFullAccess](#) IAM-Richtlinie oder eine Richtlinie angehängt worden sein, die die Berechtigung enthält. `iam:CreateServiceLinkedRole` Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie ein Mesh erstellen, erstellt App Mesh die serviceverknüpfte Rolle erneut für Sie. Wenn Ihr Konto nur Meshes enthält, die vor dem 5. Juni 2019 erstellt wurden, und Sie die serviceverknüpfte Rolle mit diesen Meshes verwenden möchten, können Sie die Rolle mit der IAM-Konsole erstellen.

Sie können die IAM-Konsole verwenden, um eine serviceverknüpfte Rolle mit dem Anwendungsfall App Mesh zu erstellen. Erstellen Sie in der AWS CLI oder der AWS-API eine servicegebundene Rolle mit dem Servicenamen `appmesh.amazonaws.com`. Weitere Informationen finden Sie unter [Erstellen einer servicegebundenen Rolle](#) im IAM-Leitfaden. Wenn Sie diese servicegebundene Rolle löschen, können Sie mit demselben Verfahren die Rolle erneut erstellen.

Bearbeiten einer serviceverknüpften Rolle für App Mesh

App Mesh erlaubt es Ihnen nicht, die `AWSServiceRoleForAppMesh` dienstverknüpfte Rolle zu bearbeiten. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollename nach dem Erstellen einer serviceverknüpften Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter [Bearbeiten einer serviceverknüpften Rolle](#) im IAM-Benutzerhandbuch.

Löschen einer dienstverknüpften Rolle für App Mesh

Wenn Sie ein Feature oder einen Service, die bzw. der eine servicegebundene Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch die Ressourcen für Ihre serviceverknüpften Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

Note

Wenn der App Mesh Mesh-Dienst die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt das Löschen möglicherweise fehl. Wenn dies passiert, warten Sie einige Minuten und versuchen Sie es erneut.

Um App Mesh Mesh-Ressourcen zu löschen, die verwendet werden von `AWSServiceRoleForAppMesh`

1. Löscht alle [Routen](#), die für alle Router im Mesh definiert sind.
2. Löschen Sie alle [virtuellen Router im Mesh](#).
3. Löschen Sie alle [virtuellen Dienste](#) im Mesh.
4. Löschen Sie alle [virtuellen Knoten](#) im Mesh.
5. Löschen Sie das [Netz](#).

Führen Sie die vorherigen Schritte für alle Meshes in Ihrem Konto aus.

So löschen Sie die servicegebundene Rolle mit IAM

Verwenden Sie die IAM-Konsole, die oder die AWS API/AWS CLI, um die `AWSServiceRoleForAppMesh` serviceverknüpfte Rolle zu löschen. Weitere Informationen finden Sie unter [Löschen einer serviceverknüpften Rolle](#) im IAM-Benutzerhandbuch.

Unterstützte Regionen für dienstverknüpfte App Mesh Mesh-Rollen

App Mesh unterstützt die Verwendung von dienstbezogenen Rollen in allen Regionen, in denen der Dienst verfügbar ist. Weitere Informationen finden Sie unter [App Mesh Mesh-Endpunkte und Kontingente](#).

En) Proxy-Autorisierung

Die Proxy-Autorisierung autorisiert den [Envoy-Proxy](#), der innerhalb einer Amazon ECS-Aufgabe, in einem Kubernetes-Pod, der auf Amazon EKS läuft, oder der auf einer Amazon EC2 EC2-Instance ausgeführt wird, dazu, die Konfiguration eines oder mehrerer Mesh-Endpunkte aus dem App Mesh Envoy Management Service zu lesen. Für Kundenkonten, bei denen Envoys bereits vor dem 26.04.2021 mit ihrem App Mesh Mesh-Endpunkt verbunden ist, ist eine Proxy-Autorisierung für virtuelle Knoten, die [Transport Layer Security \(TLS\)](#) verwenden, und für virtuelle Gateways (mit

oder ohne TLS) erforderlich. Für Kundenkonten, die Envoys nach dem 26.04.2021 mit ihrem App Mesh Mesh-Endpunkt verbinden möchten, ist eine Proxy-Autorisierung für alle App Mesh Mesh-Funktionen erforderlich. Es wird empfohlen, dass alle Kundenkonten die Proxy-Autorisierung für alle virtuellen Knoten aktivieren, auch wenn sie kein TLS verwenden, um eine sichere und konsistente Erfahrung mit IAM für die Autorisierung für bestimmte Ressourcen zu haben. Die Proxy-Autorisierung erfordert, dass die `appmesh:StreamAggregatedResources` Berechtigung in einer IAM-Richtlinie angegeben ist. Die Richtlinie muss an eine IAM-Rolle angehängt sein, und diese IAM-Rolle muss an die Rechenressource angehängt werden, auf der Sie den Proxy hosten.

Erstellen einer IAM-Richtlinie

Wenn Sie möchten, dass alle Mesh-Endpunkte in einem Service-Mesh die Konfiguration für alle Mesh-Endpunkte lesen können, fahren Sie mit fort [Erstellen einer IAM-Rolle](#). Wenn Sie die Mesh-Endpunkte einschränken möchten, aus denen die Konfiguration von einzelnen Mesh-Endpunkten gelesen werden kann, müssen Sie eine oder mehrere IAM-Richtlinien erstellen. Es wird empfohlen, die Mesh-Endpunkte, von denen die Konfiguration gelesen werden kann, auf den Envoy-Proxy zu beschränken, der auf bestimmten Rechenressourcen läuft. Erstellen Sie eine IAM-Richtlinie und fügen Sie die `appmesh:StreamAggregatedResources` Berechtigung zu der Richtlinie hinzu. Die folgende Beispielrichtlinie ermöglicht die Konfiguration der virtuellen Knoten, die in einem Service-Mesh benannt `serviceBv1` und `serviceBv2` gelesen werden können. Die Konfiguration kann für keine anderen virtuellen Knoten gelesen werden, die im Service Mesh definiert sind. Weitere Informationen zum Erstellen oder Bearbeiten einer IAM-Richtlinie finden Sie unter [Erstellen von IAM-Richtlinien](#) und [Bearbeiten von IAM-Richtlinien](#).

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

Sie können mehrere Richtlinien erstellen, wobei jede Richtlinie den Zugriff auf verschiedene Mesh-Endpunkte einschränkt.

Erstellen einer IAM-Rolle

Wenn Sie möchten, dass alle Mesh-Endpunkte in einem Service-Mesh die Konfiguration für alle Mesh-Endpunkte lesen können, müssen Sie nur eine IAM-Rolle erstellen. Wenn Sie die Mesh-Endpunkte einschränken möchten, aus denen die Konfiguration von einzelnen Mesh-Endpunkten gelesen werden kann, müssen Sie für jede Richtlinie, die Sie im vorherigen Schritt erstellt haben, eine Rolle erstellen. Füllen Sie die Anweisungen für die Rechenressource aus, auf der der Proxy ausgeführt wird.

- Amazon EKS — Wenn Sie eine einzelne Rolle verwenden möchten, können Sie die vorhandene Rolle verwenden, die bei der Erstellung Ihres Clusters erstellt und den Worker-Knoten zugewiesen wurde. Um mehrere Rollen verwenden zu können, muss Ihr Cluster die Anforderungen erfüllen, die unter [Aktivieren von IAM-Rollen für Dienstkonten in Ihrem Cluster](#) definiert sind. Erstellen Sie die IAM-Rollen und ordnen Sie die Rollen den Kubernetes-Dienstkonten zu. Weitere Informationen finden Sie unter [Erstellen einer IAM-Rolle und -Richtlinie für Ihr Dienstkonto](#) und [Angeben einer IAM-Rolle für Ihr Dienstkonto](#).
- Amazon ECS — Wählen Sie AWSService, Elastic Container Service und dann den Anwendungsfall Elastic Container Service Task aus, wenn Sie Ihre IAM-Rolle erstellen.
- Amazon EC2 — Wählen Sie AWSService, EC2 und dann den EC2-Anwendungsfall aus, wenn Sie Ihre IAM-Rolle erstellen. Dies gilt unabhängig davon, ob Sie den Proxy direkt auf einer Amazon EC2 Instance oder auf einer Instance auf einer Instance hosten.

Weitere Informationen zum Erstellen einer IAM-Rolle finden Sie unter [Erstellen einer Rolle für einen AWS -Service](#).

Anfügen einer IAM-Richtlinie

Wenn Sie möchten, dass alle Mesh-Endpunkte in einem Service-Mesh die Konfiguration für alle Mesh-Endpunkte lesen können, hängen Sie die [AWSAppMeshEnvoyAccess](#) verwaltete IAM-Richtlinie an die IAM-Rolle an, die Sie in einem vorherigen Schritt erstellt haben. Wenn Sie die Mesh-Endpunkte einschränken möchten, aus denen die Konfiguration von einzelnen Mesh-Endpunkten gelesen werden kann, hängen Sie jede von Ihnen erstellte Richtlinie an jede von Ihnen erstellte Rolle an. Weitere Informationen zum Anhängen einer benutzerdefinierten oder verwalteten IAM-Richtlinie an eine IAM-Rolle finden Sie unter [Hinzufügen von IAM-Identitätsberechtigungen](#).

IAM-Rolle anfügen

Ordnen Sie jede IAM-Rolle der entsprechenden Rechenressource zu:

- Amazon EKS — Wenn Sie die Richtlinie an die Rolle angehängt haben, die Ihren Worker-Knoten zugewiesen ist, können Sie diesen Schritt überspringen. Wenn Sie separate Rollen erstellt haben, weisen Sie jede Rolle einem separaten Kubernetes-Dienstkonto zu und weisen Sie jedes Dienstkonto einer einzelnen Kubernetes-Pod-Bereitstellungsspezifikation zu, die den Envoy-Proxy enthält. Weitere Informationen finden Sie unter [Angeben einer IAM-Rolle für Ihr Servicekonto](#) im Amazon EKS-Benutzerhandbuch und unter [Konfigurieren von Servicekonten für Pods](#) in der Kubernetes-Dokumentation.
- Amazon ECS — Fügen Sie der Aufgabendefinition, die den Envoy-Proxy enthält, eine Amazon ECS-Aufgabenrolle hinzu. Die Aufgabe kann mit dem EC2- oder Fargate-Starttyp ausgeführt werden. Weitere Informationen zum Erstellen einer Amazon ECS-Aufgabenrolle und zum Anhängen an eine Aufgabe finden Sie unter [Angeben einer IAM-Rolle für Ihre Aufgaben](#).
- Amazon EC2 — Die IAM-Rolle muss der Amazon EC2 EC2-Instance zugeordnet sein, die den Envoy-Proxy hostet. Weitere Informationen zum Anhängen einer Rolle an eine Amazon EC2 EC2-Instance finden Sie unter [Ich habe eine IAM-Rolle erstellt und möchte sie jetzt einer EC2-Instance zuweisen](#).

Bestätigen Sie die Erlaubnis

Vergewissern Sie sich, dass die `dieappmesh:StreamAggregatedResources` Berechtigung der Rechenressource zugewiesen ist, auf der Sie den Proxy hosten, indem Sie einen der Compute-Dienstnamen auswählen.

Amazon EKS

Eine benutzerdefinierte Richtlinie kann der Rolle zugewiesen werden, die den Worker-Knoten, einzelnen Pods oder beiden zugewiesen ist. Es wird jedoch empfohlen, die Richtlinie nur einzelnen Pods zuzuweisen, sodass Sie den Zugriff einzelner Pods auf einzelne Mesh-Endpunkte beschränken können. Wenn die Richtlinie an die Rolle angehängt ist, die den Worker-Knoten zugewiesen ist, wählen Sie die Registerkarte Amazon EC2 aus und führen Sie die dort aufgeführten Schritte für Ihre Worker-Knoten-Instances aus. Gehen Sie wie folgt vor, um zu ermitteln, welche IAM-Rolle einem Kubernetes-Pod zugewiesen ist.

1. Sehen Sie sich die Details einer Kubernetes-Bereitstellung an, die den Pod enthält, dem Sie bestätigen möchten, dass ein Kubernetes-Dienstkonto zugewiesen ist. Mit dem

folgenden Befehl werden die Details für ein Deployment mit dem Namen *my-deployment* angezeigt.

```
kubectl describe deployment my-deployment
```

Notieren Sie sich in der zurückgegebenen Ausgabe den Wert rechts von `ServiceAccount:`. Wenn eine Zeile, die mit `ServiceAccount:` beginnt, nicht existiert, ist der Bereitstellung derzeit kein benutzerdefiniertes Kubernetes-Dienstkonto zugewiesen. Sie müssen eines zuweisen. Weitere Informationen finden Sie unter [Konfigurieren von Dienstkonten für Pods](#) in der Kubernetes-Dokumentation.

2. Sehen Sie sich die Details des im vorherigen Schritt zurückgegebenen Dienstkontos an. Mit dem folgenden Befehl werden die Details eines Dienstkontos mit dem Namen *my-service-account* angezeigt.

```
kubectl describe serviceaccount my-service-account
```

Vorausgesetzt, das Kubernetes-Dienstkonto ist mit einer AWS Identity and Access Management Rolle verknüpft, sieht eine der zurückgegebenen Zeilen dem folgenden Beispiel ähnlich.

```
Annotations:          eks.amazonaws.com/role-arn=arn:aws:iam::123456789012:role/  
my-deployment
```

Im vorherigen Beispiel *my-deployment* ist dies der Name der IAM-Rolle, mit der das Dienstkonto verknüpft ist. Wenn die Ausgabe des Dienstkontos keine Zeile enthält, die dem obigen Beispiel ähnelt, ist das Kubernetes-Dienstkonto keinem AWS Identity and Access Management Konto zugeordnet und Sie müssen es einem Konto zuordnen. Weitere Informationen finden Sie unter [Angaben einer IAM-Rolle für Ihr Servicekonto](#).

3. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter <https://console.aws.amazon.com/iam/>.
4. Klicken Sie im linken Navigationsbereich auf die Option Rollen. Wählen Sie den Namen der IAM-Rolle aus, den Sie in einem vorherigen Schritt notiert haben.
5. Vergewissern Sie sich, dass entweder die zuvor erstellte benutzerdefinierte Richtlinie oder die [AWSAppMeshEnvoyAccess](#) verwaltete Richtlinie aufgeführt ist. Wenn keine der Richtlinien angehängt ist, [fügen Sie der IAM-Rolle eine IAM-Richtlinie](#) hinzu. Wenn Sie eine benutzerdefinierte IAM-Richtlinie anhängen möchten, aber noch keine haben, müssen

Sie [eine benutzerdefinierte IAM-Richtlinie mit den erforderlichen Berechtigungen erstellen](#). Wenn eine benutzerdefinierte IAM-Richtlinie angehängt ist, wählen Sie die Richtlinie aus und bestätigen Sie, dass sie enthält "Action": "appmesh:StreamAggregatedResources". Ist dies nicht der Fall, müssen Sie diese Berechtigung zu Ihrer benutzerdefinierten IAM-Richtlinie hinzufügen. Sie können auch überprüfen, ob der entsprechende Amazon-Ressourcennamen (ARN) für einen bestimmten Mesh-Endpunkt aufgeführt ist. Wenn keine ARNs aufgeführt sind, können Sie die Richtlinie bearbeiten, um die aufgelisteten ARNs hinzuzufügen, zu entfernen oder zu ändern. Weitere Informationen finden Sie unter [Bearbeiten von IAM-Richtlinien](#) und [Erstellen einer IAM-Richtlinie](#).

6. Wiederholen Sie die vorherigen Schritte für jeden Kubernetes-Pod, der den Envoy-Proxy enthält.

Amazon ECS

1. Wählen Sie in der Amazon ECS-Konsole die Option Aufgabendefinitionen aus.
2. Wählen Sie Ihre Amazon ECS-Aufgabe aus.
3. Wählen Sie auf der Seite „Name der Aufgabendefinition“ Ihre Aufgabendefinition aus.
4. Wählen Sie auf der Seite Aufgabendefinition den Link mit dem Namen der IAM-Rolle aus, der sich rechts neben der Aufgabenrolle befindet. Wenn eine IAM-Rolle nicht aufgeführt ist, müssen Sie [eine IAM-Rolle erstellen](#) und sie Ihrer Aufgabe zuordnen, indem Sie [Ihre Aufgabendefinition aktualisieren](#).
5. Vergewissern Sie sich auf der Übersichtsseite auf der Registerkarte Berechtigungen, dass entweder die zuvor erstellte benutzerdefinierte Richtlinie oder die [AWSAppMeshEnvoyAccess](#) verwaltete Richtlinie aufgeführt ist. Wenn keine der Richtlinien angehängt ist, [fügen Sie der IAM-Rolle eine IAM-Richtlinie](#) hinzu. Wenn Sie eine benutzerdefinierte IAM-Richtlinie anhängen möchten, aber keine haben, müssen Sie [die benutzerdefinierte IAM-Richtlinie erstellen](#). Wenn eine benutzerdefinierte IAM-Richtlinie angehängt ist, wählen Sie die Richtlinie aus und bestätigen Sie, dass sie enthält "Action": "appmesh:StreamAggregatedResources". Ist dies nicht der Fall, müssen Sie diese Berechtigung zu Ihrer benutzerdefinierten IAM-Richtlinie hinzufügen. Sie können auch überprüfen, ob der entsprechende Amazon-Ressourcennamen (ARN) für bestimmte Mesh-Endpunkte aufgeführt ist. Wenn keine ARNs aufgeführt sind, können Sie die Richtlinie bearbeiten, um die aufgelisteten ARNs hinzuzufügen, zu entfernen oder zu ändern. Weitere Informationen finden Sie unter [Bearbeiten von IAM-Richtlinien](#) und [Erstellen einer IAM-Richtlinie](#).

