

应用服务网格  
23.3.1

# 用户指南

文档版本 01  
发布日期 2023-03-15



版权所有 © 华为技术有限公司 2024。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

## 商标声明



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

## 注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

# 目录

<b>1 产品介绍</b>	<b>1</b>
1.1 什么是应用服务网格	1
1.2 产品优势	3
1.3 应用场景	4
1.3.1 服务灰度发布	4
1.3.2 服务流量管理	4
1.3.3 端到端的透明安全	5
1.3.4 服务运行监控	6
1.3.5 传统微服务 SDK 结合	6
1.4 约束与限制	7
1.5 基本概念	8
1.6 规格推荐	9
1.7 与其他云服务的关系	10
<b>2 快速入门</b>	<b>11</b>
2.1 Bookinfo 应用的灰度发布实践	11
2.2 为集群开通 Istio	20
2.2.1 入门概述	20
2.2.2 准备工作	21
2.2.3 创建网格	23
2.3 配置式应用灰度发布	24
2.3.1 入门概述	24
2.3.2 准备工作	25
2.3.3 灰度发布	28
<b>3 用户指南</b>	<b>31</b>
3.1 创建网格	31
3.1.1 创建网格	31
3.2 网格管理	32
3.2.1 卸载网格	32
3.3 服务管理	33
3.3.1 配置诊断	33
3.3.2 手动修复项	35
3.3.2.1 所有 Pod 是否都配置了 app 和 version 标签	35

3.3.2.2 所有 Pod 的 app 和 version 标签是否都相等.....	36
3.3.2.3 所有 Pod 是否都注入了 sidecar.....	37
3.3.3 自动修复项.....	38
3.3.3.1 Service 的端口名称是否符合 istio 规范.....	39
3.3.3.2 Service 的选择器中是否配置了 version 标签.....	39
3.3.3.3 服务是否配置了默认版本的服务路由，路由配置是否正确.....	40
3.4 网关管理.....	41
3.4.1 添加网关.....	41
3.4.2 添加路由.....	43
3.5 灰度发布.....	44
3.5.1 灰度发布概述.....	44
3.5.2 创建灰度任务.....	45
3.5.3 灰度任务基本操作.....	49
3.6 网格配置.....	51
3.6.1 网格配置概述.....	51
3.6.2 sidecar 管理.....	51
3.6.3 istio 资源管理.....	53
3.6.3.1 YAML 方式配置 Istio 资源.....	53
3.6.4 升级.....	54
3.6.4.1 升级网格.....	54
3.6.4.2 1.3 版本特性.....	56
3.6.4.3 1.6 版本特性.....	56
3.6.4.4 1.8 版本特性.....	56
3.6.4.5 1.13 版本特性.....	57
3.6.4.6 1.15 版本特性.....	57
3.6.4.7 1.3 升级 1.8 VirtualService 支持 Delegate 切换.....	57
3.7 流量治理.....	60
3.7.1 流量治理概述.....	60
3.7.2 配置流量策略.....	61
3.7.3 更改流量策略.....	66
<b>4 最佳实践.....</b>	<b>67</b>
4.1 数据面 sidecar 升级不中断业务.....	67
4.2 面向 Dubbo 协议的服务治理.....	69
4.2.1 简介.....	69
4.2.2 服务发现模型.....	69
4.2.3 SDK 适配方式.....	70
4.2.3.1 PASSTHROUGH 方案.....	70
4.2.3.2 静态目标服务.....	71
<b>5 常见问题.....</b>	<b>72</b>
5.1 网格集群.....	72
5.1.1 启用服务网格后，状态一直为安装中.....	72
5.1.2 卸载服务网格后，状态一直为未就绪.....	72

5.2 网格管理.....	73
5.2.1 为什么我的集群不能启用网格? .....	73
5.2.2 Istio 卸载之后, 为什么独享节点还在? .....	74
5.2.3 如何为集群开放命名空间注入? .....	74
5.2.4 某些工作负载不注入 Sidecar, 该如何配置? .....	75
5.2.5 Sidecar 未就绪导致 Pod 启动失败.....	75
5.3 添加服务.....	78
5.3.1 添加的对外访问方式不能生效, 如何排查? .....	78
5.3.2 一键创建体验应用为什么启动很慢? .....	78
5.3.3 一键创建体验应用部署成功以后, 为何不能访问页面? .....	79
5.3.4 添加路由时, 为什么选不到对应的服务? .....	79
5.4 灰度发布.....	79
5.4.1 灰度发布部署版本为什么不能更换镜像? .....	79
5.4.2 基于请求内容发布策略对一些服务为什么没有生效? .....	79

# 1 产品介绍

## 1.1 什么是应用服务网格

### 什么是应用服务网格

应用服务网格（Application Service Mesh，简称ASM）提供非侵入式的微服务治理解决方案，支持完整的生命周期管理和流量治理，兼容Kubernetes和Istio生态，功能包括负载均衡、熔断、故障注入等多种治理能力。并内置金丝雀、蓝绿灰度发布流程，提供一站式自动化的发布管理。

### 什么是 Istio

Istio是一个提供连接、保护、控制以及观测功能的开放平台，通过提供完整的非侵入式的微服务治理解决方案，能够很好的解决云原生服务的管理、网络连接以及安全管理等服务网络治理问题。

随着微服务的大量应用，其构成的分布式应用架构在运维、调试和安全管理等维度变得更加复杂，开发者需要面临更大的挑战，如：服务发现、负载均衡、故障恢复、指标收集和监控，以及金丝雀发布、蓝绿发布、限流、访问控制、端到端认证等。

在较高的层次上，Istio有助于降低这些部署的复杂性，并减轻开发团队的压力。它是一个完全开源的服务网格，可以透明地分层到现有的分布式应用程序上。它也是一个平台，包括允许集成到任何日志记录平台、遥测或策略系统的API。Istio的多样化功能使您能够成功高效地运行分布式微服务架构，并提供保护、连接和监控微服务的统一方法。

#### 服务网格

服务网格（Service Mesh）通常用于描述构成应用程序的微服务网络以及应用之间的交互。它的需求包括服务发现、负载均衡、故障恢复、指标收集和监控以及通常更加复杂的运维需求，例如蓝绿发布、金丝雀发布、限流、访问控制和端到端认证等。

### 为什么要使用 Istio

Istio提供了一个完整的解决方案，通过为整个服务网格提供行为洞察和操作控制来满足微服务应用程序的多样化需求。

Kubernetes提供了部署、升级和有限的运行流量管理能力，但并不具备熔断、限流等能力。Istio是基于Kubernetes构建的开放平台，它很好的补齐了Kubernetes在微服务治理上的诸多能力。

想要让服务支持Istio，只需要在您的环境中部署一个特殊的Sidecar代理，使用Istio控制平面功能配置和管理代理，拦截微服务之间的所有网络通信：

- 实现HTTP、gRPC、WebSocket和TCP流量的自动负载均衡。
- 通过丰富的路由规则、重试和故障注入，可以对流量行为进行细粒度控制。
- 对出入集群入口和出口中所有流量自动度量指标、日志记录和追踪。
- 通过强大的基于身份的验证和授权，在集群中实现安全的服务间通信。

Istio旨在实现可扩展性，满足各种部署需求。

## 产品功能

### 灰度发布

- 基于请求内容灰度规则：支持基于请求内容灰度规则，可以配置Header、Cookie等多种请求信息。
- 基于流量比例灰度规则：支持基于流量比例灰度规则，根据权重比例分配流量。
- 金丝雀灰度流程：提供向导方式引导用户完成金丝雀灰度流程，包括灰度版本上线、观察灰度版本运行、配置灰度规则、观测访问情况、切分流量等。
- 蓝绿灰度流程：提供向导方式引导用户完成蓝绿灰度流程，包括灰度版本上线、观察灰度版本运行、观测访问情况、版本切换等。

### 流量治理

- 七层连接池管理：支持配置HTTP最大请求数、最大重试次数、最大等待请求数、每连接最大请求数以及连接最大空闲时间。
- 四层连接池管理：支持配置TCP最大连接数、连接超时时间、最大无响应次数、最短空闲时间以及健康检查间隔。
- 熔断：支持配置服务熔断规则，包括实例被驱逐前的连续错误数、检查周期、基础隔离时间以及最大隔离实例比例。
- 重试：支持配置HTTP重试次数、重试超时时间以及重试条件。
- 超时：支持配置HTTP请求超时时间。
- 负载均衡：支持配置随机调度、轮询调度、最少连接和一致性哈希多种负载均衡算法。
- HTTP头域：可以灵活添加、修改和删除指定HTTP头域，包括将HTTP请求转发到目标服务之前对Headers的操作，以及将HTTP响应回复给客户端前，对Headers的操作。
- 故障注入：支持配置延时故障和中断故障。

### 安全

- 对端认证：对端认证定义了流量如何通过隧道（或者不通过隧道）传输到当前服务的实例，当前支持配置默认模式（UNSET）、宽容模式（PERMISSIVE）和严格模式（STRICT）三种认证策略。
- 访问授权：访问授权用来实现对网格中服务的访问控制功能，即判断一个请求是否允许发送到当前的服务。

### 可观察性

- 应用访问拓扑：支持网格应用访问拓扑，体现服务间依赖。
- 服务运行监控：支持服务访问信息，包括服务和各个版本的QPS和延时等指标。
- 访问日志：支持收集和检索服务的访问日志。

### 网格数据面服务框架

- Spring Cloud：支持Spring Cloud SDK开发的服务在网格上统一管理。
- Dubbo：支持Dubbo SDK开发的服务在网格上统一管理。

### 兼容性和扩展

- 社区版本兼容：API完全兼容Istio。
- 社区插件支持：支持Tracing、Prometheus、Kiali、Grafana。

## 1.2 产品优势

### 简单易用

无需修改任何服务代码，也无需手动安装代理，只需开启应用服务网格功能，即可实现丰富的无侵入服务治理能力。

### 内置金丝雀、蓝绿灰度发布流程

- 灰度版本一键部署，流量切换一键生效。
- 灰度策略可配置，支持流量比例、请求内容（Cookie、OS、浏览器等）。
- 一站式健康、性能、流量监控，实现灰度发布过程量化、智能化、可视化。

### 策略化的智能路由与弹性流量管理

支持对服务配置负载均衡、服务路由、故障注入、熔断等治理规则，并结合一站式治理系统，提供实时的、可视化的微服务流量管理；无侵入智能流量治理，应用无需任何改造，即可进行动态的智能路由和弹性流量管理。

- 权重、内容等路由规则，实现应用灵活灰度发布。
- 负载均衡，满足业务处理高可用性诉求。
- 熔断，实现服务间链路稳定、可靠。
- 网络长连接管理降低资源损耗，提升网络吞吐量。
- 服务安全认证、鉴权、审计等，提供服务安全保障基石。

### 性能增强，可靠性增强

控制面和数据面在社区版本基础上进行可靠性加固和性能优化。

### 多基础设施

提供免运维的托管控制面，提供全局统一的服务治理、灰度、安全和服务运行监控能力，并支持容器和VM等多种基础设施的统一服务发现和管理。



## 协议扩展

社区通用的HTTP、gRPC、TCP、TLS外，扩展对Dubbo协议的支持。

## 传统 SDK 集成

提供Spring Cloud、Dubbo等传统微服务SDK的集成解决方案，传统的微服务SDK开发的业务代码无需大的代码修改即可迁移到云原生的容器和网格运行环境上来。

# 1.3 应用场景

## 1.3.1 服务灰度发布

### 适用场景

通常产品优化迭代的方式，是直接将某版本上线发布给全部用户，一旦遇到线上事故（或BUG），对用户的影响极大，解决问题周期较长，甚至有时不得不回滚到前一版本，严重影响了用户体验。

灰度发布是版本升级平滑过渡的一种方式，当版本升级时，使部分用户使用高版本，其他用户继续使用低版本，待高版本稳定后，逐步扩大范围把所有用户流量都迁移到高版本上面来。

