

Application Service Mesh

FAQs

Issue	02
Date	2023-07-03



Copyright © Huawei Technologies Co., Ltd. 2024. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Security Declaration

Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process*. For details about this process, visit the following web page:

<https://www.huawei.com/en/psirt/vul-response-process>

For vulnerability information, enterprise customers can visit the following web page:

<https://securitybulletin.huawei.com/enterprise/en/security-advisory>

Contents

1 Service Mesh Cluster.....	1
1.1 Why Does a Service Mesh Remain in the Installing Status for a Long Time After I Enable It for a Cluster?.....	1
1.2 Why Does a Service Mesh Remain in the Unready Status for a Long Time After I Uninstall It?.....	1
1.3 Why Is an otel-collector Workload Created Alongside a Service Mesh?.....	2
2 Mesh Management.....	7
2.1 Why Cannot I Create a Mesh for My Cluster?.....	7
2.2 Why Are Exclusive Nodes Still Exist After Istio Is Uninstalled?.....	7
2.3 How Do I Upgrade ICAgent?.....	8
2.4 How Do I Enable Namespace Injection for a Cluster?.....	8
2.5 How Do I Disable Sidecar Injection for Workloads?.....	9
2.6 What Can I Do If A Pod Cannot Be Started Due to Unready Sidecar?.....	9
3 Adding a Service.....	14
3.1 What Do I Do If an Added Gateway Does Not Take Effect?.....	14
3.2 Why Does It Take a Long Time to Start the Demo Application in Experiencing Service Mesh in One Click?.....	14
3.3 Why Cannot I Access the page of the Demo Application After It Is Successfully Deployed?.....	15
3.4 Why Cannot I Select the Corresponding Service When Adding a Route?.....	15
4 Performing Grayscale Release.....	16
4.1 Why Can't I Change the Image Used for the Grayscale Version When Performing Grayscale Release?.....	16
4.2 Why Does Not a Grayscale Policy that Based on Request Content Take Effect for Some Services?.....	16
4.3 InvalidRequestBody Is Reported When a Grayscale Release Task Is Created for a Service with Multiple Ports.....	17
5 Managing Traffic.....	18
5.1 Why Are the Created Clusters, Namespaces, and Applications Not Displayed on the Traffic Management Page?.....	18
5.2 How Do I Change the Resource Requests of the istio-proxy Container?.....	18
5.3 Does ASM Support HTTP/1.0?.....	19
5.4 How Can I Block Access from Some IP Address Ranges or Ports for a Service Mesh?.....	20
5.5 How Do I Configure max_concurrent_streams for a Gateway?.....	23
6 Monitoring Traffic.....	25

6.1 Jaeger/Zipkin Installation Guide on OSC..... 25

1 Service Mesh Cluster

1.1 Why Does a Service Mesh Remain in the Installing Status for a Long Time After I Enable It for a Cluster?

Symptom

After I create a service mesh (that is, buy a Dedicated mesh) for a CCE cluster, the mesh remains in the installing status for a long time and a message is displayed, indicating that the user security group rules are successfully enabled.

Fault Diagnosis

Log in to the CCE console, choose **Resource Management** > **Namespaces**, and check whether the **istio-system** namespace exists.

Analysis

Residual **istio-system** namespaces exist.

Solution

Delete the residual **istio-system** namespaces and install the mesh again.

1.2 Why Does a Service Mesh Remain in the Unready Status for a Long Time After I Uninstall It?

Symptom

On the ASM console, after I uninstall a service mesh, the mesh remains in the unready status for a long time.

Fault Diagnosis

Step 1 Log in to the CCE console, click the target cluster to go to its details page, and choose **O&M > Charts** in the navigation pane on the left.

Step 2 Click **Releases** and check the releases and the latest events about uninstallation failure.

The **Status** of **istio-master** is **Uninstallation Failed**, and the following message is displayed.

```
deletion failed with 1 error(s): clusterroles.rbac.authorization.k8s.io "istio-cleanup-secrets-istio-system" already exists
```

----End

Analysis

Abnormal operations cause the Helm chart of Istio stuck during uninstallation. Residual resources lead to an uninstallation failure.

Solution

Step 1 Connect to the CCE cluster using kubectl.

Step 2 Run the following commands to clear Istio resources:

```
kubectl delete ServiceAccount -n istio-system `kubectl get ServiceAccount -n istio-system | grep istio | awk '{print $1}'`  
kubectl delete ClusterRole -n istio-system `kubectl get ClusterRole -n istio-system | grep istio | awk '{print $1}'`  
kubectl delete ClusterRoleBinding -n istio-system `kubectl get ClusterRoleBinding -n istio-system | grep istio | awk '{print $1}'`  
kubectl delete job -n istio-system `kubectl get job -n istio-system | grep istio | awk '{print $1}'`  
kubectl delete crd -n istio-system `kubectl get crd -n istio-system | grep istio | awk '{print $1}'`  
kubectl delete mutatingwebhookconfigurations -n istio-system `kubectl get mutatingwebhookconfigurations -n istio-system | grep istio | awk '{print $1}'`
```

Step 3 Log in to the ASM console and uninstall the mesh again.