6. Wiederholen Sie die vorherigen Schritte für jede Aufgabendefinition, die den Envoy-Proxy enthält.

Amazon EC2

1. Klicken Sie in der Amazon EC2 EC2-Konsole auf die Option Instances im linken Navigationsbereich.
2. Wählen Sie eine Ihrer Instances aus, die den Envoy-Proxy hostet.
3. Wählen Sie auf der Registerkarte Beschreibung den Link mit dem Namen der IAM-Rolle aus, der sich rechts neben der IAM-Rolle befindet. Wenn eine IAM-Rolle nicht aufgeführt ist, müssen Sie [eine IAM-Rolle erstellen](#).
4. Vergewissern Sie sich auf der Übersichtsseite auf der Registerkarte Berechtigungen, dass entweder die zuvor erstellte benutzerdefinierte Richtlinie oder die [AWSAppMeshEnvoyAccess](#) verwaltete Richtlinie aufgeführt ist. Wenn keine der Richtlinien angehängt ist, [hängen Sie die IAM-Richtlinie](#) an die IAM-Rolle an. Wenn Sie eine benutzerdefinierte IAM-Richtlinie anhängen möchten, aber keine haben, müssen Sie [die benutzerdefinierte IAM-Richtlinie erstellen](#). Wenn eine benutzerdefinierte IAM-Richtlinie angehängt ist, wählen Sie die Richtlinie aus und bestätigen Sie, dass sie enthält "Action" : "appmesh:StreamAggregatedResources". Ist dies nicht der Fall, müssen Sie diese Berechtigung zu Ihrer benutzerdefinierten IAM-Richtlinie hinzufügen. Sie können auch überprüfen, ob der entsprechende Amazon-Ressourcennamen (ARN) für bestimmte Mesh-Endpunkte aufgeführt ist. Wenn keine ARNs aufgeführt sind, können Sie die Richtlinie bearbeiten, um die aufgelisteten ARNs hinzuzufügen, zu entfernen oder zu ändern. Weitere Informationen finden Sie unter [Bearbeiten von IAM-Richtlinien](#) und [Erstellen einer IAM-Richtlinie](#).
5. Wiederholen Sie die vorherigen Schritte für jede Instanz, auf der Sie den Envoy-Proxy hosten.

Problembehandlung bei AWS App Mesh Identität und Zugriff

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit App Mesh und IAM auftreten können.

Themen

- [Ich bin nicht berechtigt, eine Aktion in App Mesh durchzuführen](#)

- [Ich möchte Personen außerhalb meines AWS Kontos den Zugriff auf meine App Mesh Mesh-Ressourcen ermöglichen](#)

Ich bin nicht berechtigt, eine Aktion in App Mesh durchzuführen

Wenn Ihnen AWS Management Console mitgeteilt wird, dass Sie nicht berechtigt sind, eine Aktion auszuführen, müssen Sie sich an Ihren Administrator wenden, um Unterstützung zu erhalten. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Der folgende Fehler tritt auf, wenn der mateojackson IAM-Benutzer versucht, mit der Konsole einen virtuellen Knoten mit dem Namen my-mesh zu erstellen, der *my-virtual-node* im Mesh *my-mesh* benannt ist, aber nicht über die `appmesh:CreateVirtualNode` entsprechende Berechtigung verfügt.

```
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 diesem Fall bittet Mateo seinen Administrator, seine Richtlinien zu aktualisieren, damit er mithilfe der Aktion einen virtuellen Knoten erstellen kann. `appmesh:CreateVirtualNode`

Note

Da ein virtueller Knoten innerhalb eines Meshs erstellt wird, benötigt Mateos Konto auch die `appmesh:ListMeshes` Aktionen `appmesh:DescribeMesh` und, um den virtuellen Knoten in der Konsole zu erstellen.

Ich möchte Personen außerhalb meines AWS Kontos den Zugriff auf meine App Mesh Mesh-Ressourcen ermöglichen

Sie können eine Rolle erstellen, die Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation für den Zugriff auf Ihre Ressourcen verwenden können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Im Fall von Diensten, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (Access Control Lists, ACLs) verwenden, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen darüber, ob App Mesh diese Funktionen unterstützt, finden Sie unter [Wie AWS App Mesh funktioniert mit IAM](#).
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, AWS-Konten die Ihnen gehören, finden Sie im IAM-Benutzerhandbuch unter [Gewähren des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen](#).
- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie [AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte](#).
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter [Gewähren von Zugriff für extern authentifizierte Benutzer \(Identitätsverbund\)](#) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter [So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien](#) im IAM-Benutzerhandbuch.

Protokollieren von AWS App Mesh API-Aufrufen mit AWS CloudTrail

AWS App Mesh ist in einen Dienst integriert [AWS CloudTrail](#), der eine Aufzeichnung der von einem Benutzer, einer Rolle oder einem ausgeführten Aktionen bereitstellt AWS-Service. CloudTrail erfasst alle API-Aufrufe für App Mesh als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der App Mesh Mesh-Konsole und Code-Aufrufe der App Mesh Mesh-API-Operationen. Anhand der von gesammelten Informationen können Sie die Anfrage CloudTrail, die an App Mesh gestellt wurde, die IP-Adresse, von der aus die Anfrage gestellt wurde, wann sie gestellt wurde, und weitere Details ermitteln.

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Ob die Anfrage mit Anmeldeinformationen des Root-Benutzers oder des Benutzers gestellt wurde.
- Ob die Anfrage im Namen eines IAM Identity Center-Benutzers gestellt wurde.
- Gibt an, ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen Verbundbenutzer gesendet wurde.
- Ob die Anforderung aus einem anderen AWS-Service gesendet wurde.

CloudTrail ist in Ihrem aktiv AWS-Konto , wenn Sie das Konto erstellen, und Sie haben automatisch Zugriff auf den CloudTrail Eventverlauf. Der CloudTrail Ereignisverlauf bietet eine einsehbare, durchsuchbare, herunterladbare und unveränderliche Aufzeichnung der aufgezeichneten Verwaltungsereignisse der letzten 90 Tage in einem. AWS-Region Weitere Informationen finden Sie im AWS CloudTrail Benutzerhandbuch unter [Arbeiten mit dem CloudTrail Ereignisverlauf](#). Für die Anzeige des Ereignisverlaufs CloudTrail fallen keine Gebühren an.

Für eine fortlaufende Aufzeichnung der Ereignisse in AWS-Konto den letzten 90 Tagen erstellen Sie einen Trail- oder [CloudTrailLake-Event-Datenspeicher](#).

CloudTrail Pfade

Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket. Alle mit dem erstellten Pfade AWS Management Console sind regionsübergreifend. Sie können einen Pfad mit einer oder mehreren Regionen erstellen, indem Sie den verwenden. AWS CLI Es wird empfohlen, einen Trail mit mehreren Regionen zu erstellen, da Sie alle Aktivitäten in Ihrem Konto AWS-Regionen erfassen. Wenn du einen Trail mit nur einer Region erstellst, kannst du dir nur die Ereignisse ansehen, die in den Trails protokolliert wurden. AWS-Region Weitere Informationen zu Trails finden Sie unter [Einen Trail für Sie erstellen AWS-Konto und Einen Trail für eine Organisation](#) erstellen im AWS CloudTrail Benutzerhandbuch.

Sie können eine Kopie Ihrer laufenden Verwaltungsereignisse kostenlos an Ihren Amazon S3 S3-Bucket senden, CloudTrail indem Sie einen Trail erstellen. Es fallen jedoch Amazon S3 S3-Speichergebühren an. Weitere Informationen zur CloudTrail Preisgestaltung finden Sie unter [AWS CloudTrail Preise](#). Informationen zu Amazon-S3-Preisen finden Sie unter [Amazon S3-Preise](#).

CloudTrail Datenspeicher für Ereignisse in Lake

CloudTrail Mit Lake können Sie SQL-basierte Abfragen für Ihre Ereignisse ausführen. CloudTrail [Lake konvertiert bestehende Ereignisse im zeilenbasierten JSON-Format in das Apache ORC-Format](#). ORC ist ein spaltenförmiges Speicherformat, das für den schnellen Abruf von Daten optimiert ist. Die Ereignisse werden in Ereignisdatenspeichern zusammengefasst, bei denen es sich um unveränderliche Sammlungen von Ereignissen handelt, die auf Kriterien basieren, die Sie mit Hilfe von [erweiterten Ereignisselektoren](#) auswählen. Die Selektoren, die Sie auf einen Ereignisdatenspeicher anwenden, steuern, welche Ereignisse bestehen bleiben und für Sie zur Abfrage verfügbar sind. Weitere Informationen zu CloudTrail Lake finden Sie im Benutzerhandbuch unter [Arbeiten mit AWS CloudTrail Lake](#).AWS CloudTrail

CloudTrail Für das Speichern und Abfragen von Ereignisdaten in Lake fallen Kosten an. Beim Erstellen eines Ereignisdatenspeichers wählen Sie die [Preisoption](#) aus, die für den

Ereignisdatenspeicher genutzt werden soll. Die Preisoption bestimmt die Kosten für die Erfassung und Speicherung von Ereignissen sowie die standardmäßige und maximale Aufbewahrungsdauer für den Ereignisdatenspeicher. Weitere Informationen zur CloudTrail Preisgestaltung finden Sie unter [AWS CloudTrail Preise](#).

App Mesh Mesh-Verwaltungsereignisse in CloudTrail

[Verwaltungsereignisse](#) enthalten Informationen zu Verwaltungsvorgängen, die für Ressourcen in Ihrem ausgeführt werden AWS-Konto. Sie werden auch als Vorgänge auf Steuerebene bezeichnet. CloudTrail protokolliert standardmäßig Verwaltungsereignisse.

AWS App Mesh protokolliert alle App Mesh-Stuerebenenoperationen als Verwaltungsereignisse. Eine Liste der AWS App Mesh Steuerebenenoperationen, bei denen App Mesh protokolliert CloudTrail, finden Sie in der [AWS App Mesh API-Referenz](#).

Beispiele für App Mesh Mesh-Ereignisse

Ein Ereignis stellt eine einzelne Anfrage aus einer beliebigen Quelle dar und enthält Informationen über den angeforderten API-Vorgang, Datum und Uhrzeit des Vorgangs, Anforderungsparameter usw. CloudTrail Protokolldateien sind kein geordneter Stack-Trace der öffentlichen API-Aufrufe, sodass Ereignisse nicht in einer bestimmten Reihenfolge angezeigt werden.

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der die `StreamAggregatedResources` Aktion demonstriert.