### 价值

应用服务网格为应用治理提供多种灰度发布功能，在初始灰度的时候可以发现、调整问题，以降低影响度，保证整体系统稳定，高效地推动企业应用的迭代升级。

### 优势

- **内置灰度流程：**基于细粒度的分流规则，在ASM中内置了多种典型的灰度发布流程，提供一个灰度发布的向导，方便用户便捷的进行灰度发布实践。在一个服务版本正常工作，正常处理流量的同时，用户可以创建一个新的灰度版本。当灰度版本启动成功后，引导用户配置灰度规则来切分流量。
- **灵活的灰度策略：**灰度规则可以是基于权重的按比例切分流量，也可以根据服务访问的内容来将特定内容的请求发往灰度版本，对于常用的HTTP协议，如请求中的OS、浏览器、Cookie和Header信息等，在配置了灰度规则后，可以实时的观察到多个线上版本的运行和访问信息，从而在向导中一键式完成版本选择，将所有流量都切换到最终选定的版本上。

## 1.3.2 服务流量管理

### 适用场景

流量治理是一个非常宽泛的话题，例如：

- 动态修改服务间访问的负载均衡策略，如配置一致性哈希将流量转发到特定的服务实例上。
- 同一个服务有两个版本在线，将一部分流量切到某个版本上。
- 对服务进行保护，例如限制并发连接数、限制请求数、隔离故障服务实例等。

- 动态修改服务中的内容，或者模拟一个服务运行故障等。

## 价值

在Istio中实现这些服务治理功能时无须修改任何应用的代码。

应用服务网格ASM基于Istio可以为管理的服务提供非侵入的流量治理能力。根据服务的协议，提供策略化、场景化的网络连接管理。在应用拓扑上对选定服务的选定端口，根据需要配置各种不同的治理规则。

## 优势

- **重试**：服务访问失败自动重试，从而提高总体访问成功率和质量。支持配置HTTP请求重试次数、重试超时时间和重试条件。
- **超时**：服务访问超时自动处理，快速失败，从而避免资源锁定和请求卡顿。支持配置HTTP请求超时时间。
- **连接池**：通过连接池管理，可以对四层协议配置TCP的最大连接数、连接超时时间、最大无响应次数、最短空闲时间和健康检查间隔，对七层协议配置HTTP最大请求数、最大重试次数、最大等待请求数、每连接最大请求数、连接最大空闲时间，从而防止一个服务的失败级联影响到整个应用。
- **熔断**：通过熔断配置实例被驱逐前的连续错误次数、驱逐间隔时长、最小驱逐时间、最大驱逐比例等参数，从而定期考察被访问的服务实例的工作情况，如果连续出现访问异常，则将服务实例标记为异常并进行隔离，在一段时间内不为其分配流量。过一段时间后，被隔离的服务实例会再次被解除隔离，尝试处理请求，如果还不正常，则被隔离更长的时间。从而实现异常服务实例的故障隔离和自动故障恢复。
- **负载均衡**：配置各种负载均衡策略，如随机、轮询、最少连接，还可以配置一致性哈希将流量转发到特定的服务实例上。
- **HTTP头域**：灵活增加、修改和删除指定HTTP头域，包括将HTTP请求转发到目标服务之前对Headers的操作，以及将HTTP响应回复给客户端前，对Headers的操作，以非侵入方式管理请求内容。
- **故障注入**：通过对选定的服务注入中断故障、延时故障来构造故障场景，无需修改代码即可进行故障测试。

### 1.3.3 端到端的透明安全

#### 适用场景

众所周知，将传统的单体应用拆分为一个个微服务固然带来了各种好处，包括更好的灵活性、可伸缩性、重用性，但微服务也同样面临着特殊的安全需求，如下所示：

- 为了抵御中间人攻击，需要用到流量加密。
- 为了提供灵活的服务访问控制，需要用到TLS和细粒度访问策略。
- 为了决定哪些人在哪些时间可以做哪些事，需要用到审计工具。

面对这些需求应用服务网格提供全面的安全解决方案，包括身份验证策略，透明的TLS加密以及授权和审计工具。

#### 价值

- **默认的安全性**：无需修改即可保证应用程序代码和架构的安全性。

- **纵深防御**：与现有的安全系统结合并提供多层防御。
- **零信任网络**：在不受信任的网络上构建安全解决方案。

## 优势

- **非侵入安全**：应用服务网格是以一种安全基础设施的方式向用户提供透明的安全能力，让不涉及安全问题的代码安全运行，让不太懂安全的人可以开发和运维安全的服务，不用修改业务代码就能提供服务访问安全。应用服务网格提供了一个透明的分布式安全层，并提供了底层安全的通信通道，管理服务通信的认证、授权和加密，提供Pod到Pod、服务到服务的通信安全。开发人员在这个安全基础设施层上只需专注于应用程序级别的安全性。
- **细粒度授权**：在认证的基础上，就可以进行服务间的访问授权管理，可以控制某个服务，或者服务的一个特定接口进行授权管理。如只开放给特定的一个Namespace下的服务，或者开放给某个特定的服务。源服务和目标服务可以在不同的集群，甚至源服务的不同实例在不同的集群，目标服务的不同实例在不同的集群。

### 1.3.4 服务运行监控

#### 适用场景

运营容器化的基础设施带来了一系列新的挑战。因此需要增强容器、评估API端点的性能以及识别出基础设施中的有害部分。Istio服务网格可在不修改代码的情况下实现API增强，并且不会带来服务延迟。

#### 价值

应用服务网格为网格内的所有服务通信生成详细的遥测，这种遥测技术提供了服务行为的可观察性，允许运营商对其应用程序进行故障排除、维护和优化，而不会给服务开发人员带来任何额外负担。通过应用服务网格，运营商可以全面了解被监控的服务如何与其他服务以及组件本身进行交互。

#### 优势

- **非侵入监控数据采集**：在复杂应用的场景下，服务间的访问拓扑，监控等都是对服务整体运行状况进行管理，服务访问异常时进行定位定界的必要手段。服务网格技术的一项重要能力就是以应用非侵入的方式提供这些监控数据的采集，用户只需关注自己的业务开发，无需额外关注监控数据的生成。
- **灵活的服务运行管理**：在拓扑图上通过服务的访问数据，可以直观地观察到服务的健康状况，服务间的依赖情况。并且可以对关心的服务进行下钻，从服务级别下钻到服务版本级别，还可以进一步下钻到服务实例级别。通过实例级别的拓扑可以观察到配置了熔断规则后，网格如何隔离故障实例，使其逐渐接收不到流量。并且可以在故障实例正常时，如何进行实例的故障恢复，自动给恢复的实例重新分配流量。

### 1.3.5 传统微服务 SDK 结合

#### 适用场景

- 传统SDK开发的服务希望使用服务网格能力。
- 希望将Istio与微服务平台集成，并以Istio为基础打造一个微服务管控中心。

## 价值

提供Spring Cloud、Dubbo等传统微服务SDK的集成解决方案，传统的微服务SDK开发的业务代码无需大的修改，即可方便的迁移到云原生的容器和网格运行环境上来。

## 优势

- **无侵入的迁移方案：**对于大量当前使用传统SDK开发的服务，当要使用服务网格能力时，ASM提供了一套迁移业务无侵入的迁移方案。在服务调用方将Outbound的流量引流到网格的数据面上来。即短路原有SDK里的服务发现和负载均衡，直接通过Kubernetes的服务名访问，使用Kubernetes的服务发现，SDK里的治理逻辑也可逐步的被网格替换。这样服务的运行治理能力都下沉到由Kubernetes和服务网格提供的基础设施上。
- **统一策略管理：**控制面使用ASM统一的控制面做服务发现和治理规则管理，不需要独立的注册中心和配置中心；数据面的服务发现、负载均衡和各种治理都在ASM数据面Envoy上执行，SDK作为开发框架，回归到开发框架的本来职能，作为一个纯净轻量的应用开发框架供用户开发代码。
- **多种基础设施：**在方案中，数据面可以是容器，也可以是VM。服务可以是各种语言，本身开发框架也没有限制。统一通过网格的控制面下发流量规则，对所有形态的数据面进行一致的管理。

## 1.4 约束与限制

### 集群限制

启用应用服务网格前，您需要创建或已有一个可用集群，并确保集群版本为v1.15、v1.17或v1.19。

ASM 1.18之前的版本不支持CCE Turbo集群中如下操作系统节点上的容器添加至网格。

- Ubuntu 22.04

### 网格功能约束

使用网格进行服务治理时，服务和 workload（Deployment）必须是一一对应关系，不允许多个服务对应一个 workload，因为可能出现灰度发布、网关访问等功能异常。

### 旧版 ASM 与新版 ASM 区别

对于同一个网格，建议不要在旧版ASM页面和新版ASM页面交替使用，因为会有一些数据兼容性问题。

旧版ASM与新版ASM的区别如下：

- Sidecar注入方式不同。旧版ASM创建的网格没有开启Sidecar的命名空间注入，新版ASM创建的网格开启了Sidecar的命名空间注入，命名空间注入详见：<https://istio.io/latest/docs/setup/additional-setup/sidecar-injection/>。
- Istio资源格式不同。旧版ASM创建的网格和新版ASM创建的网格管理的Istio资源（VirtualService和DestinationRule）格式不同。
- 灰度发布功能不兼容。例如：在新版ASM加入网格的服务不支持在旧版ASM进行灰度发布；在新版ASM创建的灰度发布任务无法在旧版ASM显示。

- 流量治理功能不兼容。例如：新版ASM配置的流量治理无法在旧版ASM页面显示或配置。

## 1.5 基本概念

### 工作负载

工作负载即Kubernetes对一组Pod的抽象模型，用于描述业务的运行载体，包括Deployment、Statefulset、Job、Deamonset等。

- 无状态工作负载（即Kubernetes中的“Deployments”）：Pod之间完全独立、功能相同，具有弹性伸缩、滚动升级等特性。如：Nginx、WordPress。
- 有状态工作负载（即Kubernetes中的“StatefulSets”）：Pod之间不完全独立，具有稳定的持久化存储和网络标示，以及有序的部署、收缩和删除等特性。如：mysql-HA、etcd。

### 实例（Pod）

Pod是Kubernetes部署应用或服务的最小的基本单位。一个Pod 封装多个应用容器（也可以只有一个容器）、存储资源、一个独立的网络 IP 以及管理控制容器运行方式的策略选项。

### 金丝雀发布

又称灰度发布，是迭代的软件产品在生产环境安全上线的一种重要手段。在生产环境上引一部分实际流量对一个新版本进行测试，测试新版本的性能和表现，在保证系统整体稳定运行的前提下，尽早发现新版本在实际环境上的问题。

### 蓝绿发布

蓝绿发布提供了一种零宕机的部署方式。不停老版本，部署新版本进行测试，确认运行正常后，将流量切到新版本，然后老版本同时也升级到新版本。升级过程中始终有两个版本同时在线，有问题可以快速切换。

### 流量治理

应用流量治理提供可视化云原生应用的网络状态监控，并实现在线的网络连接和安全策略的管理和配置，当前支持连接池、熔断、负载均衡、HTTP头域、故障注入等能力。

### 连接池管理

配置TCP和HTTP的连接和请求池相关阈值，保护目标服务，避免对服务的过载访问。

### 熔断

配置快速响应和隔离服务访问故障，防止网络和服务调用故障级联发生，限制故障影响范围，防止故障蔓延导致系统整体性能下降或者雪崩。

## 1.6 规格推荐

### 独享节点规格推荐

应用服务网格性能与集群控制面（Master）节点资源息息相关，请您根据您的业务需求，选择合适的独享节点规格，以提高应用服务网格的可用性。

QPS（每秒请求数）总数	0~20000	20000~60000
节点规格	8U16G	16U32G

#### 说明

- 其中QPS总数为集群中所有应用的所有组件QPS总和。
- 即将提供应用服务实例个数与控制面内存资源的匹配推荐。

### ingressgateway 实例资源消耗参考

每个ingressgateway实例的资源消耗与连接类型、连接数量、QPS有关，可以参考以下数据：

表 1-1 长连接内存消耗

连接数量	内存消耗（MB）
1	0.055
1000	55
10000	550

表 1-2 短连接 CPU 和内存消耗

QPS	CPU消耗（m）	内存消耗（MB）
100	30	100
1000	300	100
10000	3000	150

以上数据仅供参考，具体的资源消耗和实际的业务模型有关，以实际测试结果为准。

## 1.7 与其他云服务的关系

应用服务网格与周边服务的依赖关系如图1-1所示。

图 1-1 应用服务网格与其他云服务关系



### 云容器引擎 CCE

云容器引擎（Cloud Container Engine，CCE）提供高可靠高性能的企业级容器应用管理服务，支持Kubernetes社区原生应用和工具，简化云上自动化容器运行环境搭建。