----End

1.3 Why Is an otel-collector Workload Created Alongside a Service Mesh?

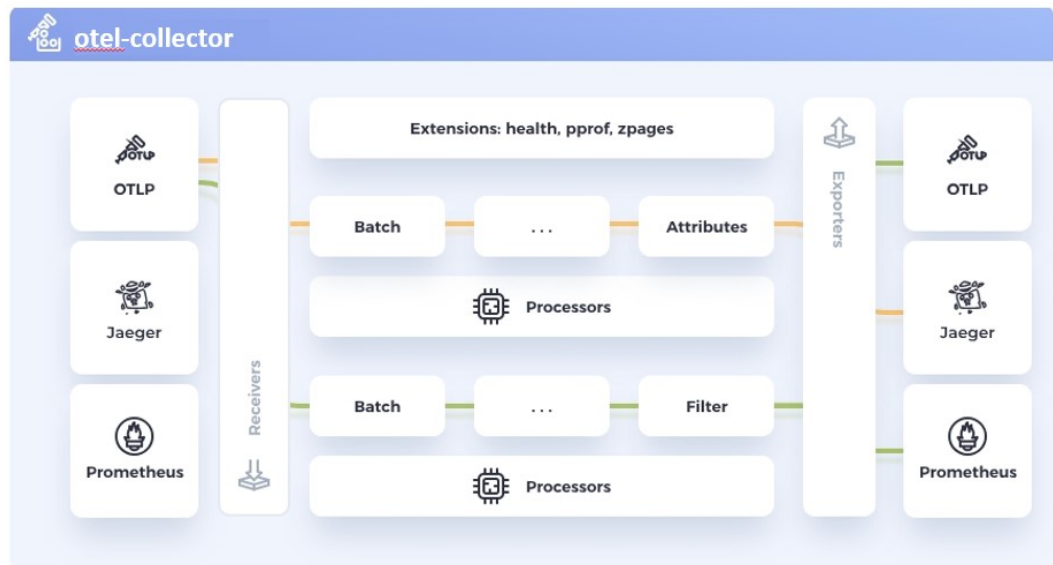
Symptom

An otel-collector workload is automatically created when a service mesh is created.

Analysis

After a cluster is connected to a service mesh, an otel-collector workload is automatically created in the **monitoring** namespace to collect telemetry data (traces, logs, and metrics) from Envoy proxies, process the data, and export the data to one or more backends for mesh observability.

otel-collector Architecture

Figure 1-1 otel-collector architecture

otel-collector consists of four modules:

- **Receivers**
A receiver, which can be push or pull based, is how data gets into otel-collector. Receivers can receive telemetry data in multiple formats, such as OTLP, Jaeger, and Prometheus in the preceding figure.
- **Processors**
Processors process data collected by receivers. For example, a common batch processor processes telemetry data in batches.
- **Exporters**
An exporter is how you send telemetry data to one or more backends. It allows a visual display of telemetry data for data analysis.
- **Extensions**
Extensions are available primarily for tasks that do not involve processing telemetry data. Extensions are optional. For example, you can add the `health_check` extension to check the health of otel-collector.

Using otel-collector on ASM of the Basic Edition

You can run the following command to obtain the settings of otel-collector.

```
[root@localhost ~]# kubectl get cm -n monitoring otel-collector-conf -oyaml
apiVersion: v1
data:
  otel-collector-config: |-
    receivers:
      zipkin: { }
      prometheus:
        config:
          scrape_configs:
            - job_name: 'istio-mesh'
              scrape_interval: 15s
              metrics_path: /stats/prometheus
              kubernetes_sd_configs:
                - role: pod
              relabel_configs:
                - source_labels: [ __meta_kubernetes_pod_container_port_name ]
                  action: keep
                  regex: http-envoy-prom
              metric_relabel_configs:
                - source_labels: [ __name__ ]
                  action: keep
                  regex: istio.*
                - source_labels: [ __name__ ]
                  regex: 'istio_build'
                  action: drop
                - source_labels: [ __name__ ]
                  regex: 'istio_response_bytes.*'
                  action: drop
                - source_labels: [ __name__ ]
                  regex: 'istio_request_bytes.*'
                  action: drop
    processors:
      batch:
      memory_limiter:
```

The following uses the configuration file obtained from ASM of the basic edition as an example:

- **receivers** defines that telemetry data can be obtained from Envoy proxies using **zipkin** and **prometheus**. **prometheus** defines that data is captured from **/stats/prometheus** every 15 seconds.

```
otel-collector-config: |-
  receivers:
    zipkin: { }
    prometheus:
      config:
        scrape_configs:
          - job_name: 'istio-mesh'
            scrape_interval: 15s
            metrics_path: /stats/prometheus
            kubernetes_sd_configs:
              - role: pod
            relabel_configs:
              - source_labels: [ __meta_kubernetes_pod_container_port_name ]
                action: keep
                regex: http-envoy-prom
            metric_relabel_configs:
              - source_labels: [ __name__ ]
                action: keep
                regex: istio.*
              - source_labels: [ __name__ ]
                regex: 'istio_build'
                action: drop
              - source_labels: [ __name__ ]
                regex: 'istio_response_bytes.*'
                action: drop
              - source_labels: [ __name__ ]
                regex: 'istio_request_bytes.*'
                action: drop
```

