Application Service Mesh 23.3.1

User Guide

Issue	01
Date	2023-03-15

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Huawei Cloud Computing Technologies Co., Ltd.

Address:	Huawei Cloud Data Center Jiaoxinggong Road Qianzhong Avenue Gui'an New District Gui Zhou 550029 People's Republic of China
Website:	https://www.huaweicloud.com/intl/en-us/

Contents

1 Service Overview

1.1 Introduction

What Is Application Service Mesh?

Application Service Mesh (ASM) is a non-intrusive solution for you to manage microservice lifecycle and traffic. It is compatible with the Kubernetes and Istio ecosystems and hosts a wide range of features such as load balancing, outlier detection, and fault injection. It also provides diversified built-in grayscale releases, including canary release and blue-green deployment, for one-stop, automated release.

What Is Istio?

Istio is an open platform that provides connection, protection, control, and observation functions. By providing a complete non-intrusive microservice governance solution, Istio can well resolve service network governance issues such as cloud-native service management, network connection, and security management.

With the popularization of microservices, greater challenges emerge in the basic operations and advanced O&M of the distributed microservice architecture.

At a high level, Istio helps reduce the complexity of application deployments, and eases the strain on your development teams. It is a fully open-source service mesh that can be transparently layered onto existing distributed applications. It is also a platform, including APIs that let it integrate into any logging platform, or telemetry or policy system. Istio's diverse feature set lets you successfully and efficiently run a distributed microservice architecture, and provides a unified way to secure, connect, and monitor microservices.

Service Mesh

The term service mesh is used to describe the network of microservices that make up applications and the interactions between applications. As a service mesh grows in size and complexity, it can become harder to understand and manage. You need to take care of basic operations, such as service discovery, load balancing, failure recovery, metrics, and monitoring. Advanced O&M includes blue-green deployment, canary release, rate limiting, access control, and end-to-end authentication.

Why Use Istio?

Istio provides behavioral insights and operational control over the service mesh as a whole, offering a complete solution to satisfy the diverse requirements of microservice applications.

Kubernetes allows you to deploy and upgrade applications, and manage running traffic. However, capabilities such as outlier detection and rate limiting are not supported. Istio, as an open platform built based on Kubernetes, provides complementary capabilities to Kubernetes in microservice governance.

You add Istio support to services by deploying a special sidecar proxy throughout your environment that intercepts all networking requests between microservices, then configure and manage Istio using its control plane functionality, which includes:

- Automatic load balancing for HTTP, gRPC, WebSocket, and TCP traffic.
- Fine-grained control of traffic behavior with rich routing rules, retries, and fault injection.
- Automatic metrics, logs, and traces for all traffic within a cluster, including cluster ingress and egress.
- Secure service-to-service communication in a cluster with strong identity-based authentication and authorization.

Istio aims to achieve scalability and meet various deployment requirements.

Features

Grayscale release

- Grayscale policies based on request content: You can set criteria based on request content, such as header and cookie. Only requests meeting the criteria will be distributed to the grayscale version.
- Grayscale policies based on traffic ratio: You can set specific ratio for the traffic to be distributed to the grayscale version.
- Canary release: Guidance will be provided to help you perform canary release on a service, including rolling out a grayscale version, observing the running and traffic of the grayscale version, configuring grayscale release policies, and diverging the traffic.
- Blue-green deployment: Guidance will be provided to help you perform blue-green deployment on a service, including rolling out a grayscale version, observing the running of the grayscale version, observing the traffic, and switching the traffic.

Traffic management

- Layer-7 connection pool management: You can set the maximum number of HTTP requests, maximum number of retry times, maximum number of pending requests, maximum number of requests for each connection, and maximum connection idle period.

- Layer-4 connection pool management: You can set the maximum TCP connections, connection timeout duration, maximum non-responses, minimum idle period, and health check interval.
- Outlier detection: You can configure outlier detection rules, such as the number of consecutive errors allowed before a pod is evicted, check period, base ejection time, and maximum percentage of ejected pods.
- Retry: You can configure the number of HTTP retry times, retry timeout duration, and retry condition.
- Timeout: You can configure the HTTP request timeout duration.
- Load balancing: You can configure multiple load balancing policies, such as random, round robin, least connections, and consistent hashing.
- HTTP header: You can flexibly add, edit, and remove HTTP headers, including the operations on the HTTP headers before the request is forwarded to the destination service and before the response is returned to the client.
- Fault injection: You can configure delay and abort faults.

Security

- Peer authentication: Peer authentication defines how traffic reaches the current service pod through the tunnel (or not through the tunnel). Currently, three authentication policies are supported: **UNSET**, **PERMISSIVE**, and **STRICT**.
- Access authorization: Access authorization controls the access to services in the mesh and determines whether a request can be sent to the current service.

Observability

- Application access topology: An application access topology shows the dependencies between services.
- Service running monitoring: Service access information, including service information, different versions of the service, QPS, and latency can be monitored.
- Access logs: Service access logs can be collected and searched.

Framework of the mesh data plane

- Spring Cloud: supports unified management of services developed using Spring Cloud SDK.
- Dubbo: supports unified management of services developed using Dubbo SDK.

Compatibility and extension

- Community compatibility: ASM APIs are fully compatible with the Istio community.
- Support for community add-ons: Tracing, Prometheus, Kiali, and Grafana are supported.

1.2 Advantages

Ease of Use

The out-of-the-box usability allows you to use a service mesh without code rewrite or manual installation.

Built-in Canary Release and Blue-Green Deployment

- Deployment of the grayscale version and traffic switchover with a few clicks
- Configurable grayscale policy that can be set based on traffic ratio and request content (cookies, OSs, and browsers)
- One-stop health, performance, and traffic monitoring, achieving quantified, intelligent, and visualized grayscale release

Policy-based Intelligent Routing and Flexible Traffic Management

Load balancing, service routing, fault injection, and outlier detection policies can be intuitively configured. Microservice traffic management can be real-time, visualized, intelligent, and automated, requiring no modifications on your applications.

- Routing rules can be set based on weight and content to implement flexible grayscale release of applications.
- Load balancing achieves high availability for service processing.
- Outlier detection ensures stable and reliable links between services.
- Network persistent connection management saves resources and improves network throughput.
- Service security certification, authentication, and audit lay a solid foundation for service security assurance.

Enhanced Performance and Reliability

The performance and reliability of the control plane and data plane are enhanced based on the community version.

Multi-infrastructure

An O&M-free hosting control plane is provided. Unified service governance, grayscale release, security, and service running monitoring capabilities are supported. Unified service discovery and management of multiple infrastructure resources such as containers and VMs are provided.

Protocol Extension

The HTTP, gRPC, TCP, TLS, and Dubbo protocols are supported.

Traditional SDK Integration

Integration solutions for traditional microservice SDKs such as Spring Cloud and Dubbo are provided. Services developed using traditional microservice SDKs can be migrated to cloud-native containers and mesh running environments without major code modification.

1.3 Application Scenarios

1.3.1 Grayscale Release

Application Scenarios

In traditional iterations, a new service version is directly released to all users at a time. This is risky, because once an online accident or bug occurs, the impact on users is great. It could take a long time to fix the issue. Sometimes, the version has to be rolled back, which severely affects user experience.

Grayscale release is a smooth iteration mode for version upgrade. During the upgrade, some users use the new version, while other users continue to use the old version. After the new version is stable and ready, it gradually takes over all the live traffic.

Product Benefits

ASM provides multiple grayscale release functions for application governance. It allows you to detect and fix issues at the early stage and ensure that the iteration goes smoothly and efficiently.

Product Advantages

- **Built-in grayscale release process**: Multiple typical grayscale release processes are provided based on fine-grained distribution rules. A grayscale release wizard is provided for you to conveniently perform grayscale release practices. When a service version processes traffic, you can create a grayscale version. After the grayscale version is rolled out successfully, you can configure grayscale policies and diverge live traffic to the grayscale version.
- **Flexible grayscale policies**: You can set grayscale policies based on traffic ratio or request content, such as the OS, browser, cookie, and header information in an HTTP request. After configuring grayscale policies, you can view the running status and access information on multiple online versions in real time. You can select a stable version and switch all live traffic to this version in one click.

1.3.2 Service Traffic Management

Application Scenarios

Traffic management entails a wide range of operations, including:

- Dynamically modifying load balancing policies for cross-service access, such as configuring consistent hashing to send traffic to specific service pods
- Distributing a certain proportion of traffic to a specific version of a service when the service has two online versions

- Protecting services, for example, limiting the number of concurrent connections and requests, and isolating faulty service pods
- Dynamically modifying the content of a service or simulating a service running fault

Product Benefits

No code refactoring is required when you use ASM to manage traffic.

Non-intrusive traffic management capabilities are provided based on Istio. Policy- and scenario-based network connection management is provided to suit different service protocols. Different management rules can be configured for different service APIs on the topology to meet your service requirements.

Product Advantages

- **Retry:** Auto retries upon service access failures improve the access quality and success rate. You can set the number of HTTP retry times, retry timeout duration, and retry condition.
- **Timeout:** Auto processing and quickly failure return upon service access timeout eliminate resource locking and request freezing. You can set the HTTP request timeout duration.
- **Connection pool management**: You can configure the maximum TCP connections, connection timeout duration, maximum non-responses, minimum idle period, and health check interval for layer-4 protocols, and configure the maximum number of HTTP requests, maximum number of retry times, maximum number of pending requests, maximum number of requests for each connection, and maximum connection idle period for layer-7 protocols. In this way, the failure of a service will not cascade and affect the entire application.
- **Outlier detection**: You can configure the number of consecutive errors allowed before pod eviction, eviction interval, minimum eviction time, and maximum eviction ratio as outlier detection. In this way, you can check the running status of service pods on a regular basis. If access exceptions occur frequently, the pod is marked as abnormal and isolated accordingly. No traffic will be distributed to it in a specific period of time. After the isolation time, requests will be distributed to the pod. If the pod still runs abnormally, it will be isolated for a longer time. This is how pod isolation and automatic fault recovery work.
- **Load balancing**: Diverse load balancing policies, such as random, round robin, and least connection, are provided. You can configure consistent hashing to send traffic to specific service pods.
- **HTTP header:** You can flexibly add, edit, and remove HTTP headers, including the operations on the HTTP headers before the request is forwarded to the destination service and before the response is returned to the client.
- **Fault injection**: Abort and delay faults can be injected to specified services to test their resilience. No code refactoring is required.

1.3.3 End-to-End Transparency and Security

Application Scenarios

Splitting traditional monolithic applications into microservices brings various benefits, including better flexibility, scalability, and reusability. The new security requirements microservices have are as follows:

- Traffic encryption is required to defend against man-in-the-middle attacks.
- TLS and fine-grained access control policies are required for flexible service access control.
- Audit tools are needed to determine who can do what at what time.

ASM provides a comprehensive security solution, including authentication policies, transparent TLS encryption, and authorization and audit tools, to address these requirements.

Product Benefits

- **Default security**: No modification is required on application code and architecture to ensure security.
- **In-depth defense**: ASM can integrate with existing security systems to provide comprehensive defense.
- **Zero-trust network**: The security solution is built assuming that all the network is untrusted.

Product Advantages

- **Non-intrusive security**: ASM provides service meshes as infrastructure with built-in security capabilities. It allows you to focus more on the development and O&M of your services. No code refactoring is required to ensure service access security. ASM provides a transparent, distributed security layer and underlying secure communication channels, which manage authentication, authorization, and encryption for service communication. ASM provides communication security between pods and services. Developers only need to focus on application-level security based on this security infrastructure layer.
- **Fine-grained authorization**: After authentication, access authorization between services can be managed. Authorization management can be performed on a specific service or a specific API of a service. For example, you can authorize all services in a specific namespace or only a specific service. The source service and destination service can be in different clusters. Pods of the source service can be in different clusters. Pods of the destination service can be in different clusters.

1.3.4 Service Running Monitoring

Application Scenarios

Container-based infrastructure brings a series of new challenges. It is necessary to evaluate and enhance the performance of API endpoints and identify potential risks of infrastructure. Istio service mesh enables you to enhance API performance with no code refactoring and service delay.

Product Benefits

ASM generates detailed telemetry for all service communications within the mesh. It provides observability of service behaviors and allows operators to easily troubleshoot, maintain, and optimize their applications. With ASM, operators can better understand how services interact with other services and their components.

Product Advantages

- **Non-intrusive collection of monitoring data**: To manage and troubleshoot a complex application, it is important to obtain its access topology between services and monitoring data. ASM collects the monitoring data in a non-intrusive way. It allows you to focus on service development rather than the generation of monitoring data.
- **Flexible service running management**: You can view the health status and dependencies between services using the access data in a topology. The topology allows you to unfold a service and observe the running status of each version of the service and each pod of a version. This pod-level topology enables you to observe how outlier detection policies work, for example, how a pod is isolated and how it recovers. When the pod is isolated, no requests are distributed to it. After it runs normally, requests are automatically distributed to it again.

1.3.5 Integration with Traditional Microservice SDKs

Application Scenarios

- Using service meshes to manage services developed using traditional SDKs.
- Integrating Istio with microservice platforms and building a microservice management and control center based on Istio.

Product Benefits

Integration solutions for traditional microservice SDKs, such as Spring Cloud and Dubbo are provided. Services developed using traditional microservice SDKs can be migrated to cloud-native containers and meshes without major code modification.

Product Advantages

- **Non-intrusive migration solution**: ASM provides a non-intrusive solution to help you migrate services developed using traditional SDKs to service meshes. The service invoker diverts outbound traffic to the mesh data plane. That is, the service discovery and load balancer in the original SDK are short-circuited. The Kubernetes service name is used for access, and the service discovery in the Kubernetes is used. The governance logic in the SDK can be gradually replaced by the mesh. In this way, the service running and governance capabilities are provided directly by the infrastructure based on Kubernetes and service meshes.
- **Unified policy management**: The unified ASM control plane is used to discover services and manage governance rules. No independent registration center or configuration center is required. Service discovery, load balancing, and governance on the data plane are all performed using the data plane

Envoy. SDKs only function as a lightweight application development framework.

- **Multiple infrastructure resources**: In the solution, the data plane can be deployed on containers or VMs. Services can be developed in various languages. There is no restriction on the development framework. Traffic rules are delivered through the mesh control plane and all forms of data planes are managed in a unified manner.

1.4 Constraints

Constraints on Clusters

Before creating a service mesh, ensure that you have an available cluster v1.15, v1.17, or v1.19.

Constraints on Service Meshes

When you use service meshes for service governance, a Deployment can only match with one service to avoid abnormal grayscale release, gateway access, or other functions.

1.5 Basic Concepts

Workload

A workload is an abstract model of a group of pods in Kubernetes. Workloads defined in Kubernetes include Deployments, StatefulSets, jobs, and DaemonSets.

- **Deployment**: Pods of a Deployment are completely independent of each other and deliver the same functions. Deployments support auto scaling and rolling updates. Typical examples include Nginx and WordPress.
- **StatefulSet**: Pods in a StatefulSet are not completely independent of each other. StatefulSets have stable persistent storage and unique network identifiers. They support ordered, graceful deployment, scaling, and deletion. Typical examples include MySQL-HA and etcd.

Pod

A pod is the smallest and simplest unit in the Kubernetes object model that you create or deploy. A pod encapsulates an application container (in some cases, multiple containers), storage resources, a unique network IP address, and options that govern how the container should run.

Canary Release

Grayscale release is essential for smooth rollout of iterative software products in production environments. A certain proportion of production traffic on the live network is distributed to a new version of the application to test the version's performance and detect faults while ensuring stable running of the system.

Blue-Green Deployment

Blue-green deployment is a zero-downtime deployment mode. A new version of an application is deployed and tested in a production environment while the live environment continues to serve all production traffic. When you confirm that the new version is functioning properly, traffic is then distributed to the new version. At the same time, the old version is upgraded to the new version. Blue-green deployment allows you to quickly switch between the two versions to effectively prevent service disruption during the upgrade.

Traffic Management

Traffic management provides you with visualized network statuses of cloud native applications and allows you to manage and configure network connections and security policies online. Currently, it supports connection pool, outlier detection, load balancing, HTTP header, fault injection, etc.

Connection Pool Management

Thresholds for TCP and HTTP connections and request pools to prevent a service from overloading.

Outlier Detection

Quick response and service access fault isolation are configured to prevent a cascade of network and service calling faults. In this way, the fault impact is curbed. The overall system performance deterioration or avalanche is prevented.

1.6 Recommended Node Specifications

Recommended Specifications for Exclusive Nodes:

The performance of ASM is closely related to the cluster control plane (master nodes). Select appropriate node specifications based on your service requirements.

Total QPS (requests per second)	0-20,000	20,000-60,000
Specifications	8 vCPUs and 16 GB memory	16 vCPUs and 32 GB memory

NOTE

- The total QPS is the sum of requests of all services in all applications per second in a cluster.
- The function of recommending a proper amount of memory resources on the control plane based on the number of application service instances is coming soon.

ingressgateway Resource Consumption Reference

The resource consumed by each ingressgateway pod is affected by the connection type, number of connections, and QPS. For details, see the following tables:

Table 1-1 Memory consumption of persistent connections

Connections	Memory Usage (MB)
1	0.055
1,000	55
10,000	550

Table 1-2 CPU and memory usage of short connections

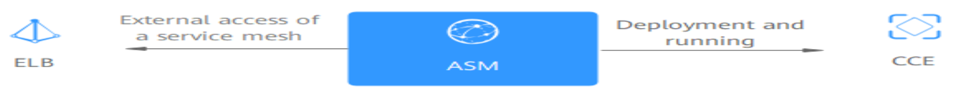
QPS	CPU Usage (m)	Memory Usage (MB)
100	30	100
1,000	300	100
10,000	3,000	150

The preceding data is for reference only. The actual resource consumption depends on the actual service model.

1.7 Related Services

Figure 1-1 shows the dependencies between ASM and other services.

Figure 1-1 Dependencies between ASM and other services



Cloud Container Engine (CCE)

Cloud Container Engine (CCE) is a high-performance, high-reliability service through which enterprises can manage containerized applications. CCE supports native Kubernetes applications and tools, allowing you to easily set up a container runtime environment on the cloud.

You can enable the service mesh function for CCE clusters to manage the services in the clusters.

Elastic Load Balance (ELB)

ELB automatically distributes access traffic to multiple cloud servers to balance the loads. It enhances an application's fault tolerance and service continuity.

You can use ELB to access ASM from outside.

2 Getting Started

2.1 Grayscale Release Practices of Bookinfo

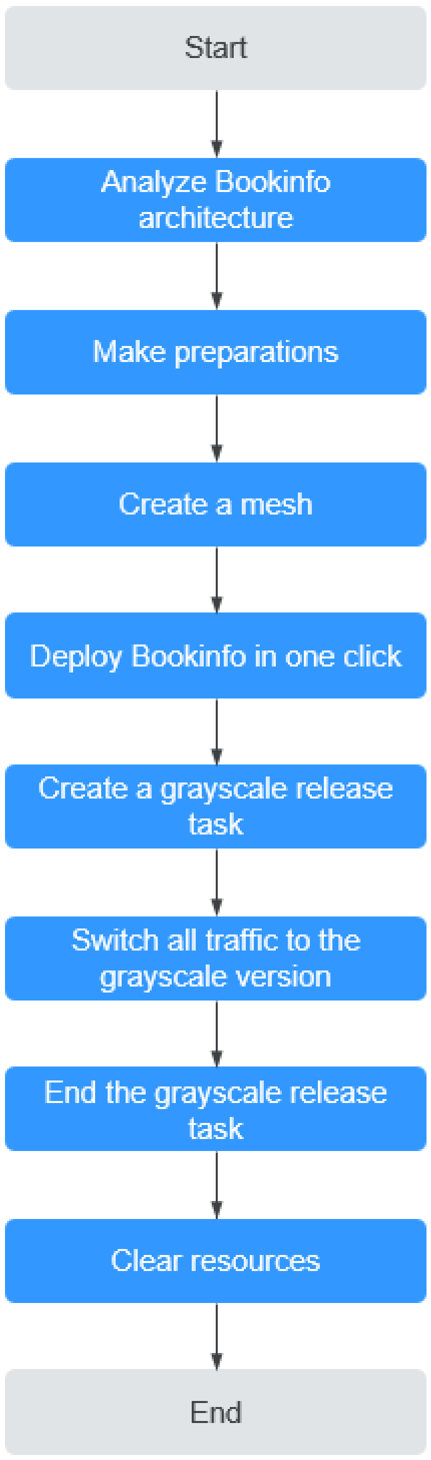
Application Service Mesh (ASM) is a service mesh platform developed based on Istio and seamlessly interconnects with Cloud Container Engine (CCE). With better usability, reliability, and visualization, ASM provides you with out-of-the-box features and enhanced user experience.

Introduction

Grayscale releases enable smooth iteration of software products in production environments. This section takes Bookinfo as an example to illustrate Istio-based service governance using ASM.

The grayscale release process of Bookinfo is as follows.

Figure 2-1 Grayscale release process of Bookinfo


Start
Analyze Bookinfo architecture
Make preparations
Create a mesh
Deploy Bookinfo in one click
Create a grayscale release task
Switch all traffic to the grayscale version
End the grayscale release task
Clear resources
End

Architecture Analysis of Bookinfo

Bookinfo is an application that functions as an online bookstore that displays each book with its description, details (such as pages), and reviews.