您可以为CCE集群启用服务网格功能，对集群中的服务进行治理。

### 弹性负载均衡 ELB

弹性负载均衡（Elastic Load Balance，ELB）将访问流量自动分发到多台云服务器，扩展应用系统对外的服务能力，实现更高水平的应用容错。

您可以通过弹性负载均衡从外部访问ASM。

# 2 快速入门

---

## 2.1 Bookinfo 应用的灰度发布实践

应用服务网格（Application Service Mesh，简称ASM）是基于开源Istio推出的服务网格平台，它深度、无缝对接了企业级Kubernetes集群服务云容器引擎（CCE），在易用性、可靠性、可视化等方面进行了一系列增强，可为客户提供开箱即用的上手体验。

### 入门指引

灰度发布是迭代软件产品在生产环境安全上线的一种重要手段。本教程以Bookinfo应用为例，向您讲解服务网格基于Istio提供的服务治理能力。

Bookinfo应用灰度发布流程包含以下步骤：



图 2-1 入门流程



## Bookinfo 应用分析

Bookinfo是一个模仿在线书店的应用，页面上会显示一本书籍的描述，书籍的细节（如页数），以及关于书籍的一些评论。

Bookinfo应用由四个单独的服务构成，几个服务是由不同的语言编写的。这些服务对应用服务网格ASM并无依赖，但是构成了一个有代表性的服务网格的例子，即由多个服务、多个语言构成，且reviews服务具有多个版本。这四个服务的说明如下：

- productpage：会调用details和reviews两个服务，用来生成页面。
- details：包含了书籍的信息。

- reviews: 包含了书籍相关的评论, 同时会调用ratings服务。
- ratings: 包含了由书籍评价组成的评级信息。

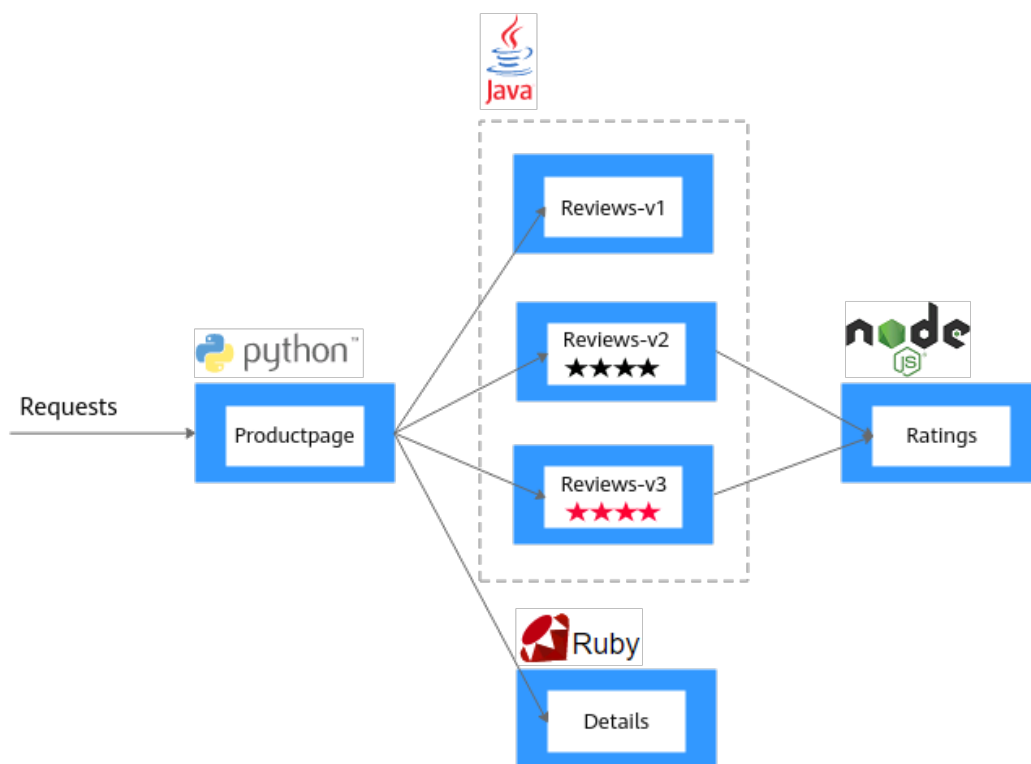
其中, reviews服务有3个版本:

- v1 ( 1.17.0 ) 版本不会调用ratings服务。
- v2 ( 1.17.1 ) 版本会调用ratings服务, 并使用1到5个黑色星形图标来显示评分信息。
- v3 ( 1.17.2 ) 版本会调用ratings服务, 并使用1到5个红色星形图标来显示评分信息。

#### 📖 说明

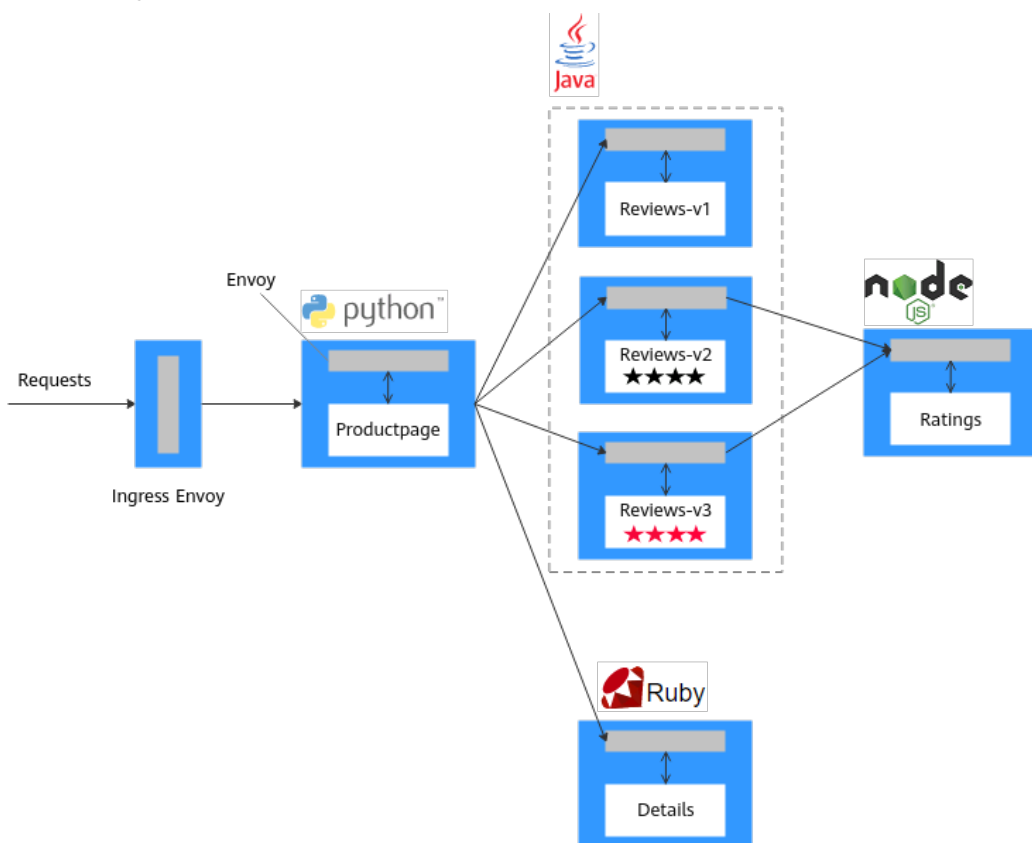
为了直观的展示灰度版本之间流量切换情况, 本教程以reviews服务的1.17.1版本 ( 黑星形 )、1.17.2版本 ( 红星形 ) 为例进行说明。

图 2-2 Bookinfo 应用的端到端架构



在ASM中运行Bookinfo应用, 无需对应用自身做出任何改变, 只需简单的在ASM环境中对服务进行配置和运行, 即把Envoy Sidecar注入到每个服务之中。最终的部署结果如图2-3所示。

图 2-3 Envoy Sidecar 注入之后的 Bookinfo 应用



所有的服务都和Envoy Sidecar集成在一起，被集成服务的所有出入流量都被Sidecar所劫持，这样就可以利用ASM为应用提供服务路由、遥测数据收集以及策略实施等功能。

## 准备工作

在开始之前，您需要完成如下的准备工作。

### 步骤1 创建虚拟私有云和子网。

虚拟私有云（Virtual Private Cloud，简称VPC）提供一个隔离的、用户自主配置和管理的虚拟网络环境，可以提升资源的安全性，简化用户的网络部署。

1. 登录虚拟私有云VPC控制台。
2. 单击右上角“创建虚拟私有云”。
3. 根据界面提示完成参数填写，单击“立即创建”。

### 步骤2 （可选）创建密钥对。

创建集群节点时，如果使用密钥对登录方式，需要提前创建好密钥对。

1. 登录弹性云服务器ECS控制台。
2. 选择左侧导航中的“密钥对”，单击右上角“创建密钥对”。
3. 输入密钥对名称后，单击“确定”。
4. 您的浏览器会提示您下载或自动下载私钥文件。文件名是您为密钥对指定的名称，文件扩展名为“.pem”。请将私钥文件保存在安全位置。然后在系统弹出的提示框中单击“确定”。

### 📖 说明

为保证安全，私钥只能下载一次，请妥善保管，否则将无法登录节点。

#### 步骤3 创建负载均衡。

弹性负载均衡将作为服务网格对外访问入口，被服务网格管理的应用流量，均从此实例进入并分发到后端服务。

1. 登录弹性负载均衡ELB控制台。
2. 单击右上角的“创建弹性负载均衡”。
3. 所属VPC、子网请选择**步骤1**中创建的虚拟私有云和子网，其他参数根据界面提示填写，单击“立即申请”。

#### 步骤4 创建集群。

1. 登录云容器引擎CCE控制台。
2. 选择左侧导航中的“资源管理 > 集群管理”，单击右上角“创建CCE集群”。
3. 在“服务选型”页面设置如下参数，其余参数均采用默认值。
  - 集群名称：用户自行输入，此处设置为“cce-asm”。
  - 虚拟私有云、所在子网：选择**步骤1**中创建的虚拟私有云和子网。
4. 单击“下一步：创建节点”，配置添加节点参数。除节点规格和登录方式外，其余参数保持默认。
  - 节点规格：vCPUs为4核，内存为8GB。

### 📖 说明

此规格为部署Bookinfo应用所需的最小资源。

- 登录方式：选择**步骤2**中创建的密钥对，用于远程登录节点时的身份认证。
5. 单击“下一步：安装插件”，在“安装插件”步骤中选择要安装的插件。“系统资源插件”为必装插件，“高级功能插件”可根据实际需求进行选择安装。
  6. 单击“下一步：配置确认”，阅读使用说明并勾选“我已知晓上述限制”，确认所设置的服务选型参数、规格等信息。
  7. 确认订单无误后，单击“提交”，集群开始创建。

集群创建预计需要6-10分钟，您可以单击“返回集群管理”进行其他操作或单击“查看集群事件列表”后查看集群详情。

----结束

## 创建网络

**步骤1** 登录应用服务网格ASM控制台。

**步骤2** 单击右上角“创建网络”。

**步骤3** 设置如下参数，其余参数均采用默认值。

- **网络类型**  
默认为基础版。
- **网络名称**  
设置网络的名称。

- **Istio版本**  
网格支持的Istio版本。
- **集群**  
选择**步骤4**中创建的集群。
- **Istio控制面节点**  
如果需要高可用，建议选择两个或以上不同可用区的节点。

**步骤4** 设置完成后，在右侧的配置清单中确认网格配置，单击“提交”。

创建时间预计需要1~3分钟，请耐心等待。当网格状态从“安装中”变为“运行中”，表示网格创建成功。

----结束

## 一键创建 Bookinfo 应用

为集群开启应用服务网格功能后，可以通过“体验任务”创建一个Bookinfo应用Demo，具体操作如下：

**步骤1** 登录应用服务网格ASM控制台。

**步骤2** 单击网格名称，进入详情页面。

**步骤3** 选择左侧导航中的“体验任务”，单击Bookinfo任务中的“安装”。

**步骤4** 在右侧页面设置Bookinfo应用所在的集群，在“负载均衡”中选择与所选集群处于同一VPC和子网的负载均衡实例，并设置一个对外端口，单击“安装”。

图 2-4 安装 Bookinfo

点击下方“安装”按钮，为您一键式创建 bookinfo 体验模板，包含 productpage、details、reviews、ratings等服务。

网格

所在集群

负载均衡   [创建负载均衡](#)