- **processors** defines two data processing modes: batch and memory_limiter.

```
processors:  
  batch:  
  memory_limiter:  
    check_interval: 1s  
    limit_percentage: 80  
    spike_limit_percentage: 20
```

- **exporters** defines how processed telemetry data is exported to the APM server.

```
exporters:  
  apm:  
    address: "100.79.1.215:8923"  
    project_id: 719217bc273743ea8d7ac1ae8bc34480  
    cluster_id: d7491b95-5111-11ee-8779-0255ac100b05
```

- **extensions** defines the health_check extension, which is used to check the health of otel-collector.

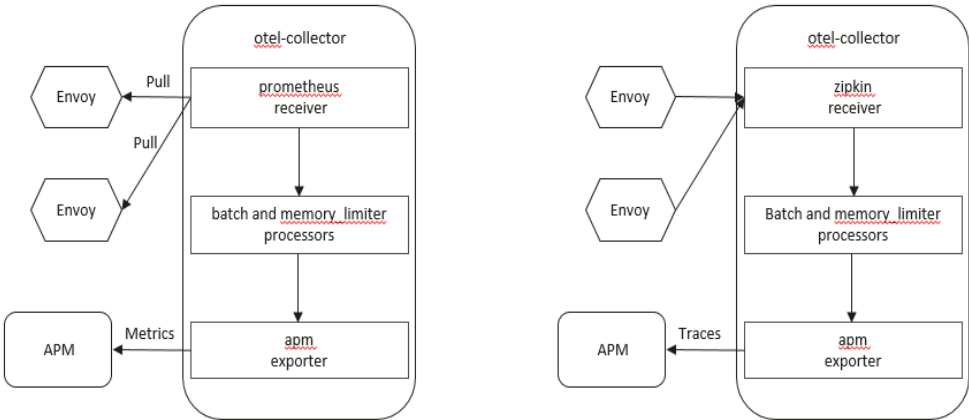
```
extensions:  
  health_check:  
    endpoint: 127.0.0.1:13133
```

- **service** is used to configure the preceding defined configuration items used by otel-collector.

```
service:  
  telemetry:  
    logs:  
      level: info  
    extensions: [ health_check ]  
    pipelines:  
      metrics/apm:  
        receivers: [ prometheus ]  
        processors: [ memory_limiter, batch ]  
        exporters: [ apm ]  
      traces/apm:  
        receivers: [ zipkin ]  
        processors: [ memory_limiter, batch ]  
        exporters: [ apm ]
```

For example, in the preceding configuration file, two pipelines are configured for processing metrics and traces. (A pipeline consists of a receiver, a processor, and an exporter.) The log level is set to INFO or higher. The following figure shows the architecture used for processing metrics and traces.

Figure 1-2 Metrics and traces processing architecture



Solution

No action is required.

2 Mesh Management

2.1 Why Cannot I Create a Mesh for My Cluster?

Symptom

I cannot create a mesh for my cluster.

Analysis

Currently, clusters of versions earlier than 1.15 cannot be managed by meshes.

Solution

- Step 1** Check the cluster version. Currently, only clusters of v1.15, v1.17, v1.19, v1.21, or v1.23 can be managed by meshes.
- Step 2** Check your browser. Chrome is recommended. The button for mesh creation may be unavailable when you are using other browsers, such as Firefox, due to adaptation problems.

----End

2.2 Why Are Exclusive Nodes Still Exist After Istio Is Uninstalled?

Symptom

After Istio is uninstalled, exclusive nodes still exist.

Analysis

Only Istio control plane workloads will be deleted when you uninstall Istio for a cluster. Node resources will not be deleted automatically.

Solution

Nodes from which Istio are uninstalled can be used as common nodes. If these nodes are no longer required, log in to the CCE console, click the target cluster to go to its details page, and choose **Resources** > **Nodes** in the navigation pane to delete the nodes.

2.3 How Do I Upgrade ICAgent?

- Step 1** Log in to the ASM console. In the navigation pane, choose **Monitoring Center** to go to the APM console.
- Step 2** On the APM console, choose **Agent** > **Management** in the navigation pane, select the target cluster, and click **Upgrade ICAgent**.