```
{
  "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"
}
```

Informationen zu CloudTrail Datensatzinhalten finden Sie im AWS CloudTrail Benutzerhandbuch unter [CloudTrailDatensatzinhalte](#).

Datenschutz in AWS App Mesh

Das [Modell der AWS gemeinsamen Verantwortung](#) und geteilter Verantwortung gilt für den Datenschutz in AWS App Mesh. Wie in diesem Modell beschrieben, AWS ist verantwortlich für den Schutz der globalen Infrastruktur, auf der alle Systeme laufen AWS Cloud. Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Sie sind auch für die Sicherheitskonfiguration und die Verwaltungsaufgaben für die von Ihnen verwendeten AWS-Services verantwortlich. Weitere Informationen zum Datenschutz finden Sie unter [Häufig gestellte Fragen zum Datenschutz](#). Informationen zum Datenschutz in Europa finden Sie im Blog-Beitrag [AWS -Modell der geteilten Verantwortung und in der DSGVO](#) im AWS -Sicherheitsblog.

Aus Datenschutzgründen empfehlen wir, dass Sie AWS-Konto Anmeldeinformationen schützen und einzelne Benutzer mit AWS IAM Identity Center oder AWS Identity and Access Management (IAM) einrichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem empfehlen wir, die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
- Verwenden Sie SSL/TLS, um mit Ressourcen zu kommunizieren. AWS Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein. AWS CloudTrail

- Verwenden Sie AWS Verschlüsselungslösungen zusammen mit allen darin enthaltenen Standardsicherheitskontrollen AWS-Services.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu schützen.
- Wenn Sie für den Zugriff AWS über eine Befehlszeilenschnittstelle oder eine API FIPS 140-2-validierte kryptografische Module benötigen, verwenden Sie einen FIPS-Endpunkt. Weitere Informationen über verfügbare FIPS-Endpunkte finden Sie unter [Federal Information Processing Standard \(FIPS\) 140-2](#).

Wir empfehlen dringend, in Freitextfeldern, z. B. im Feld Name, keine vertraulichen oder sensiblen Informationen wie die E-Mail-Adressen Ihrer Kunden einzugeben. Dies gilt auch, wenn Sie mit App Mesh oder anderen Geräten arbeiten und die Konsole, die API oder AWS SDKs AWS-Services verwenden. AWS CLI Alle Daten, die Sie in Tags oder Freitextfelder eingeben, die für Namen verwendet werden, können für Abrechnungs- oder Diagnoseprotokolle verwendet werden. Wenn Sie eine URL für einen externen Server bereitstellen, empfehlen wir dringend, keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL einzuschließen.

Datenverschlüsselung

Ihre Daten werden bei der Verwendung von App Mesh verschlüsselt.

Verschlüsselung im Ruhezustand

Standardmäßig werden die App Mesh Mesh-Konfigurationen, die Sie erstellen, im Ruhezustand verschlüsselt.

Verschlüsselung während der Übertragung

App Mesh Mesh-Dienstendpunkte verwenden das HTTPS-Protokoll. Die gesamte Kommunikation zwischen dem Envoy-Proxy und dem App Mesh Envoy Management Service ist verschlüsselt. Wenn Sie eine FIPS-konforme Verschlüsselung für die Kommunikation zwischen dem Envoy-Proxy und dem App Mesh Envoy Management Service benötigen, können Sie eine FIPS-Variante des Envoy-Proxy-Container-Images verwenden. Weitere Informationen finden Sie unter [Bild des Gesandten](#).

Die Kommunikation zwischen Containern innerhalb virtueller Knoten ist nicht verschlüsselt, aber dieser Verkehr verlässt den Netzwerk-Namespace nicht.

Compliance-Validierung für AWS App Mesh

Informationen darüber, ob ein AWS-Service in den Geltungsbereich bestimmter Compliance-Programme fällt, finden Sie unter [AWS-Services in Geltungsbereich nach Compliance-Programm](#). Wählen Sie das Compliance-Programm, das Sie interessiert. Allgemeine Informationen finden Sie unter [AWS-Compliance-Programme](#).

Sie können Auditberichte von Drittanbietern unter AWS Artifact herunterladen. Weitere Informationen finden Sie unter [Berichte herunterladen in AWS Artifact](#).

Ihre Compliance-Verantwortung bei der Verwendung von AWS-Services ist von der Sensibilität Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften abhängig. AWS stellt die folgenden Ressourcen zur Unterstützung der Compliance bereit:

- [Kurzanleitungen für Sicherheit und Compliance](#) – In diesen Bereitstellungsleitfäden werden Überlegungen zur Architektur erörtert und Schritte zum Bereitstellen von Basisumgebungen auf AWS zur Verfügung gestellt, die auf Sicherheit und Compliance ausgerichtet sind.
- [Erstellung einer Architektur mit HIPAA-konformer Sicherheit und Compliance in Amazon Web Services](#) – In diesem Whitepaper wird beschrieben, wie Unternehmen mithilfe von AWS HIPAA-berechtigte Anwendungen erstellen können.

Note

Nicht alle AWS-Services sind HIPAA-berechtigt. Weitere Informationen finden Sie in der [Referenz für HIPAA-berechtigte Services](#).

- [AWS-Compliance-Ressourcen](#) – Diese Arbeitsbücher und Leitfäden könnten für Ihre Branche und Ihren Standort relevant sein.
- [AWS-Compliance-Leitfäden für Kunden](#) – Verstehen Sie das Modell der geteilten Verantwortung aus dem Blickwinkel der Einhaltung von Vorschriften. In den Leitfäden werden die bewährten Methoden zum Schutz von AWS-Services zusammengefasst und die Leitlinien den Sicherheitskontrollen in verschiedenen Frameworks (einschließlich des National Institute of Standards and Technology (NIST), des Payment Card Industry Security Standards Council (PCI) und der International Organization for Standardization (ISO)) zugeordnet.
- [Auswertung von Ressourcen mit Regeln](#) im AWS ConfigEntwicklerhandbuch – Der AWS Config-Service bewertet, wie gut Ihre Ressourcenkonfigurationen mit internen Praktiken, Branchenrichtlinien und Vorschriften übereinstimmen.

- [AWS Security Hub](#) – Dieser AWS-Service bietet einen umfassenden Überblick über Ihren Sicherheitsstatus innerhalb von AWS. Security Hub verwendet Sicherheitskontrollen, um Ihre AWS-Ressourcen zu bewerten und Ihre Einhaltung von Sicherheitsstandards und bewährten Methoden zu überprüfen. Eine Liste der unterstützten Services und Kontrollen finden Sie in der [Security-Hub-Steuerungsreferenz](#).
- [AWS Audit Manager](#) – Dieser AWS-Service hilft Ihnen, Ihre AWS-Nutzung kontinuierlich zu überprüfen, um den Umgang mit Risiken und die Compliance von Branchenstandards zu vereinfachen.

Infrastruktursicherheit in AWS App Mesh

Als verwalteter Dienst AWS App Mesh ist er durch AWS globale Netzwerksicherheit geschützt. Informationen zu AWS Sicherheitsdiensten und zum AWS Schutz der Infrastruktur finden Sie unter [AWS Cloud-Sicherheit](#). Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter [Infrastructure Protection](#) in Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS veröffentlichte API-Aufrufe, um über das Netzwerk auf App Mesh zuzugreifen. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit [AWS Security Token Service](#) (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Sie können den Sicherheitsstatus Ihrer VPC verbessern, indem Sie App Mesh für die Verwendung eines VPC-Endpunkts mit Schnittstelle konfigurieren. Weitere Informationen finden Sie unter [VPC-Endpunkte mit App Mesh Mesh-Schnittstelle \(\)AWS PrivateLink](#).

VPC-Endpunkte mit App Mesh Mesh-Schnittstelle (AWS PrivateLink)

Sie können die Sicherheitslage Ihrer Amazon VPC verbessern, indem Sie App Mesh so konfigurieren, dass es einen VPC-Endpunkt mit Schnittstelle verwendet. Schnittstellenendpunkte werden von einer Technologie unterstützt AWS PrivateLink, mit der Sie privat auf App Mesh Mesh-APIs zugreifen können, indem Sie private IP-Adressen verwenden. PrivateLinkschränkt den gesamten Netzwerkverkehr zwischen Ihrer Amazon VPC und App Mesh auf das Amazon-Netzwerk ein.

Sie müssen es nicht konfigurieren PrivateLink, aber wir empfehlen es. Weitere Informationen zu VPC-Endpunkten PrivateLink und deren Schnittstelle finden Sie unter [Zugreifen auf Dienste](#) über AWS PrivateLink

Überlegungen zu VPC-Endpunkten mit App Mesh Mesh-Schnittstelle

Bevor Sie VPC-Schnittstellen-Endpunkte für App Mesh einrichten, sollten Sie die folgenden Überlegungen beachten:

- Wenn Ihre Amazon VPC kein Internet-Gateway hat und Ihre Aufgaben den `awslogs` Protokolltreiber verwenden, um CloudWatch Protokollinformationen an Logs zu senden, müssen Sie einen VPC-Schnittstellen-Endpunkt für CloudWatch Logs erstellen. Weitere Informationen finden Sie unter [Using CloudWatch Logs with Interface VPC Endpoints](#) im Amazon CloudWatch Logs-Benutzerhandbuch.
- VPC-Endpunkte unterstützen keine AWS regionsübergreifenden Anfragen. Stellen Sie sicher, dass Sie Ihren Endpunkt in derselben Region erstellen, in der Sie Ihre API-Aufrufe an App Mesh tätigen möchten.
- VPC-Endpunkte unterstützen nur von Amazon bereitgestellten DNS über Amazon Route 53. Wenn Sie Ihre eigene DNS verwenden möchten, können Sie die bedingte DNS-Weiterleitung nutzen. Weitere Informationen finden Sie unter [DHCP Options Sets](#) im Amazon VPC-Benutzerhandbuch.
- Die mit dem VPC-Endpunkt verbundene Sicherheitsgruppe muss eingehende Verbindungen über Port 443 aus dem privaten Subnetz der Amazon VPC zulassen.

Note

Die Steuerung des Zugriffs auf App Mesh durch Anhängen einer Endpunktrichtlinie an den VPC-Endpunkt (z. B. mithilfe des Dienstnamens `com.amazonaws.Region.appmesh-envoy-management`) wird für die Envoy-Verbindung nicht unterstützt.

Weitere Überlegungen und Einschränkungen finden Sie unter [Überlegungen zur Availability Zone für Benutzeroberflächenendpunkte](#) und [Eigenschaften und Einschränkungen von Schnittstellenendpunkten](#).

Erstellen Sie den VPC-Endpunkt der Schnittstelle für App Mesh

Um den VPC-Schnittstellen-Endpunkt für den App Mesh-Service zu [erstellen, verwenden Sie das Verfahren Creating an Interface Endpoint](#) im Amazon VPC-Benutzerhandbuch. Geben Sie `com.amazonaws.Region.appmesh-envoy-management` den Servicenamen für Ihren Envoy-Proxy an, um eine Verbindung zum öffentlichen Envoy-Verwaltungsservice von App Mesh herzustellen, und `com.amazonaws.Region.appmesh` für Mesh-Operationen.

Note

Region steht für die Regionskennung für eine AWS Region, die von App Mesh unterstützt wird, z. B. `us-east-2` für die Region USA Ost (Ohio).

Sie können zwar in jeder Region, in der App Mesh unterstützt wird, einen VPC-Schnittstellen-Endpunkt für App Mesh definieren, Sie können jedoch möglicherweise keinen Endpunkt für alle Availability Zones in jeder Region definieren. Um herauszufinden, welche Availability Zones mit VPC-Schnittstellen-Endpunkten in einer Region unterstützt werden, verwenden Sie den [describe-vpc-endpoint-services](#) Befehl oder den AWS Management Console. Mit den folgenden Befehlen werden beispielsweise die Verfügbarkeitszonen zurückgegeben, in denen Sie VPC-Endpunkte mit App Mesh Mesh-Schnittstelle in der Region USA Ost (Ohio) bereitstellen können:

```
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[]'
```

Ausfallsicherheit in AWS App Mesh

Im Zentrum der globalen AWS-Infrastruktur stehen die AWS-Regionen und Availability Zones. AWS Regionen stellen mehrere physisch getrennte und isolierte Availability Zones bereit, die mit Netzwerken mit geringer Latenz, hohem Durchsatz und hochredundanten Vernetzungen verbunden

sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Availability Zones ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser hoch verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

App Mesh führt Control-Plane-Instances für mehrere -Availability-Zones aus, um eine hohe Verfügbarkeit zu gewährleisten. App Mesh erkennt und ersetzt fehlerhafte Steuerebenen-Instances und bietet automatisierte Versionsaktualisierungen und Patches für sie.

Notfallwiederherstellung in AWS App Mesh

Der App Mesh Mesh-Dienst verwaltet Backups von Kundendaten. Sie müssen nichts tun, um Backups zu verwalten. Die gesicherten Daten sind verschlüsselt.

Konfiguration und Schwachstellenanalyse in AWS App Mesh

App Mesh verkauft ein verwaltetes [Envoy-Proxy-Docker-Container-Image](#), das Sie mit Ihren Microservices bereitstellen. App Mesh stellt sicher, dass das Container-Image mit den neuesten Sicherheitslücken- und Leistungspatches gepatcht wird. App Mesh testet neue Envoy-Proxy-Versionen anhand des App Mesh Mesh-Funktionsumfangs, bevor Ihnen die Bilder zur Verfügung gestellt werden.

Sie müssen Ihre Microservices aktualisieren, um die aktualisierte Container-Image-Version verwenden zu können. Im Folgenden finden Sie die neueste Version des Images.

```
840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-envoy:v1.27.3.0-prod
```

Fehlerbehebung bei App Mesh

In diesem Kapitel werden bewährte Methoden zur Fehlerbehebung und häufig auftretende Probleme bei der Verwendung von App Mesh behandelt. Wählen Sie einen der folgenden Bereiche aus, um bewährte Methoden und häufige Probleme für diesen Bereich zu überprüfen.

Themen

- [Bewährte Methoden zur Fehlerbehebung bei App Mesh](#)
- [Fehlerbehebung bei der Einrichtung von App Mesh](#)
- [Fehlerbehebung bei der App-Mesh-Konnektivität](#)
- [Fehlerbehebung bei der App Mesh-Skalierung](#)
- [Fehlerbehebung bei der Beobachtbarkeit von App Mesh](#)
- [Fehlerbehebung bei der App-Mesh-Sicherheit](#)
- [Fehlerbehebung bei App Mesh Kubernetes](#)

Bewährte Methoden zur Fehlerbehebung bei App Mesh

Wir empfehlen Ihnen, die bewährten Methoden in diesem Thema zu befolgen, um Probleme bei der Verwendung von App Mesh zu beheben.

Aktivieren der Envoy-Proxy-Verwaltungsschnittstelle

Der Envoy-Proxy wird mit einer -Verwaltungsschnittstelle geliefert, mit der Sie Konfiguration und Statistiken ermitteln und andere administrative Funktionen wie Connection Draining ausführen können. Weitere Informationen finden Sie unter [Administrationsschnittstelle](#) in der Envoy-Dokumentation.

Wenn Sie die verwaltete verwenden [Bild des Gesandten](#), ist der Verwaltungsendpunkt standardmäßig auf Port 9901 aktiviert. In den Beispielen in wird die Beispiel-Administrationsendpunkt-URL als [Fehlerbehebung bei der Einrichtung von App Mesh](#) angezeigt `http://my-app.default.svc.cluster.local:9901/`.

Note

Der Verwaltungsendpunkt sollte niemals dem öffentlichen Internet zugänglich gemacht werden. Darüber hinaus empfehlen wir, die Verwaltungsendpunktprotokolle zu

überwachen, die von der `ENVOY_ADMIN_ACCESS_LOG_FILE` Umgebungsvariablen `/tmp/envoy_admin_access.log` standardmäßig auf festgelegt werden.

Aktivieren der Envoy DogStats-D-Integration für Metrik-Offload

Der Envoy-Proxy kann so konfiguriert werden, dass er Statistiken für OSI-Layer-4- und -Layer-7-Datenverkehr und für den internen Prozessstatus auslagert. Dieses Thema zeigt zwar, wie Sie diese Statistiken verwenden, ohne die Metriken wie CloudWatch Metriken und Prometheus auf Senken wie Metriken und Prometheus auszulagern. Wenn Sie diese Statistiken an einem zentralen Ort für alle Ihre Anwendungen haben, können Sie Probleme diagnostizieren und das Verhalten schneller bestätigen. Weitere Informationen finden Sie unter [Verwenden von Amazon CloudWatch-Metriken](#) und in der [Prometheus-Dokumentation](#).

Sie können DogStatsD-Metriken konfigurieren, indem Sie die in definierten Parameter festlegen [DogStatsD-Variablen](#). Weitere Informationen zu DogStatsD finden Sie in der [DogStatsD-Dokumentation](#). Eine Demonstration der Metrikauslagerung auf AWS CloudWatch Metriken finden Sie im [App Mesh mit Amazon-ECS-Grundlagen-Walkthrough](#) auf GitHub.

Aktivieren der Zugriffsprotokolle

Wir empfehlen, Zugriffsprotokolle für Ihr [Virtuelle Gateways](#) zu aktivieren [Virtuelle Knoten](#) und Details zur Übertragung des Datenverkehrs zwischen Ihren Anwendungen zu ermitteln. Weitere Informationen finden Sie unter [Zugriffsprotokollierung](#) in der Envoy-Dokumentation. Die Protokolle enthalten detaillierte Informationen zum Verhalten des OSI Layer 4- und Layer 7-Datenverkehrs. Wenn Sie das Standardformat von Envoy verwenden, können Sie die Zugriffsprotokolle mit [CloudWatch Logs Insights](#) mithilfe der folgenden Parse-Anweisung analysieren.

```
parse @message "[*] \"* * *\" * * * * * * * * * * *\" as StartTime, Method, Path, Protocol, ResponseCode, ResponseFlags, BytesReceived, BytesSent, DurationMillis, UpstreamServiceTimeMillis, ForwardedFor, UserAgent, RequestId, Authority, UpstreamHost
```

Aktivieren der Envoy-Debug-Protokollierung in Vorproduktionsumgebungen

Wir empfehlen, die Protokollebene des Envoy-Proxys debug in einer Vorproduktionsumgebung auf festzulegen. Debug-Protokolle können Ihnen helfen, Probleme zu identifizieren, bevor Sie die zugehörige App-Mesh-Konfiguration in Ihre Produktionsumgebung hochstufen.

Wenn Sie das [Envoy-Image](#) verwenden, können Sie die Protokollebene debug über die `ENVOY_LOG_LEVEL` Umgebungsvariable auf festlegen.

Note

Wir empfehlen nicht, die `-debug`Ebene in Produktionsumgebungen zu verwenden. Wenn Sie die Stufe auf festlegen, debug erhöht sich die Protokollierung und kann sich auf die Leistung und die Gesamtkosten von Protokollen auswirken, die auf Lösungen wie [CloudWatch Logs](#) ausgelagert werden.

Wenn Sie das Standardformat von Envoy verwenden, können Sie die Prozessprotokolle mit [CloudWatch Logs Insights](#) mithilfe der folgenden Parse-Anweisung analysieren:

```
parse @message "[*][*][*][*] [*] *" as Time, Thread, Level, Name, Source, Message
```

Überwachen der Envoy-Proxy-Konnektivität mit der App-Mesh-Steuerebene

Wir empfehlen Ihnen, die Envoy-Metriken zu überwachen, `control_plane.connected_state` um sicherzustellen, dass der Envoy-Proxy mit der App-Mesh-Steuerebene kommuniziert, um die dynamischen Konfigurationsressourcen abzurufen. Weitere Informationen finden Sie unter [Management Server](#).

Fehlerbehebung bei der Einrichtung von App Mesh

In diesem Thema werden häufige Probleme beschrieben, die bei der Einrichtung von App Mesh auftreten können.

Envoy-Container-Image kann nicht abgerufen werden

Symptome

Sie erhalten die folgende Fehlermeldung in einer Amazon-ECS-Aufgabe. Die Amazon-ECR-*Konto-ID* und *-Region* in der folgenden Nachricht können unterschiedlich sein, je nachdem, aus welchem Amazon-ECR-Repository Sie das Container-Image abgerufen haben.