仅支持集群所在 VPC vpc-default 下的公网负载均衡实例，查询结果已自动过滤

对外端口

**步骤5** 等待Bookinfo应用创建完成。创建完成后进入“服务管理”页面，配置诊断栏将显示为“正常”，Bookinfo应用包含productpage、details、reviews、ratings四个服务。

图 2-5 服务列表

服务名称	配置诊断	访问地址
details	正常	http://details.default.svc:9080 HTTP
productpage	正常	http://:8080/productpage HTTP
		http://:8080/ HTTP
		http://productpage.default.svc:9080/ HTTP
ratings	正常	http://ratings.default.svc:9080 HTTP
reviews	正常	http://reviews.default.svc:9080 HTTP

----结束

## 为服务添加灰度版本

本步骤将为Bookinfo应用的“reviews”服务添加新的灰度版本，并配置相应的灰度策略，将原有生产环境的默认版本的流量引流一部分到新版本中。

下面将以为“reviews”服务添加一个v3新版本，且v3新版本接收Bookinfo应用的30%流量为例进行配置。

### 部署灰度版本

**步骤1** 在左侧导航中选择“灰度发布”，在金丝雀发布下，单击“立即发布”。

**步骤2** 配置灰度发布基本信息。

- 灰度任务名称：用户自定义，此处设置为reviews-v3。
- 命名空间：选择服务所在命名空间。
- 灰度发布服务：在下拉框中选择reviews。
- 工作负载：选择服务所属的工作负载。

**步骤3** 配置灰度版本信息。

- 部署集群：选择服务所属的集群。
- 版本号：配置为v3。
- 实例数量：使用默认。
- 实例配置：镜像版本选择1.17.2，其他参数保持默认。

**步骤4** 单击“发布”，待启动进度为100%，表明灰度版本部署成功。

----结束

### 配置流量策略

为灰度版本设置流量策略，灰度版本会根据配置的流量配比引流老版本中的部分或全部流量。

**步骤1** 灰度版本部署成功后，单击“配置流量策略”。

**步骤2** 设置流量策略。

策略类型分为“基于流量比例”和“基于请求内容”，通过页签选择确定。

- 基于流量比例：根据流量比例配置规则，将从原版本中切分指定比例的流量到灰度版本。例如80%的流量走原版本，20%的流量走灰度版本。

- 基于请求内容：根据请求内容配置规则，只有请求内容中满足特定条件的流量会切分到灰度版本上。例如只有在Windows操作系统上的用户可以访问灰度版本。

以“基于流量比例”为例，且v3版本流量配比为20%。

图 2-6 流量策略

* 流量配比	版本	流量配比	当前实例数
	v1	80 %	1
	v3 (灰度版本)	20 %	1

**步骤3** 单击“策略下发”。

**步骤4** 在“服务列表”页面，单击productpage服务中的“访问地址”。不断刷新页面，页面在v1和v3版本之间来回切换，并且比例大致接近4:1。

图 2-7 v1 版本页面

The screenshot shows a web page for 'The Comedy of Errors'. The page has a dark header with 'BookInfo Sample' and a 'Sign in' button. The main content area is white and features the title 'The Comedy of Errors' in blue. Below the title is a summary: '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.' There are two tabs: 'Book Details' and 'Book Reviews'. The 'Book Details' tab is active, showing: Type: paperback, Pages: 200, Publisher: PublisherA, Language: English, ISBN-10: 1234567890, ISBN-13: 123-1234567890. The 'Book Reviews' tab shows two reviews. The first review is by 'Reviewer1' and has a rating of 5 stars (★★★★★). The second review is by 'Reviewer2' and has a rating of 4 stars (★★★★☆).

图 2-8 v3 版本页面

The screenshot shows the same web page as in Figure 2-7, but for the v3 version. The layout is identical, but the 'Book Reviews' tab is active. The first review by 'Reviewer1' now has a rating of 4 stars (★★★★☆), and the second review by 'Reviewer2' has a rating of 3 stars (★★★☆☆). This indicates that the v3 version is receiving a significant portion of the traffic, as evidenced by the change in the number of reviews.

----结束

## 灰度版本切换

检查v3版本的资源数与v1版本是否相匹配，确认其能承接v1的所有流量后，即可将v1流量全部切换到v3。

**步骤1** 在“监测与处理”页面，单击v3版本后的“全流量接管”。

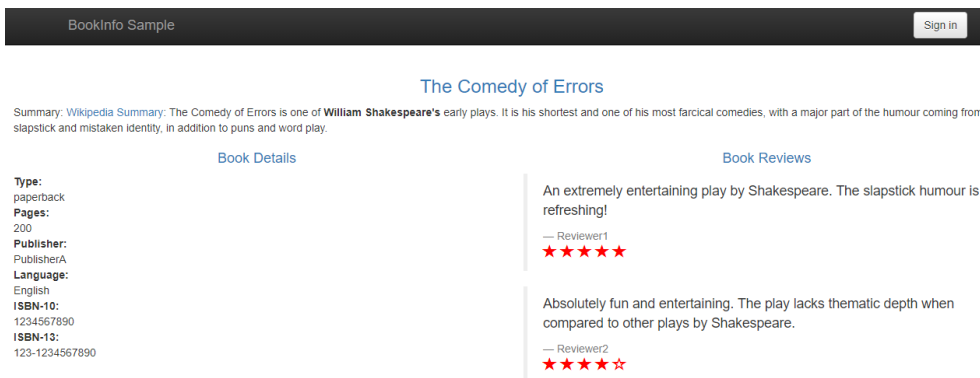
图 2-9 全流量接管



**步骤2** 单击“确定”。

页面右上角会提示全流量接管成功。在Bookinfo应用页面，不断刷新页面，页面仅显示v3版本信息，即星形图标全部为红色。

图 2-10 v3 版本页面



----结束

## 结束灰度任务

v3承接v1版本所有流量后，即可结束灰度任务，释放v1版本的资源。

**步骤1** 在“监测与处理”页面，单击“结束灰度任务”。

**步骤2** 单击“确定”，结束灰度任务将下线原版本，并删除灰度任务。

图 2-11 结束灰度任务



下线服务版本，会将包含的工作负载和Istio相关配置资源全部删除。

----结束



## 清除资源

到此本Demo已全部操作完成，为了避免资源的浪费，请及时删除应用和节点。

**步骤1** 选择左侧导航中的“体验任务”，单击Bookinfo任务中的“卸载”。

**步骤2** 单击“确定”。卸载Bookinfo体验任务，会自动删除productpage、details、reviews、ratings服务及相关资源。

图 2-12 卸载体验任务



### 📖 说明

卸载体验任务后，已完成灰度发布的服务，其灰度版本对应的负载需要手动在CCE控制台删除。

----结束

## 2.2 为集群开通 Istio

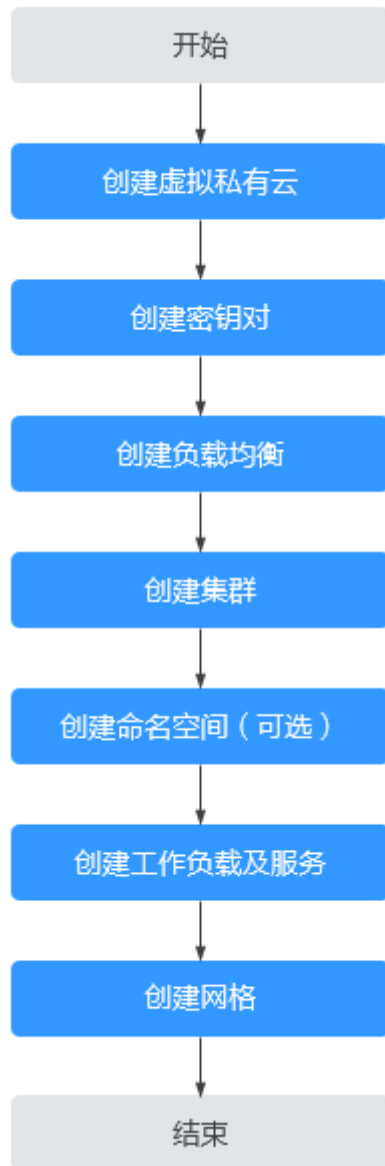
### 2.2.1 入门概述

应用服务网格提供非侵入式的微服务治理解决方案，支持完整的生命周期管理和流量治理，兼容Kubernetes和Istio生态，其功能包括负载均衡、熔断、限流等多种治理能力。

### 流程说明

为集群开通Istio的流程包含以下步骤：

图 2-13 为集群开通 Istio 的流程图



## 2.2.2 准备工作

为集群开通Istio之前，您需要完成如下的准备工作。

### 创建虚拟私有云

虚拟私有云（Virtual Private Cloud，简称VPC）提供一个隔离的、用户自主配置和管理的虚拟网络环境，可以提升资源的安全性，简化用户的网络部署。

**步骤1** 登录虚拟私有云VPC控制台。

**步骤2** 单击右上角“创建虚拟私有云”。

**步骤3** 根据界面提示完成参数填写，单击“立即创建”。

----结束

## 创建密钥对

新建一个密钥对，用于远程登录节点时的身份认证。

- 步骤1** 登录弹性云服务器ECS控制台。
- 步骤2** 选择左侧导航中的“密钥对”，单击右上角“创建密钥对”。
- 步骤3** 输入密钥对名称后，单击“确定”。
- 步骤4** 您的浏览器会提示您下载或自动下载私钥文件。文件名是您为密钥对指定的名称，文件扩展名为“.pem”。请将私钥文件保存在安全位置。然后在系统弹出的提示框中单击“确定”。

### 说明

为保证安全，私钥只能下载一次，请妥善保管，否则将无法登录节点。

----结束

## 创建负载均衡

弹性负载均衡将作为服务网格对外访问入口，被服务网格管理的应用流量，均从此实例进入并分发到后端服务。

- 步骤1** 登录弹性负载均衡ELB控制台。
- 步骤2** 单击右上角的“创建弹性负载均衡”。
- 步骤3** 所属VPC、子网请选择[创建虚拟私有云](#)中创建的虚拟私有云和子网，其他参数根据界面提示填写，单击“立即申请”。

----结束

## 创建集群

- 步骤1** 登录云容器引擎CCE控制台。
- 步骤2** 选择左侧导航中的“资源管理 > 集群管理”，单击右上角“创建CCE集群”。
- 步骤3** 在“服务选型”页面设置如下参数，其余参数均采用默认值。
  - 集群名称：用户自行输入，此处设置为“cluster-test”。
  - 虚拟私有云、所在子网：选择[创建虚拟私有云](#)中创建的虚拟私有云和子网。
- 步骤4** 单击“下一步：创建节点”，配置添加节点的参数。除节点规格和登录方式外，其余参数保持默认。
  - 节点规格：vCPUs为4核，内存为8GB。
  - 登录方式：选择[创建密钥对](#)中创建的密钥对，用于远程登录节点时的身份认证。
- 步骤5** 单击“下一步：安装插件”，在“安装插件”步骤中选择要安装的插件。

“系统资源插件”为必装插件，“高级功能插件”可根据实际需求进行选择安装。
- 步骤6** 单击“下一步：配置确认”，阅读使用说明并勾选“我已知晓上述限制”，确认所设置的服务选型参数、规格等信息。
- 步骤7** 确认订单无误后，单击“提交”，集群开始创建。

集群创建预计需要6-10分钟，您可以单击“返回集群管理”进行其他操作或单击“查看集群事件列表”后查看集群详情。

----结束

## 创建命名空间（可选）

**步骤1** 登录云容器引擎CCE控制台。

**步骤2** 选择左侧导航中的“资源管理 > 命名空间”，单击右上角“创建命名空间”。

**步骤3** 输入命名空间的名称，并选择已创建的集群。

**步骤4** 单击“确定”。

----结束

## 创建工作负载及服务

**步骤1** 登录云容器引擎CCE控制台。

**步骤2** 选择左侧导航中的“工作负载 > 无状态负载 Deployment”，单击右上角“创建无状态工作负载”。

**步骤3** 参考《云容器引擎 用户指南》中的指导，创建工作负载和服务。

----结束

## 2.2.3 创建网格

ASM支持创建基础版网格，该网格为标准版本网格，提供商用级网格服务。

### 操作步骤

**步骤1** 登录应用服务网格ASM控制台，单击右上角“创建网格”。

**步骤2** 设置如下参数，其余参数均采用默认值。

- **网格类型**  
默认为基础版。
- **网格名称**  
设置网格的名称。
- **Istio版本**  
网格支持的Istio版本。
- **集群**  
选择[创建集群](#)中创建的集群。
- **Istio控制面节点**  
如果需要高可用，建议选择两个或以上不同可用区的节点。