----End

2.4 How Do I Enable Namespace Injection for a Cluster?

When injecting a sidecar to the namespace of a cluster, if the namespace injection is not enabled in the cluster, perform the following steps:

- Step 1** Connect to the cluster using `kubectl`.
- Step 2** Run the `kubectl get iop -nistio-system` command to query iop resources.
- If the following information is displayed, the iop resource exists. Go to [Step 3](#).

```
user@dts2fot109u4ymb-machine:~$ kubectl get iop -nistio-system
NAME          REVISION  STATUS    AGE
data-plane    1         HEALTHY   69d
```

- If the following information is displayed, no iop resources exist. Go to [Step 4](#).

```
web-terminal-7b778fc945-9m2hf:~# kubectl get iop -nistio-system
No resources found in istio-system namespace.
```

- Step 3** Run the `kubectl edit iop -nistio-system data-plane` command to modify the **autoInject** configuration item. In the preceding command, *data-plane* indicates the name of the iop resource queried in the previous step. Replace it with the actual value.

```
global:
  defaultPodDisruptionBudget:
    enabled: true
  hub: 100.79.1.215:20202/asm
  logging:
    level: default:info
  meshID: test-payment
  multiCluster:
    clusterName: test-yy
    network: test-yy-network
  proxy:
    autoInject: enabled
  remotePilotAddress: 10.252.2.34
  tag: 1.8.6-r1-20220512225026
```

- Step 4** Run the `kubectl edit cm -nistio-system istio-sidecar-injector` command to modify the **istio-sidecar-injector** configuration item.

```
data:
  config: |-
    policy: enabled
```

----End

2.5 How Do I Disable Sidecar Injection for Workloads?

After sidecar injection is enabled for a namespace of a cluster, sidecars are automatically injected for pods of all workloads in the namespace. You can configure sidecars not to be injected into some workloads:

- Step 1** Log in to the CCE console, click the target cluster to go to its details page, and choose **Resources > Workloads** in the navigation pane on the left.
- Step 2** Click **Edit YAML** in the **Operation** column of the target workload.
- Step 3** Find the **spec.template.metadata.annotations** field and add **sidecar.istio.io/inject: 'false'**.

```
annotations:
  sidecar.istio.io/inject: 'false'
```

```
107 spec:
108   replicas: 1
109   selector:
110     matchLabels:
111       app: reviews
112       version: v1
113   template:
114     metadata:
115       creationTimestamp: null
116     labels:
117       app: reviews
118       release: istio-bookinfo
119       version: v1
120     annotations:
121       sidecar.istio.io/inject: 'false'
```

For more details about sidecar injection, see [Automatic Sidecar Injection](#).

----End

2.6 What Can I Do If A Pod Cannot Be Started Due to Unready Sidecar

Description

Pods of services managed by a mesh may fail to be started and keep restarting. When the service container communicates with external systems, the traffic passes through the **istio-proxy** container. However, the service container is started earlier than the **istio-proxy** container. As a result, the communication with external systems fails and the pod keeps restarting.

Solution

In Istio 1.7 and later versions, the community adds a switch named **HoldApplicationUntilProxyStarts** to the **istio-injector** injection logic. After the switch is enabled, the proxy is injected to the first container and the **istio-proxy** container is started earlier than the service container.

The switch can be configured globally or locally. The following describes two ways to enable the switch.

NOTICE

After this switch is enabled, the service container cannot be started until the sidecar is fully ready, which slows down pod startup and reduces scalability for burst traffic. You are advised to evaluate service scenarios and enable this switch only for required services.

- **Global Configuration**

- a. Run the following command to edit the IOP CR resource:

kubectl edit iop private-data-plane -n istio-system

Add the following command to the **spec.values.global.proxy** field:

```
holdApplicationUntilProxyStarts: true
```

```
values:
  gateways:
    istio-egressgateway:
      autoscaleEnabled: false
      labels:
        app: istio-egressgateway
      tolerations:
        - effect: NoExecute
          key: istio
          operator: Exists
    istio-ingressgateway:
      autoscaleEnabled: false
      customService: true
      labels:
        app: istio-ingressgateway
      replicaCount: 1
      tolerations:
        - effect: NoExecute
          key: istio
          operator: Exists
  global:
    defaultPodDisruptionBudget:
      enabled: true
    hub: swr.cn-north-7.myhuaweicloud.com/asm
    logging:
      level: default:info
    meshID: envoy-critical
    multiCluster:
      clusterName: test-yyl-multi
    proxy:
      autoInject: enabled
      holdApplicationUntilProxyStarts: true
```

- b. Run the following command to check whether the latest logs contain no error information:

```
kubectl logs -n istio-operator $(kubectl get po -n istio-operator | awk '{print $1}' | grep -v NAME)
```

- c. Run the following command to check whether the IOP CR is normal:

```
kubectl get iop -n istio-system
```

```
[root@lx666-14467 ~]# kubectl get iop -n istio-system
NAME                REVISION  STATUS    AGE
private-data-plane  1         HEALTHY   6d2h
[root@lx666-14467 ~]#
```

- d. Run the following command to upgrade the services in the mesh in a rolling manner:

```
kubectl rollout restart deployment nginx -n default
```

where, **nginx** is an example service, and **default** is the namespace. Replace them with the actual values.