Bookinfo consists of four independent services developed in different languages. These services demonstrate the features of a typical service mesh. They are described as follows:

- productpage: calls the details and reviews services to generate a page.
- details: contains book information.
- reviews: contains book reviews and calls the ratings service.
- ratings: contains book rating information based on reviews.

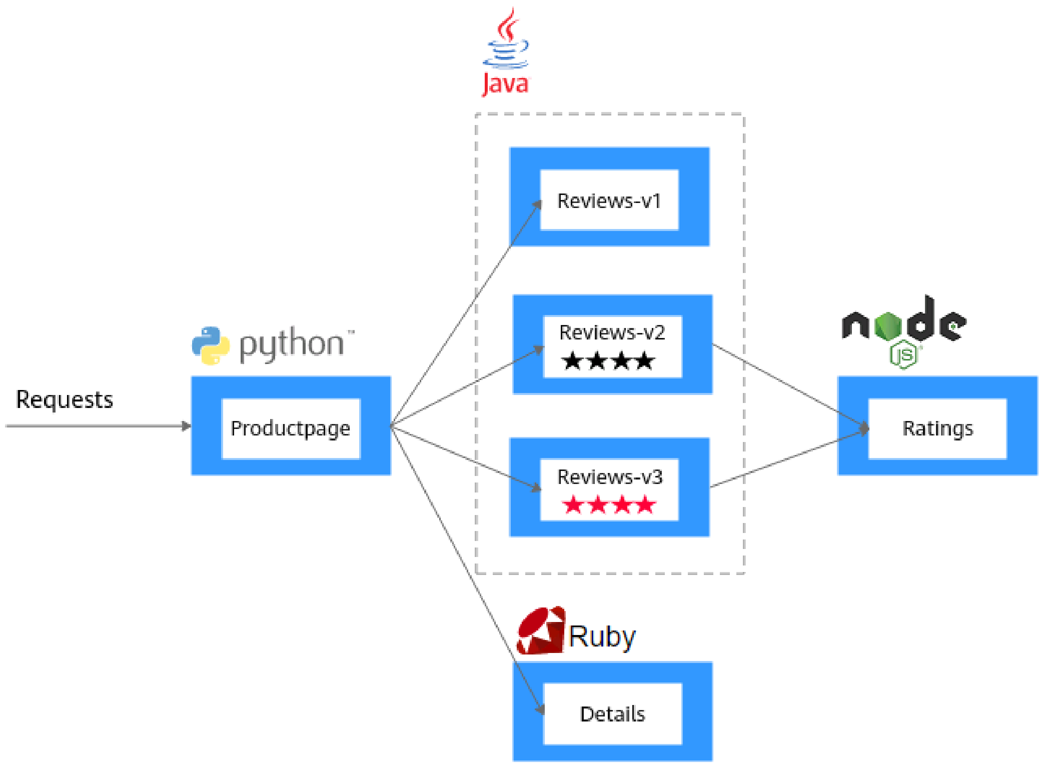
The reviews service has three versions:

- The v1 (1.17.0) version does not call the ratings service.
- The v2 (1.17.1) version calls the ratings service and uses one to five black stars to show ratings.
- The v3 (1.17.2) version calls the ratings service and uses one to five red stars to show ratings.

NOTE

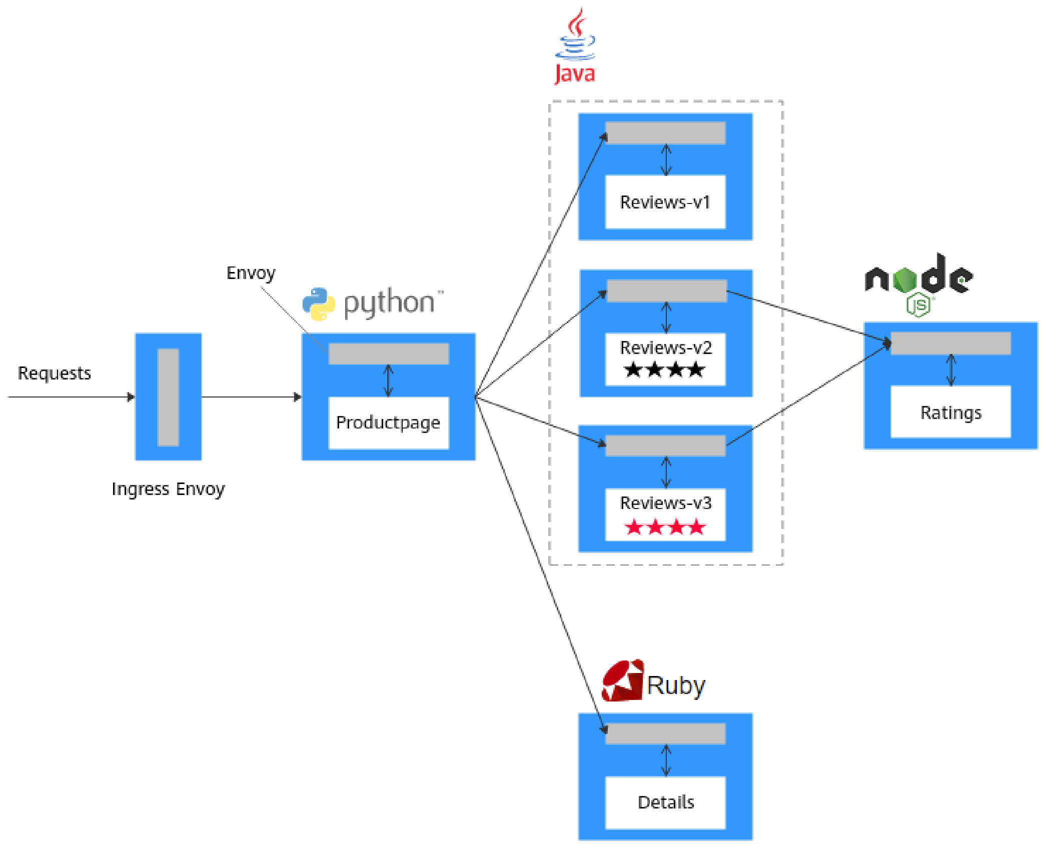
To demonstrate traffic switching between versions, this section takes v2 (rating with black stars) and v3 (rating with red stars) of the reviews service as examples.

Figure 2-2 End-to-end architecture of Bookinfo



Running Bookinfo with ASM does not require any changes on the application itself. Simply configure and run the services in the ASM environment, that is, inject an Envoy sidecar into each service. **Figure 2-3** shows the final deployment.

Figure 2-3 Bookinfo with Envoy sidecars injected



All services are integrated with Envoy sidecars. All inbound and outbound traffic of the integrated services is intercepted by sidecars. In this way, ASM can provide service routing, telemetry data collection, and policy implementation for Bookinfo.

Preparations

Perform the following operations:

Step 1 Create a VPC and subnet.

Virtual Private Cloud (VPC) provides a logically isolated, configurable, and manageable virtual network environment, improving resource security and simplifying network deployment.

1. Log in to the VPC console.
2. Click Create VPC in the upper right corner.
3. Set the parameters as prompted and click Create Now.

Step 2 (Optional) Create a key pair.

To log in to a cluster node using a key pair, create a key pair in advance.

1. Log in to the Elastic Cloud Server (ECS) console.
2. In the navigation pane, choose **Key Pair**. On the **Key Pair** page, click **Create Key Pair** in the upper right corner.
3. Enter a key pair name and click **OK**.

4. Manually or automatically download the private key file. The file name is the specified key pair name with a suffix of **.pem**. Securely store the private key file. In the dialog box displayed, click **OK**.

   NOTE

   For security purposes, a key pair can be downloaded only once. Keep it secure to ensure successful login.

Step 3 Create a load balancer.

A load balancer will be used as the external access entry of a service mesh, which will route the traffic to backend services.

1. Log in to the Elastic Load Balance (ELB) console.
2. Click Create Elastic Load Balancer in the upper right corner.
3. **VPC** and **Subnet**: Select the VPC and subnet created in **Step 1**, configure other parameters as prompted, and click Apply Now.

Step 4 Create a cluster.

1. Log in to the Cloud Container Engine (CCE) console.
2. In the navigation pane, choose **Resource Management** > **Clusters**. Then, click **Create CCE Cluster** in the upper right corner.
3. On the **Configure** page, configure the following parameters and retain the default values for other parameters.
   – **Cluster Name**: Enter a cluster name, for example, **cce-asm**.
   – **VPC** and **Subnet**: Select the VPC and subnet created in **Step 1**.
4. Click **Next: Create Node**, configure the following parameters, and retain the default values for other parameters.
   – **Specifications**: 4 vCPUs and 8 GB memory.

     NOTE

     This is the minimum specifications for deploying Bookinfo.
   – **Login Mode**: Select the key pair created in **Step 2** for identity authentication upon remote node login.
5. Click **Next: Install Add-on** and select the add-ons to be installed in the **Install Add-on** step.

   System resource add-on must be installed. **Advanced functional add-on** is optional.
6. Click **Next: Confirm**. Read the product constraints and select **I am aware of the above limitations**. Review the configured parameters and specifications.
7. Submit the order.

   It takes about 6 to 10 minutes to create a cluster. You can click **Back to Cluster List** to perform other operations on the cluster or click **Go to Cluster Events** to view the cluster details.

----End

Creating a Mesh

Step 1 Log in to the ASM console.

Step 2 Click **Create Mesh** in the upper right corner.

Step 3 Configure the following parameters and retain the default values for other parameters.

- **Mesh Edition**
  The default value is **Basic edition**.
- **Mesh Name**
  Enter the mesh name.
- **Istio Version**
  Select the Istio version supported by the mesh.
- **Cluster**
  Select the cluster created in **Step 4**.
- **Mesh Control Plane Node**
  To achieve HA, select two or more nodes from different AZs.

Step 4 Review the mesh configuration in the **Configuration List** on the right of the page and click **Submit**.

It takes about 1 to 3 minutes to create a mesh. If the mesh status changes from **Installing** to **Running**, the mesh is successfully created.

----End

Deploying Bookinfo in One Click

After the mesh is enabled for the cluster, you can quickly create a Bookinfo demo.

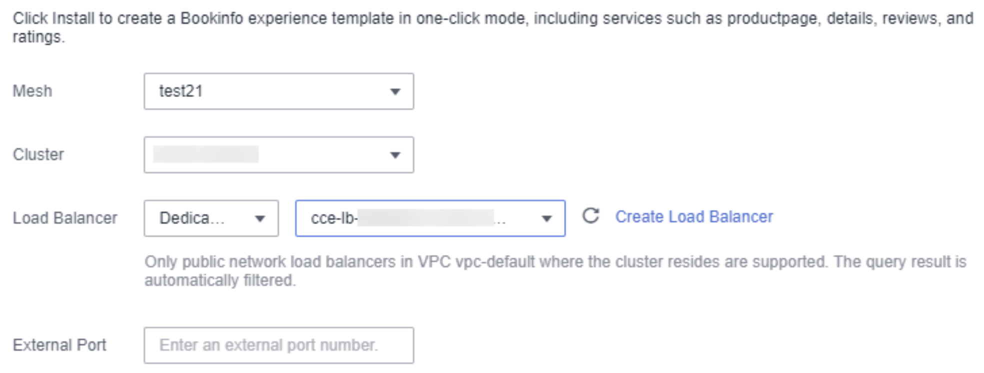
Step 1 Log in to the ASM console.

Step 2 Click the target mesh to view its details.

Step 3 In the navigation pane, choose **Experience Tasks** and click **Try Now** in the Bookinfo task.

Step 4 On the right of the page, set **Cluster** to the cluster where Bookinfo resides, set **Load Balancer** to a load balancer that is in the same VPC and subnet as the selected cluster, set an external port, and click **Install**.

Figure 2-4 Installing Bookinfo



Click Install to create a Bookinfo experience template in one-click mode, including services such as productpage, details, reviews, and ratings.

Mesh: test21

Cluster: [redacted]

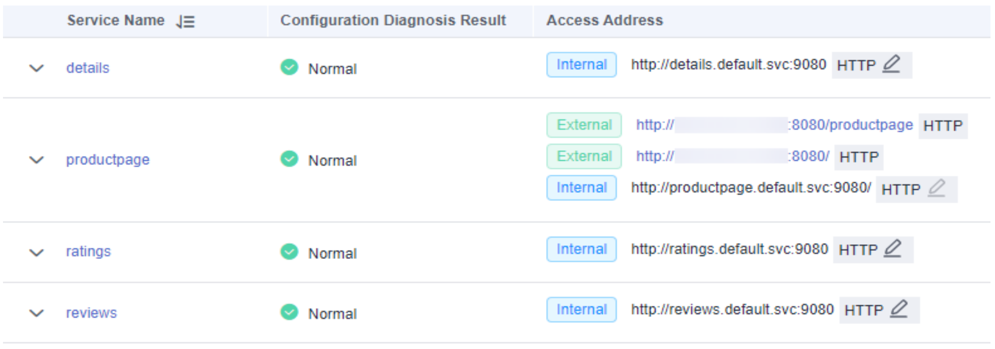
Load Balancer: Dedica... | cce-lb-[redacted]...

Only public network load balancers in VPC vpc-default where the cluster resides are supported. The query result is automatically filtered.

External Port: Enter an external port number.

Step 5 Wait until Bookinfo is created. Click **Service Management** and ensure that the value in the **Configuration Diagnosis Result** column is **Normal**. The Bookinfo contains the productpage, details, reviews, and ratings services.

Figure 2-5 Service list



Service Name	Configuration Diagnosis Result	Access Address
details	Normal	Internal http://details.default.svc:9080 HTTP
productpage	Normal	External http:// :8080/productpage HTTP External http:// :8080/ HTTP Internal http://productpage.default.svc:9080/ HTTP
ratings	Normal	Internal http://ratings.default.svc:9080 HTTP
reviews	Normal	Internal http://reviews.default.svc:9080 HTTP

----End

Creating a Grayscale Release Task

A new grayscale version of the **reviews** service of Bookinfo will be created. A grayscale policy will be configured to divert traffic of the default version to the new version.

The following steps will guide you to create a new version (v3) of the **reviews** service and divert 30% traffic of Bookinfo to this version.

Deploying a grayscale version

Step 1 In the navigation pane, choose **Grayscale Release**. On the **Canary Release** area of the displayed page, click **Create Release Task**.

Step 2 Configure basic information.

- **Task Name**: Enter a task name, for example, **reviews-v3**.
- **Namespace**: Select the namespace to which the service belongs.
- **Service**: Select **reviews** from the drop-down list box.
- **Workload**: Select the workload to which the service belongs.

Step 3 Configure grayscale version information.

- **Cluster**: Select the cluster to which the service belongs.
- **Version**: Set this parameter to **v3**.
- **Pods**: Retain the default value.
- **Pod Configuration**: Set the image tag to **1.17.2** and retain the default values for other parameters.

Step 4 Click **Release**. If the progress reaches 100%, the grayscale version is successfully released.

----End

Configuring a traffic policy

Configure a grayscale policy for the grayscale version. A specified percentage of traffic will be diverted from the original version to the grayscale version.

Step 1 After the grayscale version is deployed, click **Configure Traffic Policy**.

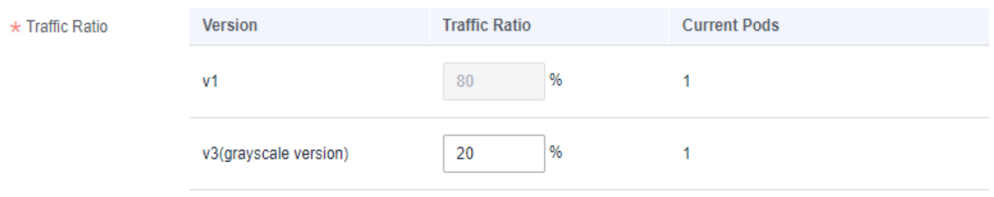
Step 2 Configure a traffic policy.

Policy Type: The value can be **Based on traffic ratio** or **Based on request content**.

- **Based on traffic ratio**: A specified percentage of traffic will be directed to the grayscale version. For example, 80% of the traffic is directed to the original version, and 20% is directed to the grayscale version.
- **Based on request content**: Only the traffic that meets specific conditions will be directed to the grayscale version. For example, only users on the Windows operating system can access the grayscale version.

In this example, configure a traffic policy **Based on traffic ratio** and set the traffic percentage of v3 to **20%**.

Figure 2-6 Traffic policy



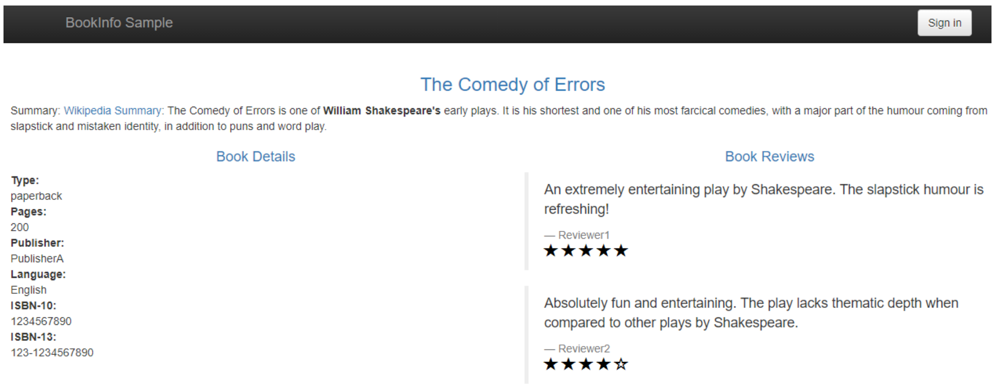
* Traffic Ratio

Version	Traffic Ratio	Current Pods
v1	80 %	1
v3(grayscale version)	20 %	1

Step 3 Click **Deliver Policy**.

Step 4 On the **Service List** page, click the **Access Address** of the productpage service. Frequently refresh the book information page. You can find that the **Book Reviews** area is switching between black stars (v1) and red stars (v3) and the ratio is nearly 4 to 1.

Figure 2-7 v1 page



BookInfo Sample Sign in

The Comedy of Errors

Summary: Wikipedia Summary: The Comedy of Errors is one of **William Shakespeare's** early plays. It is his shortest and one of his most farcical comedies, with a major part of the humour coming from slapstick and mistaken identity, in addition to puns and word play.

Book Details

Type:
paperback
Pages:
200
Publisher:
PublisherA
Language:
English
ISBN-10:
1234567890
ISBN-13:
123-1234567890

Book Reviews

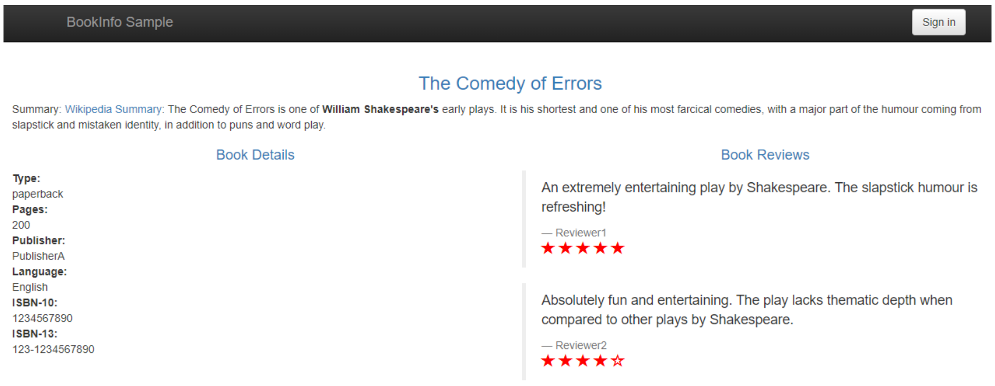
An extremely entertaining play by Shakespeare. The slapstick humour is refreshing!

— Reviewer1
★★★★★

Absolutely fun and entertaining. The play lacks thematic depth when compared to other plays by Shakespeare.

— Reviewer2
★★★★☆

Figure 2-8 v3 page



The Comedy of Errors

Summary: Wikipedia Summary: The Comedy of Errors is one of **William Shakespeare's** early plays. It is his shortest and one of his most farcical comedies, with a major part of the humour coming from slapstick and mistaken identity, in addition to puns and word play.

Book Details

Type: paperback
Pages: 200
Publisher: PublisherA
Language: English
ISBN-10: 1234567890
ISBN-13: 123-1234567890

Book Reviews

An extremely entertaining play by Shakespeare. The slapstick humour is refreshing!
— Reviewer1
★★★★★

Absolutely fun and entertaining. The play lacks thematic depth when compared to other plays by Shakespeare.
— Reviewer2
★★★★☆

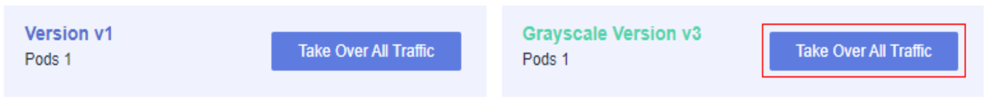
----End

Switching All Traffic to the Grayscale Version

Check whether the number of resources in v3 matches that in v1. After confirming that v3 is able to serve all the traffic of v1, switch all the traffic from v1 to v3.