```
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'
```

Auflösung

Dieser Fehler weist darauf hin, dass die verwendete Aufgabenausführungsrolle nicht über die Berechtigung zur Kommunikation mit Amazon ECR verfügt und das Envoy-Container-Image nicht aus dem Repository abrufen kann. Die Ihrer Amazon-ECS-Aufgabe zugewiesene Aufgabenausführungsrolle benötigt eine IAM-Richtlinie mit den folgenden Anweisungen:

```
{
  "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"
}
```

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Es kann keine Verbindung zum App Mesh Envoy Management Service hergestellt werden

Symptome

Ihr Envoy-Proxy kann keine Verbindung zum App Mesh Envoy-Verwaltungsservice herstellen. Sie sehen:

- Verbindung zurückgewiesene Fehler
- Verbindungs-Timeouts
- Fehler beim Beheben des Verwaltungsservice-Endpunkts von App Mesh Envoy
- gRPC-Fehler

Auflösung

Stellen Sie sicher, dass Ihr Envoy-Proxy Zugriff auf das Internet oder einen privaten [VPC-Endpunkt](#) hat und dass Ihre [Sicherheitsgruppen](#) ausgehenden Datenverkehr auf Port 443 zulassen. Die öffentlichen Endpunkte des Envoy Management Service von App Mesh folgen dem FQDN-Format (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
```

Mit dem folgenden Befehl können Sie Ihre Verbindung zu debuggen. Dadurch wird eine gültige, aber leere gRPC-Anforderung an den Envoy Management Service gesendet.

```
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
```

Wenn Sie diese Nachrichten zurückerhalten, funktioniert Ihre Verbindung zum Envoy Management Service. Informationen zum Debuggen von gRPC-bezogenen Fehlern finden Sie unter Fehler in [Envoy vom App Mesh Envoy Management Service getrennt mit Fehlertext](#).

```
grpc-status: 16
grpc-message: Missing Authentication Token
```

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Envoy wurde mit Fehlertext vom App Mesh Envoy Management Service getrennt

Symptome

Ihr Envoy-Proxy kann keine Verbindung zum App Mesh Envoy-Verwaltungsservice herstellen und seine Konfiguration empfangen. Ihre Envoy-Proxy-Protokolle enthalten einen Protokolleintrag wie den folgenden.

```
gRPC config stream closed: gRPC status code, message
```

Auflösung

In den meisten Fällen sollte der Nachrichtenteil des Protokolls auf das Problem hinweisen. In der folgenden Tabelle sind die gängigsten gRPC-Statuscodes aufgeführt, die Sie möglicherweise sehen, ihre Ursachen und ihre Lösungen.

gRPC-Statuscode	Ursache	Auflösung
0	Korrekte Trennung vom Envoy-Verwaltungsservice.	Es gibt kein Problem. App Mesh trennt gelegentlich Envoy-Proxys mit diesem Statuscode. Envoy stellt erneut eine Verbindung her und erhält weiterhin Updates.
3	Der Mesh-Endpunkt (virtueller Knoten oder virtuelles Gateway) oder eine der zugehörigen Ressourcen konnte nicht gefunden werden.	Überprüfen Sie Ihre Envoy-Konfiguration erneut, um sicherzustellen, dass sie den entsprechenden Namen der App Mesh-Ressource hat, für die sie steht. Wenn Ihre App-Mesh-Ressource in andere AWS Ressourcen wie AWS Cloud Map Namespaces oder ACM-Zertifikate integriert ist, stellen Sie sicher, dass diese Ressourcen vorhanden sind.
7	Der Envoy-Proxy ist nicht autorisiert, eine Aktion auszuführen, z. B. eine Verbindung zum Envoy-Verwaltungsservice herzustellen oder zugehörige Ressourcen abzurufen.	Stellen Sie sicher, dass Sie eine IAM-Richtlinie mit den entsprechenden Richtlinienanweisungen für App Mesh und andere -Services erstellen und diese Richtlinie an den IAM-Benutzer oder die IAM-Rolle anfügen, die Ihr Envoy-Proxy verwendet, um eine

gRPC-Statuscode	Ursache	Auflösung
		Verbindung zum Envoy-Verwaltungsservice herzustellen.
8	Die Anzahl der Envoy-Proxys für eine bestimmte App-Mesh-Ressource überschreitet das Servicekontingent auf Kontoebene.	App Mesh service quotas Informationen zu Standard-Kontokontingenten und zum Anfordern einer Kontingenterhöhung finden Sie unter .

gRPC-Statuscode	Ursache	Auflösung
16	Der Envoy-Proxy hat keine gültigen Authentifizierungsanmeldeinformationen für AWS.	Stellen Sie sicher, dass Envoy über die entsprechenden Anmeldeinformationen verfügt, um über einen IAM-Benutzer oder eine IAM-Rolle eine Verbindung zu -AWSServices herzustellen. Ein bekanntes Problem, #24136 , in Envoy für Version v1.24 und früher kann die Anmeldeinformationen nicht abrufen, wenn der Envoy-Prozess über 1024 Dateideskriptoren verwendet. Dies geschieht, wenn Envoy ein hohes Datenverkehrsvolumen bedient. Sie können dieses Problem bestätigen, indem Sie Envoy-Protokolle auf Debug-Ebene auf den Text „ <code>libcurl function was given a bad argument</code> “ überprüfen. Um dieses Problem zu beheben, aktualisieren Sie auf Envoy Version v1.25.1.0-prod oder höher.

Sie können die Statuscodes und Nachrichten von Ihrem Envoy-Proxy mit [Amazon CloudWatch Insights](#) mithilfe der folgenden Abfrage beobachten:

```
filter @message like /gRPC config stream closed/  
| parse @message "gRPC config stream closed: *, *" as StatusCode, Message
```

Wenn die angegebene Fehlermeldung nicht hilfreich war oder Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen.

Fehler bei der Zustandsprüfung des Envoy-Containers, der Bereitschafts-Sonde oder der Liveliness-Sonde

Symptome

Ihr Envoy-Proxy schlägt Zustandsprüfungen in einer Amazon-ECS-Aufgabe, einer Amazon EC2-Instance oder einem Kubernetes-Pod fehl. Sie fragen beispielsweise die Envoy-Verwaltungsschnittstelle mit dem folgenden Befehl ab und erhalten einen anderen Status als LIVE.

```
curl -s http://my-app.default.svc.cluster.local:9901/server_info | jq '.state'
```

Auflösung

Im Folgenden finden Sie eine Liste der Korrekturmaßnahmen, abhängig vom Status, der vom Envoy-Proxy zurückgegeben wird.

- **PRE_INITIALIZING** oder **INITIALIZING** – Der Envoy-Proxy muss noch eine Konfiguration empfangen oder kann keine Verbindung zum App Mesh Envoy Management Service herstellen und abrufen. Der Envoy erhält möglicherweise einen Fehler vom Envoy-Verwaltungsservice, wenn er versucht, eine Verbindung herzustellen. Weitere Informationen finden Sie unter Fehler in [Envoy wurde mit Fehlertext vom App Mesh Envoy Management Service getrennt](#).
- **DRAINING** – Der Envoy-Proxy hat damit begonnen, Verbindungen als Reaktion auf eine `- /healthcheck/fail` oder `-/drain_listeners`Anforderung auf der Envoy-Verwaltungsschnittstelle zu leeren. Wir empfehlen nicht, diese Pfade auf der Verwaltungsschnittstelle aufzurufen, es sei denn, Sie beenden Ihre Amazon-ECS-Aufgabe, Ihre Amazon EC2-Instance oder Ihren Kubernetes-Pod.

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Zustandsprüfung vom Load Balancer zum Mesh-Endpunkt schlägt fehl

Symptome

Ihr Mesh-Endpunkt wird durch die Container-Zustandsprüfung oder Bereitschafts-Sonde als fehlerfrei angesehen, aber die Zustandsprüfung vom Load Balancer zum Mesh-Endpunkt schlägt fehl.

Auflösung

Führen Sie die folgenden Schritte aus, um das Problem zu beheben.

- Stellen Sie sicher, dass die [Sicherheitsgruppe](#), die Ihrem Mesh-Endpoint zugeordnet ist, eingehenden Datenverkehr auf dem Port akzeptiert, den Sie für Ihre Zustandsprüfung konfiguriert haben.
- Stellen Sie sicher, dass die Zustandsprüfung bei manueller Anforderung konsistent erfolgreich ist, z. B. von einem [Bastion-Host innerhalb Ihrer VPC](#).
- Wenn Sie eine Zustandsprüfung für einen virtuellen Knoten konfigurieren, empfehlen wir, einen Endpoint für die Zustandsprüfung in Ihrer Anwendung zu implementieren, z. B. /ping für HTTP. Dadurch wird sichergestellt, dass sowohl der Envoy-Proxy als auch Ihre Anwendung vom Load Balancer aus weitergeleitet werden können.
- Sie können je nach den benötigten Funktionen jeden Elastic Load Balancer-Typ für den virtuellen Knoten verwenden. Weitere Informationen finden Sie unter [Elastic Load Balancing-Funktionen](#).
- Wenn Sie eine Zustandsprüfung für ein [virtuelles Gateway](#) konfigurieren, empfehlen wir, einen [Network Load Balancer](#) mit einer TCP- oder TLS-Zustandsprüfung am Listener-Port des virtuellen Gateways zu verwenden. Dadurch wird sichergestellt, dass der virtuelle Gateway-Listener gestartet wird und Verbindungen akzeptieren kann.

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an den [AWS Support](#) wenden.

Das virtuelle Gateway akzeptiert keinen Datenverkehr über Ports 1024 oder weniger

Symptome

Ihr virtuelles Gateway akzeptiert keinen Datenverkehr auf Port 1024 oder weniger, akzeptiert jedoch Datenverkehr auf einer Portnummer, die größer als 1024 ist. Sie fragen beispielsweise die Envoy-Statistiken mit dem folgenden Befehl ab und erhalten einen anderen Wert als Null.

```
curl -s http://my-app.default.svc.cluster.local:9901/stats | grep "update_rejected"
```

Möglicherweise wird in Ihren Protokollen Text ähnlich dem folgenden Text angezeigt, in dem ein Fehler beim Binden an einen privilegierten Port beschrieben wird:

```
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
```

Auflösung

Um das Problem zu beheben, muss der für das Gateway angegebene Benutzer über die Linux-Funktion verfügen `CAP_NET_BIND_SERVICE`. Weitere Informationen finden Sie unter [Funktionen](#) im Linux-Programmierhandbuch, [unter Linux-Parameter](#) in ECS-Aufgabendefinitionsparametern und [unter Festlegen von Funktionen für einen Container](#) in der Kubernetes-Dokumentation.

Important

Fargate muss einen Portwert größer als 1024 verwenden.

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Fehlerbehebung bei der App-Mesh-Konnektivität

In diesem Thema werden häufige Probleme beschrieben, die bei der App-Mesh-Konnektivität auftreten können.

DNS-Name für einen virtuellen Service kann nicht aufgelöst werden

Symptome

Ihre Anwendung kann den DNS-Namen eines virtuellen Services, mit dem sie eine Verbindung herstellen möchte, nicht auflösen.

Auflösung

Dies ist ein bekanntes Problem. Weitere Informationen finden Sie unter [Name VirtualServices nach einem beliebigen Hostnamen- oder FQDN](#) GitHub -Problem. Virtuelle Services in App Mesh können beliebig benannt werden. Solange es einen DNS-ADatensatz für den Namen des virtuellen Services gibt und die Anwendung den Namen des virtuellen Services auflösen kann, wird die Anforderung von Envoy als Proxy gesendet und an das entsprechende Ziel weitergeleitet. Um das Problem zu beheben, fügen Sie einen DNS-ADatensatz zu jeder Nicht-Loopback-IP-Adresse hinzu,

z. B. `10.10.10.10` für den Namen des virtuellen Services. Der DNS-ADatensatz kann unter den folgenden Bedingungen hinzugefügt werden:

- In Amazon Route 53, wenn der Name durch Ihren privaten gehosteten Zonennamen mit Suffix versehen ist
- Innerhalb der `/etc/hosts` Datei des Anwendungscontainers
- Auf einem DNS-Server eines Drittanbieters, den Sie verwalten

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Es konnte keine Verbindung zu einem virtuellen Service-Backend hergestellt werden

Symptome

Ihre Anwendung kann keine Verbindung zu einem virtuellen Service herstellen, der als Backend auf Ihrem virtuellen Knoten definiert ist. Beim Versuch, eine Verbindung herzustellen, schlägt die Verbindung möglicherweise vollständig fehl, oder die Anforderung aus Sicht der Anwendung schlägt möglicherweise mit einem HTTP 503 Antwortcode fehl.

Auflösung

Wenn die Anwendung überhaupt keine Verbindung herstellen kann (kein HTTP 503 Antwortcode zurückgegeben), gehen Sie wie folgt vor:

- Stellen Sie sicher, dass Ihre Datenverarbeitungsumgebung für die Arbeit mit App Mesh eingerichtet wurde.
 - Stellen Sie für Amazon ECS sicher, dass Sie die entsprechende [Proxy-Konfiguration](#) aktiviert haben. Eine end-to-end exemplarische Vorgehensweise finden Sie unter [Erste Schritte mit App Mesh und Amazon ECS](#).
 - Stellen Sie für Kubernetes, einschließlich Amazon EKS, sicher, dass Sie den neuesten App-Mesh-Controller über Helm installiert haben. Weitere Informationen finden Sie unter [App Mesh Controller](#) auf Helm Hub oder [Tutorial: Konfigurieren der App Mesh-Integration mit Kubernetes](#).
 - Stellen Sie für Amazon EC2 sicher, dass Sie Ihre Amazon EC2-Instance für den Proxy von App-Mesh-Datenverkehr eingerichtet haben. Weitere Informationen finden Sie unter [Services aktualisieren](#).

- Stellen Sie sicher, dass der Envoy-Container, der auf Ihrem Datenverarbeitungsservice ausgeführt wird, erfolgreich eine Verbindung zum App Mesh Envoy-Verwaltungsservice hergestellt hat. Sie können dies bestätigen, indem Sie Envoy-Statistiken für das Feld überprüfen `control_plane.connected_state`. Weitere Informationen zu `control_plane.connected_state` finden Sie unter [Überwachen der Envoy-Proxy-Konnektivität](#) in unserem Leitfaden zur Fehlerbehebung bei bewährten Methoden.

Wenn Envoy die Verbindung zunächst herstellen konnte, später aber getrennt und nie wieder verbunden wurde, finden Sie unter [Envoy hat die Verbindung zum App Mesh Envoy Management Service mit Fehlertext getrennt](#), um zu beheben, warum die Verbindung getrennt wurde.

Wenn die Anwendung eine Verbindung herstellt, die Anforderung jedoch mit einem HTTP 503 Antwortcode fehlschlägt, versuchen Sie Folgendes:

- Stellen Sie sicher, dass der virtuelle Service, mit dem Sie eine Verbindung herstellen, im Gitter vorhanden ist.
- Stellen Sie sicher, dass der virtuelle Service über einen Anbieter verfügt (einen virtuellen Router oder virtuellen Knoten).
- Wenn Sie Envoy als HTTP-Proxy verwenden und ausgehenden Datenverkehr über Envoy-Statistiken in `cluster.cds_egress*_mesh-allow-all` anstelle des richtigen Ziels sehen, leitet Envoy Anfragen höchstwahrscheinlich nicht ordnungsgemäß über `weiterfilter_chains`. Dies kann das Ergebnis der Verwendung eines unqualifizierten virtuellen Servicenamens sein. Wir empfehlen Ihnen, den Service Discovery-Namen des tatsächlichen Services als virtuellen Servicenamen zu verwenden, da der Envoy-Proxy über seine Namen mit anderen virtuellen Services kommuniziert.

Weitere Informationen finden Sie unter [Virtuelle Services](#).

- Überprüfen Sie die Envoy-Proxy-Protokolle auf eine der folgenden Fehlermeldungen:
 - `No healthy upstream` – Der virtuelle Knoten, an den der Envoy-Proxy zu leiten versucht, hat keine aufgelösten Endpunkte oder hat keine fehlerfreien Endpunkte. Stellen Sie sicher, dass der virtuelle Zielknoten über die richtigen Einstellungen für Service Discovery und Zustandsprüfung verfügt.

Wenn Anfragen an den Service während einer Bereitstellung oder Skalierung des virtuellen Backend-Services fehlschlagen, befolgen Sie die Anweisungen unter [Einige Anfragen schlagen mit HTTP-Statuscode fehl503, wenn ein virtueller Service über einen virtuellen Knotenanbieter verfügt](#).

- `No cluster match for URL` – Dies wird höchstwahrscheinlich verursacht, wenn eine Anforderung an einen virtuellen Service gesendet wird, der nicht den von einer der Routen definierten Kriterien entspricht, die unter einem virtuellen Routeranbieter definiert sind. Stellen Sie sicher, dass die Anfragen von der Anwendung an eine unterstützte Route gesendet werden, indem Sie sicherstellen, dass der Pfad und die HTTP-Anfrage-Header korrekt sind.
- `No matching filter chain found` – Dies wird höchstwahrscheinlich verursacht, wenn eine Anforderung an einen virtuellen Service auf einem ungültigen Port gesendet wird. Stellen Sie sicher, dass die Anforderungen von der Anwendung denselben Port verwenden, der auf dem virtuellen Router angegeben ist.

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Es konnte keine Verbindung zu einem externen Service hergestellt werden

Symptome

Ihre Anwendung kann keine Verbindung zu einem Service außerhalb des Gitters herstellen, z. B. `amazon.com`.

Auflösung

Standardmäßig lässt App Mesh keinen ausgehenden Datenverkehr von Anwendungen innerhalb des Gitters zu einem Ziel außerhalb des Gitters zu. Um die Kommunikation mit einem externen Service zu ermöglichen, gibt es zwei Möglichkeiten:

- Legen Sie den [ausgehenden Filter](#) für die Mesh-Ressource auf `festALLOW_ALL`. Diese Einstellung ermöglicht es jeder Anwendung innerhalb des Gitters, mit jeder Ziel-IP-Adresse innerhalb oder außerhalb des Gitters zu kommunizieren.
- Modellieren Sie den externen Service im Gitter mit einem virtuellen Service, virtuellen Router, Route und virtuellen Knoten. Um beispielsweise den externen Service zu modellieren `example.com`, können Sie einen virtuellen Service `example.com` namens mit einem virtuellen Router und einer Route erstellen, die den gesamten Datenverkehr an einen virtuellen Knoten mit dem Hostnamen der DNS-Serviceerkennung `sendexample.com`.