- e. Run the following command to check whether the pod is restarted:

```
kubectl get pod -n default | grep nginx
```

```
[root@lx666-14467 ~]# kubectl get pod -n default | grep nginx
nginx-6b4959fffb-pr8t8    2/2    Running    0    14s
[root@lx666-14467 ~]#
```

- f. Run the following command to check whether **postStart lifecycle** is added to the pod and whether the **istio-proxy** container is placed in the first position:

kubectl edit pod nginx-7bc96f87b9-l4dbl

```
- name: ISTIO_META_CLUSTER_ID
  value: test-yy1-multi
image: swr.cn-north-7.myhuaweicloud.com/asm/proxyv2:1.13.9-r1-20221110212800
imagePullPolicy: IfNotPresent
lifecycle:
  postStart:
    exec:
      command:
        - pilot-agent
        - wait
name: istio-proxy
ports:
```

- **Local Configuration**

For Istio 1.8 or later versions, you can label the pods for which this function needs to be enabled with **proxy.istio.io/config** and set **holdApplicationUntilProxyStarts** to true.

The following uses the **nginx** service in the **default** namespace as an example. The operations for other services are similar.

kubectl edit deploy nginx -n default

Add the following commands to the **spec.template.metadata.annotations** field:

```
proxy.istio.io/config: |
holdApplicationUntilProxyStarts: true
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "6"
    description: ""
  creationTimestamp: "2022-11-24T07:55:31Z"
  generation: 6
  labels:
    appgroup: ""
    version: v1
  name: tomcat
  namespace: default
  resourceVersion: "55550644"
  uid: cd5dbfe8-83cc-4964-86fc-f657c85e852d
spec:
  progressDeadlineSeconds: 600
  replicas: 1
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: tomcat
      version: v1
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      annotations:
        kubectrl.kubernetes.io/restartedAt: "2022-11-25T10:35:02+08:00"
        proxy.istio.io/config: |
          holdApplicationUntilProxyStarts: true
    creationTimestamp: null
```

3 Adding a Service

3.1 What Do I Do If an Added Gateway Does Not Take Effect?

The possible cause is that the Gateway-related resource configurations are missing or incorrect. Do as follows to locate the fault:

- Log in to the Elastic Load Balance console, check whether the external port and ECSs are successfully listened by the load balancer.
- Log in to the cluster and run the **kubectrl get gateway -n istio-system** command to check whether the IP address, domain name, and port number are configured for the Gateway. Run the **kubectrl get svc -n istio-system** command to check whether the ingress Gateway has the corresponding IP address and port and is not in the pending status.
- Check whether the internal access protocol of the service added to the service mesh is consistent with the external access protocol configured for the service's Gateway.
- If the **ERR_UNSAFE_PORT** error is displayed when you use a browser to access the service, that is because the port is identified as a dangerous port by the browser. In this case, you need to use another external port.

3.2 Why Does It Take a Long Time to Start the Demo Application in Experiencing Service Mesh in One Click?

The demo application contains the productpage, details, ratings, and reviews services. All related workloads and Istio resources including DestinationRule, VirtualService, and Gateway need to be created. Therefore, the creation takes a comparatively long time.

3.3 Why Cannot I Access the page of the Demo Application After It Is Successfully Deployed?

Symptom

The page of the demo application cannot be accessed after the application is successfully deployed.

Analysis

The load balancer configured for the application does not listen to the port.

Solution

Log in to the Elastic Load Balance console. Check whether the port listener has been created and whether the health status of the backend server is normal. For details about how to create a load balancer, see [Listener](#).

3.4 Why Cannot I Select the Corresponding Service When Adding a Route?

During adding a route, the target service is filtered based on the corresponding gateway protocol. The filtering rules are as follows:

- For an HTTP gateway, select an HTTP service.
- For a TCP gateway, select a TCP service.
- For a gRPC gateway, select a gRPC service.
- For an HTTPS gateway, select either an HTTP or a gRPC service.
- For a TLS gateway which TLS termination is enabled, select a TCP service. If TLS termination for a TLS gateway is disabled, select a TLS service.

4 Performing Grayscale Release

4.1 Why Can't I Change the Image Used for the Grayscale Version When Performing Grayscale Release?

Description

When I perform grayscale release, the image used for the grayscale version cannot be changed.

Analysis

When performing grayscale release on a service, you create a new version of the same service. Therefore, the image used by the service cannot be changed. Only image tags can be changed.

Solution

Pack the required image into a different tag of the same image and push it to the image repository. Then, select the newly pushed image tag when you perform grayscale release on the service.

4.2 Why Does Not a Grayscale Policy that Based on Request Content Take Effect for Some Services?

Symptom

A grayscale policy that based on request content does not take effect on some services.

Analysis

A grayscale policy based on request content is valid only for the entry service that is directly accessed.

Solution

If you want a grayscale policy to be applied to all services in an application, the header information of the HTTP request needs to be transferred in the service code.

4.3 InvalidRequestBody Is Reported When a Grayscale Release Task Is Created for a Service with Multiple Ports

Symptom

When a grayscale release task is created for a service with multiple ports, "ASM.0002 InvalidRequestBody" is reported.