Step 1 On the **Monitor and Manage Traffic** page, click **Take Over All Traffic** next to v3.

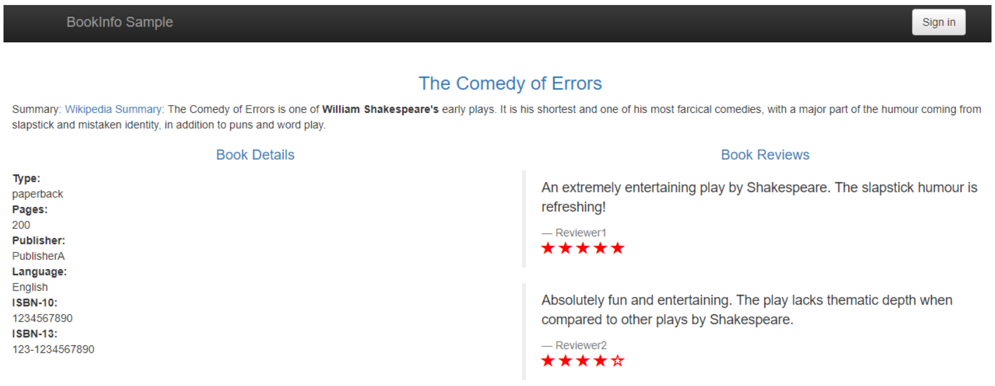
Figure 2-9 Taking over all traffic



Step 2 Click **OK**.

A message indicating that the traffic is successfully switched is displayed in the upper right corner. Frequently refresh the Bookinfo page. You can find that only red stars (v3) are used in the **Book Reviews** area.

Figure 2-10 v3 page



The Comedy of Errors

Summary: Wikipedia Summary: The Comedy of Errors is one of **William Shakespeare's** early plays. It is his shortest and one of his most farcical comedies, with a major part of the humour coming from slapstick and mistaken identity, in addition to puns and word play.

Book Details

Type: paperback
Pages: 200
Publisher: PublisherA
Language: English
ISBN-10: 1234567890
ISBN-13: 123-1234567890

Book Reviews

An extremely entertaining play by Shakespeare. The slapstick humour is refreshing!
— Reviewer1
★★★★★

Absolutely fun and entertaining. The play lacks thematic depth when compared to other plays by Shakespeare.
— Reviewer2
★★★★☆

----End

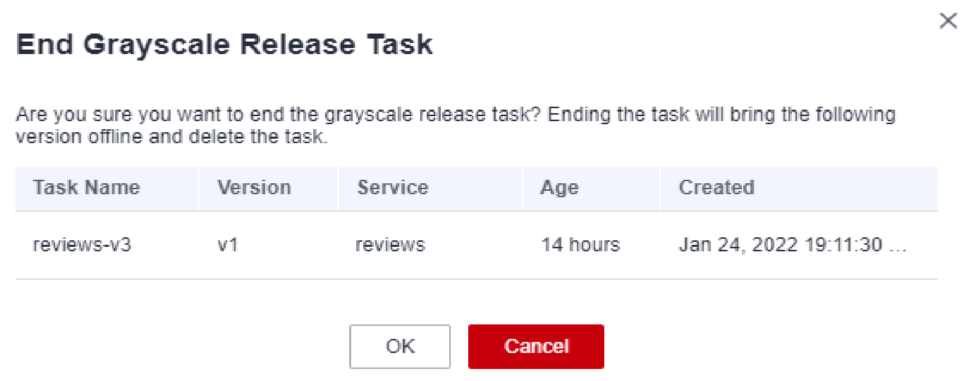
Ending the Grayscale Release Task

After v3 takes over all the traffic from v1, bring v1 offline to release its resources.

Step 1 On the **Monitor and Manage Traffic** page, click **End Task**.

Step 2 Click **OK** to end the task, bring the original version offline, and delete the task.

Figure 2-11 Ending the grayscale release task



End Grayscale Release Task

Are you sure you want to end the grayscale release task? Ending the task will bring the following version offline and delete the task.

Task Name	Version	Service	Age	Created
reviews-v3	v1	reviews	14 hours	Jan 24, 2022 19:11:30 ...

Bringing a version offline will delete all its workloads and Istio configuration resources.

----End

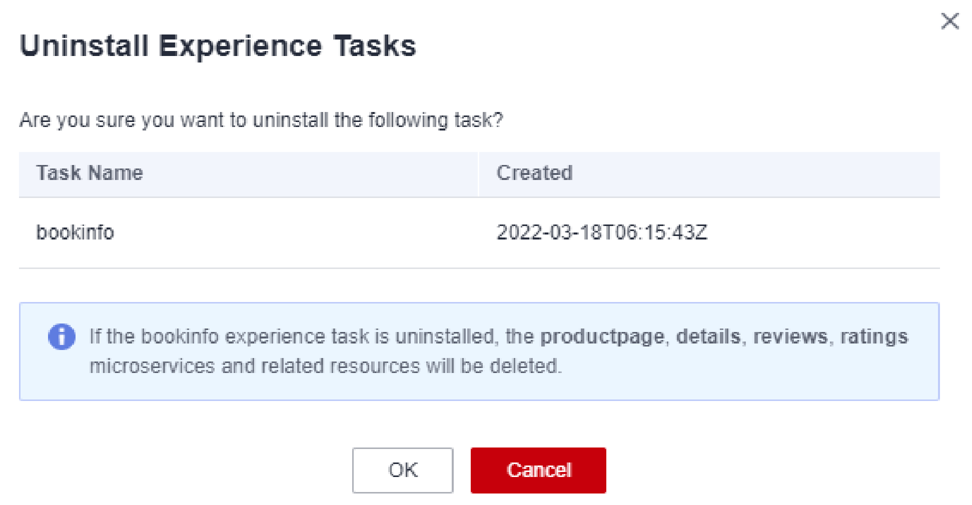
Clearing Resources

This is the end of the demo of performing the grayscale release using ASM. Delete applications and nodes in time to avoid unnecessary fees.

Step 1 In the navigation pane, choose **Experience Tasks** and click **Uninstall** in the Bookinfo task.

Step 2 Click **OK**. After the Bookinfo experience task is uninstalled, the productpage, details, reviews, and ratings services and related resources are automatically deleted.

Figure 2-12 Uninstalling experience tasks



Uninstall Experience Tasks

Are you sure you want to uninstall the following task?

Task Name	Created
bookinfo	2022-03-18T06:15:43Z

If the bookinfo experience task is uninstalled, the **productpage**, **details**, **reviews**, **ratings** microservices and related resources will be deleted.

NOTE

After an experience task is uninstalled, go to the CCE console and manually delete the workloads corresponding to the grayscale version of the service for which grayscale release has been completed.

----End

2.2 Enabling Istio for a Cluster

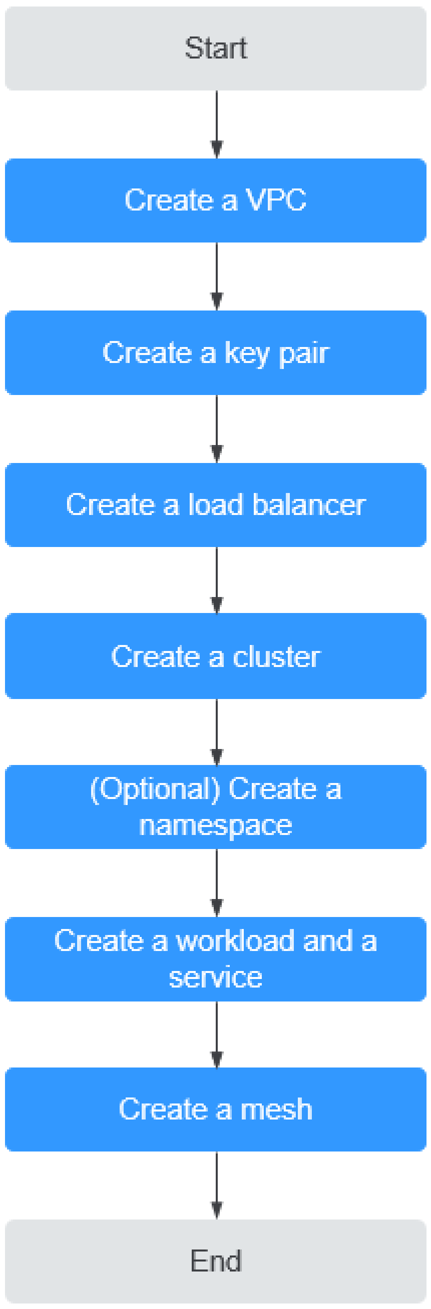
2.2.1 Overview

Providing a non-intrusive microservice governance solution, ASM supports full-lifecycle management and traffic management and is compatible with the Kubernetes and Istio ecosystems. It hosts functions such as load balancing, outlier detection, and rate limiting.

Process Description

The process of enabling Istio for a cluster is shown in the following figure.

Figure 2-13 Process of enabling Istio for a cluster



2.2.2 Preparations

Before enabling Istio for a cluster, perform the following operations.

Creating a VPC

Virtual Private Cloud (VPC) provides a logically isolated, configurable, and manageable virtual network environment, improving resource security and simplifying network deployment.

Step 1 Log in to the VPC console.

Step 2 Click Create VPC in the upper right corner.

Step 3 Set the parameters as prompted and click Create Now.

----End

Creating a Key Pair

Create a key pair for identity authentication upon remote node login.

Step 1 Log in to the Elastic Cloud Server (ECS) console.

Step 2 In the navigation pane, choose **Key Pair**. On the **Key Pair** page, click **Create Key Pair** in the upper right corner.

Step 3 Enter a key pair name and click **OK**.

Step 4 Manually or automatically download the private key file. The file name is the specified key pair name with a suffix of **.pem**. Securely store the private key file. In the dialog box displayed, click **OK**.

NOTE

For security purposes, a key pair can be downloaded only once. Keep it secure to ensure successful login.

----End

Creating a Load Balancer

A load balancer will be used as the external access entry of a service mesh, which will route the traffic to backend services.

Step 1 Log in to the Elastic Load Balance (ELB) console.

Step 2 Click Create Elastic Load Balancer in the upper right corner.

Step 3 **VPC** and **Subnet**: Select the VPC and subnet created in **Creating a VPC**, configure other parameters as prompted, and click Apply Now.

----End

Creating a Cluster

Step 1 Log in to the Cloud Container Engine (CCE) console.

Step 2 In the navigation pane, choose **Resource Management** > **Clusters**. Then, click **Create CCE Cluster** in the upper right corner.

Step 3 On the **Configure** page, configure the following parameters and retain the default values for other parameters.

- **Cluster Name**: Enter a cluster name, for example, **cluster-test**.
- **VPC** and **Subnet**: Select the VPC and subnet created in **Creating a VPC**.

Step 4 Click **Next: Create Node**, configure the following parameters, and retain the default values for other parameters.

- **Specifications**: 4 vCPUs and 8 GB memory.

- **Login Mode**: Select the key pair created in **Creating a Key Pair** for identity authentication upon remote node login.

Step 5 Click **Next: Install Add-on** and select the add-ons to be installed in the **Install Add-on** step.

System resource add-on must be installed. **Advanced functional add-on** is optional.

Step 6 Click **Next: Confirm**. Read the product constraints and select **I am aware of the above limitations**. Review the configured parameters and specifications.

Step 7 **Submit** the order.

It takes about 6 to 10 minutes to create a cluster. You can click **Back to Cluster List** to perform other operations on the cluster or click **Go to Cluster Events** to view the cluster details.

----End

(Optional) Creating a Namespace

Step 1 Log in to the CCE console.

Step 2 In the navigation pane, choose **Resource Management** > **Namespaces**. Then, click **Create Namespace** in the upper right corner.

Step 3 Enter the namespace name and select the created cluster.

Step 4 Click **OK**.

----End

Creating a Workload and a Service

Step 1 Log in to the CCE console.

Step 2 Choose **Workloads** from the navigation pane, click the **Deployments** tab, click **Create Deployment** in the upper right corner.

Step 3 Create a workload and a Service by referring to *CCE User Guide*.

----End

2.2.3 Creating a Mesh

ASM allows you to create a service mesh of the Basic edition, which is a standard service mesh available for commercial use.

Procedure

Step 1 Log in to the ASM console and click **Create Mesh**.

Step 2 Configure the following parameters and retain the default values for other parameters.

- **Mesh Edition**

  The default value is **Basic edition**.

- **Mesh Name**
  Enter the mesh name.
- **Istio Version**
  Select the Istio version supported by the mesh.
- **Cluster**
  Select the cluster created in **Creating a Cluster**.
- **Mesh Control Plane Node**
  To achieve HA, select two or more nodes from different AZs.

Step 3 Review the mesh configuration in the **Configuration List** on the right of the page and click **Submit**.

It takes about 1 to 3 minutes to create a mesh. If the mesh status changes from **Installing** to **Running**, the mesh is successfully created.

----End

2.3 Configurable Grayscale Release

2.3.1 Overview

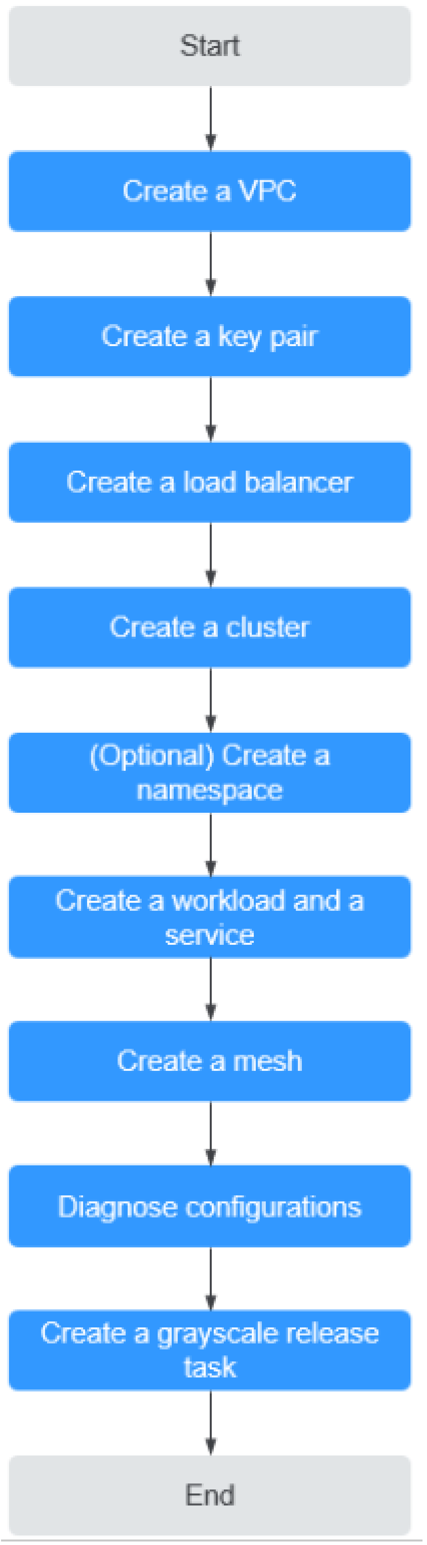
A grayscale release is a smooth iteration mode for version upgrade. During the upgrade, some users use the new version, while other users continue to use the old version. After the new version is stable and ready, it gradually takes over all the live traffic.

This section describes how to create a VPC and a grayscale version to complete a grayscale release.

Process Description

The following figure shows the grayscale release process.

Figure 2-14 Process of creating a grayscale release task



2.3.2 Preparations

Before creating a grayscale release task, complete the following preparations.

Creating a VPC

Virtual Private Cloud (VPC) provides a logically isolated, configurable, and manageable virtual network environment, improving resource security and simplifying network deployment.

Step 1 Log in to the VPC console.

Step 2 Click Create VPC in the upper right corner.

Step 3 Set the parameters as prompted and click Create Now.

----End

Creating a Key Pair

Create a key pair for identity authentication upon remote node login.

Step 1 Log in to the Elastic Cloud Server (ECS) console.

Step 2 In the navigation pane, choose **Key Pair**. On the **Key Pair** page, click **Create Key Pair** in the upper right corner.

Step 3 Enter a key pair name and click **OK**.

Step 4 Manually or automatically download the private key file. The file name is the specified key pair name with a suffix of **.pem**. Securely store the private key file. In the dialog box displayed, click **OK**.

NOTE

For security purposes, a key pair can be downloaded only once. Keep it secure to ensure successful login.

----End

Creating a Load Balancer

A load balancer will be used as the external access entry of a service mesh, which will route the traffic to backend services.

Step 1 Log in to the Elastic Load Balance (ELB) console.

Step 2 Click Create Elastic Load Balancer in the upper right corner.

Step 3 **VPC** and **Subnet**: Select the VPC and subnet created in **Creating a VPC**, configure other parameters as prompted, and click Apply Now.

----End

Creating a Cluster

Step 1 Log in to the Cloud Container Engine (CCE) console.

Step 2 In the navigation pane, choose **Resource Management** > **Clusters**. Then, click **Create CCE Cluster** in the upper right corner.

Step 3 On the **Configure** page, configure the following parameters and retain the default values for other parameters.

- **Cluster Name**: Enter a cluster name, for example, **cluster-test**.
- **VPC** and **Subnet**: Select the VPC and subnet created in **Creating a VPC**.

Step 4 Click **Next: Create Node**, configure the following parameters, and retain the default values for other parameters.

- **Specifications**: 4 vCPUs and 8 GB memory.
- **Login Mode**: Select the key pair created in **Creating a Key Pair** for identity authentication upon remote node login.

Step 5 Click **Next: Install Add-on** and select the add-ons to be installed in the **Install Add-on** step.

System resource add-on must be installed. **Advanced functional add-on** is optional.

Step 6 Click **Next: Confirm**. Read the product constraints and select **I am aware of the above limitations**. Review the configured parameters and specifications.

Step 7 **Submit** the order.

It takes about 6 to 10 minutes to create a cluster. You can click **Back to Cluster List** to perform other operations on the cluster or click **Go to Cluster Events** to view the cluster details.

----End

(Optional) Creating a Namespace

Step 1 Log in to the CCE console.

Step 2 In the navigation pane, choose **Resource Management** > **Namespaces**. Then, click **Create Namespace** in the upper right corner.

Step 3 Enter the namespace name and select the created cluster.

Step 4 Click **OK**.

----End

Creating a Workload and a Service

Step 1 Log in to the CCE console.

Step 2 Choose **Workloads** from the navigation pane, click the **Deployments** tab, click **Create Deployment** in the upper right corner.

Step 3 Create a workload and a Service by referring to *CCE User Guide*.

----End

Creating a Service Mesh

Step 1 Log in to the ASM console and click **Create Mesh**.

Step 2 Set **Mesh Name** to **asmtest**, select the cluster named **cluster-test** created in **Creating a Cluster**, and select a node on which the Istio control plane is installed. Select two or more nodes in different AZs.

Step 3 Click **Show Advanced Settings**. In the **Sidecar Configuration** area, select the namespace named **default** and enable **Restart Existing Services**.

Step 4 Review the mesh configuration in the **Configuration List** on the right of the page and click **Submit**.

It takes about 1 to 3 minutes to create a service mesh. If the service mesh status changes from **Installing** to **Running**, the service mesh is successfully created.

----End

Diagnosing Configurations

ASM diagnoses all services in a managed cluster. Grayscale release can be performed only for services that are diagnosed as normal.

Step 1 Log in to the ASM console, click the service mesh name **asmtest** to access its details page.

Step 2 In the navigation pane on the left, choose **Service Management**, select **Namespace: default** from the drop-down list box, and view the **Configuration Diagnosis Result** of **servicetest**.

Step 3 If the **Configuration Diagnosis Result** is **Abnormal**, click **Fix** to fix the issues.