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Verbindung zu einem MySQL- oder SMTP-Server nicht möglich

Symptome

Wenn Sie ausgehenden Datenverkehr zu allen Zielen zulassen (Mesh EgressFilter `type=ALLOW_ALL`), z. B. zu einem SMTP-Server oder einer MySQL-Datenbank mithilfe einer virtuellen Knotendefinition, schlägt die Verbindung von Ihrer Anwendung fehl. Im Folgenden finden Sie beispielsweise eine Fehlermeldung beim Versuch, eine Verbindung zu einem MySQL-Server herzustellen.

```
ERROR 2013 (HY000): Lost connection to MySQL server at 'reading initial communication packet', system error: 0
```

Auflösung

Dies ist ein bekanntes Problem, das mit App Mesh-Image-Version 1.15.0 oder höher behoben wird. Weitere Informationen finden Sie unter dem Problem [Verbindung zu MySQL mit App Mesh nicht möglich](#) GitHub . Dieser Fehler tritt auf, weil der von App Mesh konfigurierte ausgehende Listener in Envoy den Envoy TLS Inspector Listener-Filter hinzufügt. Weitere Informationen finden Sie unter [TLS Inspector](#) in der Envoy-Dokumentation. Dieser Filter bewertet, ob eine Verbindung TLS verwendet, indem das erste vom Client gesendete Paket untersucht wird. Bei MySQL und SMTP sendet der Server jedoch das erste Paket nach der Verbindung. Weitere Informationen zu MySQL finden Sie unter [Anfänglicher Handshake](#) in der MySQL-Dokumentation. Da der Server das erste Paket sendet, schlägt die Überprüfung am Filter fehl.

So umgehen Sie dieses Problem je nach Version von Envoy:

- Wenn Ihre App-Mesh-Image-Envoy-Version 1.15.0 oder höher ist, modellieren Sie keine externen Services wie MySQL, SMTP, MSSQL usw. als Backend für den virtuellen Knoten Ihrer Anwendung.
- Wenn Ihre App-Mesh-Image-Envoy-Version vor 1.15.0 liegt, fügen Sie Port 3306 zur Liste der Werte für `APPMESH_EGRESS_IGNORED_PORTS` in Ihren Services für MySQL und als Port hinzu, den Sie für SMTP verwenden.

Important

Während die Standard-SMTP-Ports 25, und sind 465, sollten Sie nur den Port 587 hinzufügen, den Sie verwenden, `APPMESH_EGRESS_IGNORED_PORTS` und nicht alle drei.

Weitere Informationen finden Sie unter [Services für Kubernetes aktualisieren](#), [Services für Amazon ECS aktualisieren](#) oder [Services für Amazon EC2 aktualisieren](#). Amazon EC2

Wenn Ihr Problem immer noch nicht behoben ist, können Sie uns Details dazu zur Verfügung stellen, was Sie bei der Verwendung des vorhandenen [GitHub Problems](#) feststellen, oder sich an den [AWS - Support](#) wenden.

Es konnte keine Verbindung zu einem Service hergestellt werden, der als virtueller TCP-Knoten oder virtueller Router in App Mesh modelliert wurde.

Symptome

Ihre Anwendung kann keine Verbindung zu einem Backend herstellen, das die TCP-Protokolleinstellung in der App Mesh-[PortMapping](#) Definition verwendet.

Auflösung

Dies ist ein bekanntes Problem. Weitere Informationen finden Sie unter [Routing zu mehreren TCP-Zielen auf demselben Port](#) auf GitHub. App Mesh erlaubt derzeit nicht, dass mehrere als TCP modellierte Backend-Ziele denselben Port verwenden, da die Informationen, die dem Envoy-Proxy auf OSI Layer 4 bereitgestellt werden, eingeschränkt sind. Gehen Sie wie folgt vor, um sicherzustellen, dass TCP-Datenverkehr für alle Backend-Ziele ordnungsgemäß weitergeleitet werden kann:

- Stellen Sie sicher, dass alle Ziele einen eindeutigen Port verwenden. Wenn Sie einen virtuellen Routeranbieter für den virtuellen Backend-Service verwenden, können Sie den virtuellen Router-Port ändern, ohne den Port auf den virtuellen Knoten zu ändern, an die er weitergeleitet wird. Auf diese Weise können die Anwendungen Verbindungen am virtuellen Router-Port öffnen, während der Envoy-Proxy weiterhin den im virtuellen Knoten definierten Port verwendet.
- Wenn das als TCP modellierte Ziel ein MySQL-Server oder ein anderes TCP-basiertes Protokoll ist, in dem der Server die ersten Pakete nach der Verbindung sendet, finden Sie weitere Informationen unter [Verbindung zu einem MySQL- oder SMTP-Server nicht möglich](#).

Wenn Ihr Problem immer noch nicht behoben ist, können Sie uns Details dazu bereitstellen, was Sie bei der Verwendung des vorhandenen [GitHub Problems](#) feststellen, oder sich an den [AWS -Support](#) wenden.

Die Verbindung zu einem Service, der nicht als virtuelles Service-Backend für einen virtuellen Knoten aufgeführt ist, ist erfolgreich

Symptome

Ihre Anwendung kann eine Verbindung herstellen und Datenverkehr an ein Ziel senden, das nicht als virtuelles Service-Backend auf Ihrem virtuellen Knoten angegeben ist.

Auflösung

Wenn Anforderungen an ein Ziel erfolgreich sind, das in den App Mesh-APIs nicht modelliert wurde, besteht die wahrscheinlichste Ursache darin, dass der [ausgehende Filtertyp](#) des Gitters auf festgelegt wurde `ALLOW_ALL`. Wenn der ausgehende Filter auf festgelegt ist `ALLOW_ALL`, wird eine ausgehende Anfrage von Ihrer Anwendung, die nicht mit einem modellierten Ziel (Backend) übereinstimmt, an die von der Anwendung festgelegte Ziel-IP-Adresse gesendet.

Wenn Sie den Datenverkehr zu Zielen verweigern möchten, die nicht im Gitter modelliert sind, sollten Sie den Wert des ausgehenden Filters auf `setzen` `DROP_ALL`.

Note

Das Festlegen des ausgehenden Filterwerts für das Gitter wirkt sich auf alle virtuellen Knoten innerhalb des Gitters aus.

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Einige Anfragen schlagen mit HTTP-Statuscode fehl **503**, wenn ein virtueller Service über einen virtuellen Knotenanbieter verfügt

Symptome

Ein Teil der Anforderungen Ihrer Anwendung schlägt an ein virtuelles Service-Backend fehl, das einen virtuellen Knotenanbieter anstelle eines virtuellen Routeranbieters verwendet. Wenn Sie einen virtuellen Routeranbieter für den virtuellen Service verwenden, schlagen Anforderungen nicht fehl.

Auflösung

Dies ist ein bekanntes Problem. Weitere Informationen finden Sie unter [Wiederholungsrichtlinie für den Anbieter virtueller Knoten für einen virtuellen Service](#) auf GitHub. Wenn Sie einen virtuellen Knoten als Anbieter für einen virtuellen Service verwenden, können Sie nicht die Standard-Wiederholungsrichtlinie angeben, die die Clients Ihres virtuellen Services verwenden sollen. Im Vergleich dazu ermöglichen virtuelle Routeranbieter die Angabe von Wiederholungsrichtlinien, da sie eine Eigenschaft der untergeordneten Routenressourcen sind.

Um Anforderungsfehler an virtuelle Knotenanbieter zu reduzieren, verwenden Sie stattdessen einen virtuellen Routeranbieter und geben Sie eine Wiederholungsrichtlinie für ihre Routen an. Weitere Möglichkeiten zur Reduzierung von Anforderungsfehlern für Ihre Anwendungen finden Sie unter [Bewährte Methoden für App Mesh](#).

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Es konnte keine Verbindung zu einem Amazon-EFS-Dateisystem hergestellt werden

Symptome

Beim Konfigurieren einer Amazon-ECS-Aufgabe mit einem Amazon-EFS-Dateisystem als Volume kann die Aufgabe nicht mit dem folgenden Fehler gestartet werden.

```
ResourceInitializationError: failed to invoke EFS utils commands to set up EFS volumes:
  stderr: mount.nfs4: Connection refused : unsuccessful EFS utils command execution;
  code: 32
```

Auflösung

Dies ist ein bekanntes Problem. Dieser Fehler tritt auf, weil die NFS-Verbindung zu Amazon EFS auftritt, bevor Container in Ihrer Aufgabe gestartet werden. Dieser Datenverkehr wird von der Proxy-Konfiguration an Envoy weitergeleitet, das zu diesem Zeitpunkt nicht ausgeführt wird. Aufgrund der Reihenfolge des Startups kann der NFS-Client keine Verbindung zum Amazon-EFS-Dateisystem herstellen und die Aufgabe kann nicht gestartet werden. Um das Problem 2049 zu beheben, fügen Sie der Liste der Werte für die `EgressIgnoredPorts` Einstellung in der Proxy-Konfiguration Ihrer Amazon-ECS-Aufgabendefinition Port hinzu. Weitere Informationen finden Sie unter [Proxy-Konfiguration](#).

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Die Verbindung zum -Service ist erfolgreich, aber die eingehende Anforderung wird nicht in Zugriffsprotokollen, Ablaufverfolgungen oder Metriken für Envoy angezeigt

Symptome

Obwohl Ihre Anwendung eine Verbindung herstellen und Anforderungen an eine andere Anwendung senden kann, können Sie entweder keine eingehenden Anforderungen in den Zugriffsprotokollen oder in den Nachverfolgungsinformationen für den Envoy-Proxy sehen.

Auflösung

Dies ist ein bekanntes Problem. Weitere Informationen finden Sie unter Problem mit der [Einrichtung von iptables-Regeln](#) auf Github. Der Envoy-Proxy fängt nur eingehenden Datenverkehr an den Port ab, von dem der entsprechende virtuelle Knoten überwacht wird. Anforderungen an einen anderen Port umgehen den Envoy-Proxy und erreichen direkt den Service dahinter. Damit der Envoy-Proxy den eingehenden Datenverkehr für Ihren Service abfangen kann, müssen Sie Ihren virtuellen Knoten und Service so einstellen, dass er auf demselben Port überwacht.

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Das Festlegen der Umgebungsvariablen **HTTP_PROXY/HTTPS_PROXY** auf Containerebene funktioniert nicht wie erwartet.

Symptome

Wenn HTTP_PROXY/HTTPS_PROXY als Umgebungsvariable an der folgenden Stelle festgelegt ist:

- App-Container in der Aufgabendefinition mit aktiviertem App Mesh, Anfragen, die an den Namespace der App-Mesh-Services gesendet werden, erhalten HTTP 500 Fehlermeldungen vom Envoy-Sidecar.
- Envoy container in der Aufgabendefinition mit aktiviertem App Mesh, Anfragen, die aus dem Envoy-Sidecar kommen, werden nicht über den HTTP/HTTPS-Proxy-Server geleitet und die Umgebungsvariable funktioniert nicht.

Auflösung

Für den App-Container:

App-Mesh-Funktionen, indem der Datenverkehr innerhalb Ihrer Aufgabe durch den Envoy-Proxy geleitet wird. Die HTTP_PROXY/HTTPS_PROXY-Konfiguration überschreibt dieses Verhalten, indem der Container-Datenverkehr so konfiguriert wird, dass er über einen anderen externen Proxy geleitet wird. Der Datenverkehr wird weiterhin von Envoy abgefangen, unterstützt jedoch nicht das Proxying des Mesh-Datenverkehrs mit einem externen Proxy.

Wenn Sie den gesamten Nicht-Mesh-Datenverkehr leiten möchten, legen Sie so fest, NO_PROXY dass er den CIDR/Namespace Ihres Gitters, den Localhost und die Endpunkte der Anmeldeinformationen wie im folgenden Beispiel enthält.

```
NO_PROXY=localhost,127.0.0.1,169.254.169.254,169.254.170.2,10.0.0.0/16
```

Für den Envoy-Container:

Envoy unterstützt keinen generischen Proxy. Wir empfehlen nicht, diese Variablen festzulegen.

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Upstream-Anforderungs-Timeouts auch nach Einstellung des Timeouts für Routen.

Symptome

Sie haben das Timeout für Folgendes definiert:

- Die Routen, aber Sie erhalten immer noch einen Upstream-Anforderungs-Timeout-Fehler.
- Der Listener des virtuellen Knotens sowie das Timeout und das Timeout für Wiederholungsversuche für die Routen, aber Sie erhalten immer noch einen Upstream-Anforderungs-Timeout-Fehler.

Auflösung

Damit Anforderungen mit hoher Latenz von mehr als 15 Sekunden erfolgreich abgeschlossen werden können, müssen Sie ein Timeout sowohl auf Routen- als auch auf virtueller Knoten-Listener-Ebene angeben.

Wenn Sie ein Routen-Timeout angeben, das größer als die standardmäßigen 15 Sekunden ist, stellen Sie sicher, dass das Timeout auch für den Listener für alle teilnehmenden virtuellen Knoten angegeben ist. Wenn Sie das Timeout jedoch auf einen Wert verringern, der niedriger als der Standardwert ist, ist es optional, die Timeouts auf virtuellen Knoten zu aktualisieren. Weitere Informationen zu Optionen beim Einrichten virtueller Knoten und Routen finden Sie unter [Virtuelle Knoten](#) und [Routen](#).

Wenn Sie eine Wiederholungsrichtlinie angegeben haben, sollte die Dauer, die Sie für das Anforderungs-Timeout angeben, immer größer oder gleich dem Wiederholungs-Timeout sein, multipliziert mit den maximalen Wiederholungen, die Sie in der Wiederholungsrichtlinie definiert haben. Auf diese Weise kann Ihre Anfrage mit allen Wiederholungen erfolgreich abgeschlossen werden. Weitere Informationen finden Sie unter [Routen](#).

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Envoy antwortet mit einer HTTP-Bad-Anforderung.

Symptome

Envoy antwortet mit einer HTTP 400 Bad-Anforderung für alle Anforderungen, die über den Network Load Balancer (NLB) gesendet werden. Wenn wir die Envoy-Protokolle überprüfen, sehen wir:

- Debuggen von Protokollen:

```
dispatch error: http/1.1 protocol error: HPE_INVALID_METHOD
```

- Zugriffsprotokolle:

```
"- - HTTP/1.1" 400 DPE 0 11 0 - "-" "-" "-" "-" "-"
```

Auflösung

Die Auflösung besteht darin, das Proxy-Protokoll Version 2 (PPv2) für die [Zielgruppenattribute](#) Ihres NLB zu deaktivieren.

Seit heute wird PPv2 vom virtuellen Gateway und vom virtuellen Knoten Envoy, die mit der App Mesh-Steuerebene ausgeführt werden, nicht unterstützt. Wenn Sie NLB mit dem AWS Load Balancer Controller auf Kubernetes bereitstellen, deaktivieren Sie PPv2, indem Sie das folgende Attribut auf `false` setzen:

```
service.beta.kubernetes.io/aws-load-balancer-target-group-attributes:  
  proxy_protocol_v2.enabled
```

Weitere Informationen zu NLB-Ressourcenattributen finden Sie unter [AWS Load Balancer Controller Annotations](#).

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Das Timeout kann nicht richtig konfiguriert werden.

Symptome

Ihre Anfrage wird innerhalb von 15 Sekunden beendet, auch nachdem das Timeout auf dem Listener des virtuellen Knotens und das Timeout auf der Route zum Backend des virtuellen Knotens konfiguriert wurde.

Auflösung

Stellen Sie sicher, dass der richtige virtuelle Service in der Backend-Liste enthalten ist.

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Fehlerbehebung bei der App Mesh-Skalierung

In diesem Thema werden häufige Probleme beschrieben, die bei der App Mesh-Skalierung auftreten können.

Konnektivität schlägt fehl und Container-Zustandsprüfungen schlagen fehl, wenn die Skalierung über 50 Replikat für einen virtuellen Knoten/ein virtuelles Gateway hinaus skaliert wird

Symptome

Wenn Sie die Anzahl der Replikat, z. B. Amazon-ECS-Aufgaben, Kubernetes-Pods oder Amazon EC2-Instances, für einen virtuellen Knoten/ein virtuelles Gateway über 50 hinaus skalieren, schlagen die Zustandsprüfungen des Envoy-Containers für neue und aktuell ausgeführte Envoys fehl.

Downstream-Anwendungen, die Datenverkehr an den virtuellen Knoten/das virtuelle Gateway senden, sehen mit dem HTTP-Statuscode Anforderungsfehler503.