**步骤3** 设置完成后，在右侧的配置清单中确认网格配置，单击“提交”。

创建时间预计需要1~3分钟，请耐心等待。当网格状态从“安装中”变为“运行中”，表示网格创建成功。

----结束

## 2.3 配置式应用灰度发布

### 2.3.1 入门概述

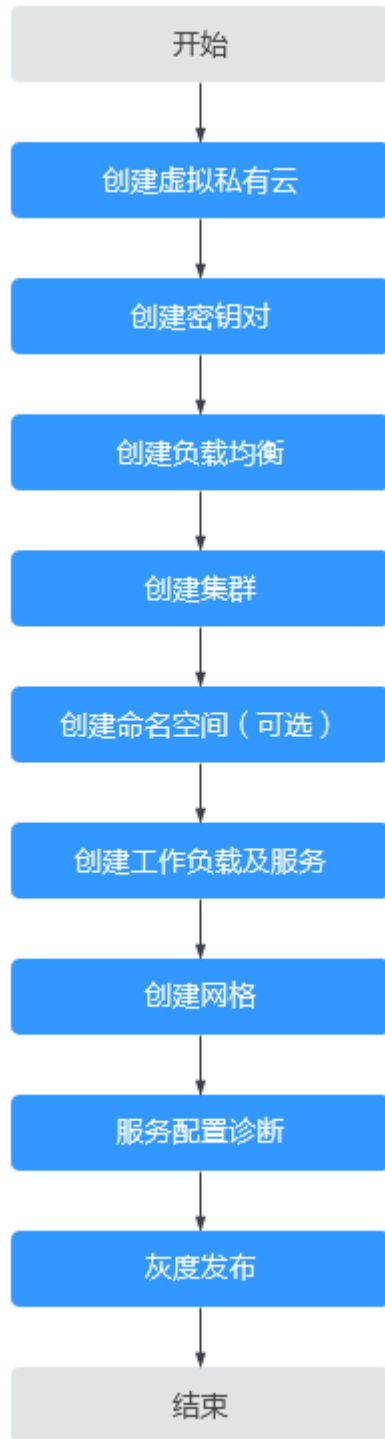
灰度发布是版本升级平滑过渡的一种方式，当版本升级时，使部分用户使用高版本，其他用户继续使用低版本，待高版本稳定后，逐步扩大范围把所有用户流量都迁移到高版本上面来。

本章将从创建虚拟私有云开始，到创建一个灰度版本，讲解如何实现一次灰度发布。

#### 流程说明

实现灰度发布的流程包含以下步骤：

图 2-14 实现灰度发布的流程图



## 2.3.2 准备工作

应用灰度发布之前，您需要完成如下的准备工作。

### 创建虚拟私有云

虚拟私有云（Virtual Private Cloud，简称VPC）提供一个隔离的、用户自主配置和管理的虚拟网络环境，可以提升资源的安全性，简化用户的网络部署。

- 步骤1** 登录虚拟私有云VPC控制台。
- 步骤2** 单击右上角“创建虚拟私有云”。
- 步骤3** 根据界面提示完成参数填写，单击“立即创建”。

----结束

## 创建密钥对

新建一个密钥对，用于远程登录节点时的身份认证。

- 步骤1** 登录弹性云服务器ECS控制台。
- 步骤2** 选择左侧导航中的“密钥对”，单击右上角“创建密钥对”。
- 步骤3** 输入密钥对名称后，单击“确定”。
- 步骤4** 您的浏览器会提示您下载或自动下载私钥文件。文件名是您为密钥对指定的名称，文件扩展名为“.pem”。请将私钥文件保存在安全位置。然后在系统弹出的提示框中单击“确定”。

### 说明

为保证安全，私钥只能下载一次，请妥善保管，否则将无法登录节点。

----结束

## 创建负载均衡

弹性负载均衡将作为服务网格对外访问入口，被服务网格管理的应用流量，均从此实例进入并分发到后端服务。

- 步骤1** 登录弹性负载均衡ELB控制台。
- 步骤2** 单击右上角的“创建弹性负载均衡”。
- 步骤3** 所属VPC、子网请选择[创建虚拟私有云](#)中创建的虚拟私有云和子网，其他参数根据界面提示填写，单击“立即申请”。

----结束

## 创建集群

- 步骤1** 登录云容器引擎CCE控制台。
- 步骤2** 选择左侧导航中的“资源管理 > 集群管理”，单击右上角“创建CCE集群”。
- 步骤3** 在“服务选型”页面设置如下参数，其余参数均采用默认值。
  - 集群名称：用户自行输入，此处设置为“cluster-test”。
  - 虚拟私有云、所在子网：选择[创建虚拟私有云](#)中创建的虚拟私有云和子网。
- 步骤4** 单击“下一步：创建节点”，配置添加节点的参数。除节点规格和登录方式外，其余参数保持默认。
  - 节点规格：vCPUs为4核，内存为8GB。
  - 登录方式：选择[创建密钥对](#)中创建的密钥对，用于远程登录节点时的身份认证。

**步骤5** 单击“下一步：安装插件”，在“安装插件”步骤中选择要安装的插件。

“系统资源插件”为必装插件，“高级功能插件”可根据实际需求进行选择性安装。

**步骤6** 单击“下一步：配置确认”，阅读使用说明并勾选“我已知晓上述限制”，确认所设置的服务选型参数、规格等信息。

**步骤7** 确认订单无误后，单击“提交”，集群开始创建。

集群创建预计需要6-10分钟，您可以单击“返回集群管理”进行其他操作或单击“查看集群事件列表”后查看集群详情。

----结束

## 创建命名空间（可选）

**步骤1** 登录云容器引擎CCE控制台。

**步骤2** 选择左侧导航中的“资源管理 > 命名空间”，单击右上角“创建命名空间”。

**步骤3** 输入命名空间的名称，并选择已创建的集群。

**步骤4** 单击“确定”。

----结束

## 创建工作负载及服务

**步骤1** 登录云容器引擎CCE控制台。

**步骤2** 选择左侧导航中的“工作负载 > 无状态负载 Deployment”，单击右上角“创建无状态工作负载”。

**步骤3** 参考《云容器引擎 用户指南》中的指导，创建工作负载和服务。

----结束

## 创建网格

**步骤1** 登录应用服务网格ASM控制台，单击右上角“创建网格”。

**步骤2** 设置网格名称为“asmtest”，选择在[创建集群](#)中创建的名称为“cluster-test”的集群，并选择安装Istio控制面的节点，建议选择两个或以上不同可用区的节点。

**步骤3** 展开高级配置，在“sidecar配置”中勾选名称为“default”的命名空间，选择重启已有服务。

**步骤4** 设置完成后，在页面右侧配置清单确认网格配置，确认无误后，单击“提交”。

创建网格预计需要1~3分钟，请耐心等待。当网格状态从“安装中”变为“运行中”，表示网格创建成功。

----结束

## 服务配置诊断

应用服务网格会对管理集群下的所有服务进行诊断，诊断结果为正常的服务才能进行灰度发布。



- 步骤1** 登录应用服务网格ASM控制台，单击名称为“asmtest”的网格，进入网格详情页面。
- 步骤2** 在左侧导航栏选择“服务管理”，选择“命名空间：default”并查看“servicetest”的配置诊断状态。
- 步骤3** 如果配置诊断为异常，单击“处理”，根据修复指导的操作修复，直到服务的配置诊断状态为正常。

----结束

## 2.3.3 灰度发布

### 为服务添加灰度版本


- 步骤1** 登录应用服务网格ASM控制台，单击“asmtest”网格内的 。
- 步骤2** 创建灰度任务名称为“test”的灰度任务，并配置基本信息及灰度版本信息，灰度发布服务选择[创建工作负载及服务](#)中创建的名称为“servicetest”的服务，工作负载会自动关联“deptest”，单击“发布”。

图 2-15 创建灰度任务

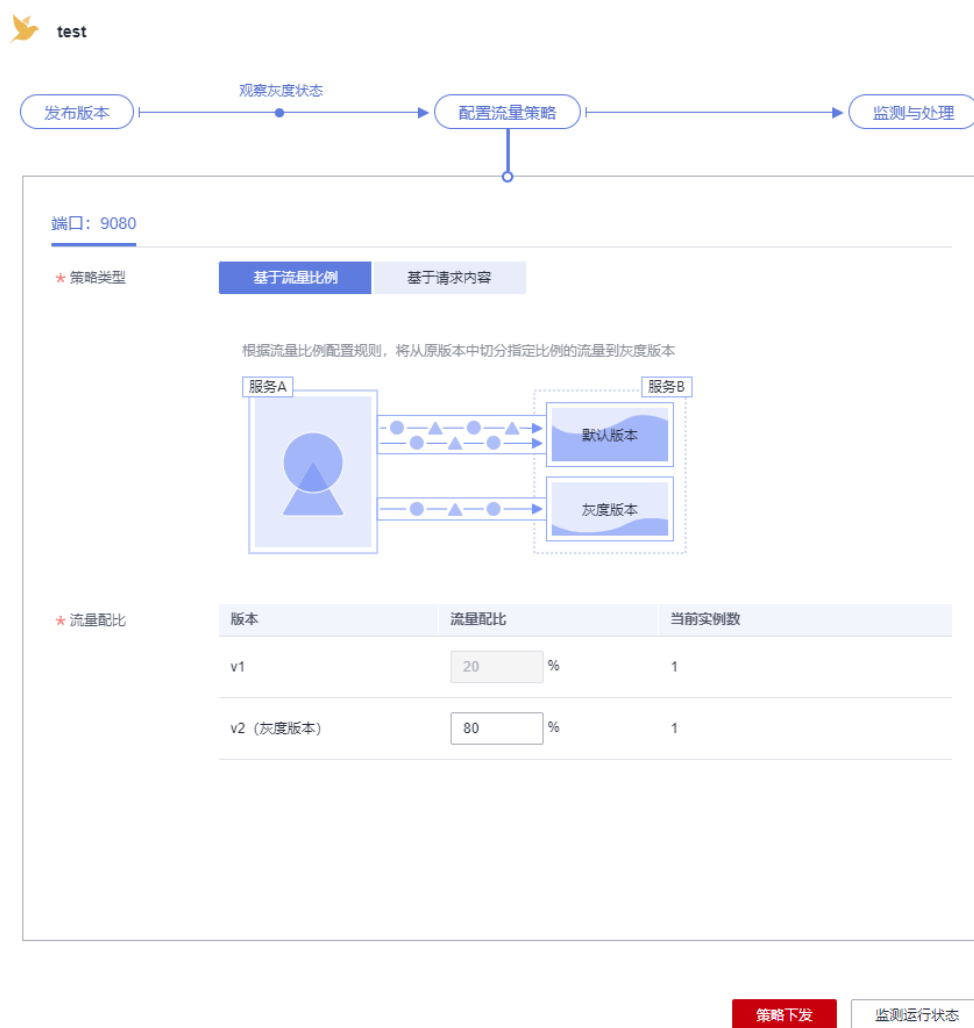


### 说明

如果服务“servicetest”无法选择，需要确认服务是否异常，修复后才能选择。

**步骤3** 单击“配置流量策略”，选择“基于流量比例”策略类型，为v2（灰度版本）配置“流量配比”为80%。

图 2-16 配置流量策略



**步骤4** 单击“策略下发”。

灰度策略的生效需要几秒的时间，您可以在监测灰度运行状态页面，观察灰度版本的运行状态。

----结束

## 灰度版本切换

检查v2版本的资源数与v1版本是否相匹配，确认其能承接v1的所有流量后，即可将v1流量全部切换到v2。

**步骤1** 在“灰度发布”页面，单击灰度任务名为“test”后的“监测与处理”。

**步骤2** 单击灰度版本 v2后的“全流量接管”。

**图 2-17** 全流量接管



**步骤3** 在弹出的“全流量接管”窗口单击“确定”，页面右上角会提示全流量接管成功。

----结束

## 将原版本下线

v2承接v1所有流量后，即可删除v1版本，释放v1版本的资源。

**步骤1** 在“灰度发布”页面，单击灰度任务名为“test”后的“监测与处理”。

**步骤2** 在“监测与处理”页面中，当v2版本中的流量占比为100%时，表明v2已承接v1所有流量。单击“结束灰度任务”。

**步骤3** 确认无误后，单击“确定”。

结束灰度任务将下线v1版本，并删除test灰度任务。

----结束

# 3 用户指南

## 3.1 创建网格

### 3.1.1 创建网格

ASM支持创建基础版网格，该网格为标准版本网格，提供商用级网格服务。

#### 前提条件

已创建CCE集群，如果未创建，请先创建集群。

#### 约束与限制

- 应用服务网格依赖集群CoreDNS的域名解析能力，请确保集群拥有足够资源，且CoreDNS插件运行正常
- 集群启用Istio时，需要开通node节点（计算节点/工作节点）所在安全组的入方向7443端口规则，用于Sidecar自动注入回调。如果您使用CCE创建的默认安全组，此端口会自动开通。如果您自建安全组规则，请手动开通7443端口，以确保Istio自动注入功能正常