----End

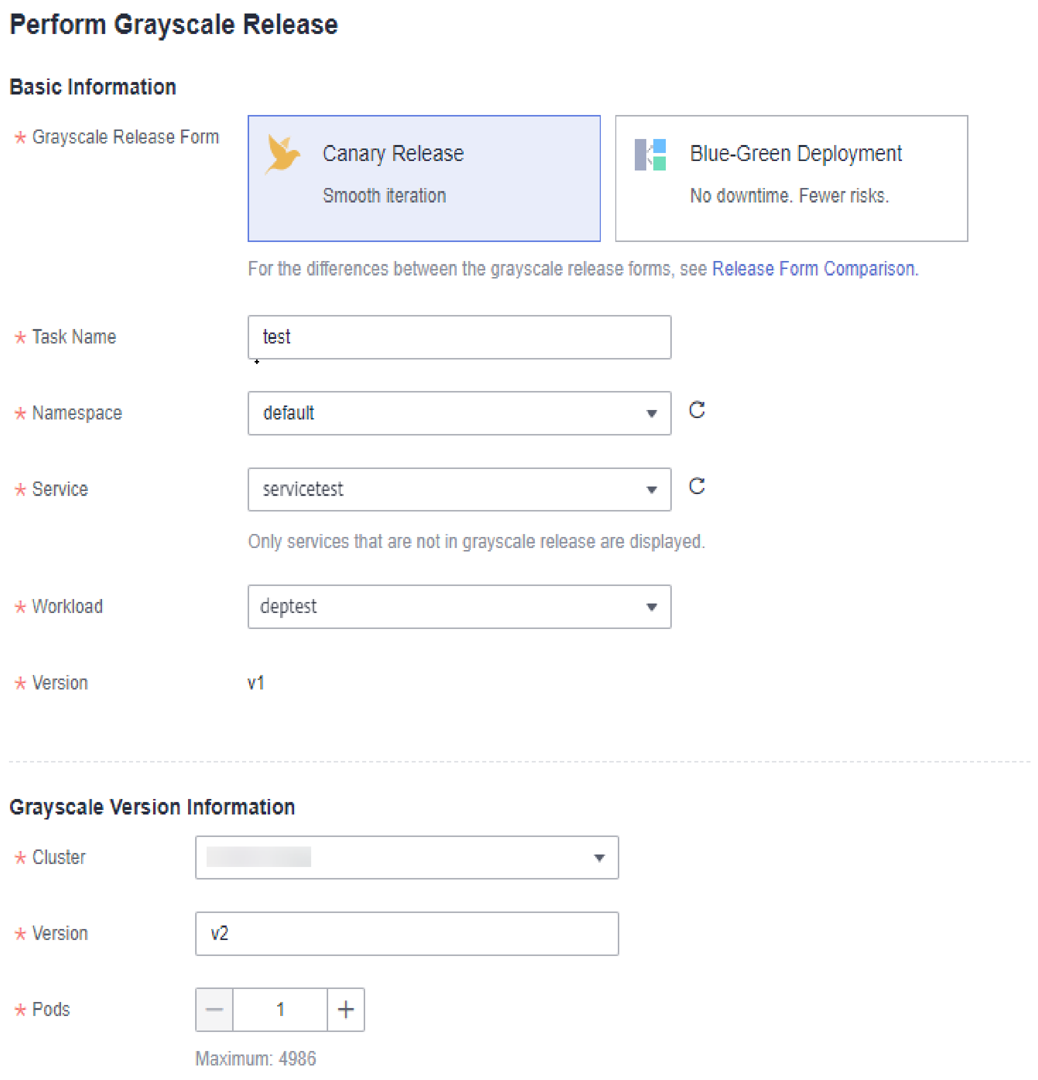
2.3.3 Grayscale Release

Creating a Grayscale Release Task

Step 1 Log in to the ASM console and click in the **asmtest** mesh.

Step 2 Set **Task Name** to **test** and select **servicetest** created in Creating a Workload and a Service for **Service**. (**Workload** is automatically set to **deptest**.) Configure other basic information and grayscale version information and click **Release**.

Figure 2-15 Creating a grayscale release task



Perform Grayscale Release

Basic Information

* Grayscale Release Form	[✓ Canary Release — Smooth iteration] Blue-Green Deployment — No downtime. Fewer risks.
	For the differences between the grayscale release forms, see Release Form Comparison.
* Task Name	test
* Namespace	default
* Service	servicetest
	Only services that are not in grayscale release are displayed.
* Workload	deptest
* Version	v1

Grayscale Version Information

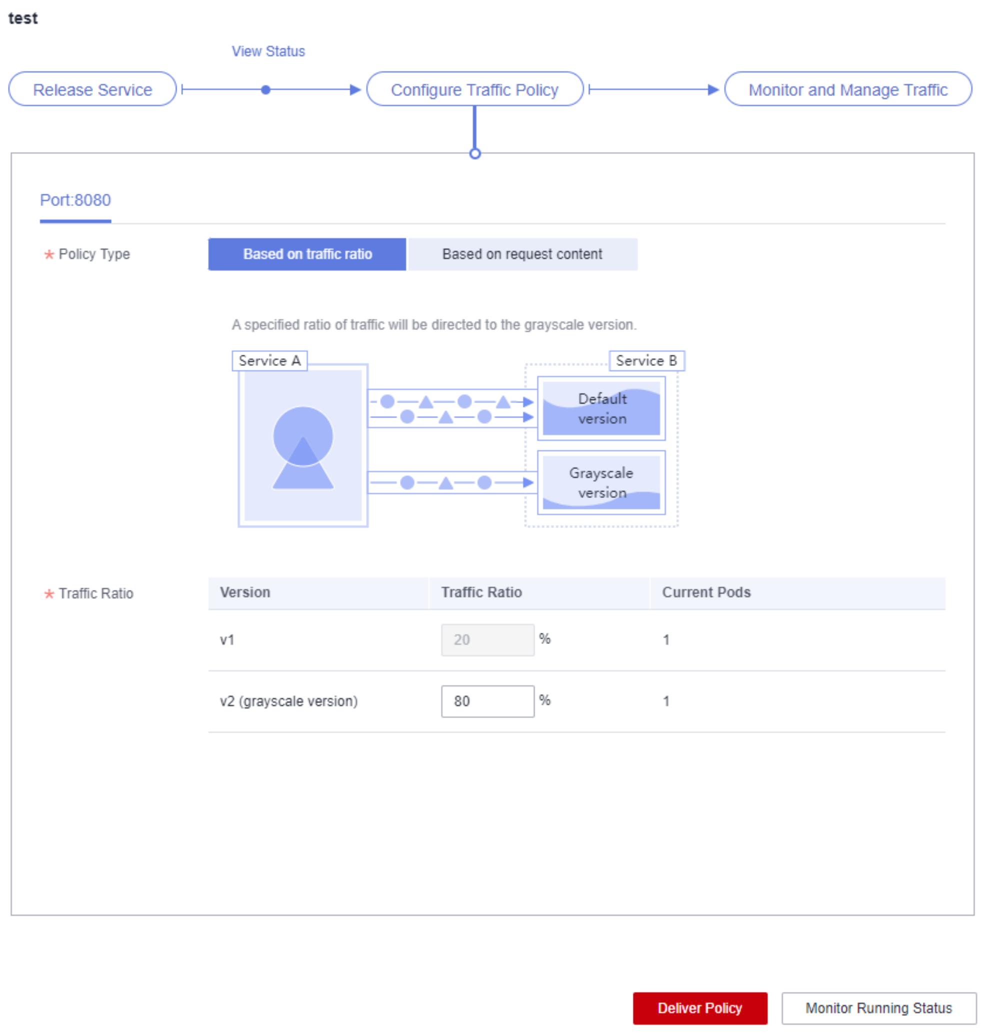
* Cluster	
* Version	v2
* Pods	1
	Maximum: 4986

NOTE

If you cannot select **servicetest**, check whether the service is normal. If the service is abnormal, fix the issues and try again.

Step 3 Click **Configure Traffic Policy**, set the policy type to **Based on traffic ratio**, and set the **traffic ratio** of v2 to 80%.

Figure 2-16 Configuring a traffic policy



* Policy Type: Based on traffic ratio | Based on request content

A specified ratio of traffic will be directed to the grayscale version.

* Traffic Ratio

Version	Traffic Ratio	Current Pods
v1	20 %	1
v2 (grayscale version)	80 %	1

Step 4 Click **Deliver Policy**.

It takes several seconds for the traffic policy to take effect. You can view the running of the grayscale version on the **View Status** page.

----End

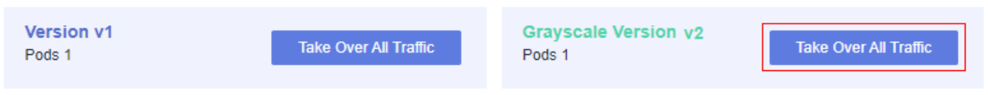
Switching All Traffic to the Grayscale Version

Check whether the number of resources in v2 matches that in v1. After confirming that v2 is able to serve all the traffic of v1, switch all the traffic from v1 to v2.

Step 1 On the **Grayscale Release** page, click **test** and then click **Monitor and Manage Traffic**.

Step 2 Click **Take Over All Traffic** next to v2.

Figure 2-17 Taking over all traffic



Step 3 Click **OK**. A message is displayed in the upper right corner, indicating that the traffic is taken over successfully.

----End

Bringing the Original Version Offline

After v2 takes over all the traffic from v1, bring v1 offline to release its resources.

Step 1 On the **Grayscale Release** page, click **test** and then click **Monitor and Manage Traffic**.

Step 2 On the displayed page, when the traffic percentage of v2 is **100%**, v2 has taken over all traffic of v1. Click **Terminate Task**.

Step 3 Click **OK**.

The v1 version is brought offline and the test grayscale task is deleted.

----End

3 User Guide

3.1 Creating a Mesh

3.1.1 Creating a Service Mesh

ASM allows you to create a service mesh of the Basic edition, which is a standard service mesh available for commercial use.

Prerequisites

A CCE cluster is available.

Constraints

- ASM depends on the domain name resolution of CoreDNS. Before creating a service mesh for a cluster, ensure that the cluster has required resources and CoreDNS is running normally.
- When you enable Istio for a cluster, you must enable port 7443 in the inbound direction of the security group to which the worker node belongs, for automatic sidecar injection and callback. If you use the default security group created by CCE, this port is automatically enabled. If you create a security group rule, manually enable port 7443 to ensure that sidecars can be automatically injected.

Procedure

Step 1 Log in to the ASM console.

Step 2 Click Create Mesh in the upper right corner.

Step 3 Configure the following parameters.

- **Mesh Edition**

  Only service meshes of the Basic edition are supported.

- **Mesh Name**

Enter a service mesh name, which consists of 4 to 64 characters. It must start with a lowercase letter and cannot end with a hyphen (-). Only lowercase letters, digits, and hyphens (-) are allowed.

Service mesh names under the same account must be unique and cannot be modified after creation.

- **Istio Version**

  Select the Istio version supported by the service mesh.

- **Cluster**

  Select the target cluster from the cluster list or enter the target cluster name in the upper right corner of the list to search for it. You can select only the clusters which versions are supported by the current mesh version.

- **Mesh Control Plane Node**

  To install the control plane components for the service mesh of the Basic edition in your cluster, you need to select a node for installation. If HA is required, you can select two or more nodes from different AZs.

  The selected node is labeled with **istio:master**, and the components are scheduled to this node.

Step 4 (Optional) Configure advanced settings.

- **Sidecar Configuration**

  Select a namespace and label it with **istio-injection=enabled**. All pods in the namespace will be injected with an istio-proxy sidecar.

  You can inject a sidecar in **Mesh Configuration** > **Sidecar Management** after the mesh is created. For details, see **Injecting a Sidecar**.

- **Restart Existing Services**

  : Pods of the existing services in the namespace will be restarted, which will temporarily interrupt your services. The **istio-proxy** sidecar is automatically injected into the pods of the existing services.

  : The **istio-proxy** sidecar cannot be automatically injected into the pods of the existing services. You need to manually restart the workloads on the CCE console to inject the sidecar.

Step 5 Review the service mesh configuration in the **Configuration List** on the right of the page and click **Submit**.

It takes about 1 to 3 minutes to create a service mesh. If the service mesh status changes from **Installing** to **Running**, the service mesh is successfully created.

NOTE

When the service mesh is enabled, the following operations are performed:

- Helm orchestrates the application into a Release as the resource of the service mesh control plane.
- A security group is enabled for the nodes to allow the inbound traffic for port 7443 to support automatic sidecar injection.

----End

3.2 Mesh Management

3.2.1 Uninstalling a Mesh

Scenario

When a mesh is no longer needed, you can uninstall it.

Constraints

- To uninstall a mesh in which a grayscale release task is running, you need to complete the grayscale release first.
- You need to ensure available nodes exist in the clusters for running the cleanup task to avoid uninstallation failure.

Procedure

Step 1 Log in to the ASM console.

Step 2 Click in the target mesh.

Step 3 On the dialogue box displayed, select whether to restart existing services and read the precautions.

By default, existing services are not restarted during the uninstallation. The injected istio-poxy sidecar is removed only after the existing services are restarted. If you want to restart the services, select **Yes**. Restarting the services will interrupt your services temporarily.

NOTE

You are advised to restart existing services to avoid the following exceptions: If the cluster enables the current mesh again after it is uninstalled, gateway access failed.

- Uninstalling a mesh will uninstall its control plane components and data plane sidecars.
- After the uninstallation, the gateway of the application cannot be used anymore. Use Service to expose the service.

  To update the gateway, log in to the CCE console and choose **Resource Management** > **Network** > **Services** to expose the service.
- Uninstalling a mesh will delete the labels of the mesh exclusive nodes, but the Istio-master node will not be automatically deleted. You can delete it on the CCE console.

  To view node information, choose **Resource Management** > **Nodes** on the CCE console.

----End

3.3 Service Management

3.3.1 Configuration Diagnosis

ASM diagnoses all services in a managed cluster. Traffic management and grayscale release are available only for normal services.

Constraints

- If multiple services correspond to one deployment, these services cannot be added to the mesh. Otherwise, functions such as grayscale release or gateway access may fail.
- If the workload of a service uses the host network mode (**hostNetwork: true** is configured for the pod), sidecars cannot be injected for the service.

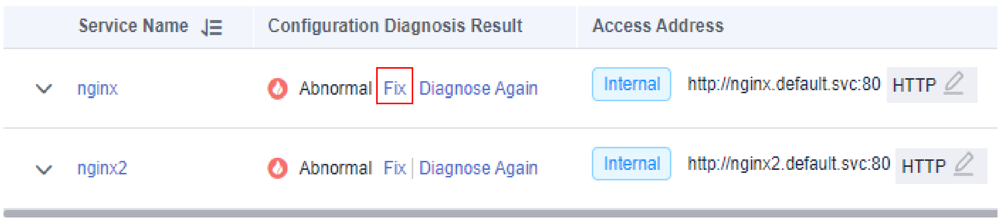
Service Diagnosis

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane, choose **Service Management**. The diagnosis results of services are displayed in the **Configuration Diagnosis Result** column.

If a service is abnormal, click **Fix** to fix the issues. For details, see **Service Issue Fixing**.

Figure 3-1 Service diagnosis



Service Name	Configuration Diagnosis Result	Access Address
nginx	Abnormal Fix Diagnose Again	Internal http://nginx.default.svc:80 HTTP
nginx2	Abnormal Fix | Diagnose Again	Internal http://nginx2.default.svc:80 HTTP

Step 3 After the issues are fixed, you can click **Diagnose Again** to diagnose the service again.

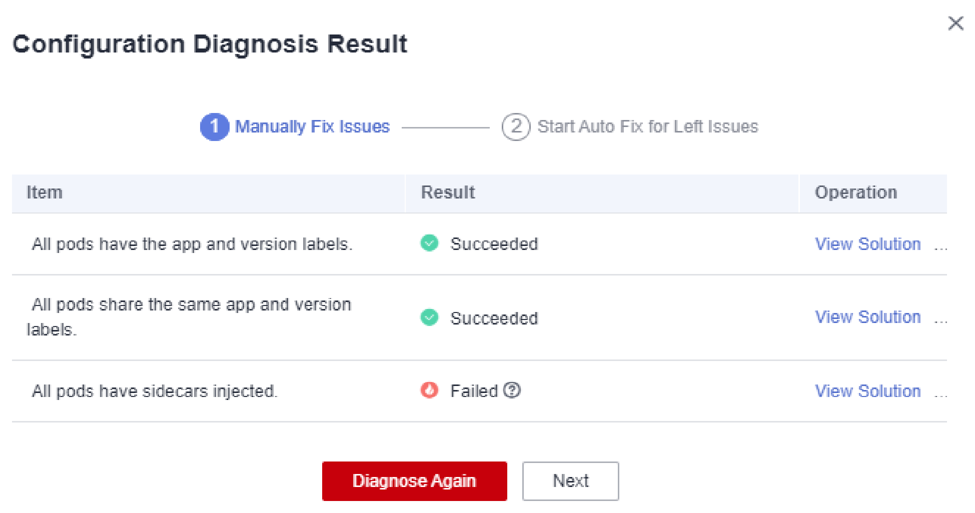
----End

Service Issue Fixing

If a service is abnormal, you need to manually fix the abnormal items and then perform auto fix for left issues.

Step 1 Click **Fix** in the row of the abnormal service. If issues to be fixed manually exist, click **View Solution** to see how to fix them.

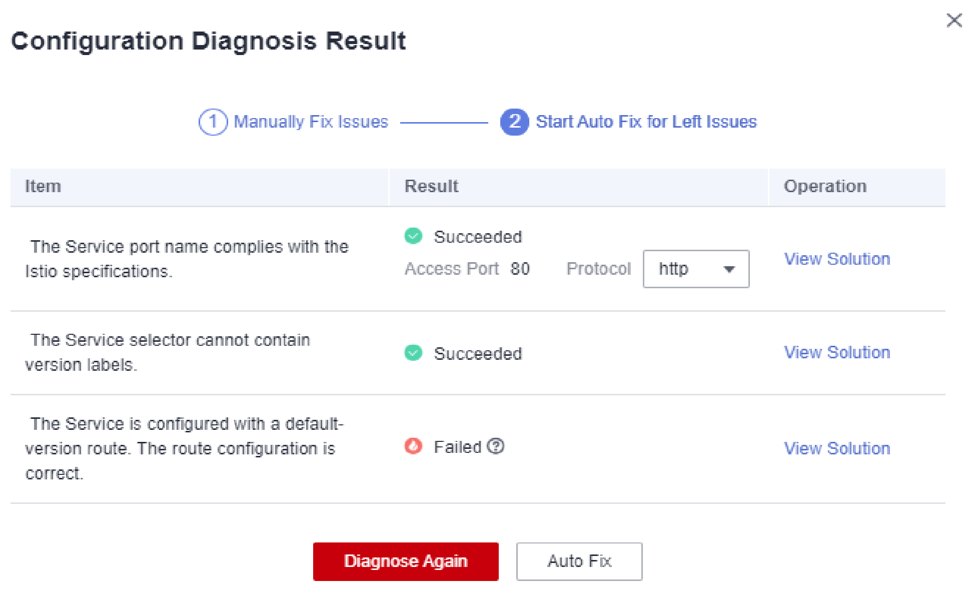
Figure 3-2 Manually fixing issues


Configuration Diagnosis Result

Item	Result	Operation
All pods have the app and version labels.	Succeeded	View Solution ...
All pods share the same app and version labels.	Succeeded	View Solution ...
All pods have sidecars injected.	Failed	View Solution ...

Step 2 Click **Next** to go to the auto fix page, and click **Auto Fix** to automatically fix left issues.

Figure 3-3 Starting auto fix for left issues


Configuration Diagnosis Result

Item	Result	Operation
The Service port name complies with the Istio specifications.	Succeeded Access Port 80 Protocol http	View Solution
The Service selector cannot contain version labels.	Succeeded	View Solution
The Service is configured with a default-version route. The route configuration is correct.	Failed	View Solution

NOTE

- If left issues cannot be fix automatically, click **View Solution** and fix them manually.
- Auto fix does not support Services which have configured gateways or have created grayscale release tasks.
- If the service is not displayed in the service list, check whether the corresponding workload exists.

----End

3.3.2 Manual Fixing Items

3.3.2.1 All Pods Have the app and version Labels

Description

All pods of a Service must be labeled with **app** and **version**. **app** traces traffic in traffic monitoring, and **version** distinguishes different versions in grayscale release. If a pod is not labeled with **app** or **version**, this item is abnormal.

Rectification Guide

The labels of pods are configured in **spec.template.metadata.labels** of the Deployment. The recommended configuration is as follows:

```
labels:
  app: {serviceName}
  version: v1
```

CAUTION

Modifying or deleting the Deployment will trigger pod rolling upgrade, which may cause temporary service interruption. Therefore, perform the operation at a proper time.

Step 1 Copy the original workload configuration and save it as a YAML file.

kubectl get deployment {deploymentName} **-n** {namespace} **-o yaml >** {deploymentName}**-deployment.yaml**

For example:

kubectl get deployment productpage -n default -o yaml > productpage-deployment.yaml

Step 2 Modify the **productpage-deployment.yaml** file. If the file does not contain **app** and **version**, add them. You are advised to set **app** to the Service name and the **version** to **v1**.

Step 3 Delete the original workload.

kubectl delete deployment {oldDeploymentName} **-n** {namespace}

Step 4 Apply the new workload configuration.

kubectl apply -f productpage-deployment.yaml

----End

NOTE

For clusters of v1.15 or earlier, you can modify pod labels on the CCE console, but residual ReplicaSets may exist.

To check whether there are residual ReplicaSets, perform the following steps:

1. Query the ReplicaSets of the Deployment.

   kubectl get replicaset | grep {deploymentName}

2. Find the ReplicaSets containing more than one pod. These ReplicaSets may be residual after label modification. You need to delete the old ReplicaSets.

   kubectl delete replicaset {replicaSetName} **-n** {namespace}

3.3.2.2 All Pods Share the Same app and version Labels

Description

All pods of a Service must share the same **app** and **version** labels. **app** traces traffic in traffic monitoring, and **version** distinguishes different versions in grayscale release. If pods with different **app** or **version** labels exist, this item is abnormal.

Rectification Guide

The labels of pods are configured in **spec.template.metadata.labels** of the Deployment. The recommended configuration is as follows:

```
labels:
  app: {serviceName}
  version: v1
```

To modify the labels of multiple pods to the same value, perform the following steps:

Step 1 View the labels configured for **spec.selector**.

kubectl get svc {serviceName} **-o yaml**

For example, the labels are **app: ratings** and **release: istio-bookinfo**.