Auflösung

Das Standardkontingent von App Mesh für die Anzahl der Envoy's pro virtuellem Knoten/virtuellem Gateway beträgt 50. Wenn die Anzahl der laufenden Envoy's dieses Kontingent überschreitet, können neue und aktuell ausgeführte Envoy's keine Verbindung zum Envoy Management Service von App Mesh mit dem gRPC-Statuscode 8 () herstellenRESOURCE_EXHAUSTED. Dieses Kontingent kann erhöht werden. Weitere Informationen finden Sie unter [App Mesh service quotas](#).

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Anforderungen schlagen mit fehl503, wenn ein virtuelles Service-Backend horizontal auf- oder abskaliert wird

Symptome

Wenn ein virtueller Backend-Service horizontal auf- oder abskaliert wird, schlagen Anforderungen von nachgelagerten Anwendungen mit einem -HTTP 503Statuscode fehl.

Auflösung

App Mesh empfiehlt mehrere Ansätze zur Minderung von Fehlerfällen bei der horizontalen Skalierung von Anwendungen. Ausführliche Informationen zur Vermeidung dieser Fehler finden Sie unter [Bewährte Methoden für App Mesh](#).

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Envoy-Container stürzt bei erhöhter Last mit segfault ab

Symptome

Bei einer hohen Datenverkehrslast stürzt der Envoy-Proxy aufgrund eines Segmentierungsfehlers ab (Linux-Beendigungscode 139). Die Envoy-Prozessprotokolle enthalten eine Anweisung wie die folgende.

```
Caught Segmentation fault, suspect faulting address 0x0"
```

Auflösung

Der Envoy-Proxy hat wahrscheinlich das Standard-Ulimit für nofile des Betriebssystems überschritten, das Limit für die Anzahl der Dateien, die ein Prozess gleichzeitig geöffnet haben kann. Diese Verletzung ist auf den Datenverkehr zurückzuführen, der mehr Verbindungen verursacht, die zusätzliche Betriebssystem-Sockets verbrauchen. Um dieses Problem zu beheben, erhöhen Sie den Wert für ulimit nofile auf dem Host-Betriebssystem. Wenn Sie Amazon ECS verwenden, kann dieses Limit über die [Ulimit-Einstellungen](#) in den [Ressourcenlimit-Einstellungen](#) der Aufgabendefinition geändert werden.

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Die Erhöhung der Standardressourcen spiegelt sich nicht in den Service Limits wider

Symptome

Nachdem Sie das Standardlimit von App Mesh-Ressourcen erhöht haben, wird der neue Wert nicht wiedergegeben, wenn Sie sich Ihre Servicelimits ansehen.

Auflösung

Die neuen Limits werden derzeit nicht angezeigt, Kunden können sie jedoch weiterhin trainieren.

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Die Anwendung stürzt aufgrund einer großen Anzahl von Aufrufen für Zustandsprüfungen ab.

Symptome

Nachdem Sie aktive Zustandsprüfungen für einen virtuellen Knoten aktiviert haben, gibt es eine Erhöhung der Anzahl der Zustandsprüfungsaufrufe. Die Anwendung stürzt aufgrund der stark erhöhten Anzahl von Aufrufen für Zustandsprüfungen ab, die an die Anwendung getätigt wurden.

Auflösung

Wenn die aktive Zustandsprüfung aktiviert ist, sendet jeder Envoy-Endpunkt des Downstreams (Clients) Zustandsanforderungen an jeden Endpunkt des Upstream-Clusters (Servers), um Routing-

Entscheidungen zu treffen. Daher wäre die Gesamtzahl der Anforderungen für Zustandsprüfungen $\text{number of client Envoys} * \text{number of server Envoys} * \text{active health check frequency}$.

Um dieses Problem zu beheben, ändern Sie die Häufigkeit der Zustandsprüfungs-Sonde, was das Gesamtvolumen der Zustandsprüfungs-Sonden reduzieren würde. Zusätzlich zu den aktiven Zustandsprüfungen ermöglicht App Mesh die Konfiguration der [Ausreißerererkennung](#) als Mittel der passiven Zustandsprüfung. Verwenden Sie die Ausreißerererkennung, um zu konfigurieren, wann ein bestimmter Host basierend auf aufeinanderfolgenden 5xx Antworten entfernt werden soll.

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Fehlerbehebung bei der Beobachtbarkeit von App Mesh

In diesem Thema werden häufige Probleme beschrieben, die bei der Beobachtbarkeit von App Mesh auftreten können.

AWS X-Ray Ablaufverfolgungen für meine Anwendungen konnten nicht angezeigt werden

Symptome

Ihre Anwendung in App Mesh zeigt keine X-Ray-Ablaufverfolgungsinformationen in der X-Ray-Konsole oder APIs APIs.

Auflösung

Um X-Ray in App Mesh verwenden zu können, müssen Sie Komponenten korrekt konfigurieren, um die Kommunikation zwischen Ihrer Anwendung, Sidecar-Containern und dem X-Ray-Service zu ermöglichen. Führen Sie die folgenden Schritte aus, um zu bestätigen, dass X-Ray korrekt eingerichtet wurde:

- Stellen Sie sicher, dass das App Mesh Virtual Node Listener-Protokoll nicht auf festgelegt ist TCP.
- Stellen Sie sicher, dass der X-Ray-Container, der mit Ihrer Anwendung bereitgestellt wird, den UDP-Port freigibt 2000 und als Benutzer ausgeführt wird 1337. Weitere Informationen finden Sie im [Amazon-ECS-X-Ray-Beispiel](#) auf GitHub.
- Stellen Sie sicher, dass für den Envoy-Container die Nachverfolgung aktiviert ist. Wenn Sie das [App Mesh Envoy-Image](#) verwenden, können Sie X-Ray aktivieren, indem Sie

die `ENABLE_ENVOY_XRAY_TRACING` Umgebungsvariable auf einen Wert von 1 und die `XRAY_DAEMON_PORT` Umgebungsvariable auf `2000` setzen.

- Wenn Sie X-Ray in Ihrem Anwendungscode mit einem der [sprachspezifischen SDKs](#) instrumentiert haben, stellen Sie sicher, dass es korrekt konfiguriert ist, indem Sie den Anleitungen für Ihre Sprache folgen.
- Wenn alle vorherigen Elemente korrekt konfiguriert sind, überprüfen Sie die X-Ray-Containerprotokolle auf Fehler und befolgen Sie die Anweisungen unter [Fehlerbehebung. AWS X-Ray](#). Eine ausführlichere Erläuterung der X-Ray-Integration in App Mesh finden Sie unter [Integrieren von X-Ray mit App Mesh](#).

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Envoy-Metriken für meine Anwendungen können in Amazon- CloudWatch Metriken nicht angezeigt werden

Symptome

Ihre Anwendung in App Mesh gibt keine vom Envoy-Proxy generierten Metriken an CloudWatch Metriken aus.

Auflösung

Wenn Sie CloudWatch Metriken in App Mesh verwenden, müssen Sie mehrere Komponenten korrekt konfigurieren, um die Kommunikation zwischen Ihrem Envoy-Proxy, dem CloudWatch Kundendienstmitarbeiter-Sidecar und dem CloudWatch Metrikservice zu ermöglichen. Führen Sie die folgenden Schritte aus, um zu bestätigen, dass die CloudWatch Metriken für den Envoy-Proxy korrekt eingerichtet wurden:

- Stellen Sie sicher, dass Sie das CloudWatch Agenten-Image für App Mesh verwenden. Weitere Informationen finden Sie unter [App- CloudWatchMesh-Agent](#) auf GitHub.
- Stellen Sie sicher, dass Sie den CloudWatch Agenten für App Mesh entsprechend konfiguriert haben, indem Sie die plattformspezifischen Nutzungsanweisungen befolgen. Weitere Informationen finden Sie unter [App- CloudWatchMesh-Agent](#) auf GitHub.
- Wenn alle vorherigen Elemente korrekt konfiguriert sind, überprüfen Sie die CloudWatch Agent-Container-Protokolle auf Fehler und befolgen Sie die Anweisungen unter [Fehlerbehebung für den CloudWatch Agenten](#).

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Benutzerdefinierte Samplingregeln für AWS X-Ray Ablaufverfolgungen können nicht konfiguriert werden

Symptome

Ihre Anwendung verwendet X-Ray-Ablaufverfolgung, aber Sie können keine Samplingregeln für Ihre Ablaufverfolgungen konfigurieren.

Auflösung

Da App Mesh Envoy derzeit keine dynamische X-Ray-Sampling-Konfiguration unterstützt, sind die folgenden Problemumgehungen verfügbar.

Wenn Ihre Envoy-Version 1.19.1 oder höher ist, haben Sie die folgenden Optionen.

- Um nur die Abtastrate festzulegen, verwenden Sie die `XRAY_SAMPLING_RATE` Umgebungsvariable im Envoy-Container. Der Wert sollte als Dezimalzahl zwischen 0 und 1.00 (100 %) angegeben werden. Weitere Informationen finden Sie unter [AWS X-Ray Variablen](#).
- Um die lokalisierten benutzerdefinierten Samplingregeln für den X-Ray-Tracer zu konfigurieren, verwenden Sie die `XRAY_SAMPLING_RULE_MANIFEST` Umgebungsvariable, um einen Dateipfad im Envoy-Container-Dateisystem anzugeben. Weitere Informationen finden Sie unter [Samplingregeln](#) im AWS X-Ray -Entwicklerhandbuch.

Wenn Ihre Envoy-Version vorliegt 1.19.1, gehen Sie wie folgt vor.

- Verwenden Sie die `ENVOY_TRACING_CFG_FILE` Umgebungsvariable, um Ihre Abtastrate zu ändern. Weitere Informationen finden Sie unter [Envoy-Konfigurationsvariablen](#). Geben Sie eine benutzerdefinierte Tracing-Konfiguration an und definieren Sie lokale Samplingregeln. Weitere Informationen finden Sie unter [Envoy X-Ray config](#).
- Beispiel für eine benutzerdefinierte Ablaufverfolgungskonfiguration für die `ENVOY_TRACING_CFG_FILE` Umgebungsvariable:

```
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
```

- Weitere Informationen zur Konfiguration für das Samplingregelmanifest in der `-samplingRuleManifest`Eigenschaft finden Sie unter [Konfigurieren des X-Ray-SDK für Go](#).

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Fehlerbehebung bei der App-Mesh-Sicherheit

In diesem Thema werden häufige Probleme beschrieben, die bei der App-Mesh-Sicherheit auftreten können.

Herstellen einer Verbindung zu einem virtuellen Backend-Service mit einer TLS-Client-Richtlinie nicht möglich

Symptome

Wenn Sie einem virtuellen Service-Backend in einem virtuellen Knoten eine TLS-Client-Richtlinie hinzufügen, schlägt die Konnektivität zu diesem Backend fehl. Beim Versuch, Datenverkehr an den Backend-Service zu senden, schlagen die Anforderungen mit einem HTTP 503 Antwortcode und der Fehlermeldung fehl: `upstream connect error or disconnect/reset before headers. reset reason: connection failure`.

Auflösung

Um die Ursache des Problems zu ermitteln, empfehlen wir, die Envoy-Proxy-Prozessprotokolle zu verwenden, um Ihnen bei der Diagnose des Problems zu helfen. Weitere Informationen finden Sie

unter [Aktivieren der Envoy-Debug-Protokollierung in Vorproduktionsumgebungen](#). Verwenden Sie die folgende Liste, um die Ursache des Verbindungsfehlers zu ermitteln:

- Stellen Sie sicher, dass die Verbindung zum Backend erfolgreich ist, indem Sie die unter genannten Fehler aufheben [Es konnte keine Verbindung zu einem virtuellen Service-Backend hergestellt werden](#).
- Suchen Sie in den Envoy-Prozessprotokollen nach den folgenden Fehlern (auf Debug-Ebene protokolliert).

```
TLS error: 268435581:SSL routines:OPENSSL_internal:CERTIFICATE_VERIFY_FAILED
```

Dieser Fehler wird durch einen oder mehrere der folgenden Gründe verursacht:

- Das Zertifikat wurde nicht von einer der Zertifizierungsstellen signiert, die im Vertrauenspaket für TLS-Client-Richtlinien definiert sind.
- Das Zertifikat ist nicht mehr gültig (abgelaufen).
- Der Subject Alternative Name (SAN) stimmt nicht mit dem angeforderten DNS-Hostnamen überein.
- Stellen Sie sicher, dass das vom Backend-Service angebotene Zertifikat gültig ist, dass es von einer der Zertifizierungsstellen in Ihren TLS-Clientrichtlinien als Vertrauenspaket signiert ist und dass es die in definierten Kriterien erfüllt [Transport Layer Security \(TLS\)](#).
- Wenn der Fehler, den Sie erhalten, mit dem unten angegebenen übereinstimmt, bedeutet dies, dass die Anforderung den Envoy-Proxy umgeht und die Anwendung direkt erreicht. Beim Senden von Datenverkehr ändern sich die Statistiken auf Envoy nicht, was darauf hinweist, dass Envoy nicht auf dem Weg ist, den Datenverkehr zu entschlüsseln. Stellen Sie in der Proxy-Konfiguration des virtuellen Knotens sicher, dass der den richtigen Wert `AppPorts` enthält, den die Anwendung überwacht.

```
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
```

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden. Wenn Sie glauben, dass Sie eine Sicherheitslücke gefunden haben oder Fragen zur Sicherheit von App Mesh haben, lesen Sie die [AWS Richtlinien zur Meldung von Schwachstellen](#).

Verbindung zu einem virtuellen Backend-Service nicht möglich, wenn die Anwendung TLS erzeugt

Symptome

Beim Ursprung einer TLS-Sitzung aus einer Anwendung und nicht aus dem Envoy-Proxy schlägt die Konnektivität zu einem virtuellen Backend-Service fehl.

Auflösung

Dies ist ein bekanntes Problem. Weitere Informationen finden Sie unter [Feature Request: TLS-Aushandlung zwischen der Downstream-Anwendung und dem Upstream-Proxy](#) GitHub -Problem. In App Mesh wird die TLS-Ursprungsfunktion derzeit vom Envoy-Proxy, aber nicht von der Anwendung unterstützt. Um die TLS-Ursprungsunterstützung in der Envoy zu verwenden, deaktivieren Sie die TLS-Ursprungsfunktion in der Anwendung. Auf diese Weise kann Envoy die ausgehenden Anfrage-Header lesen und die Anfrage über eine TLS-Sitzung an das entsprechende Ziel weiterleiten. Weitere Informationen finden Sie unter [Transport Layer Security \(TLS\)](#).

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden. Wenn Sie der Meinung sind, dass Sie eine Sicherheitslücke gefunden haben oder Fragen zur Sicherheit von App Mesh haben, lesen Sie die [AWS Richtlinien zur Meldung von Schwachstellen](#).

Es konnte nicht bestätigt werden, dass die Konnektivität zwischen Envoy-Proxys TLS verwendet

Symptome

Ihre Anwendung hat die TLS-Beendigung auf dem virtuellen Knoten oder dem Listener des virtuellen Gateways oder die TLS-Ursprungsfunktion auf der Backend-TLS-Clientrichtlinie aktiviert, aber Sie können nicht bestätigen, dass die Konnektivität zwischen Envoy-Proxys über eine TLS-verhandelte Sitzung erfolgt.

Auflösung

Die in dieser Auflösung definierten Schritte verwenden die Envoy-Verwaltungsschnittstelle und die Envoy-Statistiken. Hilfe bei der Konfiguration dieser finden Sie unter [Aktivieren der Envoy-Proxy-Verwaltungsschnittstelle](#) und [Aktivieren der Envoy DogStats-D-Integration für Metrik-Offload](#). In den folgenden Statistikbeispielen wird der Einfachheit halber die Verwaltungsschnittstelle verwendet.

- Für den Envoy-Proxy, der die TLS-Beendigung durchführt:
 - Stellen Sie sicher, dass das TLS-Zertifikat in der Envoy-Konfiguration mit dem folgenden Befehl gestartet wurde.

```
curl http://my-app.default.svc.cluster.local:9901/certs
```

In der zurückgegebenen Ausgabe sollte mindestens ein Eintrag unter `certificates[].cert_chain` für das Zertifikat angezeigt werden, das bei der TLS-Beendigung verwendet wird.

- Stellen Sie sicher, dass die Anzahl der erfolgreichen eingehenden Verbindungen zum Listener des Proxys genau der Anzahl der SSL-Handshakes plus der Anzahl der erneut verwendeten SSL-Sitzungen entspricht, wie in den folgenden Beispielbefehlen und -ausgaben gezeigt.

```
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)
```

- Für den Envoy-Proxy, der die TLS-Ursprungsfunktion ausführt:
 - Stellen Sie sicher, dass der TLS-Vertrauensspeicher in der Envoy-Konfiguration mit dem folgenden Befehl gestartet wurde.

```
curl http://my-app.default.svc.cluster.local:9901/certs
```

Sie sollten unter mindestens einen Eintrag `certificates[].ca_certs` für die Zertifikate sehen, die bei der Validierung des Backend-Zertifikats während der TLS-Ursprungsfunktion verwendet werden.