Fault Location

Log in to the ASM console, press **F12**, and switch to the **Network** tab to view APIs. Error code 400 is returned when each POST request is sent to create a grayscale release task. The returned information is as follows:

some ports of the service have been configured with routes, ports=[%v]

Analysis

Some ports are deleted from the service that is properly configured. For example, service01 has two ports 80 and 81, and port 81 is deleted on the CCE console.

Solution

Restore the deleted service ports.

5 Managing Traffic

5.1 Why Are the Created Clusters, Namespaces, and Applications Not Displayed on the Traffic Management Page?

1. Check whether Istio has been enabled for your cluster.
2. Check whether at least one service has been added to the **Service List** page and is in the **Running** state.
3. Check whether you have uninstalled the ICAgent in the cluster.

5.2 How Do I Change the Resource Requests of the istio-proxy Container?

The default resources of the **istio-proxy** container are as follows. You can change the resources as required.

```
resources:
  limits:
    cpu: "2"
    memory: 512Mi
  requests:
    cpu: "1"
    memory: 512Mi
```

Method 1: Modify the Configuration for All Services in the Mesh

Do as follow to change the resource requests configuration of the **istio-proxy** container for all services in the mesh at a time:

Step 1 Run the following command to modify the ConfigMap:

kubectl edit cm istio-sidecar-injector -n istio-system

```
315     resources:
316       {{ if or (isset .ObjectMeta.Annotations `sidecar.istio.io/proxyCPU`) (isset .ObjectMeta
ar.istio.io/proxyLimitCPU`) (isset .ObjectMeta.Annotations `sidecar.istio.io/proxyLimitMemory`) -
317     requests:
318       {{ if (isset .ObjectMeta.Annotations `sidecar.istio.io/proxyCPU`) -}}
319       cpu: "{{ index .ObjectMeta.Annotations `sidecar.istio.io/proxyCPU` }}"
320       {{ end }}
321       {{ if (isset .ObjectMeta.Annotations `sidecar.istio.io/proxyMemory`) -}}
322       memory: "{{ index .ObjectMeta.Annotations `sidecar.istio.io/proxyMemory` }}"
323       {{ end }}
324     limits:
325       {{ if (isset .ObjectMeta.Annotations `sidecar.istio.io/proxyLimitCPU`) -}}
326       cpu: "{{ index .ObjectMeta.Annotations `sidecar.istio.io/proxyLimitCPU` }}"
327       {{ end }}
328       {{ if (isset .ObjectMeta.Annotations `sidecar.istio.io/proxyLimitMemory`) -}}
329       memory: "{{ index .ObjectMeta.Annotations `sidecar.istio.io/proxyLimitMemory` }}"
330       {{ end }}
331     {{ else -}}
332     {{- if .Values.global.proxy.resources }}
333     #{{ toYaml .Values.global.proxy.resources | indent 4 }}
334     requests:
335     cpu: xxx
336     memory: xxx
337     limits:
338     cpu: xxx
339     memory: xxx
340     {{- end }}
341     {{ end -}}
```

Step 2 Restart the **istio-sidecar-injector** pod in the **istio-system** namespace.

Step 3 Restart service pods in the rolling upgrade mode to avoid service interruption.

----End

Method 2: Modify the Configuration for A Specific Service in the Mesh

Step 1 Modify the YAML file of the service.

kubectl edit deploy <nginx> -n <namespace>

Step 2 Add the following configuration lines to **spec.template.metadata.annotations** (you can change the resource size as required).

```
sidecar.istio.io/proxyCPU: 500m
sidecar.istio.io/proxyLimitCPU: 500m
sidecar.istio.io/proxyLimitMemory: 1024Mi
sidecar.istio.io/proxyMemory: 1024Mi
```

For meshes of Istio 1.8, add the following configuration lines:

```
sidecar.istio.io/proxyCPU: 500m
sidecar.istio.io/proxyCPULimit: 500m
sidecar.istio.io/proxyMemoryLimit: 1024Mi
sidecar.istio.io/proxyMemory: 1024Mi
```

Step 3 After the modification, restart service pods in the rolling upgrade mode to avoid service interruption.

----End

5.3 Does ASM Support HTTP/1.0?

Description

By default, Istio does not support HTTP/1.0.

Cause Analysis

In Istio, Envoy forwards traffic and Pilot allocates rules. The **PILOT_HTTP10** (environment variable) of **pilot** is set to **0** by default, that is, **HTTP/1.0** is not supported by default.

Solution

Edit the **..values.pilot.env.PILOT_HTTP10** environment variable in the **iop** file and transfer the **PILOT_HTTP10** environment variable to **pilot**. Example:

```
pilot:
  autoscaleEnabled: false
  enableHttp10: false
  env:
    PILOT_HTTP10: 1
  replicaCount: 1
  resources:
    limits:
      cpu: 2000m
      memory: 4096Mi
    requests:
      cpu: 100m
      memory: 128Mi
  tolerations:
```

5.4 How Can I Block Access from Some IP Address Ranges or Ports for a Service Mesh?

Scenarios

In some scenarios, you may want to specify IP address ranges that need to be blocked by a service mesh proxy. In some scenarios, you may want to specify ports to block requests. This section describes how to block access from some IP address ranges and ports.