Step 2 Search for the pods of a Service by label.

kubectl get pod -n {namespace} **-l app=ratings,release=istio-bookinfo**

NOTE

{namespace} is the same as the namespace of the Service.

Step 3 Find the workload of a pod by the pod name.

kubectl get deployment {deploymentName} **-n** {namespace}

NOTE

- Generally, the pod name is in the format of **{deploymentName}-{random character string}-{random character string}**.
- If no workload of a pod is found by the pod name, you need to delete residual ReplicaSets.

  To check whether there are residual ReplicaSets, perform the following steps:

  1. Query the ReplicaSets of the Deployment.

     kubectl get replicaset | grep {deploymentName}

  2. Find the ReplicaSets containing more than one pod. These ReplicaSets may be residual after label modification. You need to delete the old ReplicaSets.

     kubectl delete replicaset {replicaSetName} **-n** {namespace}

Step 4 For details about how to modify the **app** and **version** labels of a pod, see **Rectification Guide**.

----End

3.3.2.3 All Pods Have Sidecars Injected

Description

An **istio-proxy** container must exist in all pods of a Service. Otherwise, this item is abnormal.

Rectification Guide

Step 1 Log in to the ASM console and click the name of the service mesh that the Service is added to. Choose **Mesh Configuration** in the navigation pane, click the **Sidecar Management** tab, and check whether a sidecar is injected into the namespace that the Service belongs to.

- If no, go to **Step 2**.
- If yes, go to **Step 3**.

Step 2 Inject a sidecar.

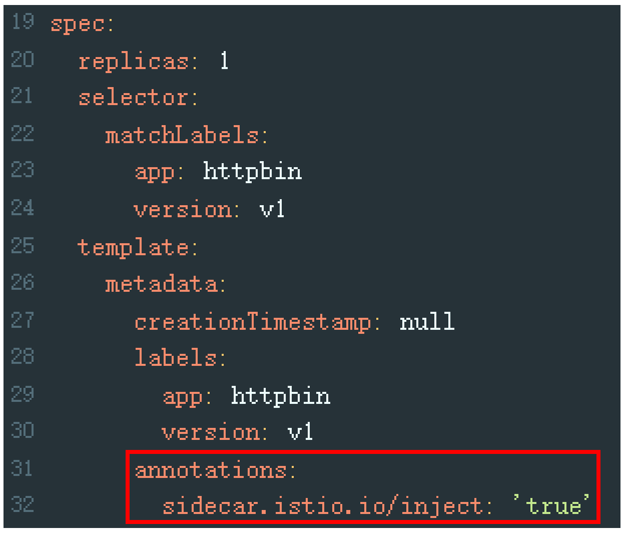
You can inject sidecars for pods of all workloads in the namespace. For details, see **Injecting a Sidecar**. You can also inject sidecars for a workload as follows:

1. Label the namespace where the workload is located with **istio-injection=enabled**.

   kubectl label ns <namespace> **istio-injection=enabled**

2. Add the **annotations** field for the workload on the CCE console.

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



```
spec:
  replicas: 1
  selector:
    matchLabels:
      app: httpbin
      version: v1
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: httpbin
        version: v1
      annotations:
        sidecar.istio.io/inject: 'true'
```

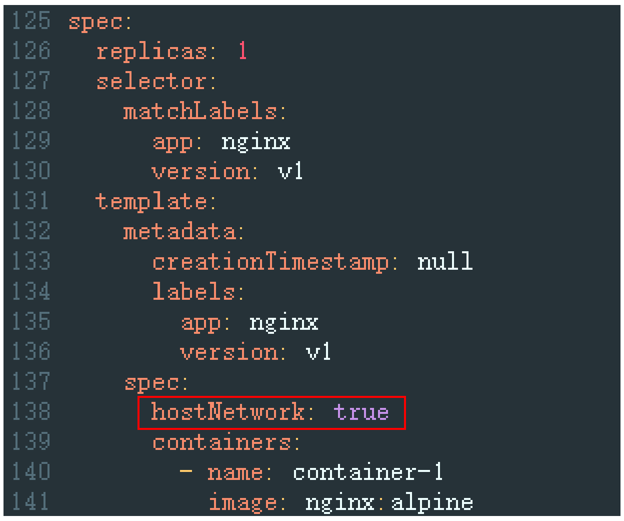
For more details about sidecar injection, see **Installing the Sidecar**.

Step 3 If namespace injection is enabled for the cluster but no sidecar is injected into the pod, you need to manually restart the pod on the CCE console as follows:

On the CCE console, choose **More** > **Redeploy** in the **Operation** column of the target workload.

Step 4 Check whether the host network mode is configured for the workload as follows:

On the CCE console, choose **More** > **Edit YAML** in the **Operation** column of the target workload, and check whether **spec.template.spec.hostNetwork: true** is configured. If yes, check whether this field can be deleted or set to **false**. Otherwise, sidecars cannot be injected.



```
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
      version: v1
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: nginx
        version: v1
    spec:
      hostNetwork: true
      containers:
        - name: container-1
          image: nginx:alpine
```

Step 5 Check whether the number of pods exceeds the service mesh scale.

If the number exceeds 200, the excess pods cannot be injected with sidecars.

----End

3.3.3 Auto Fixing Items

3.3.3.1 The Service Port Name Complies with the Istio Specifications

Description

The Service port name must contain the specified protocol and prefix and must be in the following format:

```
name: <protocol>[-<suffix>]
```

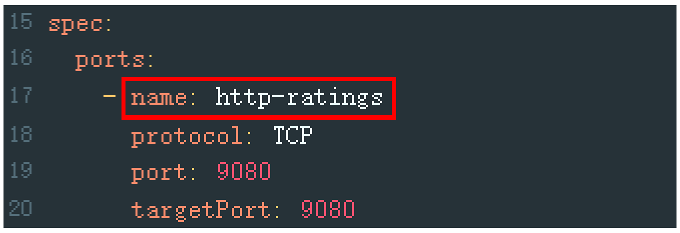
<protocol> can be **http**, **tcp**, or **grpc**. Istio provides routing capabilities based on protocols defined on ports. For example, **name: http-service0** and **name: tcp** are valid port names, while **name: httpforecast** is not.

If the Service port name is invalid, this item is abnormal.

Rectification Guide

Step 1 Log in to the CCE console.

Step 2 In the navigation pane on the left, choose **Resource Management** > **Network**, find the target service by cluster name and namespace in **Services** tab page, click **Edit YAML**, view its Service protocol, and add a protocol type before the target service name as shown in the following figure.



```
spec:
  ports:
    - name: http-ratings
      protocol: TCP
      port: 9080
      targetPort: 9080
```

Step 3 Click **OK**.

----End

3.3.3.2 The Service Selector Cannot Contain version Labels

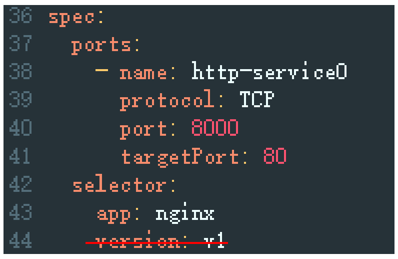
Description

The **spec.selector** of a Service cannot be labeled with **version**. Otherwise, this item is abnormal.

Rectification Guide

Step 1 Log in to the CCE console.

Step 2 In the navigation pane on the left, choose **Resource Management** > **Network**, find the target Service by cluster name and namespace in **Services** tab page, click **Edit YAML**, view its spec.selector, and delete the **version** label.



```
spec:
  ports:
    - name: http-service0
      protocol: TCP
      port: 8000
      targetPort: 80
  selector:
    app: nginx
   ~~version: v1~~
```

----End

3.3.3.3 The Service Is Configured with a Default-version Route and The Route Configuration Is Correct

Description

Istio defines service traffic routing rules in **VirtualService** and **DestinationRule**. Therefore, you need to configure **VirtualService** and **DestinationRule** for each service. The following rules must be met:

- All ports of a Service must be configured in **VirtualService**.
- The protocol type in **VirtualService** must be the same as that of the ports of a Service.
- The default service version must be configured in **VirtualService** and **DestinationRule**.

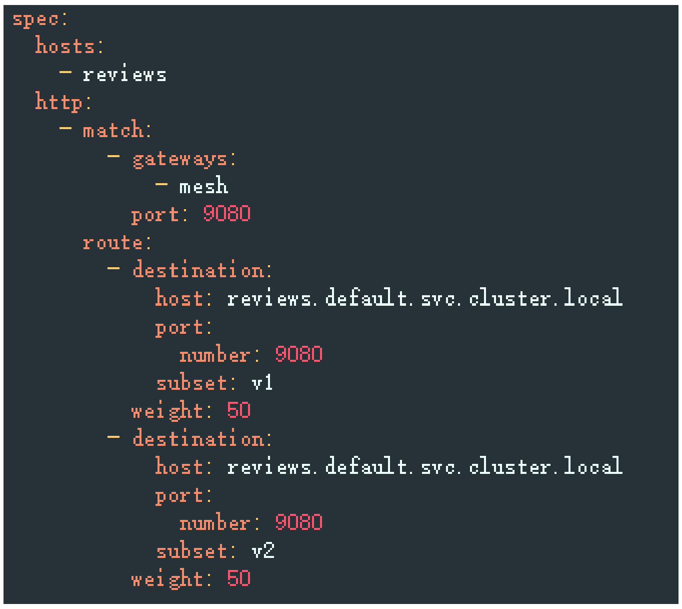
NOTE

If the check result changes, the port number or port name of a Service may be changed.

Rectification Guide

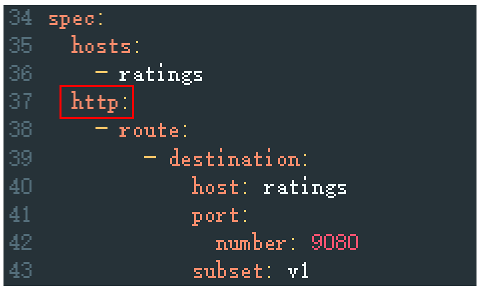
Step 1 Log in to the ASM console. Select the mesh where the service is located. In the navigation pane on the left, choose **Mesh Configuration**, click **Istio Resource Management**, and select **Istio resources: virtualservices** and the namespace to which the service belongs.

Step 2 Ensure that all ports of the Service are configured in **VirtualService**.



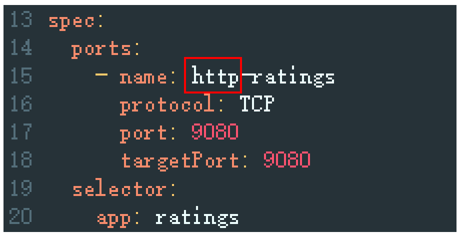
```
spec:
  hosts:
    - reviews
  http:
    - match:
        - gateways:
            - mesh
          port: 9080
      route:
        - destination:
            host: reviews.default.svc.cluster.local
            port:
              number: 9080
            subset: v1
          weight: 50
        - destination:
            host: reviews.default.svc.cluster.local
            port:
              number: 9080
            subset: v2
          weight: 50
```

Step 3 Ensure that the protocol type in **VirtualService** is the same as that of the ports of the Service.



```
spec:
  hosts:
    - ratings
  http:
    - route:
        - destination:
            host: ratings
            port:
              number: 9080
            subset: v1
```

Figure 3-4 Protocol type in VirtualService



```
spec:
  ports:
    - name: http-ratings
      protocol: TCP
      port: 9080
      targetPort: 9080
  selector:
    app: ratings
```

Figure 3-5 Port protocol type of the Service

----End

3.4 Gateway Management

3.4.1 Adding a Gateway

A gateway enables unified entry, traffic management, security, and service isolation.

Prerequisites

Gateways use load balancers of ELB to provide network access. Before adding a gateway, you need to create a load balancer.

When creating a load balancer, you need to ensure that it belongs to the same VPC as the cluster.

Procedure

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane on the left, choose **Gateway Management** and click **Add Gateway**.

Step 3 Configure the following parameters.

- **Gateway Name**

  Enter a gateway name. Enter 4 to 59 characters starting with a lowercase letter and ending with a lowercase letter or digit. Only lowercase letters, digits, and hyphens (-) are allowed.

- **Cluster**

  Select the cluster to which the gateway belongs.

- **Load Balancer**
- **Listener**

  Gateways configure a listener for the load balancer, which listens to requests from the load balancer and distributes traffic.

  – **External Protocol**

    Select one to match the protocol type of your service. **HTTP**, **gRPC**, **TCP**, **TLS**, and **HTTPS** are supported.

  – **External Port**

    Enter the port number exposed in the Load Balancer Service address. The port number can be specified randomly.

  – **TLS Termination**

    If **External Protocol** is **HTTPS**, **TLS Termination** is enabled and cannot be disabled.

    If **External Protocol** is **TLS**, you can enable or disable **TLS Termination**. If you enable TLS termination, bind a certificate to support TLS-based data transmission encryption and authentication. If you disable TLS termination, encrypted TLS data will be directly forwarded.

  – **Secret Certificate**

    ▪ When configuring a TLS protocol with TLS termination enabled, you need to bind a certificate to support TLS-based data transmission encryption and authentication.

    ▪ When configuring the HTTPS protocol, you need to bind a secret certificate.

  – **Earliest TLS Version Supported/Latest TLS Version Supported**

When configuring a TLS protocol with TLS termination enabled or an HTTPS protocol, you can select the earliest and latest TLS versions.

Step 4 (Optional) Configure routing parameters.

When the access address of a request matches the forwarding policy (which consists of a domain name and URL. If the domain name is left empty, the ELB IP address is used by default), the request is forwarded to the corresponding target Service for processing. Click +. The **Add Route** dialog box is displayed.

- **Domain Name**

  Enter the external domain name of the service. If this parameter is left blank, the IP address of the load balancer is used by default. If you enable TLS termination, enter a domain name configured in the certificate for SNI domain name verification.

- **URL Matching Rule**
  - **Prefix**: A URL can be accessed if its prefix is the same as that you configure. For example, **/healthz/v1** and **/healthz/v2**.
  - **Exact**: Only the URL that fully matches the values you set can be accessed. For example, if the URL is set to **/healthz**, only **/healthz** can be accessed.

- **URL**

  Mapping URL supported by the service, for example, **/example**.

- **Namespace**

  Select the namespace to which the gateway belongs.

- **Target Service**

  Service of the gateway. Select a value from the drop-down list box. The target service is filtered based on the corresponding gateway protocol. For details about the filtering rules, see **Why Cannot I Select the Corresponding Service When Adding a Route?**

  The service which configuration diagnosis fails cannot be selected. You need to fix the issues first. For details, see **Manual Fixing Items** or **Auto Fixing Items**.

- **Access Port**

  Only ports that match external protocols are displayed.

- **Rewrite**

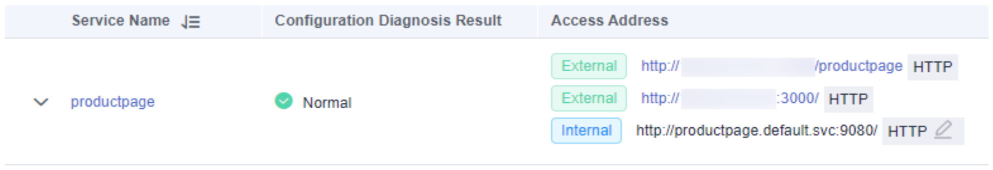
  (This parameter is configurable when the external protocol is HTTP.)

  Rewrite the HTTP URI and host/authority header before forwarding. Disabled by default. To enable it, configure the following parameters:
  - URI: This value is used to rewrite the URI or prefix.
  - Host/Authority Header: This value is used to rewrite the HTTP host/authority header.

Step 5 Click **OK**.

You can obtain the external network access address of the service in the **Service Management** page.

Figure 3-6 External network access address of the service



Service Name	Configuration Diagnosis Result	Access Address
productpage	Normal	External http://[redacted]/productpage HTTP External http://[redacted]:3000/ HTTP Internal http://productpage.default.svc:9080/ HTTP

----End

3.4.2 Adding a Route

Scenario

You can add multiple routes and configure multiple forwarding policies for a created gateway.

Procedure

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane on the left, choose **Gateway Management**, select the target gateway, click **Add Route** in the **Operation** column, and configure the following parameters:

- **Domain Name**

  Enter the external domain name of the service. If this parameter is left blank, the IP address of the load balancer is used by default. If you enable TLS termination, enter a domain name configured in the certificate for SNI domain name verification.

- **URL Matching Rule**
  - **Prefix**: A URL can be accessed if its prefix is the same as that you configure. For example, **/healthz/v1** and **/healthz/v2**.
  - **Exact**: Only the URL that fully matches the values you set can be accessed. For example, if the URL is set to **/healthz**, only **/healthz** can be accessed.

- **URL**

  Mapping URL supported by the service, for example, **/example**.

  NOTE

  The URLs of the same gateway must be unique.

- **Namespace**

  Select the namespace to which the gateway belongs.

- **Target Service**

  Service of the gateway. Select a value from the drop-down list box. The target service is filtered based on the corresponding gateway protocol. For details about the filtering rules, see **Why Cannot I Select the Corresponding Service When Adding a Route?**.

  The service which configuration diagnosis fails cannot be selected. You need to fix the issues first. For details, see **Manual Fixing Items** or **Auto Fixing Items**.

- **Access Port**

  Only ports that match external protocols are displayed.
- **Rewrite**

  (This parameter is configurable when the external protocol is HTTP.)

  Rewrite the HTTP URI and host/authority header before forwarding. Disabled by default. To enable it, configure the following parameters:
  - URI: This value is used to rewrite the URI or prefix.
  - Host/Authority Header: This value is used to rewrite the HTTP host/ authority header.

Step 3 Click **OK**.

----End

3.5 Grayscale Release

3.5.1 Grayscale Release Overview

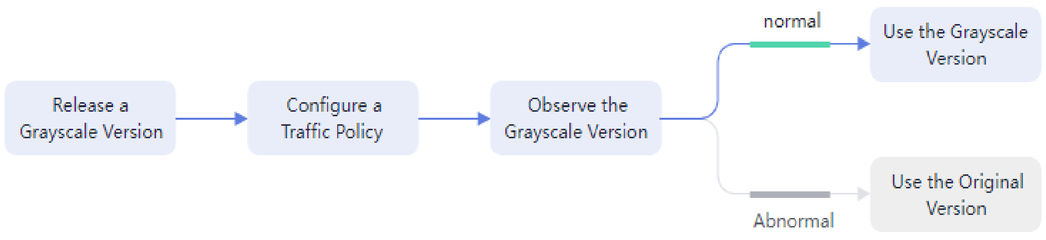
When switching between old and new services, you may be challenged in ensuring the system service continuity. If a new service version is directly released to all users at a time, it can be risky because once an online accident or bug occurs, the impact on users is great. It could take a long time to fix the issue. Sometimes, the version has to be rolled back, which severely affects user experience.

Several release policies are developed for service upgrade: canary release, blue-green deployment, A/B testing, rolling upgrade, and batch suspension of release. Traffic loss or service unavailability caused by releases can be avoided as much as possible. Currently, ASM supports canary release and blue-green deployment.

Canary Release

Canary release is also called grayscale release. It is a smooth iteration mode for version upgrade. During the upgrade, some users use the new version, while other users continue to use the old version. After the new version is stable and ready, it gradually takes over all the live traffic. In this way, service risks brought by the release of the new version can be minimized, the impact of faults can be reduced, and quick rollback is supported.

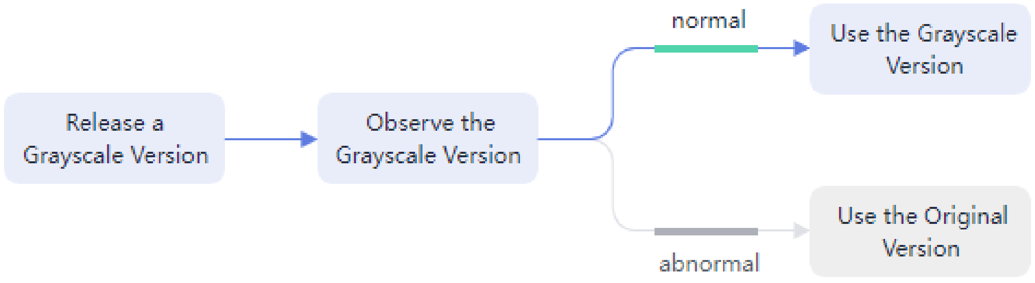
Figure 3-7 Canary release process


Release a Grayscale Version
Configure a Traffic Policy
Observe the Grayscale Version
normal
Use the Grayscale Version
Abnormal
Use the Original Version

Blue-Green Deployment

Blue-green deployment provides a zero-downtime, predictable manner for releasing applications to reduce service interruption during the release. A new version is deployed while the old version is retained. The two versions are online at the same time. The new and old versions work in hot backup mode. The route weight is switched (0 or 100) to enable different versions to go online or offline. If a problem occurs, the version can be quickly rolled back.

Figure 3-8 Blue-green deployment process


Release a Grayscale Version
Observe the Grayscale Version
normal
Use the Grayscale Version
abnormal
Use the Original Version

3.5.2 Creating a Grayscale Release Task

Basic Concepts

- Grayscale version

  Only one grayscale version can be released for a service. You can configure grayscale policies for the version.