#### 操作步骤

**步骤1** 登录应用服务网格控制台。

**步骤2** 单击右上角“创建网格”。

**步骤3** 设置网格参数。

- **网格类型**  
仅支持基础版，该网格为标准版本网格，提供商用级网格服务。
- **网格名称**  
网格的名称，取值必须以小写字母开头，由小写字母、数字、中划线（-）组成，且不能以中划线（-）结尾，长度范围为4~64个字符。  
同一账号下网格不可重名，且网格名称创建后不可修改。
- **Istio版本**

网格支持的Istio版本。

- **集群**

在集群列表中选择集群，或在列表右上角输入集群名称搜索需要的集群。仅可选择当前网格版本支持的集群版本。

- **Istio控制面节点**

基础版网格的控制面组件安装在用户集群，因此需要选择用于安装Istio控制面的节点。如果需要高可用，建议选择两个或以上不同可用区的节点。

所选节点会被添加istio:master标签，网格组件会调度到该节点上。

#### 步骤4 （可选）高级配置。


- **sidecar配置**

选择命名空间，为命名空间设置标签istio-injection=enabled，其中的Pod在重启后会自动注入istio-proxy sidecar。

如果不进行sidecar配置，可在网格创建成功后在“网格配置 > sidecar管理”中注入sidecar。具体操作请参考[sidecar注入](#)。

- **是否重启已有服务**

：会重启命名空间下已有服务关联的Pod，将会暂时中断业务。只有在重启后，已有服务关联的Pod才会自动注入istio-proxy sidecar。

：已有服务关联的Pod不会自动注入istio-proxy sidecar，需要在CCE控制台，手动重启工作负载才会注入sidecar。

#### 步骤5 设置完成后，在页面右侧配置清单确认网格配置，确认无误后，单击“提交”。

创建网格预计需要1~3分钟，请耐心等待。当网格状态从“安装中”变为“运行中”，表示网格创建成功。

#### 说明

启用网格期间会操作如下资源：

- 创建一个Helm应用编排release对象，作为服务网格控制面的资源。
- 开通节点的安全组，允许7443端口的入流量，使其支持对Pod进行自动注入。

----结束

## 3.2 网格管理

### 3.2.1 卸载网格

#### 操作场景


当网格不再需要时，可以将其卸载。

#### 约束与限制

- 如果网格有正在运行的灰度发布任务，请先完成灰度发布，再卸载网格。
- 请确保集群中有可用节点，用于运行清理任务，否则将导致卸载失败。

## 操作步骤

**步骤1** 登录应用服务网格控制台。

**步骤2** 单击对应网格下的  图标。

**步骤3** 在“卸载服务网格”页面，选择是否重启已有服务，并阅读注意事项。

卸载时，默认不会重启已有服务。只有在服务重启后才会去除注入的istio-proxy sidecar，如需重启，请选择“是”，重启服务将会暂时中断您的业务。

### 说明

建议您选择重启已有服务，如果不重启，会引起如下异常：当前网格卸载后，集群重新启用网格的情况下，网关会访问失败。

- 卸载服务网格将会卸载Istio控制面组件及数据面sidecar。
- 卸载后，应用的对外访问方式将无法继续使用，请将旧的对外访问方式改为用service方式对外发布。

如需更新对外访问方式，请在CCE控制台“资源管理 > 网络管理 > Service”页面创建服务暴露对外访问方式。

- 卸载时将自动为您清理istio独享节点的相关标签，但不会删除istio-master节点，请到CCE界面删除，避免资源浪费。

如需查看节点信息，请在CCE控制台“资源管理 > 节点管理”页面中查看。

----结束

## 3.3 服务管理

### 3.3.1 配置诊断

应用服务网格会对管理集群下的所有服务进行诊断，诊断结果为正常的服务，方可进行流量治理及灰度发布等操作。

### 约束与限制

- 如果多个服务对应一个工作负载（Deployment），则不允许将这些服务加入网格进行治理，因为可能出现灰度发布、网关访问等功能异常。
- 如果服务的工作负载使用主机网络模式（Pod配置了hostNetwork: true），则不支持注入sidecar。

### 服务诊断

**步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。

**步骤2** 在左侧导航栏选择“服务管理”，服务列表中展示了各服务的诊断结果。

如果服务存在异常，请单击“处理”，根据[服务异常修复](#)进行处理。

图 3-1 服务诊断

服务名称	配置诊断	访问地址
nginx	异常 <a href="#">处理</a> <a href="#">重新诊断</a>	<a href="#">内</a> http://nginx.default.svc:80 HTTP <a href="#">🔗</a>
nginx2	异常 <a href="#">处理</a> <a href="#">重新诊断</a>	<a href="#">内</a> http://nginx2.default.svc:80 HTTP <a href="#">🔗</a>

**步骤3** 修复异常项后，您可以单击“重新诊断”对服务进行再次诊断。

----结束

## 服务异常修复

诊断异常的服务，需要先手动修复异常状态的手动处理项，再一键修复可自动处理项。

**步骤1** 在诊断状态为异常的服务下单击“处理”，若手动处理项有异常，根据修复指导进行手动修复。

图 3-2 手动处理项

### 配置诊断 ×

① 手动处理项 ————— ② 可自动处理项

检查项	检查结果	操作
所有Pod是否配置了app和version标签	成功	<a href="#">修复指导</a>
所有Pod的app和version标签是否都相等	成功	<a href="#">修复指导</a>
所有Pod是否都注入了sidecar	失败 <a href="#">?</a>	<a href="#">修复指导</a>

[重新诊断](#) [下一步](#)

**步骤2** 手动修复异常状态的手动处理项后，单击“下一步”进入自动修复项页面，单击“一键修复”，自动处理异常状态的检查项。

图 3-3 自动处理项

×

### 配置诊断

① 手动处理项 ———— ② 可自动处理项

检查项	检查结果	操作
Service 的端口名称是否符合istio规范	<span style="color: green;">●</span> 成功 访问端口 80 协议 <input type="text" value="http"/>	<a href="#">修复指导</a>
Service的选择器中是否配置了version标签	<span style="color: green;">●</span> 成功	<a href="#">修复指导</a>
服务是否配置了默认版本的服务路由，路由配置是否正确	<span style="color: red;">●</span> 失败 <span style="font-size: small;">?</span>	<a href="#">修复指导</a>

重新诊断 一键修复

#### 📖 说明

- 如果自动处理无法修复状态异常的检查项，请根据修复指导进行手动修复。
- 已配置网关或创建灰度发布的服务可能因为Service的端口名称被修改而出现异常，此时不支持进行一键修复。
- 如果服务未在服务列表中展示，请检查对应的工作负载是否存在。

---结束

## 3.3.2 手动修复项

### 3.3.2.1 所有 Pod 是否都配置了 app 和 version 标签

#### 问题描述

Service关联的所有Pod都必须配置app和version标签。app标签在流量监控中用于流量的跟踪，version标签在灰度发布中用于区分不同版本。如果存在未配置app或version标签的Pod，则报此异常。

#### 修复指导

Pod标签配置在Deployment的spec.template.metadata.labels中，建议配置为：

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

#### ⚠️ 注意

修改或删除Deployment会触发Pod滚动升级，可能会导致业务短暂中断，请根据业务场景选择适当的时间修改。



**步骤1** 复制原有工作负载配置，保存为YAML文件。

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

例如：

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

**步骤2** 修改productpage-deployment.yaml内容，如果没有app和version，需要添加。app的值建议与Service名称一致，version建议为v1。

**步骤3** 删除原工作负载。

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

**步骤4** 应用新的工作负载配置。

```
kubectl apply -f productpage-deployment.yaml
```

----结束

#### 说明

1.15及以下集群还可以在CCE控制台直接修改Pod的标签，但有可能会导致ReplicaSet残留。判断是否存在ReplicaSet残留的方法如下：

1. 查询Deployment的ReplicaSet。

```
kubectl get replicaset | grep {deploymentName}
```

2. 找到实例个数大于1的ReplicaSet，如果ReplicaSet个数大于1，可能是修改label导致ReplicaSet残留。需要删除旧配置的ReplicaSet。

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

### 3.3.2.2 所有 Pod 的 app 和 version 标签是否都相等

#### 问题描述

Service关联的所有Pod的app和version标签必须都相等。app标签在流量监控中用于流量的跟踪，version标签在灰度发布中用于区分不同版本。如果存在app或version标签不相等的Pod，则报此异常。

#### 修复指导

Pod标签配置在Deployment的spec.template.metadata.labels中，建议配置为：

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

修改多个Pod标签为相等的操作方法如下：

**步骤1** 查看Service选择器（spec.selector）配置的标签。

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

例如，标签是app: ratings和release: istio-bookinfo。

**步骤2** 根据标签查找Service关联的Pod。

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

#### 📖 说明

{namespace}和Service的namespace一致。

**步骤3** 根据Pod名称，找到其关联的工作负载。

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

#### 📖 说明

- 一般Pod名称格式为{deploymentName}-{随机字符串}-{随机字符串}。
- 如果根据Pod名称未查询到工作负载，可能是因为ReplicaSet有残留，需要将其删除。判断是否存在ReplicaSet残留的方法如下：

1. 查询Deployment的ReplicaSet。

```
kubectl get replicaset | grep {deploymentName}
```

2. 找到实例个数大于1的ReplicaSet，如果ReplicaSet个数大于1，可能是修改label导致ReplicaSet残留。需要删除旧配置的ReplicaSet。

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

**步骤4** 请参考[修复指导](#)修改工作负载中Pod的app和version标签。

----结束

### 3.3.2.3 所有 Pod 是否都注入了 sidecar

#### 问题描述

Service管理的所有Pod都必须存在istio-proxy容器，否则报此异常。

#### 修复指导

**步骤1** 登录ASM控制台，选择服务所在网格。单击左侧导航中的“网格配置”，选择“sidecar管理”页签，检查服务所在命名空间是否已注入sidecar。

- 否，执行[步骤2](#)。
- 是，执行[步骤3](#)。

**步骤2** 注入sidecar。

可参考[sidecar注入](#)，为命名空间下所有工作负载关联的Pod注入sidecar。或者只为某个工作负载注入sidecar，方法如下：

1. 为工作负载所在命名空间打上istio-injection=enabled标签。

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

2. 在CCE控制台为工作负载添加annotations字段。

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

```
19 spec:
20   replicas: 1
21   selector:
22     matchLabels:
23       app: httpbin
24       version: v1
25   template:
26     metadata:
27       creationTimestamp: null
28     labels:
29       app: httpbin
30       version: v1
31     annotations:
32       sidecar.istio.io/inject: 'true'
```