Workload Configuration for Blocking Access from Some IP Address Ranges

Modify the **deployment** file to block some IP address ranges.

Run the **kubectl edit deploy -n *user_namespace* *user_deployment*** command.

1. In **deployment.spec.template.metadata.annotations**, use **traffic.sidecar.istio.io/includeOutboundIPRanges** to specify IP address ranges to be blocked.

```
type: RollingUpdate
template:
  metadata:
    annotations:
      asm/updatedAt: "2023-03-23T03:49:21Z"
      sidecar.istio.io/proxyCPU: "0.1"
      sidecar.istio.io/proxyCPULimit: "2"
      sidecar.istio.io/proxyMemory: 128Mi
      sidecar.istio.io/proxyMemoryLimit: 2048Mi
      traffic.sidecar.istio.io/includeOutboundIPRanges: 192.168.0.1/24
    creationTimestamp: null
  labels:
    app: nginx
    version: v1
```

2. In **deployment.spec.template.metadata.annotations**, use **traffic.sidecar.istio.io/excludeOutboundIPRanges** to specify IP address ranges that are allowed.

```
template:
  metadata:
    annotations:
      asm/updatedAt: "2023-03-23T03:49:21Z"
      sidecar.istio.io/proxyCPU: "0.1"
      sidecar.istio.io/proxyCPULimit: "2"
      sidecar.istio.io/proxyMemory: 128Mi
      sidecar.istio.io/proxyMemoryLimit: 2048Mi
      traffic.sidecar.istio.io/excludeOutboundIPRanges: 192.168.0.1/24
    creationTimestamp: null
  labels:
    app: nginx
    version: v1
```

Note: The preceding operations will cause rolling upgrades of service containers.

Workload Configuration for Blocking Ingress and Egress Traffic over Some Ports

Modify the **deployment** file to block ingress and egress traffic over some ports.

Run the **kubectl edit deploy -n *user_namespace* *user_deployment*** command.

1. In **deployment.spec.template.metadata.annotations**, use **traffic.sidecar.istio.io/excludeInboundPorts** to specify the ports that allow the ingress traffic.

```
type: RollingUpdate
template:
  metadata:
    annotations:
      asm/updatedAt: "2023-06-01T01:40:56Z"
      traffic.sidecar.istio.io/excludeInboundPorts: 3306,6379
    creationTimestamp: null
  labels:
    app: echo
```

2. In **deployment.spec.template.metadata.annotations**, use **traffic.sidecar.istio.io/includeInboundPorts** to specify the ports that block the ingress traffic.

```
type: RollingUpdate
template:
  metadata:
    annotations:
      asm/updatedAt: "2023-06-01T01:40:56Z"
      traffic.sidecar.istio.io/includeInboundPorts: 3306,6379
    creationTimestamp: null
  labels:
    app: echo
```

3. In **deployment.spec.template.metadata.annotations**, use **traffic.sidecar.istio.io/excludeOutboundPorts** to specify the ports that allow the egress traffic.

```
template:
  metadata:
    annotations:
      asm/updatedAt: "2023-06-01T01:40:56Z"
      traffic.sidecar.istio.io/excludeOutboundPorts: 3306,6379
    creationTimestamp: null
  labels:
```

4. In **deployment.spec.template.metadata.annotations**, use **traffic.sidecar.istio.io/includeOutboundPorts** to specify the ports that block the egress traffic.

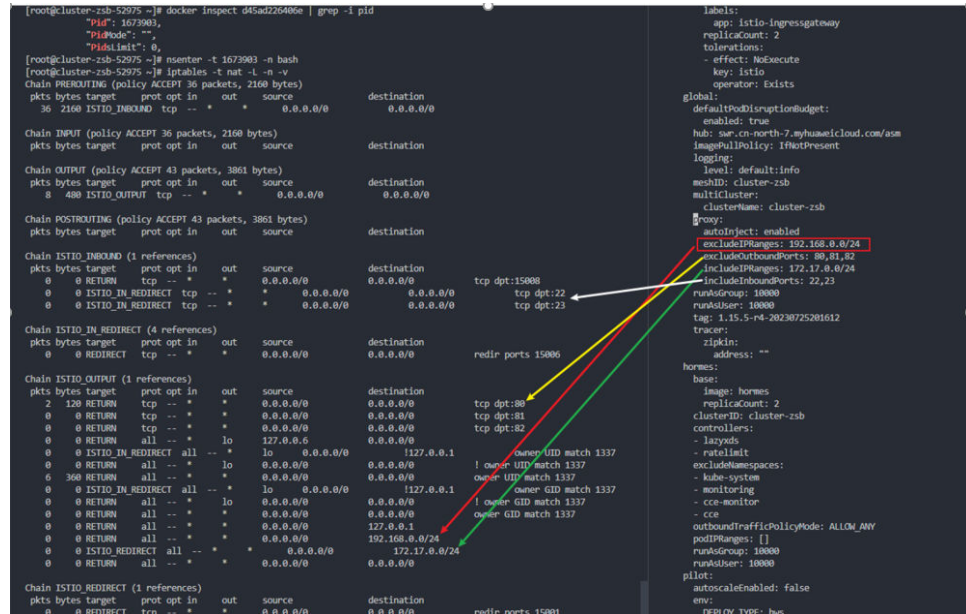
```
type: RollingUpdate
template:
  metadata:
    annotations:
      asm/updatedAt: "2023-06-01T01:40:56Z"
      traffic.sidecar.istio.io/includeOutboundPorts: 3306,6379
    creationTimestamp: null
  labels:
    app: echo
    sidecarVersion: 1.13.9-r1-1685522112
```