- Grayscale policy

  Before releasing a new service version in the production environment and letting it serve all the live traffic, you can add a grayscale version and configure grayscale policies to serve just a proportion of the traffic. After the grayscale version has run stably for a period, it can serve as the default version to take over all traffic in place of the original version in the production environment.

Creating a Grayscale Release Task

Step 1 Log in to the ASM console and go to the **Create a grayscale release task** page by one of the following ways:

- (Shortcut) In the upper right corner of an Enterprise mesh, click .
- (Shortcut) In the center of the target mesh, click **Create a grayscale release task**.
- Create a grayscale release task on the mesh details page.
  a. Click the target mesh and go to its details page, click **Grayscale Release** in the navigation pane on the left.
  b. If no grayscale task is running, click **Create Release Task** in the **Canary Release** or **Blue-Green Deployment** area. If there is an ongoing grayscale task, click **Grayscale Release** in the upper right corner.

Step 2 Configure basic information of the grayscale release task.

- **Grayscale Release Form**

Select **Canary Release** or **Blue-Green Deployment** as required. For details about the differences between the two forms, see **Grayscale Release Overview**.

- **Task Name**

  Customize a grayscale release task name. Enter 4 to 63 characters, starting with a lowercase letter and ending with a letter or digit. Only lowercase letters, digits, and hyphens (-) are allowed.

- **Namespace**

  Select the namespace to which the service belongs.

- **Service**

  Select the service to be released from the drop-down list box. Services that are running grayscale tasks cannot be selected. They are automatically filtered out from the list.

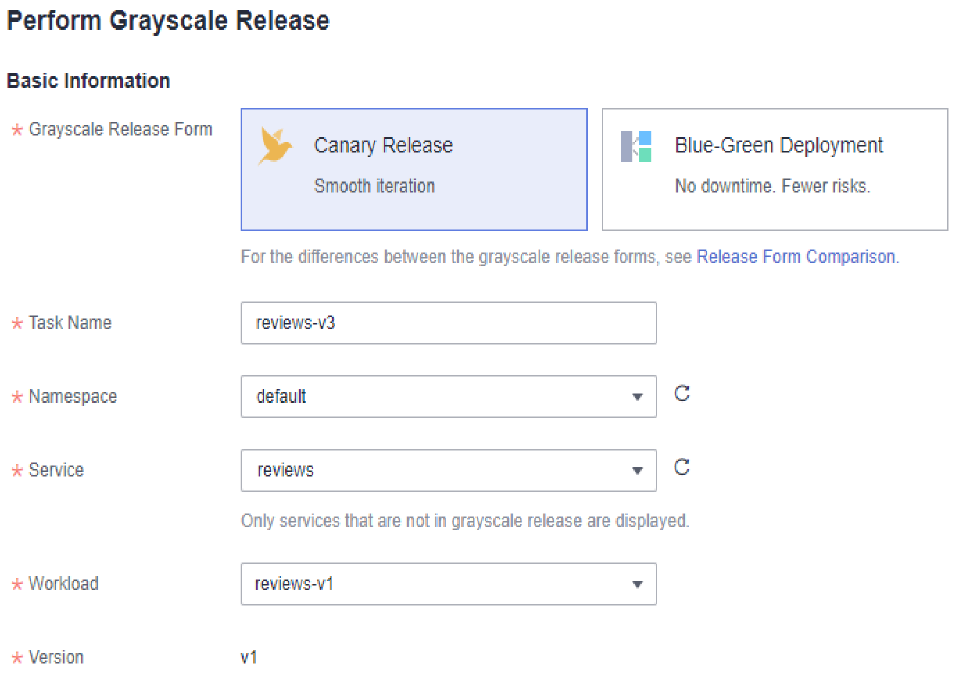
- **Workload**

  Select the workload to which the service belongs.

- **Version**

  Current service version number, which cannot be changed.

Figure 3-9 Basic information



Perform Grayscale Release

Basic Information

* Grayscale Release Form	Canary Release (Smooth iteration) / Blue-Green Deployment (No downtime. Fewer risks.)
	For the differences between the grayscale release forms, see Release Form Comparison.
* Task Name	reviews-v3
* Namespace	default
* Service	reviews
	Only services that are not in grayscale release are displayed.
* Workload	reviews-v1
* Version	v1

Step 3 Configure grayscale version information.

- **Cluster**

  Select the cluster on which the grayscale version of the service will be deployed.

- **Version**

  Enter the grayscale version number of the service.

- **Pods**

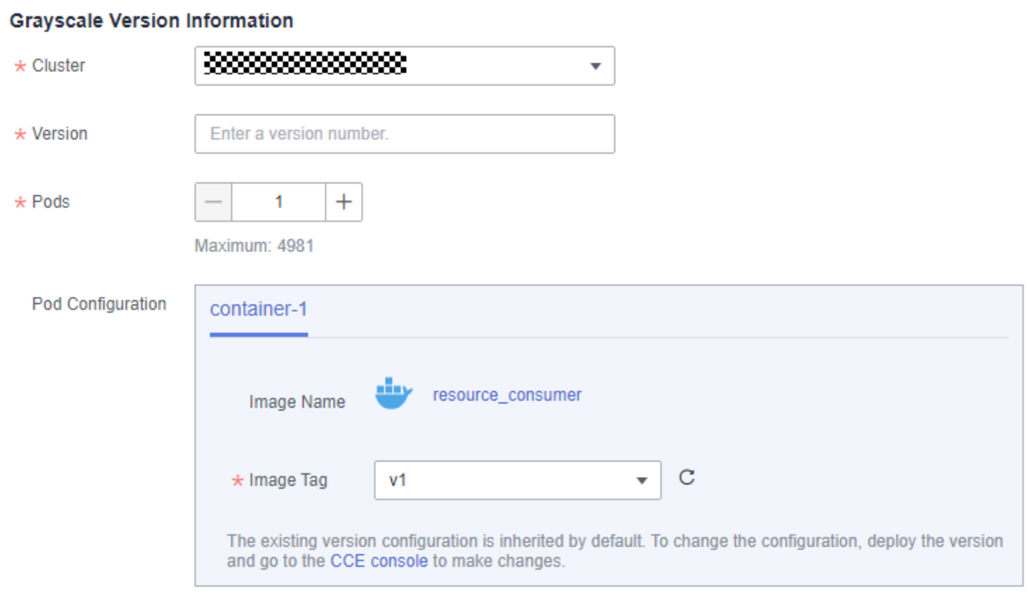
Number of pods of the grayscale version. You can modify the number as required. Each pod of the grayscale version consists of containers deployed with the same image.

- **Image Name**

  The image of the service is selected by default.
- **Image Tag**

  Select the image tag of the grayscale version.

Figure 3-10 Grayscale version information



Grayscale Version Information

* Cluster	[redacted]
* Version	Enter a version number.
* Pods	1 (Maximum: 4981)
Pod Configuration	container-1
Image Name	resource_consumer
* Image Tag	v1

The existing version configuration is inherited by default. To change the configuration, deploy the version and go to the CCE console to make changes.

Step 4 Click **Release**. Wait for the grayscale version to be created.

Ensure that all pods of the grayscale version are running normally and configure the traffic policy when the startup progress reaches 100%. You can view the pod monitoring information, including **Start Logs** and **Performance Monitoring** on the **View Status** page.

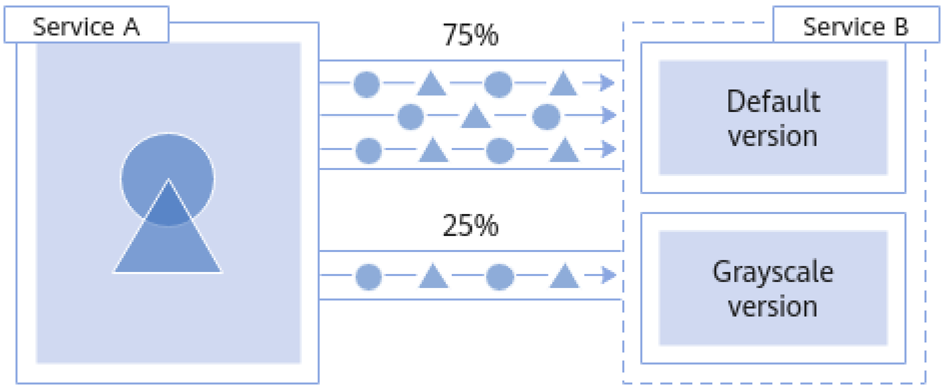
Step 5 (For canary release only) Click **Configure Traffic Policy** to configure a traffic policy.

Policy Type: The value can be **Based on traffic ratio** or **Based on request content**.

- **Based on traffic ratio**

  A specified ratio of traffic will be directed to the grayscale version. For example, 75% of the traffic is directed to the original version, and 25% is directed to the grayscale version. In actual applications, you can gradually increase the traffic ratio of the grayscale version and deliver policies to monitor the performance of the grayscale version.

Figure 3-11 Based on traffic ratio

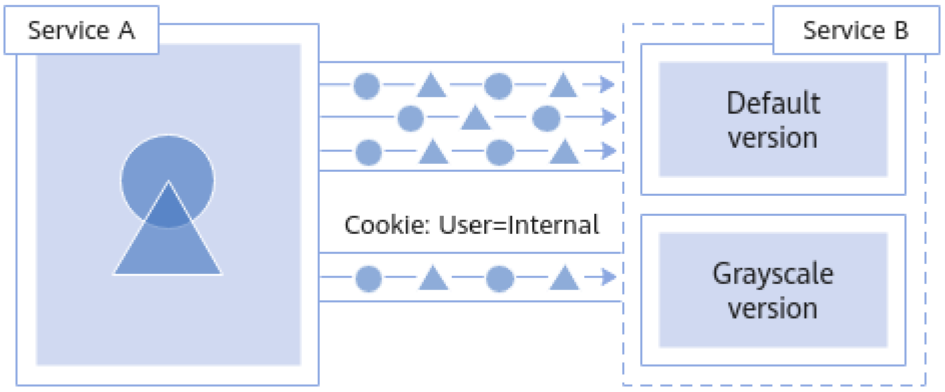


Traffic ratio: You can set the traffic ratio for the original version and grayscale version. The system distributes traffic to the two versions based on the specific traffic ratio.

- **Based on request content**

  The grayscale version can be accessed only when the traffic meets the rules based on the cookies, custom headers, queries, operating systems, and browsers. For example, only HTTP requests whose cookies meet **User=Internal** can be forwarded to the grayscale version. Other requests are still received by the original version.

Figure 3-12 Based on request content



– **Cookie**

  Regular expression: When the cookie of a request matches the configured regular expression, the request will be distributed to the grayscale version.

– **Header**

  ▪ **Full match**: Only the URL that fully matches the values you set can be accessed. For example, if **Key** is set to **User** and **Value** is set to **Internal**, only requests whose headers contain **User** with the value **Internal** are responded by the service of the grayscale version.

  ▪ **Regular expression**: When the header of a request matches the configured regular expression, the request will be distributed to the grayscale version.

You can customize the key and value for filtering. The value supports the full match and regular expression.

- **Query**
  - **Full match**: Only the URL that fully matches the values you set can be accessed. For example, if **Key** is set to **User** and **Value** is set to **Internal**, only requests whose queries contain **User** with the value **Internal** are responded by the service of the grayscale version.
  - **Regular expression**: When the query of a request matches the configured regular expression, the request will be distributed to the grayscale version.

    You can customize the key and value for filtering. The value supports the full match and regular expression.
- **Allowed OS**: Select OSs that can access the grayscale version, including iOS, Android, Windows, and macOS.
- **Allowed Browser**: Select browsers that can access the grayscale version, including Chrome and Internet Explorer.
- **Traffic management YAML**: The rule YAML is automatically generated based on the configured parameters.

NOTE

A traffic policy based on request content is valid only for the entry service that is directly accessed. If you want the traffic policy to be applied to all services, the header information of HTTP requests needs to be transferred in the service code.

For example, if you configured a grayscale policy based on the request content for service **reviews** and did not transfer the HTTP request header information in the service code, the grayscale policy will not take effect when you send requests to service **productpage**.

The reason is that when the **productpage** service calls the **reviews** service, the header information of the HTTP request you sent to **productpage** will be lost. As a result, the **reviews** service receives a request without the header information. The grayscale policy will not take effect.

Step 6 Click **Deliver Policy**.

It takes several seconds for a grayscale policy to take effect. You can view the traffic monitoring of the service and the health monitoring of the original version and grayscale version.

----End

3.5.3 Basic Operations on a Grayscale Task

Description

Basic operations on a grayscale version are performed by modifying the configuration of the DestinationRule and VirtualService resources of Istio. After the modification is complete, wait for about 10 seconds for the new policy to take effect.

Modifying the Traffic Policy of a Grayscale Version

Modifying a grayscale policy that is based on traffic ratio

For such a grayscale policy that is based on traffic ratio, you can gradually increase the traffic ratio of the grayscale version to avoid service risks caused by direct traffic switchover. To change the traffic ratio, perform the following steps:

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane, choose **Grayscale Release**. Then click the target canary release task.

Step 3 On the **Configure Traffic Policy** page, set the traffic ratio of the grayscale version.

If the traffic ratio of the grayscale version is set to **x**, the traffic ratio of the original version is automatically adjusted to **100-x**.

Step 4 Click **Deliver Policy**.

----End

Modifying a grayscale policy that is based on request content

With such a policy, a grayscale version can be accessed only when the traffic meets the rules based on Cookies, Headers, Queries, Allowed Operating Systems, and Allowed Browsers. In real-world use cases, rules may be modified for multiple times to fully verify the performance of the grayscale version.

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane on the left, choose **Grayscale Release** and click the target canary release task.

Step 3 On the **Configure Traffic Policy** page, reconfigure **Cookie**, **Header**, **Query**, **Allowed OS**, and **Allowed Browser**.

Step 4 Click **Deliver Policy**.

----End

Switching the Grayscale Policy Type

You can change the type of a grayscale policy from **based on request content** to **based on traffic ratio** and vice versa. After this operation is complete, all configured rules become invalid and all traffic is redistributed based on the new policy.

NOTICE

Grayscale policies can be changed only for running tasks. After a grayscale version is released (that is, the new version completely takes over the traffic and the old version has been brought offline), its grayscale policy cannot be reconfigured.

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane on the left, choose **Grayscale Release** and click the target canary release task.

Step 3 On the **Configure Traffic Policy** page, change the policy type.

Step 4 Click **Deliver Policy**.

----End

Taking Over All Traffic

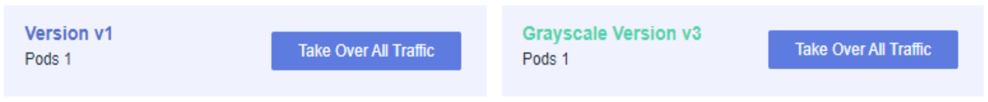
After you click **Take Over All Traffic**, the original version or grayscale version takes over all traffic.

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane on the left, choose **Grayscale Release** and click the target grayscale release task.

Step 3 On the **Monitor and Manage Traffic** page, click **Take Over All Traffic** next to the target version.

Figure 3-13 Taking over all traffic



Step 4 In the displayed dialog box, click **OK**.

----End

Terminating a Grayscale Release Task

After the grayscale version takes over all traffic, you can terminate the grayscale task. After the grayscale task is canceled, the original version will be brought offline, and all workloads and Istio configuration resources will be deleted.

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane on the left, choose **Grayscale Release** and click the target grayscale release task.

Step 3 On the **Monitor and Manage Traffic** page, click **Take Over All Traffic** next to the grayscale version.

Step 4 Click **Terminate Task** in the lower right corner.

Step 5 In the displayed dialog box, click **OK**.

You can go to the **Terminated Tasks** tab page to view the finished grayscale task. The **Release Result** is **Released successfully**.

----End

Canceling a Grayscale Release Task

After the original version takes over all traffic, you can cancel the grayscale task.

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane on the left, choose **Grayscale Release** and click the target grayscale release task.

Step 3 On the **Monitor and Manage Traffic** page, click **Take Over All Traffic** next to the original version.

Step 4 Click **Cancel Task** in the lower right corner. You can also click in the upper right corner of a task in the grayscale task list.

Step 5 In the displayed dialog box, click **OK**.

You can go to the **Terminated Tasks** tab page to view the finished grayscale task. The **Release Result** is **Released canceled**.

----End

Viewing Terminated Grayscale Release Tasks

You can view canceled and finished grayscale tasks on the **Terminated Tasks** tab page.

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane on the left, choose **Grayscale Release** and click the **Terminated Tasks** tab page.

You can view the release task name, release result, service, and release time, and delete a terminated task.

----End

3.6 Mesh Configuration

3.6.1 Overview

Mesh configuration provides cluster management, sidecar management, Istio resource management, and upgrade capabilities.

The mesh control plane workloads inject and manage sidecars of data plane pods, deliver policies and configurations, and collect monitoring data. Sidecars work with service containers in data plane pods, and they are in charge of routing and forwarding, traffic policy configuration, and monitoring data collection.

The functions of each tab page in **Mesh Configuration** are as follows:

- **Basic Information**: You can view the mesh name, ID, status, edition, version, creation time, and clusters with the mesh enabled.
- **Sidecar Management**: You can view information about all workloads injected with sidecars, perform sidecar injection, and configure sidecar resource limits. For details, see **Sidecar Management**.

- **Istio Resource Management**: You can view all Istio resources (such as VirtualService and DestinationRule), create Istio resources in YAML or JSON format, and modify existing Istio resources. For details, see **Istio Resource Management**.
- **Upgrade**: You can upgrade the version of a service mesh. For details, see **Upgrading a Mesh**.

3.6.2 Sidecar Management

On the **Sidecar Management** page, you can view information about all workloads injected with sidecars, perform sidecar injection, and configure sidecar resource limits.

Injecting a Sidecar

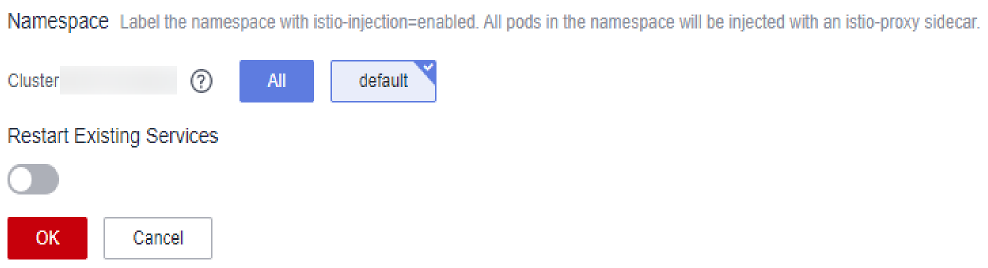
You can view the namespace and cluster to which the injected sidecar belongs. If no sidecar has been injected or you need to inject sidecar for more namespaces, perform the following operations:

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane, choose **Mesh Configuration**. Then click the **Sidecar Management** tab.

Step 3 Click **Sidecar Management**, select a namespace, determine whether to restart the existing services, and click **OK**.

Figure 3-14 Injecting a sidecar



Namespace Label the namespace with istio-injection=enabled. All pods in the namespace will be injected with an istio-proxy sidecar.

Cluster

- **Namespace**: Select one or more namespaces. The system labels the namespaces with **istio-injection=enabled**.
- **Restart Existing Services**

  : Pods of the existing services in the namespace will be restarted, which will temporarily interrupt your services. The **istio-proxy** sidecar is automatically injected into the pods of the existing services.

  : The **istio-proxy** sidecar cannot be automatically injected into the pods of the existing services. You need to manually restart the workloads on the CCE console to inject the sidecar. Whether to restart existing services affects only existing services. If the namespaces are labeled with **istio-injection=enabled**, sidecars will be automatically injected into new pods.

NOTE

- If the system displays a message indicating that modification of namespace injection is not enabled in the following clusters, you need to run the **kubectl** command to enable namespace injection. For details, see **How Do I Enable Namespace Injection for a Cluster?**.
- After sidecar injection is enabled for a namespace of a cluster, sidecars are automatically injected for pods of all workloads in the namespace. If you do not want to inject sidecars for some workloads, see **How Do I Disable Sidecar Injection for Workloads?**.
- For meshes of 1.8.4-r1 and earlier versions (including all 1.3 and 1.6 versions), you are advised to check whether the workload contains the annotation **sidecar.istio.io/inject: 'true'**. If not, add the annotation to the **spec.template.metadata.annotations** field:

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

----End

Viewing Workload Details

The list displays all workloads created in the clusters managed by a mesh. You can view the workload name, cluster to which the workload belongs, service, and sidecar information of the workload, including the sidecar name, version, status, CPU usage, and memory usage. The procedure is as follows:

Step 1 In the drop-down list and search box in the upper right corner of the workload list, select a cluster and namespace, and enter the target workload name.

Step 2 Click ˅ in front of the workload to view the sidecar information of the workload.

If the system displays a message indicating that there is no sidecar in the workload, no sidecar has been injected into the namespace to which the workload belongs. In this case, you can inject one into the namespace. For details, see **Injecting a Sidecar**.

----End

Configuring Sidecar Resource Limits

You can configure the upper and lower limits of CPU and memory resources for sidecars (istio-proxy container). If the upper and lower resource limits are not set for a workload, a resource leak of this workload will make resources unavailable for other workloads deployed on the same node. In addition, workloads that do not have upper and lower resource limits cannot be accurately monitored.