您可以单击[Installing the Sidecar](#)了解更多sidecar注入的知识。

**步骤3** 如果网格已经开启了命名空间注入，但是Pod未注入sidecar，需要在CCE控制台手动重启Pod。方法如下：

登录CCE控制台，在工作负载所在行，单击操作列的“更多 > 重新部署”。

**步骤4** 检查工作负载是否配置了主机网络模式。方法如下：

登录CCE控制台，在工作负载所在行，单击操作列的“更多 > 编辑YAML”，查看是否配置了spec.template.spec.hostNetwork: true。如果是，请确认是否可将该字段删除或者配置为false，否则不允许注入sidecar。

```
125 spec:
126   replicas: 1
127   selector:
128     matchLabels:
129       app: nginx
130       version: v1
131   template:
132     metadata:
133       creationTimestamp: null
134     labels:
135       app: nginx
136       version: v1
137     spec:
138       hostNetwork: true
139     containers:
140     - name: container-1
141       image: nginx:alpine
```

**步骤5** 检查网格实例数量是否已经超出限额。

如果实例总数已经超出 200，那么超出部分的实例将无法注入sidecar。

----结束

### 3.3.3 自动修复项

### 3.3.3.1 Service 的端口名称是否符合 istio 规范

#### 问题描述

Service端口名称必须包含指定的协议和前缀，按以下格式命名：

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

其中，<protocol>可以是http、tcp、grpc等，Istio根据在端口上定义的协议来提供对应的路由能力。例如“name: http-service0”和“name: tcp”是合法的端口名；而“name: httpforecast”是非法的端口名。

如果未按照以上格式命名，则报此异常。

#### 修复指导

**步骤1** 登录CCE控制台。

**步骤2** 在左侧导航栏选择“资源管理 > 网络管理”，在“Service”页签中按照集群名称和命名空间搜索服务，单击对应服务后的“编辑YAML”，查看Service协议，根据支持协议，修改协议，在服务名称前加协议类型，如下图。

```
15 spec:
16   ports:
17     - name: http-ratings
18       protocol: TCP
19       port: 9080
20       targetPort: 9080
```

**步骤3** 单击“确定”。

----结束

### 3.3.3.2 Service 的选择器中是否配置了 version 标签

#### 问题描述

Service的选择器（spec.selector）中不能包含version标签。如果包含，则报此异常。

#### 修复指导

**步骤1** 登录CCE控制台。

**步骤2** 在左侧导航栏选择“资源管理 > 网络管理”，在“Service”页签中按照集群名称和命名空间搜索服务，单击对应服务后的“编辑YAML”，查看Service的选择器（spec.selector），删除已配置的version标签。

```
36 spec:
37   ports:
38     - name: http-service0
39       protocol: TCP
40       port: 8000
41       targetPort: 80
42   selector:
43     app: nginx
44     version: v1
```

----结束

### 3.3.3.3 服务是否配置了默认版本的服务路由，路由配置是否正确

#### 问题描述

Istio在VirtualService和DestinationRule中定义了服务的流量路由规则，所以需要为每个服务配置VirtualService和DestinationRule，需要满足以下的规则：

- VirtualService中必须配置了Service的所有端口。
- VirtualService中的协议类型必须和Service中端口协议类型一致。
- VirtualService和DestinationRule中必须配置了默认的服务版本。

#### 📖 说明

如果检查结果发生改变，可能Service的端口号或端口名称被修改。

#### 修复指导

**步骤1** 登录ASM控制台，选择服务所在网格，单击左侧导航中的“网格配置”，选择“istio资源管理”页签，在搜索框中选择“istio资源：virtualservices”及服务所属命名空间。

**步骤2** 确保VirtualService中必须配置了Service的所有端口。

```
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
```

**步骤3** 确保VirtualService中的协议类型必须和Service中端口协议类型一致。

图 3-4 VirtualService 的协议类型

```
34 spec:
35   hosts:
36     - ratings
37     http:
38     - route:
39       - destination:
40         host: ratings
41         port:
42           number: 9080
43         subset: v1
```

图 3-5 Service 的端口协议类型

```
13 spec:
14   ports:
15     - name: httpratings
16       protocol: TCP
17       port: 9080
18       targetPort: 9080
19   selector:
20     app: ratings
```

----结束

## 3.4 网关管理

### 3.4.1 添加网关

服务网关在微服务实践中可以做到统一接入、流量管控、安全防护、业务隔离等功能。

#### 前提条件

服务网关使用弹性负载均衡服务（ELB）的负载均衡器提供网络访问，因此在添加网关前，请提前创建负载均衡。

创建负载均衡时，需要确保所属VPC与集群的VPC一致。

#### 操作步骤

**步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。

**步骤2** 在左侧导航栏选择“网关管理”，单击“添加网关”。

**步骤3** 配置网关参数。

- **网关名称**

请输入网关的名称。由小写字母、数字和中划线（-）组成，且必须以小写字母开头，小写字母或数字结尾，长度范围为4~59个字符。

- **集群选择**

选择网关所属的集群。

- **负载均衡配置**

- **监听器配置**

服务网关为负载均衡器配置监听器，监听器对负载均衡器上的请求进行监听，并分发流量。

- 对外协议

请根据业务的协议类型选择。支持HTTP、GRPC、TCP、TLS及HTTPS五种协议类型的选择。

- 对外端口

开放在负载均衡服务地址的端口，可任意指定。

- TLS终止

对外协议为HTTPS时，TLS终止为开启状态，且不可关闭。

对外协议为TLS时，可选择开启/关闭TLS终止。开启TLS终止时需要绑定证书，以支持TLS数据传输加密认证；关闭TLS终止时，网关将直接转发加密的TLS数据。

- 密钥证书


- 配置TLS协议并开启TLS时，需要绑定证书，以支持TLS数据传输加密认证。

- 配置HTTPS协议时，需要绑定密钥证书。

- TLS最低版本/TLS最高版本

配置TLS协议并开启TLS终止，或者配置HTTPS协议时，提供TLS最低版本/TLS最高版本的选择。

#### 步骤4 （可选）配置路由参数。

请求的访问地址与转发规则匹配（转发规则由域名+URL组成，域名置空时，默认为ELB IP地址）时，此请求将被转发到对应的目标服务处理。单击图标，弹出“添加路由”对话框。

- **域名**

请填写组件对外发布域名。不填时访问地址默认为负载均衡实例IP地址。如果您开启了TLS终止，则必须填写证书内认证域名，以完成SNI域名校验。

- **URL匹配规则**

- 前缀匹配：例如映射URL为/healthz，只要符合此前缀的URL均可访问。例如/healthz/v1、/healthz/v2。

- 完全匹配：只有完全匹配上才能生效。例如映射URL为/healthz，则必须为此URL才能访问。

- **URL**

服务支持的映射URL，例如/example。

- **命名空间**

服务网关所在的命名空间。

- **目标服务**

添加网关的服务，直接在下拉框中选择。目标服务会根据对应的网关协议进行过滤，过滤规则请参见[添加路由时，为什么选不到对应的服务？](#)。

配置诊断失败的服务无法选择，需要先根据[手动修复项](#)或[自动修复项](#)进行修复。

- **访问端口**

仅显示匹配对外协议的端口。

- **重写**

（对外协议为HTTP时可配置）

重写HTTP URI和Host/Authority头，于转发前执行。默认关闭。开启后，需要配置如下参数：

- URI：使用此值重写URI的路径（或前缀），如果原始URI是基于前缀匹配，那么将替换相应匹配的前缀。
- Host/Authority头：使用此值重写HTTP的Host/Authority头。

**步骤5** 配置完成后，单击“确定”。

网关添加完成后，可前往“服务管理”页面，获取服务外网访问地址。

**图 3-6** 服务外网访问地址

服务名称	配置诊断	访问地址
productpage	正常	<ul style="list-style-type: none"><li>外 http:// :3000/productpage HTTP</li><li>外 http:// :3000/ HTTP</li><li>内 http://productpage.default.svc:9080/ HTTP</li></ul>

----结束

## 3.4.2 添加路由

### 操作场景

您可以给已创建好的网关添加多个路由，配置多个转发策略。

### 操作步骤

**步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。

**步骤2** 在左侧导航栏选择“网关管理”，在需要添加路由的网关所在行，单击操作列的“添加路由”，配置如下参数。

- **域名**

请填写组件对外发布域名。不填时访问地址默认为负载均衡实例IP地址。如果您开启了TLS终止，则必须填写证书内认证域名，以完成SNI域名校验。

- **URL匹配规则**

- 前缀匹配：例如映射URL为/healthz，只要符合此前缀的URL均可访问。例如/healthz/v1、/healthz/v2。
- 完全匹配：只有完全匹配上才能生效。例如映射URL为/healthz，则必须为此URL才能访问。

- **URL**

服务支持的映射URL，例如/example。

**说明**

同一网关下的URL配置不能相同。

- **命名空间**

服务网关所在的命名空间。



- **目标服务**

添加网关的服务，直接在下拉框中选择。目标服务会根据对应的网关协议进行过滤，过滤规则请参见[添加路由时，为什么选不到对应的服务？](#)。

配置诊断失败的服务无法选择，需要先根据[手动修复项](#)或[自动修复项](#)进行修复。

- **访问端口**

仅显示匹配对外协议的端口。

- **重写**

（对外协议为HTTP时可配置）

重写HTTP的URI和Host/Authority头，于转发前执行。默认关闭。开启后，需要配置如下参数：

- URI：使用此值重写URI的路径（或前缀），如果原始URI是基于前缀匹配，那么将替换相应匹配的前缀。
- Host/Authority头：使用此值重写HTTP的Host/Authority头。

**步骤3** 配置完成后，单击“确定”。

----结束

## 3.5 灰度发布

### 3.5.1 灰度发布概述

应用程序升级面临最大挑战是新旧业务切换，将软件从测试的最后阶段带到生产环境，同时要保证系统不间断提供服务。如果直接将某版本上线发布给全部用户，一旦遇到线上事故（或BUG），对用户的影响极大，解决问题周期较长，甚至有时不得不回滚到前一版本，严重影响了用户体验。

长期以来，业务升级逐渐形成了几个发布策略：金丝雀发布、蓝绿发布、A/B测试、滚动升级以及分批暂停发布，尽可能避免因发布导致的流量丢失或服务不可用问题。ASM当前支持金丝雀发布和蓝绿发布两种发布方式。

#### 金丝雀发布

又称灰度发布，是版本升级平滑过渡的一种方式，当版本升级时，使部分用户使用新版本，其他用户继续使用老版本，待新版本稳定后，逐步扩大范围把所有用户流量都迁移到新版本上面来。这样可以最大限度地控制新版本发布带来的业务风险，降低故障带来的影响面，同时支持快速回滚。

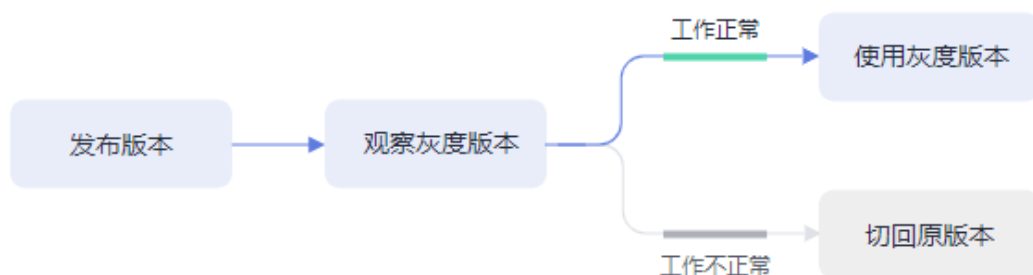
图 3-7 金丝雀发布流程



## 蓝绿发布

蓝绿发布提供了一种零宕机的部署方式，是一种以可预测的方式发布应用的技术，目的是减少发布过程中服务停止的时间。在保留老版本的同时部署新版本，将两个版本同时在线，新版本和老版本相互热备，通过切换路由权重的方式（非0即100）实现应用的不同版本上线或者下线，如果有问题可以快速地回滚到老版本。

图 3-8 蓝绿发布流程




### 3.5.2 创建灰度任务

#### 基本概念

- 灰度版本  
一个服务仅支持发布一个灰度版本，可以对灰度版本配置相应的灰度策略。
- 灰度策略  
当您需要生产环境发布一个新的待上线版本时，您可以选择添加一个灰度版本，并配置相应的灰度策略，将原有的生产环境的默认版本的流量引流一部分至待上线版本。经过评估稳定后，可以将此灰度版本接管所有流量，下线原来的版本，从而接管原有的生产环境的版本上的流量。