Note: The preceding operations will cause rolling upgrades of service containers.

Verification

The configurations take effect in iptables of containers. Run the following commands to check whether the configurations take effect.

1. Log in to the node where the workload is running and run the **docker ps** command to find the pause container and view the container ID.
2. Run the **docker inspect <CONTAINER_ID> | grep -i pid** command to view the process ID.
3. Run the **nsenter -t <PID> -n bash** command to go to the namespace of the container.
4. Run the **iptables iptables -t nat -L -n -v** command to check whether the configurations take effect for specified IP address ranges and ports.



5.5 How Do I Configure max_concurrent_streams for a Gateway?

Step 1 Log in to any node in the cluster where the gateway is located and run the following command to create resources:

```
cat>"stream-limit-envoyfilter.yaml"<<EOF
apiVersion: networking.istio.io/v1alpha3
kind: EnvoyFilter
metadata:
  name: http2-stream-limit
  namespace: istio-system
spec:
  workloadSelector:
    labels:
      istio: ingressgateway
  configPatches:
    - applyTo: NETWORK_FILTER # http connection manager is a filter in Envoy
      match:
        context: GATEWAY
        listener:
          filterChain:
            filter:
              name: "envoy.filters.network.http_connection_manager"
      patch:
        operation: MERGE
        value:
          typed_config:
            "@type": "type.googleapis.com/
envoy.extensions.filters.network.http_connection_manager.v3.HttpConnectionManager"
            http2_protocol_options:
              max_concurrent_streams: 128
EOF
```

NOTE

The **max_concurrent_streams** parameter indicates the maximum number of concurrent streams of a gateway. You can set this parameter as required.

Step 2 Run the **kubectl apply -f stream-limit-envoyfilter.yaml** command to create an EnvoyFilter.

```
root@ecs-guobaoqing-0054:~# kubectl get envoyfilter -nistio-system
NAME                AGE
http2-stream-limit  8s
```

----End

6 Monitoring Traffic

6.1 Jaeger/Zipkin Installation Guide on OSC

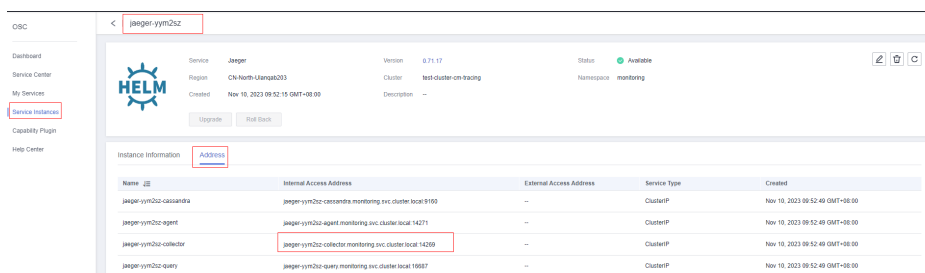
The methods for installing the Jaeger and Zipkin add-ons are the same. The following uses Jaeger as an example to describe the method.

- Step 1** Log in to the OSC console. In the navigation pane, choose **Service Center**. Search for Jaeger and click **Subscribe**.
- Step 2** In the navigation pane, choose **My Service**. On the **My Subscriptions** tab, click **Create Instance** of Jaeger.
- Step 3** Set **Environment**, **Region**, **Container Cluster**, **Namespace**, and **Service Type**. Then select the **I Understand** check box in the lower left corner.

NOTE

Set **Namespace** to **monitoring**. Otherwise, you need to configure ServiceEntry and WorkloadEntry.

- Step 4** Click **Next: Instance Parameters** and configure instance parameters using form or YAML (recommended).
- Step 5** Click **Next: Information Confirmation**, confirm the information, and click **Submit**. Wait until the instance is successfully created.
- Step 6** On the instance details page, click the **Access** tab to view the IP address and port number used by Jaeger to receive requests.



NOTICE

The address and port number that contain the **collector** keyword are the service address and port number set for the third-party Jaeger or Zipkin service in **Observability Configuration** when a service mesh is purchased on ASM.

----End