The default upper and lower limits of sidecar resources are as follows:

- CPU (core): 0.1 to 2 (included)
- MEM (MiB): 128 to 1,024 (included)

To change the value, perform the following operations:

Step 1 Click **Set Resource Limit** in the **Operation** column of the target workload. You can also select multiple workloads and click **Set Resource Limit** in the upper left corner of the workload list to configure sidecar resource limits in batches.

- Minimum CPU: CPU request, the minimum number of CPU cores required by a container. Resources are scheduled for the container based on this value.

The container can be scheduled to a node only when the total available CPU on the node is greater than or equal to the number of CPU cores applied for the container.

- Maximum CPU: CPU limit, the maximum number of CPU cores required by a container.
- Minimum memory: memory request, the minimum amount of memory required by a container. Resources are scheduled for the container based on this value. The container can be scheduled to this node only when the total available memory on the node is greater than or equal to the requested container memory.
- Maximum memory: memory limit, the maximum amount of memory required by a container. When the memory usage exceeds the specified memory limit, the pod may be restarted, which affects the normal use of the workload.

----End

3.6.3 Istio Resource Management

3.6.3.1 Configuring Istio Resources Using YAML

You can modify all Istio resources (such as VirtualService and DestinationRule) of a service in YAML or JSON format on the **Istio Resource Management** page. You can also create new Istio resources.

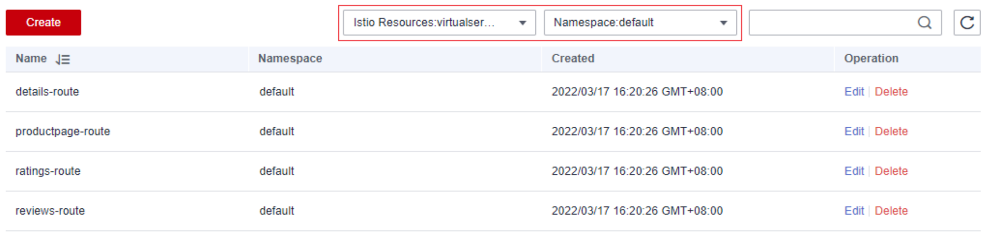
Modifying an Existing Istio Resource

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane, choose **Mesh Configuration**. Then click the **Istio Resource Management** tab.

Step 3 In the drop-down list, select the Istio resource type (for example, Istio Resources: virtualservices) and the namespace to which the resource belongs.

Figure 3-15 Filtering Istio resources



Name	Namespace	Created	Operation
details-route	default	2022/03/17 16:20:26 GMT+08:00	Edit | Delete
productpage-route	default	2022/03/17 16:20:26 GMT+08:00	Edit | Delete
ratings-route	default	2022/03/17 16:20:26 GMT+08:00	Edit | Delete
reviews-route	default	2022/03/17 16:20:26 GMT+08:00	Edit | Delete

Step 4 Click **Edit** in the **Operation** column. In the right pane, modify related configurations and click **OK**.

The configuration file can be displayed in YAML or JSON format and can be downloaded to the local PC.

----End

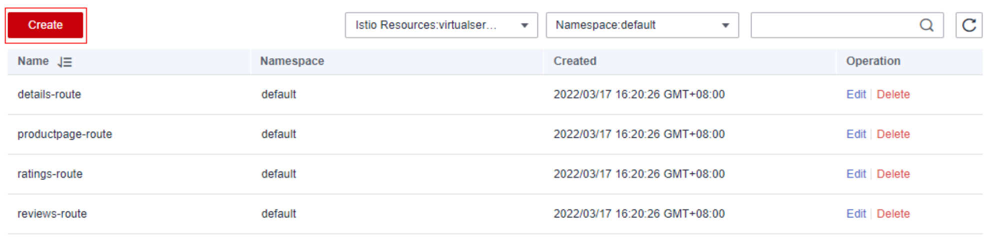
Creating an Istio Resource

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane, choose **Mesh Configuration**. Then click the **Istio Resource Management** tab.

Step 3 Click **Create** in the upper left corner of the list.

Figure 3-16 Creating an Istio resource



Name	Namespace	Created	Operation
details-route	default	2022/03/17 16:20:26 GMT+08:00	Edit | Delete
productpage-route	default	2022/03/17 16:20:26 GMT+08:00	Edit | Delete
ratings-route	default	2022/03/17 16:20:26 GMT+08:00	Edit | Delete
reviews-route	default	2022/03/17 16:20:26 GMT+08:00	Edit | Delete

Step 4 Edit the file in the right pane, or click **Import File** to upload the edited YAML or JSON file.

Step 5 Confirm the file content and click **OK**.

----End

3.6.4 Upgrade

3.6.4.1 Upgrading a Mesh

Scenario

You can upgrade a mesh of an earlier version. ASM of the Basic edition supports canary upgrades, and ASM of the Enterprise edition supports in-place patch upgrade.

Upgrade Impact

- During the upgrade, the data plane proxy of the new version is automatically injected. Service pods will be restarted in the rolling mode, which may temporarily interrupt services.
- Do not perform operations including but not limited to creating a grayscale release task or configuring a traffic rule during the upgrade.

Upgrade Path

Mesh Edition	Source Version	Target Version	Upgrade Mode
Basic edition	1.8.4-r1	1.8.4-r4	Patch update
	1.8.4-r2	1.8.4-r4	Patch update

Mesh Edition	Source Version	Target Version	Upgrade Mode
	1.8.4-r3	1.8.4-r4	Patch update
	1.8.x	1.15.5-r4	Upgrade to 1.13.*x* (canary upgrade) and then to 1.15.5-r4 (canary upgrade)
	1.13.x	1.15.5-r4	Upgrade to 1.15.5-r4 (canary upgrade)
	1.15.5-rx	1.15.5-r4	Patch update to 1.15.5-r4 (canary upgrade)

NOTE

For details about the features of each version, see **Features in v1.3**, **Features in v1.6**, **Features in v1.8**, **Features in v1.13**, and **Features in v1.15**.

Procedure

Step 1 Log in to the ASM console and check whether the meshes need to be upgraded. The specific steps are as follows:

- Check whether a message indicating the meshes to be upgraded is displayed above the list.
- Check whether **Upgrade** is displayed on the right of the mesh name.

If a service mesh can be upgraded, click its name to go to its details page.

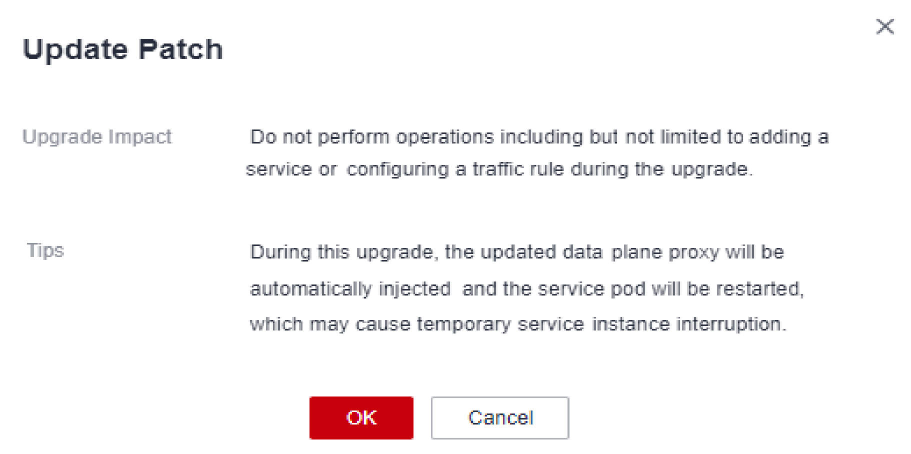
Step 2 In the navigation pane, choose **Mesh Configuration**. Then click the **Upgrade** tab.

Step 3 Select a proper upgrade mode to upgrade the service mesh according to **Upgrade Path**.

- **Upgrading the Edition**
- **Updating the Edition Patch**

  Click **Update Patch**. In the dialog box that is displayed, click **OK**.

Figure 3-17 Patch update



Upgrade Impact: Do not perform operations including but not limited to adding a service or configuring a traffic rule during the upgrade.

Tips: During this upgrade, the updated data plane proxy will be automatically injected and the service pod will be restarted, which may cause temporary service instance interruption.

----End

3.6.4.2 Features in v1.3

- Mixer-based metrics, access logs, and call chains are supported.
- SDS is supported and the pods do not need to be restarted for certificate rotation.
- Sidecar APIs are supported.
- Control-plane performance optimization is supported.
- CCE clusters v1.13, v1.15, v1.17, and v1.19 are supported.

For details, see **https://istio.io/latest/news/releases/1.3.x/announcing-1.3/change-notes/**.

3.6.4.3 Features in v1.6

- Control plane components are integrated, which simplifies control plane installation and O&M.
- Workload Entry enables operators to describe the properties of a single non-Kubernetes workload.
- SDS is enabled by default.
- Data plane–based data collection using Telemetry V2 (non-Mixer-based telemetry) is supported.
- Multi-port service governance based on the port granularity is supported.
- CCE clusters v1.15 and v1.17 are supported.

For details, see **https://istio.io/latest/news/releases/1.6.x/announcing-1.6/change-notes/**.

3.6.4.4 Features in v1.8

- The Mixer component is officially brought offline. Access logs, traces, and monitoring data are collected based on the data plane.
- **EnvoyFilter** is enhanced to support more flexible **Insert** operations.

- A new AuthorizationPolicy-based authorization policy is enabled.
- CCE clusters v1.15, v1.17, v1.19, and v1.21 are supported.

For details, see **https://istio.io/latest/news/releases/1.8.x/announcing-1.8/change-notes/**.

3.6.4.5 Features in v1.13

- Istio 1.13.9 is supported.
- CCE Turbo clusters v1.21 and v1.23 are supported.
- CCE clusters v1.21 and v1.23 are supported.
- **Istio sidecar proxy** can be configured through the ProxyConfig API.
- In Istio 1.10 and later versions, Sidecar does not redirect traffic to the loopback interface (lo). Instead, it redirects traffic to the eth0 interface of applications. If your service pod listens the lo, change the listening port to eth0 or all-zero. Otherwise, the service cannot be accessed after the upgrade. **Learn more**.

For details, see **https://istio.io/latest/news/releases/1.13.x/announcing-1.13/change-notes/**.

3.6.4.6 Features in v1.15

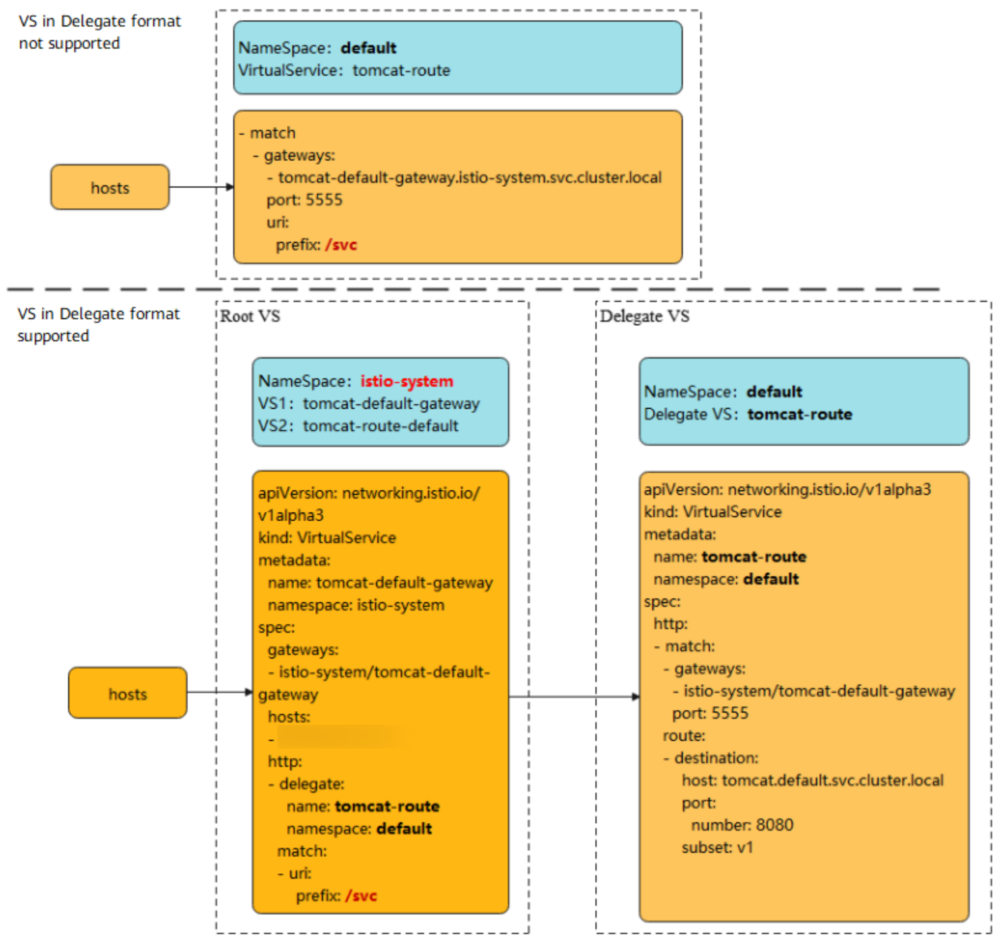
- Istio 1.15.7 is supported.
- CCE Turbo clusters v1.21, v1.23, v1.25, and v1.27 are supported.
- CCE clusters v1.21, v1.23, v1.25, and v1.27 are supported.
- Security vulnerabilities such as CVE-2023-44487, CVE-2023-39325 and CVE-2023-27487 are fixed.

For details, visit **https://istio.io/latest/news/releases/1.15.x/announcing-1.15.7/**.

3.6.4.7 Upgrading v1.3 to v1.8 to Allow VirtualService to Support Delegate Switchover

Scenario

By default, the mesh of v1.8 supports the **Delegate** function of **VirtualService**, and the ASM console supports only the **VirtualService** in **Delegate** format. The upgrade does not modify your **VirtualService**, but you cannot maintain the route on the page after the upgrade. Therefore, you need to modify your **VirtualService** according to this section.


VS in Delegate format not supported
hosts
VS in Delegate format supported
Root VS
Delegate VS
hosts

```
NameSpace: default
VirtualService: tomcat-route
```

```
- match
  - gateways:
    - tomcat-default-gateway.istio-system.svc.cluster.local
    port: 5555
    uri:
      prefix: /svc
```

```
NameSpace: istio-system
VS1: tomcat-default-gateway
VS2: tomcat-route-default
```

```
apiVersion: networking.istio.io/
v1alpha3
kind: VirtualService
metadata:
  name: tomcat-default-gateway
  namespace: istio-system
spec:
  gateways:
  - istio-system/tomcat-default-
gateway
  hosts:
  -
  http:
  - delegate:
      name: tomcat-route
      namespace: default
    match:
    - uri:
        prefix: /svc
```

```
NameSpace: default
Delegate VS: tomcat-route
```

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: tomcat-route
  namespace: default
spec:
  http:
  - match:
    - gateways:
      - istio-system/tomcat-default-gateway
      port: 5555
    route:
    - destination:
        host: tomcat.default.svc.cluster.local
        port:
          number: 8080
        subset: v1
```

NOTE

For details about **Delegate**, see the description in the Istio community.

https://istio.io/latest/docs/reference/config/networking/virtual-service/#Delegate

Constraints

- **Delegate** can be set only when **route** and **redirect** are left blank.
- ASM supports only 1-level **Delegate**. Multi-level **Delegate** does not take effect.
- **HTTPMatchRequest** of **Delegate VirtualService** must be a subset of **root VirtualService**.
- The **Delegate** feature is valid only for HTTP and gRPC protocols.

Procedure

The modification involves two scenarios. The following uses the **Tomcat** service added to a mesh as an example.

Scenario 1: If no gateway is added to the service before the upgrade, you can skip the following operations after the upgrade.

Scenario 2: If a gateway is added to the service before the upgrade, perform the following operations after the upgrade.

1. Configure the **kubectl** command for the cluster where the mesh is located. For details, see the cluster details on the CCE console.
2. Create two **VirtualService YAML** files in the **istio-system** namespace.

   File name: **tomcat**-default-gateway.yaml

   To be more specific:

   – **tomcat**: name of the service to be modified.

   – **tomcat-default-gateway**: name of the **VirtualService**. The format is **{*Service name*}-default-gateway**.

   – **tomcat-route**: name of the **VirtualService** to be modified.

   – **100.85.219.117**: ELB IP address.

   ```
   apiVersion: networking.istio.io/v1beta1
   kind: VirtualService
   metadata:
     name: tomcat-default-gateway
     namespace: istio-system
   spec:
     gateways:
     - istio-system/tomcat-default-gateway
     hosts:
     - 100.85.219.117
     http:
     - delegate:
         name: tomcat-route
         namespace: default
       match:
       - uri:
           prefix: /test
   ```

   File name: **tomcat**-route-default.yaml

   To be more specific:

   – **tomcat**: name of the service to be modified.

   – **tomcat-route-default**: name of the **VirtualService**. The format is **{*Service name*}-route-default**.

   – **tomcat-route**: name of the **VirtualService** to be modified.

   ```
   apiVersion: networking.istio.io/v1beta1
   kind: VirtualService
   metadata:
     name: tomcat-route-default
     namespace: istio-system
   spec:
     hosts:
     - tomcat.default.svc.cluster.local
     http:
     - delegate:
         name: tomcat-route
         namespace: default
       match:
       - uri:
           prefix: /
   ```

   Run the following commands to create a **VirtualService**:

   kubectl create -f tomcat-route-default.yaml

   kubectl create -f tomcat-default-gateway.yaml

3. Run the **kubectl -n{namespace} get vs** command to obtain the **VirtualService** of the service and run the **kubectl -n{namespace} edit vs tomcat-route** command to modify it as follows:

a. Delete **spec.gateways**, **spec.hosts**, and **spec.http.match.uri**.

b. Replace **tomcat-default-gateway.istio-system.svc.cluster.local** with **istio-system/tomcat-default-gateway**.

c. Change **apiVersion: networking.istio.io/v1alpha3** to **apiVersion: networking.istio.io/v1beta1**.

d. **destination.host**: The format is **{*Service name*}.{namespace}.svc.cluster.local**.