- Stellen Sie sicher, dass die Anzahl der erfolgreichen ausgehenden Verbindungen zum Backend-Cluster genau der Anzahl der SSL-Handshakes plus der Anzahl der erneut verwendeten SSL-Sitzungen entspricht, wie in den folgenden Beispielbefehlen und -ausgaben gezeigt.

```
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)
```

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden. Wenn Sie glauben, dass Sie eine Sicherheitslücke gefunden haben oder Fragen zur Sicherheit von App Mesh haben, lesen Sie die [AWS Richtlinien zur Meldung von Schwachstellen](#).

Fehlerbehebung bei TLS mit Elastic Load Balancing

Symptome

Wenn Sie versuchen, einen Application Load Balancer oder Network Load Balancer zum Verschlüsseln des Datenverkehrs zu einem virtuellen Knoten zu konfigurieren, können die Zustandsprüfungen für Konnektivität und Load Balancer fehlschlagen.

Auflösung

Um die Ursache des Problems zu ermitteln, müssen Sie Folgendes überprüfen:

- Für den Envoy-Proxy, der die TLS-Beendigung durchführt, müssen Sie jede Fehlkonfiguration ausschließen. Führen Sie die oben in der angegebenen Schritte aus [Herstellen einer Verbindung zu einem virtuellen Backend-Service mit einer TLS-Client-Richtlinie nicht möglich](#).

- Für den Load Balancer müssen Sie sich die Konfiguration des ansehen. TargetGroup:
 - Stellen Sie sicher, dass der TargetGroup Port mit dem definierten Listener-Port des virtuellen Knotens übereinstimmt.
 - Stellen Sie bei Application Load Balancern, die TLS-Verbindungen über HTTP zu Ihrem Service herstellen, sicher, dass das TargetGroup Protokoll auf festgelegt istHTTPS. Wenn Zustandsprüfungen verwendet werden, stellen Sie sicher, dass auf gesetzt HealthCheckProtocol istHTTPS.
 - Stellen Sie bei Network Load Balancern, die TLS-Verbindungen über TCP zu Ihrem Service herstellen, sicher, dass das TargetGroup Protokoll auf festgelegt istTLS. Wenn Zustandsprüfungen verwendet werden, stellen Sie sicher, dass auf gesetzt HealthCheckProtocol istTCP.

 Note

Alle Aktualisierungen von TargetGroup erfordern eine Änderung des TargetGroup Namens.

Wenn dies richtig konfiguriert ist, sollte Ihr Load Balancer eine sichere Verbindung zu Ihrem Service mithilfe des Zertifikats bereitstellen, das dem Envoy-Proxy zur Verfügung gestellt wird.

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden. Wenn Sie glauben, dass Sie eine Sicherheitslücke gefunden haben oder Fragen zur Sicherheit von App Mesh haben, lesen Sie die [AWS Richtlinien zur Meldung von Schwachstellen](#).

Fehlerbehebung bei App Mesh Kubernetes

In diesem Thema werden häufige Probleme beschrieben, die bei der Verwendung von App Mesh mit Kubernetes auftreten können.

In Kubernetes erstellte App-Mesh-Ressourcen können in App Mesh nicht gefunden werden

Symptome

Sie haben die App-Mesh-Ressourcen mit der benutzerdefinierten Ressourcendefinition (CRD) von Kubernetes erstellt, aber die von Ihnen erstellten Ressourcen sind in App Mesh nicht sichtbar, wenn Sie die - AWS Management Console oder -APIs verwenden.

Auflösung

Die wahrscheinliche Ursache ist ein Fehler im Kubernetes-Controller für App Mesh. Weitere Informationen finden Sie unter [Fehlerbehebung](#) auf GitHub. Überprüfen Sie die Controller-Protokolle auf Fehler oder Warnungen, die darauf hinweisen, dass der Controller keine Ressourcen erstellen konnte.

```
kubectl logs -n appmesh-system -f \  
$(kubectl get pods -n appmesh-system -o name | grep controller)
```

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Pods bestehen nach dem Einfügen des Envoy-Sidecars keine Bereitschafts- und Liveliness-Prüfungen

Symptome

Pods für Ihre Anwendung wurden zuvor erfolgreich ausgeführt, aber nachdem das Envoy-Sidecar in einen Pod eingefügt wurde, schlagen Bereitschafts- und Livelinessprüfungen fehl.

Auflösung

Stellen Sie sicher, dass der Envoy-Container, der in den Pod injiziert wurde, mit dem Envoy-Verwaltungsservice von App Mesh gestartet wurde. Sie können alle Fehler überprüfen, indem Sie auf die Fehlercodes in [verweisen](#) [Envoy wurde mit Fehlertext vom App Mesh Envoy Management Service getrennt](#). Sie können den folgenden Befehl verwenden, um Envoy-Protokolle auf den entsprechenden Pod zu überprüfen.

```
kubectl logs -n appmesh-system -f \  
$(kubectl get pods -n appmesh-system -o name | grep controller) \  
| grep "gRPC config stream closed"
```

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Pods, die sich nicht als Instances registrieren oder deren Registrierung aufheben AWS Cloud Map

Symptome

Ihre Kubernetes-Pods werden AWS Cloud Map im Rahmen ihres Lebenszyklus nicht in registriert oder von dieser abgemeldet. Ein Pod kann erfolgreich gestartet werden und bereit sein, Datenverkehr bereitzustellen, erhält aber keine. Wenn ein Pod beendet wird, behalten Clients möglicherweise immer noch seine IP-Adresse bei und versuchen, Datenverkehr an ihn zu senden, was fehlschlägt.

Auflösung

Dies ist ein bekanntes Problem. Weitere Informationen finden Sie unter [Pods werden in Kubernetes mit Problem nicht automatisch registriert/abgemeldet AWS Cloud Map](#) GitHub . Aufgrund der Beziehung zwischen Pods, virtuellen App-Mesh-Knoten und AWS Cloud Map Ressourcen kann der [App-Mesh-Controller für Kubernetes](#) desynchronisiert werden und Ressourcen verlieren. Dies kann beispielsweise passieren, wenn eine virtuelle Knotenressource aus Kubernetes gelöscht wird, bevor die zugehörigen Pods beendet werden.

Um dieses Problem zu beheben:

- Stellen Sie sicher, dass Sie die neueste Version des App-Mesh-Controllers für Kubernetes ausführen.
- Stellen Sie sicher, dass die AWS Cloud Map namespaceName und in Ihrer virtuellen Knotendefinition korrekt serviceName sind.
- Stellen Sie sicher, dass Sie alle zugehörigen Pods löschen, bevor Sie Ihre virtuelle Knotendefinition löschen. Wenn Sie Hilfe bei der Identifizierung benötigen, welche Pods einem virtuellen Knoten zugeordnet sind, finden Sie weitere Informationen unter [Es kann nicht festgestellt werden, wo ein Pod für eine App-Mesh-Ressource ausgeführt wird](#).
- Wenn Ihr Problem weiterhin besteht, führen Sie den folgenden Befehl aus, um Ihre Controller-Protokolle auf Fehler zu überprüfen, die dazu beitragen können, das zugrunde liegende Problem aufzudecken.

```
kubectl logs -n appmesh-system \  
$(kubectl get pods -n appmesh-system -o name | grep appmesh-controller)
```

- Erwägen Sie, den folgenden Befehl zu verwenden, um Ihre Controller-Pods neu zu starten. Dadurch können Synchronisationsprobleme behoben werden.

```
kubectl delete -n appmesh-system \  
$(kubectl get pods -n appmesh-system -o name | grep appmesh-controller)
```

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Es kann nicht festgestellt werden, wo ein Pod für eine App-Mesh-Ressource ausgeführt wird

Symptome

Wenn Sie App Mesh auf einem Kubernetes-Cluster ausführen, kann ein Operator nicht bestimmen, wo ein Workload oder Pod für eine bestimmte App-Mesh-Ressource ausgeführt wird.

Auflösung

Kubernetes-Pod-Ressourcen werden mit dem Gitter und dem virtuellen Knoten versehen, dem sie zugeordnet sind. Sie können mit dem folgenden Befehl abfragen, welche Pods für einen bestimmten virtuellen Knotennamen ausgeführt werden.

```
kubectl get pods --all-namespaces -o json | \  
jq '.items[] | { metadata } | select(.metadata.annotations."appmesh.k8s.aws/  
virtualNode" == "virtual-node-name")'
```

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Es kann nicht festgestellt werden, als welche App-Mesh-Ressource ein Pod ausgeführt wird

Symptome

Wenn App Mesh auf einem Kubernetes-Cluster ausgeführt wird, kann ein Operator nicht bestimmen, auf welcher App-Mesh-Ressource ein bestimmter Pod ausgeführt wird.

Auflösung

Kubernetes-Pod-Ressourcen werden mit dem Gitter und dem virtuellen Knoten versehen, dem sie zugeordnet sind. Sie können die Mesh- und virtuellen Knotennamen ausgeben, indem Sie den Pod direkt mit dem folgenden Befehl abfragen.

```
kubectl get pod pod-name -n namespace -o json | \
jq '{ "mesh": .metadata.annotations."appmesh.k8s.aws/mesh",
"virtualNode": .metadata.annotations."appmesh.k8s.aws/virtualNode" }'
```

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

Client-Envoys können nicht mit dem App Mesh Envoy Management Service kommunizieren, wenn IMDSv1 deaktiviert ist

Symptome

Wenn deaktiviert IMDSv1 ist, können Client-Envoys nicht mit der App-Mesh-Steuerebene (Envoy Management Service) kommunizieren. -IMDSv2 Unterstützung ist in der App-Mesh-Envoy-Version vor nicht verfügbar `v1.24.0.0-prod`.

Auflösung

Um dieses Problem zu beheben, können Sie einen dieser drei Schritte ausführen.

- Führen Sie ein Upgrade auf App Mesh Envoy Version `v1.24.0.0-prod` oder höher durch, die IMDSv2 Unterstützung bietet.
- Aktivieren Sie erneut IMDSv1 auf der Instance, in der Envoy ausgeführt wird. Anweisungen zum Wiederherstellen von finden Sie IMDSv1 unter [Konfigurieren der Instance-Metadatenoptionen](#).
- Wenn Ihre Services auf Amazon EKS ausgeführt werden, wird empfohlen, IAM-Rollen für Servicekonten (IRSA) zum Abrufen von Anmeldeinformationen zu verwenden. Anweisungen zum Aktivieren von IRSA finden Sie unter [IAM-Rollen für Servicekonten](#).

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

IRSA funktioniert nicht auf dem Anwendungscontainer, wenn App Mesh aktiviert ist und Envoy injiziert wird

Symptome

Wenn App Mesh auf einem Amazon-EKS-Cluster mithilfe des App-Mesh-Controllers für Amazon EKS aktiviert ist, werden Envoy und proxyinit Container in den Anwendungs-Pod injiziert. Die Anwendung kann nicht annehmen IRSA und übernimmt stattdessen `node` `role`. Wenn wir die Pod-Details beschreiben, sehen wir, dass entweder die `AWS_ROLE_ARN` Umgebungsvariable `AWS_WEB_IDENTITY_TOKEN_FILE` oder nicht im Anwendungscontainer enthalten sind.

Auflösung

Wenn entweder `-AWS_WEB_IDENTITY_TOKEN_FILE` oder `-AWS_ROLE_ARN` Umgebungsvariablen definiert sind, überspringt der Webhook den Pod. Geben Sie keine dieser Variablen an, und der Webhook kümmert sich um das Einfügen dieser Variablen für Sie.

```
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
    }
}
```

Wenn Ihr Problem immer noch nicht behoben ist, sollten Sie ein [GitHub Problem](#) öffnen oder sich an [AWS den -Support](#) wenden.

App Mesh Preview Channel

Der App Mesh Preview Channel ist eine eigenständige Variante des App Mesh Mesh-Dienstes, der in der us-west-2 Region bereitgestellt wird. Im Vorschaukanal werden kommende Funktionen vorgestellt, die Sie während der Entwicklung ausprobieren können. Wenn Sie Funktionen im Vorschaukanal verwenden, können Sie Feedback geben, GitHub um das Verhalten der Funktionen zu beeinflussen. Sobald ein Feature die vollständige Funktionalität im Vorschaukanal hat und alle erforderlichen Integrationen und Überprüfungen abgeschlossen sind, wird es in den App Mesh-Produktionsdienst umgestellt.

Der AWS App Mesh Vorschaukanal ist ein Beta-Service und alle Funktionen sind Vorschauen, wie diese Bedingungen in den [AWS Servicebedingungen](#) definiert sind. Ihre Teilnahme am Vorschaukanal unterliegt Ihrer Vereinbarung mit AWS und den AWS Servicebedingungen, insbesondere den Teilnahmebedingungen für den Universal- und Beta-Service, und ist vertraulich.

Die folgenden Fragen werden häufig zum Vorschaukanal gestellt.

Was ist der Vorschaukanal?

Der Preview Channel ist ein Endpunkt für öffentliche Dienste, über den Sie neue Servicefunktionen ausprobieren und Feedback dazu geben können, bevor sie allgemein verfügbar sind. Der Service-Endpunkt für den Vorschaukanal ist vom Standard-Produktionsendpunkt getrennt. Sie interagieren mit dem Endpunkt, indem Sie die AWS CLI, eine Dienstmodelldatei für den Vorschaukanal und Befehlseingabedateien für den verwenden AWS CLI. Mit dem Preview Channel können Sie neue Funktionen ausprobieren, ohne Ihre aktuelle Produktionsinfrastruktur zu beeinträchtigen. Sie werden ermutigt, dem App Mesh-Team [Feedback zu geben](#), um sicherzustellen, dass App Mesh die wichtigsten Anforderungen der Kunden erfüllt. Ihr Feedback zu Funktionen, während sie sich im Vorschaukanal befinden, kann dazu beitragen, die Funktionen von App Mesh so zu gestalten, dass wir den bestmöglichen Service bieten können.

Wie unterscheidet sich der Vorschaukanal von der Produktions-App Mesh?

In der folgenden Tabelle sind Aspekte des App Mesh Mesh-Dienstes aufgeführt, die sich vom Vorschaukanal unterscheiden.

Aspekt	App Mesh Produktionsservice	App Mesh Preview Channel-Dienst
Frontend endpoint	appmesh.us-west-2.amazonaws.com	appmesh-preview.us-west-2.amazonaws.com
Envoy management service endpoint	appmesh-envoy-management.us-west-2.amazonaws.com	appmesh-preview-envoy-management.us-west-2.amazonaws.com
CLI	AWS App Mesh list-meshes	AWS App Mesh-preview list-meshes (only available after adding the Preview Channel service model)
Signing name	appmesh	appmesh-preview
Service principal	appmesh.amazonaws.com	appmesh-preview.amazonaws.com

Note

Das Beispiel in der Tabelle für den App Mesh Mesh-Produktionsservice listet zwar die us-west-2 Region auf, der Produktionsservice ist jedoch in den meisten Regionen verfügbar. Eine Liste aller Regionen, in denen der App Mesh Mesh-Produktionsservice verfügbar ist, finden Sie unter [AWS App Mesh Endpunkte und Kontingente](#). Der App Mesh Preview Channel-Dienst ist jedoch nur in der us-west-2 Region verfügbar.

Wie kann ich Funktionen im Vorschaukanal verwenden?

1. Fügen Sie das Preview Channel-Servicemodell, das die Preview Channel-Funktion enthält, AWS CLI mit dem folgenden Befehl hinzu.

```
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
```

- Erstellen Sie eine JSON-Datei, die die Funktion enthält, basierend auf dem JSON-Beispiel und den Anweisungen im [AWS App Mesh Benutzerhandbuch](#) für die Funktion.
- Implementieren Sie die Funktion mit dem entsprechenden AWS CLI Befehl und der Befehlseingabedatei. Mit dem folgenden Befehl wird beispielsweise mithilfe der Datei *route.json* eine Route mit Preview Channel-Funktionen erstellt.

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

- Fügen Sie `APPMESH_PREVIEW = 1` als Konfigurationsvariable für den Envoy-Container hinzu, wenn Sie ihn zu Ihren Amazon ECS-Aufgabendefinitionen, Kubernetes-Pod-Spezifikationen oder Amazon EC2 EC2-Instances hinzufügen. Diese Variable ermöglicht es dem Envoy-Container, mit den Preview Channel-Endpunkten zu kommunizieren. Weitere Informationen zum Hinzufügen von Konfigurationsvariablen finden Sie unter [Dienste in Amazon ECS aktualisieren](#), [Dienste in Kubernetes aktualisieren](#) und [Dienste auf Amazon EC2 aktualisieren](#).