```
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
  name: tomcat-route
  namespace: default
spec:
  gateways:
  - tomcat-default-gateway.istio-system.svc.cluster.local
  - mesh
  hosts:
  - tomcat
 - 100.85.219.117 # spec.gateways and spec.hosts need to be deleted.
  http:
  - match:
    - gateways:
      - istio-system/tomcat-default-gateway
      port: 5555
      uri:
        prefix: /test # spec.http.match.uri needs to be deleted.
    route:
    - destination:
        host: tomcat.default.svc.cluster.local
        port:
          number: 8080
        subset: v1
  - match:
    - gateways:
      - mesh
      port: 8080
    route:
    - destination:
        host: tomcat.default.svc.cluster.local
        port:
          number: 8080
        subset: v1
```

4. Click **External Access URL** on the service list page to check whether it can be accessed normally.
5. On the **Service Gateway** page, check whether the service gateway route is displayed properly.

3.7 Traffic Management

3.7.1 Overview

Non-intrusive traffic management is a core function of Istio. With traffic management, you only need to focus on your own service logic rather than service access management. Traffic management enables you to:

- Dynamically modify load balancing policies for cross-service access, such as configuring consistent hashing to forward traffic to specific service pods.

- Distribute a certain proportion of traffic to a specific version of a service when the service has two online versions.
- Protect services, for example, limiting the number of concurrent connections and requests, and isolating faulty service pods.
- Dynamically modify the content of a service or simulate a service running fault.

ASM provides retry, timeout, connection pool, outlier detection, load balancing, HTTP header, and fault injection functions to meet traffic management requirements in most service scenarios.

Table 3-1 Common mesh functions and management roles

Mesh Function	Management Role	
	Service Initiator	Service Provider
Route management	Y	N
Load balancing	Y	N
Tracing analysis	Y	Y
Service authentication	Y	Y
Observability data	Y	Y
Retry	Y	N
Rewrite	Y	N
Redirection	Y	N
Authorization	N	Y
Fault injection	Y	N
Timeout	Y	N
Connection pool	Y	N
Outlier detection	Y	N
HTTP header	Y	N

Constraints

Traffic management cannot be performed for the service whose configuration diagnosis fails. For details about rectifying faults, see **Manual Fixing Items** or **Auto Fixing Items**.

3.7.2 Configuring a Traffic Policy

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane, choose **Service Management**. In the upper right corner of the list, select the namespace that your services belong to.

Step 3 Locate the target service and click **Manage Traffic** in the **Operation** column. In the window that slides out from the right, configure retry, timeout, connection pool, outlier detection, load balancing, HTTP header, and fault injection policies.

Retry

Auto retries upon service access failures improve the access quality and success rate.

On the **Retry** tab, click **Configure now**. In the displayed dialog box, set the parameters listed in the table below.

Table 3-2 Retry parameters

Parameter	Description	Value Range
Retries	Maximum number of retries allowed for a single request. The default retry interval is 25 ms. The actual number of retries depends on the configured timeout period and retry timeout period.	1-2147483647
Retry Timeout (s)	Timeout period of an initial or retry request. The default value is the same as the timeout period configured in the **Timeout** area below.	0.001–2592000
Retry Condition	Configure retry conditions. For details, see **Retry Policies** and **gRPC Retry Policies**.	-

Timeout

Auto processing and quickly failure return upon service access timeout eliminate resource locking and request freezing.

On the **Timeout** tab, click **Configure now**. In the displayed dialog box, set the parameters listed in the table below.

Table 3-3 Timeout parameters

Parameter	Description	Value Range
Timeout (s)	Timeout period for HTTP requests	0.001–2592000

Connection Pool

Connections and requests that exceed the thresholds are cut off to protect target services. Connection pool settings are applied to each pod of the upstream service at the TCP and HTTP levels. For details, see **Circuit Breaker**.

On the **Connection Pool** tab, click **Configure now**. In the displayed dialog box, set the parameters listed in the tables below.

Table 3-4 Parameters under TCP Settings

Parameter	Description	Value Range
Maximum Number of Connections	Maximum number of HTTP/TCP connections to the target service. The default value is **4294967295**.	1-2147483647
Maximum Number of Non-responses	Maximum number of keepalive probes to be sent before the connection is determined to be invalid. By default, the OS-level configuration is used. (The default value is **9** for Linux.)	1-2147483647
Health Check Interval (s)	Time interval between two keepalive probes. By default, the OS-level configuration is used. (The default value is **75** for Linux.)	0.001–2592000
Connection Timeout (s)	TCP connection timeout period. The default value is **10**.	0.001–2592000
Minimum Idle Period (s)	Duration in which a connection remains idle before a keepalive probe is sent. By default, the OS-level configuration is used. (The default value is **7200** for Linux, namely, 2 hours.)	0.001–2592000

Table 3-5 Parameters under HTTP Settings

Parameter	Description	Value Range
Maximum Number of Requests	Maximum number of requests that can be forwarded to a single service pod. The default value is **4294967295**.	1-2147483647
Maximum Number of Pending Requests	Maximum number of HTTP requests that can be forwarded to the target service for processing. The default value is **4294967295**.	1-2147483647

Parameter	Description	Value Range
Maximum Connection Idle Period (s)	Timeout period of an idle upstream service connection. If there is no active request within this time period, the connection will be closed. The default value is **3600** (1 hour).	0.001–2592000
Maximum Retries	Maximum number of retries of all service pods within a specified period. The default value is **4294967295**.	1-2147483647
Maximum Number of Requests Per Connection	Maximum number of requests for each connection to the backend. If this parameter is set to **1**, the keepalive function is disabled. The default value is **0**, indicating infinite. The maximum value is **536870912**.	1–536870912

Outlier Detection

Unhealthy pods are automatically isolated to improve the overall access success rate.

The traffic status of service pods is traced to determine whether the pods are healthy. Unhealthy pods will be ejected from the connection pool to improve the overall access success rate. Outlier detection can be configured for HTTP and TCP services. For HTTP services, pods that continuously return 5xx errors are considered unhealthy. For TCP services, pods whose connections time out or fail are considered unhealthy. For details, see **Outlier Detection**.

On the **Outlier Detection** tab, click **Configure now**. In the displayed dialog box, set the parameters listed in the table below.

Table 3-6 Outlier detection parameters

Parameter	Description	Value Range
Consecutive Errors	Number of consecutive errors in a specified time period. If the number of consecutive errors exceeds the parameter value, the pod will be ejected. The default value is **5**.	1-2147483647

Parameter	Description	Value Range
Base Ejection Time (s)	Base ejection time of a service pod that meets the outlier detection conditions. The actual ejection time of a service pod = Base ejection time x Number of ejection times. The value must be greater than or equal to 0.001s. The default value is **30**.	0.001–2592000
Inspection Interval (s)	If the number of errors reaches the threshold within this time period, the pod will be ejected. The value must be greater than or equal to 0.001s. The default value is **10**.	0.001–2592000
Maximum Percentage of Ejected Pods (%)	Maximum percentage of ejected service pods. The default value is **10**.	1–100

Load Balancing

You can customize a load balancing policy to target backend service pods.

On the **Load Balancing** tab, click **Configure now**. In the displayed dialog box, select one of the following load balancing algorithms as required:

- **Round robin**: The default load balancing algorithm. Each service pod in the pool gets a request in turn.
- **Least connection**: Requests are forwarded to the pod with fewer connections among two randomly selected healthy pods.
- **Random**: Requests are forwarded to a randomly selected healthy pod.
- **Consistent hashing**: There are four types, as described in **Table 3-7**.

Table 3-7 Consistent hashing algorithm types

Type	Description
Based on HTTP header	The hash value is calculated using the header of the HTTP request. Requests with the same hash value are forwarded to the same pod.
Based on cookie	The hash value is calculated using the cookie key name of the HTTP request. Requests with the same hash value are forwarded to the same pod.
Based on source IP	The hash value is calculated using the IP address of the HTTP request. Requests with the same hash value are forwarded to the same pod.

Type	Description
Based on query parameter	The hash value is calculated using the query parameter key name of the HTTP request. Requests with the same hash value are forwarded to the same pod.

HTTP Header

You can flexibly add, modify, and delete specified HTTP headers to manage request contents in non-intrusive mode.

On the **HTTP Header** tab, click **Configure now**. In the displayed dialog box, set the parameters listed in the tables below.

Table 3-8 Operations on the HTTP headers before the request is forwarded to the destination service

Parameter	Description
Add request headers.	To add a request header, you need to set **key** and **value**. You can also click ⊕ to add more request headers.
Edit request headers.	To edit an existing request header, you need to set **key** and **value**. You can also click ⊕ to edit more request headers.
Remove request headers.	To remove an existing request header, you need to set **key**. You can also click ⊕ to remove more request headers.

Table 3-9 Operations on the HTTP headers before the response is returned to the client

Parameter	Description
Add response headers.	To add a response header, you need to set **key** and **value**. You can also click ⊕ to add more response headers.
Edit response headers.	To edit an existing response header, you need to set **key** and **value**. You can also click ⊕ to edit more response headers.
Remove response headers.	To remove an existing response header, you need to set **key**. You can also click ⊕ to remove more response headers.

Fault Injection

You can inject faults into the system to check whether it can tolerate and recover from faults.

On the **Fault Injection** tab, click **Configure now**. In the displayed dialog box, set the parameters listed in the table below.

Table 3-10 Fault injection parameters

Parameter	Description	Value Range
Fault Type	The options are **Delay** and **Abort**. ● **Delay**: Service requests are delayed. ● **Abort**: A service is aborted and the preset status code is returned.	**Delay** and **Abort**
Delay (s)	This parameter needs to be set when **Fault Type** is set to **Delay**. A request will be delayed for this period of time before it is forwarded.	0.001–2592000
HTTP Status Code	This parameter needs to be set when **Fault Type** is set to **Abort**. HTTP status code returned when an abort fault occurs. The default value is **500**.	200–599
Fault Percentage (%)	Percentage of requests for which the delay or abort fault is injected.	1–100

----End

3.7.3 Viewing Traffic Monitoring

Scenario

In the traffic management window, you can view the traffic monitoring data of the last hour, including RPS, success rate, and request latency.

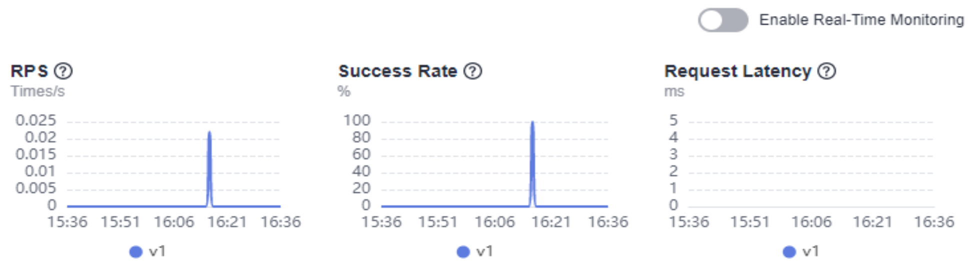
Procedure

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane, choose **Service Management**. In the upper right corner of the list, select the namespace that your services belong to.

Step 3 Locate the target service and click **Manage Traffic** in the **Operation** column. In the window that slides out from the right, view the traffic monitoring data of the last hour.

Figure 3-18 Traffic monitoring



Step 4 After real-time monitoring is enabled, data is dynamically refreshed every minute.

----End

3.7.4 Changing a Traffic Policy

Scenario

You can change the settings of a configured traffic policy. For example, you can change the load balancing algorithm from **Round robin** to **Random**.

Procedure

Step 1 Log in to the ASM console and click the name of the target service mesh to go to its details page.

Step 2 In the navigation pane, choose **Service Management**. Locate the service whose traffic policy needs to be modified and click **Manage Traffic** in the **Operation** column. In the window that slides out from the right, modify traffic policies.

----End

4 FAQs

4.1 Service Mesh Cluster

4.1.1 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, create 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.

4.1.2 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. In the navigation pane, choose **Charts** > **Sample Charts**.

Step 2 Click **Releases** and select the target cluster from the drop-down list. 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

4.2 Mesh Management

4.2.1 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, or v1.19 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

4.2.2 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 and choose **Resource Management** > **Nodes** in the navigation pane to delete the nodes.

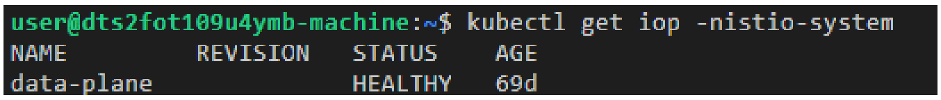
4.2.3 How Do I Enable Namespace Injection for a Cluster?

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.

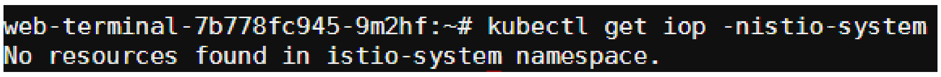
Step 2 Run the **kubectl get iop -nistio-system** command to query iop resources.

- 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                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

4.2.4 How Do I Disable Sidecar Injection for Workloads?

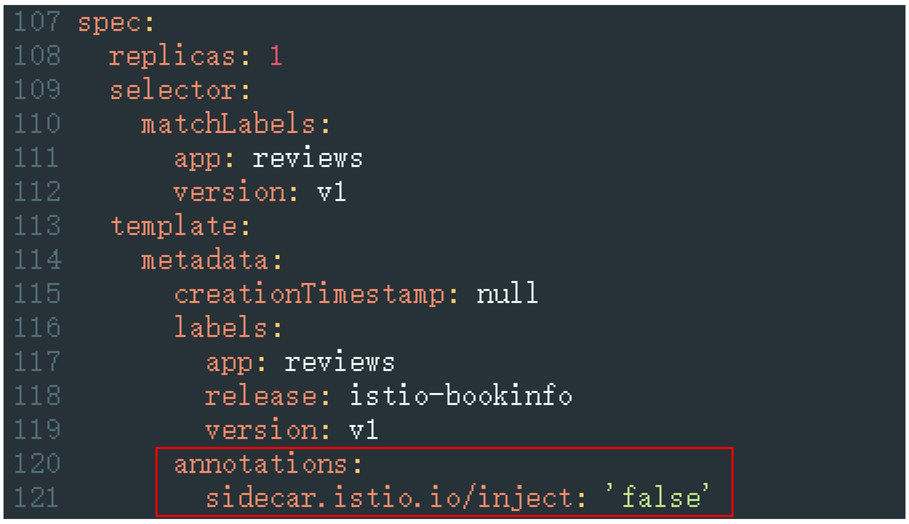
After sidecar injection is enabled for a namespace of a cluster, sidecars are automatically injected for pods of all workloads in the namespace. You can configure sidecars not to be injected into some workloads:

Step 1 Log in to the CCE console. In the navigation pane, choose **Workloads** > **Deployments**.

Step 2 Click the target cluster in the drop-down list box and 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'
```



```
spec:
  replicas: 1
  selector:
    matchLabels:
      app: reviews
      version: v1
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: reviews
        release: istio-bookinfo
        version: v1
      annotations:
        sidecar.istio.io/inject: 'false'
```

For more details about sidecar injection, see **Automatic Sidecar Injection**.

----End

4.2.5 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

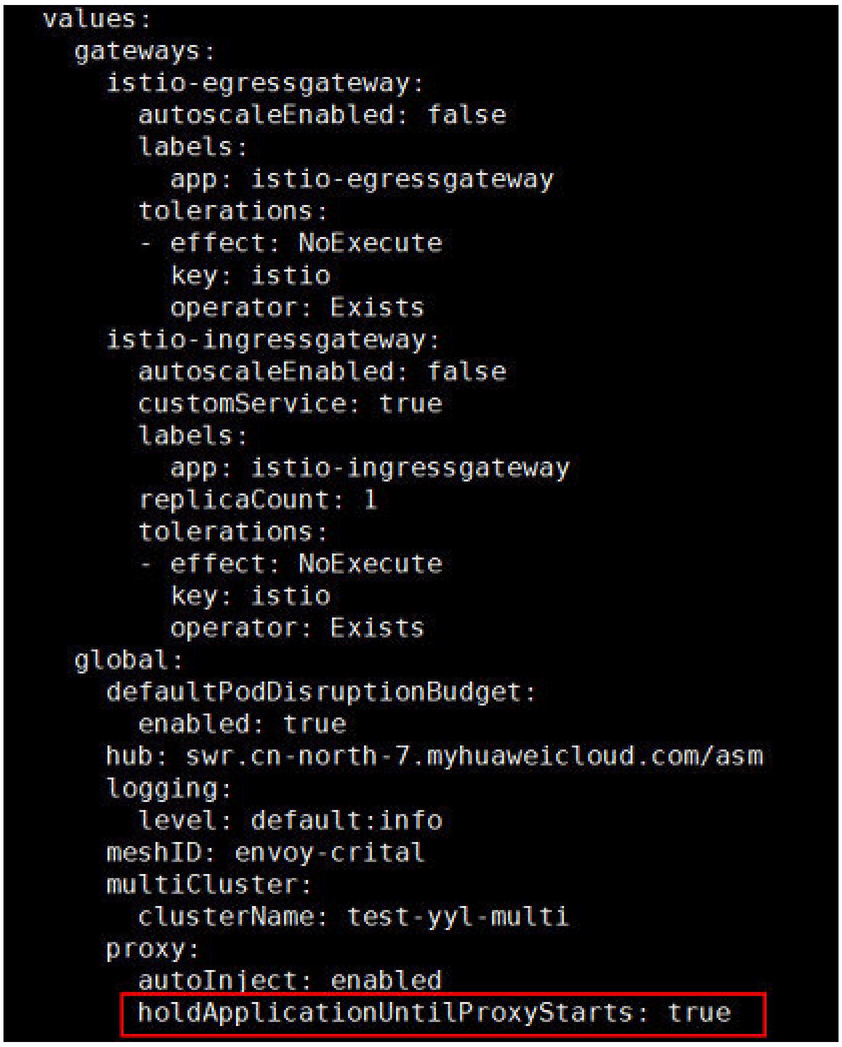
After this switch is enabled, the service container cannot be started until the sidecar is fully ready, which slows down pod startup and reduces scalability for burst traffic. You are advised to evaluate service scenarios and enable this switch only for required services.

- **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
     ```



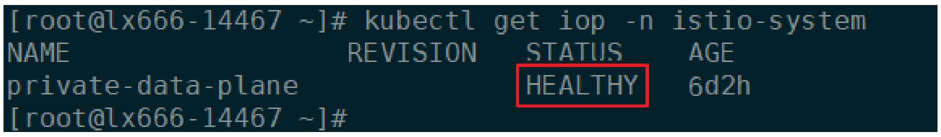
```
 values:
   gateways:
     istio-egressgateway:
       autoscaleEnabled: false
       labels:
         app: istio-egressgateway
       tolerations:
       - effect: NoExecute
         key: istio
         operator: Exists
     istio-ingressgateway:
       autoscaleEnabled: false
       customService: 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-crital
     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                HEALTHY   6d2h
[root@lx666-14467 ~]#
```

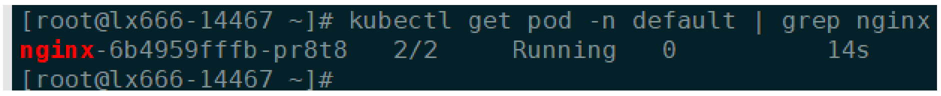
d. Run the following command to upgrade the services in the mesh in a rolling manner:

   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:

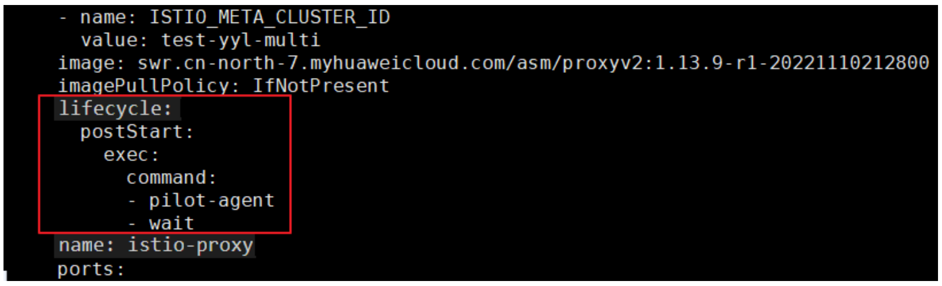
   kubectl get pod -n *default* **| grep** *nginx*



```
[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-yyl-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**

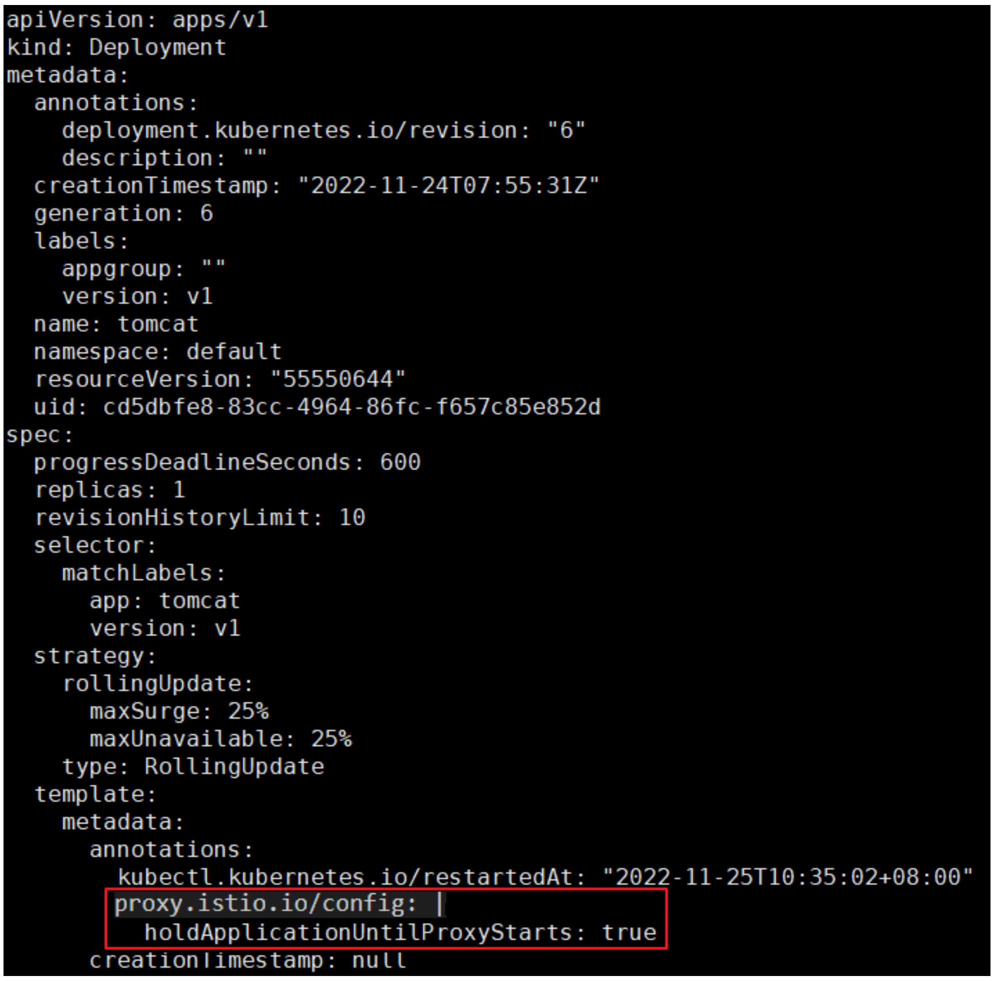
  For Istio 1.8 or later versions, you can label the pods for which this function needs to be enabled with **proxy.istio.io/config** and set **holdApplicationUntilProxyStarts** to true.

  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:
        kubectl.kubernetes.io/restartedAt: "2022-11-25T10:35:02+08:00"
        proxy.istio.io/config: |
          holdApplicationUntilProxyStarts: true
      creationTimestamp: null
```

4.3 Adding a Service

4.3.1 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 **kubectl get gateway -n istio-system** command to check whether the IP address, domain name, and port number are configured for the Gateway. Run the **kubectl 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.

4.3.2 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.

4.3.3 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.

4.3.4 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.

4.4 Performing Grayscale Release

4.4.1 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.

4.4.2 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.