Wie gebe ich Feedback?

Sie können direkt Feedback zum [App Mesh GitHub Mesh-Roadmap-Repo-Problem](#) geben, das in der Dokumentation zu der Funktion verlinkt ist.

Wie lange habe ich Zeit, um Feedback zu einer Funktion im Vorschaukanal zu geben?

Der Feedback-Zeitraum hängt von der Größe und Komplexität der eingeführten Funktion ab. Wir beabsichtigen, zwischen der Veröffentlichung einer Funktion bis zum Vorschau-Endpunkt und der Veröffentlichung der Funktion bis zur Produktion eine Kommentarfrist von 14 Tagen einzuräumen. Das App Mesh Mesh-Team kann den Feedback-Zeitraum für bestimmte Funktionen verlängern.

Welches Maß an Support wird für den Vorschaukanal angeboten?

Wir empfehlen Ihnen zwar, Feedback und Bugreports direkt zum App Mesh [GitHub Mesh-Roadmap-Problem](#) zu senden, aber wir verstehen, dass Sie möglicherweise sensible Daten teilen möchten oder dass Sie ein Problem finden, das Ihrer Meinung nach nicht sicher ist, um es öffentlich bekannt

zu geben. Bei diesen Problemen können Sie das App Mesh-Team kontaktieren [AWS Support](#) oder Feedback geben, indem Sie das App Mesh Mesh-Team direkt [per E-Mail](#) kontaktieren.

Sind meine Daten auf dem Preview Channel-Endpunkt sicher?

Ja. Der Vorschaukanal hat das gleiche Sicherheitsniveau wie der standardmäßige Produktionsendpunkt.

Wie lange wird meine Konfiguration verfügbar sein?

Sie können dreißig Tage lang mit einem Netz im Vorschaukanal arbeiten. Dreißig Tage nach der Erstellung eines Netzes können Sie das Netz nur auflisten, lesen oder löschen. Wenn Sie nach dreißig Tagen versuchen, Ressourcen zu erstellen oder zu aktualisieren, erhalten Sie eine `BadRequest` Ausnahme, in der erklärt wird, dass das Mesh archiviert wurde.

Welche Tools kann ich verwenden, um mit dem Vorschaukanal zu arbeiten?

Sie können das `AWS CLI` mit einer Preview Channel-Service Modelldatei und Befehlseingabedateien verwenden. Weitere Informationen zur Arbeit mit Funktionen finden Sie unter [Wie kann ich Funktionen im Vorschaukanal verwenden?](#). Sie können die `AWS CLI` Befehlsoptionen, die `AWS Management Console`, `SDKs` oder `AWS CloudFormation` zum Arbeiten mit den Preview Channel-Funktionen nicht verwenden. Sie können jedoch alle Tools verwenden, sobald eine Funktion für den Produktionsservice freigegeben ist.

Wird es eine Vorwärtskompatibilität der Preview Channel-APIs geben?

Nein. APIs können sich aufgrund von Feedback ändern.

Sind die Funktionen von Preview Channel vollständig?

Nein. Neue API-Objekte sind möglicherweise nicht vollständig in das `AWS Management Console`, `AWS CloudFormation`, oder integriert `AWS CloudTrail`. Sobald sich die Funktionen im Vorschaukanal festigen und fast allgemein verfügbar sind, werden die Integrationen irgendwann verfügbar sein.

Ist Dokumentation für die Funktionen von Preview Channel verfügbar?

Ja. Die Dokumentation zu den Preview Channel-Funktionen ist in der Produktionsdokumentation enthalten. Wenn beispielsweise Funktionen für die Routenressource für den Vorschaukanal freigegeben werden, befinden sich Informationen zur Verwendung der Funktionen im vorhandenen [Routenressourcendokument](#). Die Funktionen des Vorschaukanals sind als nur im Vorschaukanal verfügbar gekennzeichnet.

Wie erfahre ich, wann neue Funktionen im Vorschaukanal verfügbar sind?

Wenn neue Funktionen im Vorschaukanal eingeführt werden, wird dem [App Mesh-Dokumentverlauf](#) ein Eintrag hinzugefügt. Sie können die Seite regelmäßig überprüfen oder den [RSS-Feed zur Dokumentenhistorie von App Mesh](#) abonnieren. Darüber hinaus können Sie die [Probleme](#) für das AWS App Mesh GitHub Roadmap-Repo überprüfen. Ein Download-Link für die JSON-Datei des Preview Channel-Servicemodells wird einem Problem hinzugefügt, wenn es für den Preview Channel freigegeben wird. Weitere Informationen zur Verwendung des Modells und der Funktion finden Sie unter [Wie kann ich Funktionen im Vorschaukanal verwenden?](#).

Wie erfahre ich, wann ein Feature in den Produktionsservice übergegangen ist?

Der Text in der App Mesh-Dokumentation, der darauf hinweist, dass die Funktion nur im Vorschaukanal verfügbar ist, wird entfernt, und der [App Mesh-Dokumenthistorie](#) wird ein Eintrag hinzugefügt. Sie können die Seite regelmäßig überprüfen oder den [RSS-Feed zur Dokumentenhistorie von App Mesh](#) abonnieren.

App Mesh service quotas

AWS App Mesh ist in Service Quotas integriert, ein AWS -Service, mit dem Sie Ihre Kontingente von einem zentralen Ort aus anzeigen und verwalten können. Servicekontingente werden auch als Limits bezeichnet. Weitere Informationen zu Service Quotas finden Sie unter [Was sind Service Quotas](#) im Benutzerhandbuch für Service Quotas.

Mit Service App Mesh

App Mesh zeigen Sie AWS Management Console

1. Öffnen Sie die Service-Quotas-Konsole unter <https://console.aws.amazon.com/servicequotas/>.
2. Wählen Sie im Navigationsbereich AWS-Services.
3. Suchen Sie in der Liste der AWS-Services nach AWS App Mesh und wählen Sie es aus.

In der Liste Service Quotas wird der Name der Service Quota, der angewendete Wert (falls verfügbar) und das AWS-Standardkontingent angezeigt. Zudem wird angezeigt, ob der Kontingentwert anpassbar ist.

4. Wählen Sie den Kontingentnamen, um zusätzliche Informationen zu einem Service Quota anzuzeigen, z. B. seine Beschreibung.

Informationen zum Beantragen einer Kontingenterhöhung finden Sie unter [Beantragen einer Kontingenterhöhung](#) im Service-Quotas-Benutzerhandbuch.

App Mesh zeigen Sie AWS CLI

Führen Sie den folgenden Befehl aus.

```
aws service-quotas list-aws-default-service-quotas \
  --query 'Quotas[*]'.
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
  --service-code appmesh \
  --output table
```

Weitere Informationen zum Arbeiten mit Servicekontingenten mithilfe der AWS CLI finden Sie in der [AWS CLI-Befehlsreferenz für Service Quotas](#).

Dokumentenverlauf für App Mesh

Die folgende Tabelle beschreibt die wichtigen Updates und neuen Funktionen für das AWS App Mesh -Benutzerhandbuch. Wir aktualisieren die Dokumentation regelmäßig, um das Feedback, das Sie uns senden, einzuarbeiten.

Änderung	Beschreibung	Datum
CloudTrail Die Integrationsdokumentation wurde aktualisiert	Die Dokumentation, die die App Mesh Mesh-Integration mit CloudTrail der Protokollierung von API-Aktivitäten beschreibt, wurde aktualisiert.	28. März 2024
Aktualisierte Richtlinien	Aktualisiert <code>AWSServiceRoleForAppMesh</code> und <code>AWSAppMeshServiceRolePolicy</code> um den Zugriff auf die <code>AWS Cloud Map DiscoverInstancesRevision</code> API zu ermöglichen.	12. Oktober 2023
Unterstützung von VPC-Endpunkttrichtlinien für App Mesh	App Mesh unterstützt jetzt VPC-Endpunkttrichtlinien.	11. Mai 2023
Mehrere Listener für App Mesh	App Mesh unterstützt jetzt mehrere Listener.	18. August 2022
IPv6 für App Mesh	App Mesh unterstützt jetzt IPv6.	18. Mai 2022
CloudTrail Protokollierungsunterstützung für App Mesh Envoy Management Service	App Mesh unterstützt jetzt die CloudTrail Protokollierung für App Mesh Envoy Management Service.	18. März 2022

App Mesh Agent für Envoy	App Mesh unterstützt jetzt Agent for Envoy.	25. Februar 2022
Mehrere Listener für App Mesh	(Nur App Mesh Preview Channel) Sie können mehrere Listener für App Mesh implementieren.	23. November 2021
ARM64-Unterstützung für App Mesh	App Mesh unterstützt jetzt ARM64.	19. November 2021
Metrics-Erweiterung für App Mesh	Sie können Metrikerweiterungen für App Mesh implementieren.	29. Oktober 2021
Implementieren Sie Verbesserungen für den eingehenden Datenverkehr	Sie können den Hostnamen und den Header abgleichen und Änderungen für Hostname und Pfad implementieren.	14. Juni 2021
Implementieren Sie die gegenseitige TLS-Authentifizierung	Sie können die gegenseitige TLS-Authentifizierung implementieren.	4. Februar 2021
Start der Region in AF-South-1	App Mesh ist jetzt in der Region af-South-1 verfügbar.	22. Januar 2021
Implementieren Sie die gegenseitige TLS-Authentifizierung	(Nur App Mesh Preview Channel) Sie können die gegenseitige TLS-Authentifizierung implementieren.	23. November 2020
Implementieren Sie Verbindungspooling für einen virtuellen Gateway-Listener	Sie können Verbindungspooling für einen virtuellen Gateway-Listener implementieren.	5. November 2020

Implementieren Sie Verbindungspooling und Ausreißererkennung für einen virtuellen Node-Listener	Sie können Verbindungspooling und Ausreißererkennung für einen virtuellen Knoten-Listener implementieren.	5. November 2020
Start der Region in EU-Süd-1	App Mesh ist jetzt in der Region EU-Süd-1 verfügbar.	21. Oktober 2020
Implementieren Sie Verbindungspooling für einen virtuellen Gateway-Listener	(Nur App Mesh Preview Channel) Sie können Verbindungspooling für einen virtuellen Gateway-Listener implementieren.	28. September 2020
Implementieren Sie Verbindungspooling und Ausreißererkennung für einen virtuellen Node-Listener	(Nur App Mesh Preview Channel) Sie können Verbindungspooling und Ausreißererkennung für einen virtuellen Node-Listener implementieren.	28. September 2020
Erstellen Sie ein virtuelles Gateway und eine Gateway-Route für eingehenden Mesh-Datenverkehr	Ermöglichen Sie Ressourcen, die sich außerhalb eines Meshs befinden, die Kommunikation mit Ressourcen, die sich innerhalb eines Meshs befinden.	10. Juli 2020
Erstellen und Verwalten von App-Mesh-Ressourcen aus Kubernetes mit dem App-Mesh-Controller für Kubernetes	Sie können App-Mesh-Ressourcen in Kubernetes erstellen und verwalten. Der Controller injiziert außerdem automatisch den Envoy-Proxy und die Init-Container in Pods, die Sie bereitstellen.	18. Juni 2020

Fügen Sie einem Listener und einer Route für einen virtuellen Knoten einen Timeout-Wert hinzu	Sie können einem Listener und einer Route für einen virtuellen Knoten einen Timeout-Wert hinzufügen.	18. Juni 2020
Fügen Sie einem Listener für virtuelle Knoten einen Timeout-Wert hinzu	(Nur App Mesh Preview Channel) Sie können einem Listener für virtuelle Knoten einen Timeout-Wert hinzufügen.	29. Mai 2020
Erstellen Sie ein virtuelles Gateway für eingehenden Mesh-Datenverkehr	(Nur App Mesh Mesh-Vorschaukanal) Ermöglicht Ressourcen außerhalb eines Meshs, mit Ressourcen innerhalb eines Meshs zu kommunizieren.	8. April 2020
TLS-Verschlüsselung	(Nur App Mesh Preview Channel) Verwenden Sie Zertifikate von einer AWS Private Certificate Authority oder Ihrer eigenen Zertifizierungsstelle, um die Kommunikation zwischen virtuellen Knoten mithilfe von TLS zu verschlüsseln.	17. Januar 2020
Teilen Sie ein Mesh mit einem anderen Konto	(Nur App Mesh Preview Channel) Sie können ein Mesh mit einem anderen Konto teilen. Ressourcen, die mit einem beliebigen Konto erstellt wurden, können mit anderen Ressourcen im Mesh kommunizieren.	17. Januar 2020

Fügen Sie einer Route einen Timeout-Wert hinzu	(Nur App Mesh Mesh-Vorschaukanal) Sie können einer Route einen Timeout-Wert hinzufügen.	17. Januar 2020
Einen App Mesh Mesh-Proxy auf einem AWS Outpost erstellen	Sie können einen App Mesh Envoy-Proxy auf einem AWS Outpost erstellen.	3. Dezember 2019
HTTP/2- und gRPC-Unterstützung für Routen, virtuelle Router und virtuelle Knoten	Sie können Datenverkehr weiterleiten, der die Protokolle HTTP/2 und gRPC verwendet. Sie können virtuellen Knoten und virtuellen Routern auch einen Listener für diese Protokolle hinzufügen.	25. Oktober 2019
Richtlinie erneut versuchen	Eine Wiederholungsrichtlinie ermöglicht Clients, sich vor zeitweiligen Netzwerkausfällen oder zeitweiligen serverseitigen Ausfällen zu schützen. Sie können einer Route eine Wiederholungslogik hinzufügen.	10. September 2019
TLS-Verschlüsselung	(Nur App Mesh Preview Channel) Verschlüsseln Sie die Kommunikation zwischen virtuellen Knoten mithilfe von TLS.	6. September 2019
HTTP-Header-basiertes Routing	Leitet den Verkehr auf der Grundlage des Vorhandenseins und der Werte von HTTP-Headern in einer Anfrage weiter.	15. August 2019

[Verfügbarkeit des App Mesh Preview Channel](#)

Der App Mesh Preview Channel ist eine eigenständige Variante des App Mesh Mesh-Dienstes. Im Vorschaukanal werden zukünftige Funktionen vorgestellt, die Sie ausprobieren können, während sie entwickelt werden. Wenn Sie Funktionen im Vorschaukanal verwenden, können Sie darüber Feedback geben, GitHub um das Verhalten der Funktionen zu beeinflussen.

19. Juli 2019

[VPC-Endpunkte mit App-Mesh-Schnittstelle \(AWS PrivateLink\)](#)

Verbessern Sie die Sicherheit Ihrer VPC, indem Sie App Mesh für die Verwendung eines VPC-Endpunkts mit Schnittstelle konfigurieren. Schnittstellenendpunkte werden von einer Technologie unterstützt, die AWS PrivateLink, mit der Sie privat auf App Mesh Mesh-APIs zugreifen können, indem Sie private IP-Adressen verwenden. PrivateLink schränkt den gesamten Netzwerkverkehr zwischen Ihrer VPC und App Mesh auf das Amazon-Netzwerk ein.

14. Juni 2019

AWS Cloud Map Als Service-Discovery-Methode für virtuelle Knoten hinzugefügt	Sie können DNS oder AWS Cloud Map als Methode zur Erkennung von Diensten für virtuelle Knoten angeben. Um es AWS Cloud Map für die Diensterkennung verwenden zu können, muss Ihr Konto über die dienstverknüpfte App Mesh Mesh-Rolle verfügen.	13. Juni 2019
Automatisches Erstellen von App Mesh Mesh-Ressourcen in Kubernetes	Erstellen Sie App Mesh-Ressourcen und fügen Sie die App Mesh-Sidecar-Container-Images automatisch zu Ihren Kubernetes-Bereitstellungen hinzu, wenn Sie Ressourcen in Kubernetes erstellen.	11. Juni 2019
Allgemeine Verfügbarkeit von App Mesh	Der App Mesh Mesh-Dienst ist jetzt allgemein für Produktionszwecke verfügbar.	27. März 2019
Aktualisierung App Mesh Mesh-API	Die App Mesh Mesh-APIs wurden aktualisiert, um die Benutzerfreundlichkeit zu verbessern. Weitere Informationen finden Sie unter [FEATURE] Hinzufügen von Listenern zu virtuellen Routern und [BUG] Routes to Target Virtual Nodes with Mismatched Ports Blackhole .	7. März 2019
Erste Version von App Mesh	Erste Dokumentation für die öffentliche Vorschau des Dienstes	28. November 2018

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.