#### 创建灰度发布

**步骤1** 登录应用服务网格控制台，使用以下任意一种方式进入创建灰度任务页面。

- （快捷方式）在网格右上方，单击  图标。
- （快捷方式）在网格中心位置，单击“创建灰度任务”。
- 在网格详情页创建。
  - a. 单击网格名称，进入网格详情页，单击左侧导航栏的“灰度发布”。
  - b. 如果当前不存在发布中的灰度任务，请在金丝雀发布或蓝绿发布中单击“立即发布”；如果当前存在发布中的灰度任务，请单击右上角“灰度发布”。

**步骤2** 配置灰度发布基本信息。

- 灰度类型  
选择创建灰度发布的类型，可根据实际需求选择金丝雀发布和蓝绿发布，两者的区别可参考[灰度发布概述](#)。
- 灰度任务名称  
自定义灰度任务的名称。输入长度范围为4到63个字符，包含小写英文字母、数字和中划线（-），并以小写英文字母开头，小写英文字母或数字结尾。



- **命名空间**  
服务所在的命名空间。
- **灰度发布服务**  
在下拉列表中选择待发布的服务。正在进行灰度任务的服务不可再进行选择，列表中已自动过滤。
- **工作负载**  
选择服务所属的工作负载。
- **版本号**  
当前服务版本号，版本号不支持修改。

图 3-9 灰度发布基本信息

### 灰度发布

#### 基本信息

\* 灰度类型

 <b>金丝雀发布</b> 平滑过渡	 <b>蓝绿发布</b> 无需停机，风险较小
--	---

灰度类型区别，请查看 [类型对比](#)

\* 灰度任务名称

\* 命名空间  C

\* 灰度发布服务  C

正在进行灰度任务的服务不可重复选择创建，因此列表不再展示此类服务

\* 工作负载

\* 版本号 v1

#### 步骤3 部署灰度版本信息。

- **部署集群**  
灰度发布服务所属的集群。
- **版本号**  
输入服务的灰度版本号。
- **实例数量**  
灰度版本的实例数量。灰度版本可以有一个或多个实例，用户可根据实际需求进行修改。每个灰度版本的实例都由相同的容器部署而成。
- **镜像名称**  
默认为该服务的镜像。
- **镜像版本**  
请选择灰度版本的镜像版本。

图 3-10 灰度版本信息

**灰度版本信息**

\* 部署集群

\* 版本号

\* 实例数量

您还可以创建4991个实例，最大可用实例数 = 套餐实例数 - 已用实例数

**实例配置**

reviews

镜像名称  examples-bookinfo-reviews-v1

\* 镜像版本  C

温馨提示：已默认继承已有版本的配置，您可以在部署版本后，在 [CCE控制台](#) 更新配置

**步骤4** 单击“发布”，灰度版本开始创建。

请确保灰度版本的实例状态正常，且启动进度为100%时，再开始下一步进行流量策略的配置。发布之后进入观察灰度状态页面，可查看Pod监控，包括启动日志和性能监控信息。

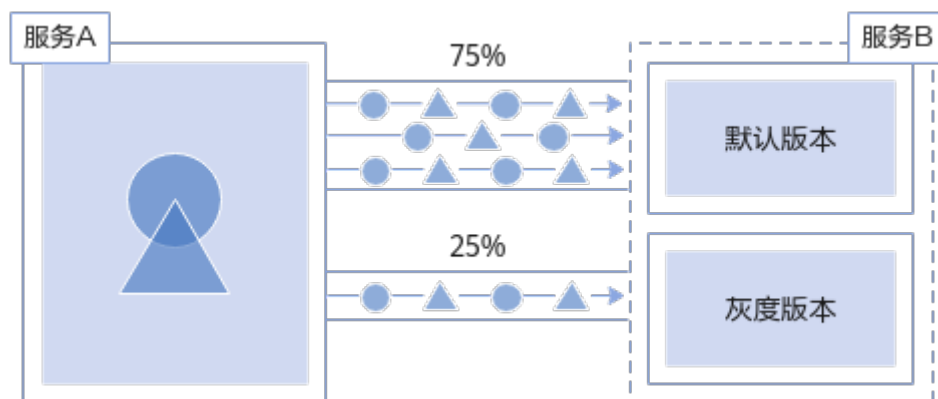
**步骤5**（仅金丝雀发布涉及）单击“配置流量策略”，进行流量策略配置。

策略类型：分为“基于流量比例”和“基于请求内容”两种类型，通过页签选择确定。

- **基于流量比例**

根据流量比例配置规则，从默认版本中切分指定比例的流量到灰度版本。例如75%的流量走默认版本，25%的流量走灰度版本。实际应用时，可根据需求将灰度版本的流量配比逐步增大并进行策略下发，来观测灰度版本的表现情况。

图 3-11 基于流量比例

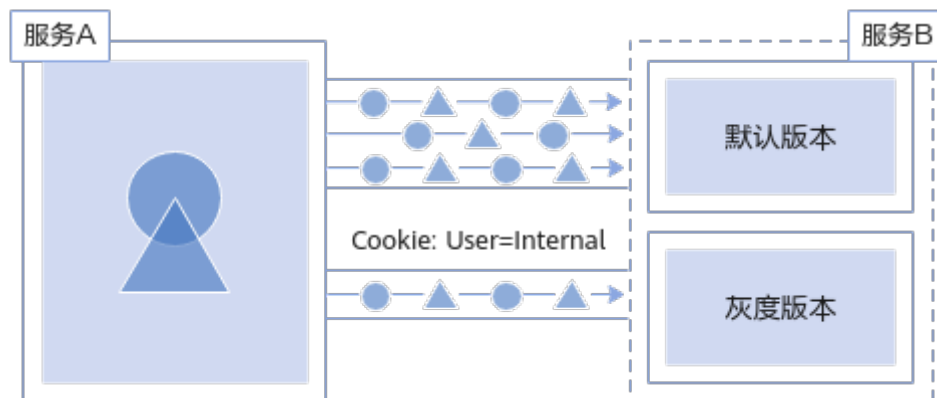


**流量配比：**可以为默认版本与灰度版本设置流量配比，系统将根据输入的流量配比来确定流量在两个版本间分发的比重。

- **基于请求内容**

目前支持基于Cookie内容、自定义Header、Query、操作系统和浏览器的规则约束，只有满足规则约束的流量才可访问到灰度版本。例如，仅Cookie满足“User=Internal”的HTTP请求才能转发到灰度版本，其余请求仍然由默认版本接收。

图 3-12 基于请求内容



- **Cookie内容**
  - 正则匹配：此处需要您使用正则表达式来匹配相应的规则。
- **自定义Header**
  - **完全匹配**：只有完全匹配上才能生效。例如：设置Header的Key=User, Value=Internal, 那么仅当Header中包含User且值为Internal的请求才由灰度版本响应。
  - **正则匹配**：此处需要您使用正则表达式来匹配相应的规则。可以自定义请求头的key和value, value支持完全匹配和正则匹配。
- **Query**
  - **完全匹配**：只有完全匹配上才能生效。例如：设置Query的Key=User, Value=Internal, 那么仅当Query中包含User且值为Internal的请求才由灰度版本响应。
  - **正则匹配**：此处需要您使用正则表达式来匹配相应的规则。可以自定义Query的key和value, value支持完全匹配和正则匹配。
- **允许访问的操作系统**：请选择允许访问的操作系统，包括iOS、Android、Windows、macOS。
- **允许访问的浏览器**：请选择允许访问的浏览器，包括Chrome、IE。
- **流量管理Yaml信息**：根据所设置的参数自动生成规则YAML。

#### 📖 说明

基于请求内容流量策略只对直接访问的入口服务有效。如果希望对所有服务有效，需要业务代码对HTTP请求的Header信息进行传播。

例如：如果您基于reviews服务，配置了基于请求内容的灰度发布策略，通过访问productpage服务的界面，是无法看到效果的。

原因是，您的客户端访问productpage服务携带了HTTP请求的Header信息，而productpage服务请求reviews服务时，将这些Header信息丢失了，从而失去了基于请求内容的灰度发布效果。

**步骤6** 设置完成后，单击“策略下发”。

灰度策略的生效需要几秒时间，您可以查看服务的流量监控，以及对原始版本及灰度版本的健康监控。

----结束

### 3.5.3 灰度任务基本操作

#### 使用说明

对灰度版本的相关操作，其原理是修改Istio的DestinationRule和VirtualService两个资源的配置信息。修改完成后，需要等待10秒左右，新的策略规则才会生效。

#### 修改灰度版本的流量策略

##### 修改基于流量比例的策略

选择基于流量比例的策略时，一般会逐步加大灰度版本的流量配比，这样可以避免直接切换带来的业务风险。修改流量配比的方法如下：

**步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。

**步骤2** 在左侧导航栏选择“灰度发布”，单击金丝雀发布任务的名称。

**步骤3** 在“配置流量策略”页面，重新输入灰度版本的流量配比。

假设将灰度版本流量配比调整至x，那么原版本的流量配比自动调整为100-x。

**步骤4** 单击“策略下发”。

----结束

##### 修改基于请求内容的策略

目前支持基于Cookie内容、自定义Header、Query、操作系统和浏览器的规则约束，只有满足规则约束的流量才可访问到灰度版本。实际应用时，可能会多次修改规则，从而充分验证灰度版本运行效果。

**步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。

**步骤2** 在左侧导航栏选择“灰度发布”，单击金丝雀发布任务的名称。

**步骤3** 在“配置流量策略”页面，重新配置Cookie内容、自定义Header、Query、允许访问的操作系统或允许访问的浏览器。

**步骤4** 单击“策略下发”。

----结束

#### 切换灰度策略类型

您可以在“基于请求内容”和“基于流量比例”的策略之间切换。策略完成切换后，之前配置的规则将全部失效，所有的流量会根据配置的新策略重新分配。

### 须知

只有状态为“运行中”的任务才支持流量策略变更，版本发布完成后（即新版本完全接管旧版本流量，且旧版本已下线），将不支持重新配置流量策略。

**步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。

**步骤2** 在左侧导航栏选择“灰度发布”，单击金丝雀发布任务的名称。

**步骤3** 在“配置流量策略”页面，切换策略类型。

**步骤4** 单击“策略下发”。

----结束

## 全流量接管

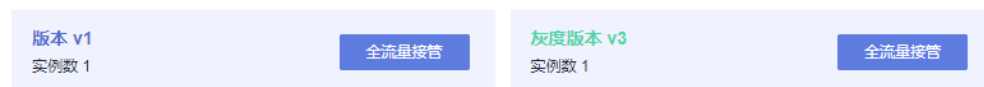
执行原版本或灰度版本后的“全流量接管”，原版本或灰度版本将接管全部流量。

**步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。

**步骤2** 在左侧导航栏选择“灰度发布”，单击灰度发布任务的名称。

**步骤3** 在“监测与处理”页面，单击版本后的“全流量接管”。

图 3-13 全流量接管



**步骤4** 在弹出的“全流量接管”窗口单击“确定”，此版本即可接管全部流量。

----结束

## 结束灰度任务

当新创建的灰度版本接管全部流量后，您可以选择结束灰度任务。结束灰度任务将下线原版本，其中包含的工作负载和Istio相关配置资源会全部删除。

**步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。

**步骤2** 在左侧导航栏选择“灰度发布”，单击灰度发布任务的名称。

**步骤3** 在“监测与处理”页面，单击灰度版本后的“全流量接管”。

**步骤4** 单击右下角“结束灰度任务”。

**步骤5** 在弹出的“结束灰度任务”窗口单击“确定”。

结束的灰度任务可以前往“已结束灰度任务”页签查看，状态显示为“发布成功”。

----结束


## 取消灰度任务

当原版本接管全部流量后，您可以选择取消灰度任务。

**步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。

**步骤2** 在左侧导航栏选择“灰度发布”，单击灰度发布任务的名称。

**步骤3** 在“监测与处理”页面，单击原版本后的“全流量接管”。

**步骤4** 单击右下角“取消灰度任务”。也可以在灰度任务列表，单击任务右上角的图标。

**步骤5** 在弹出的“取消灰度任务”窗口单击“确定”。

取消的灰度任务可以前往“已结束灰度任务”页签查看，状态显示为“发布取消”。

----结束

## 查看已结束灰度任务

取消的灰度任务，以及结束的灰度任务均可在“已结束灰度任务”页签查看。

**步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。

**步骤2** 在左侧导航栏选择“灰度发布”，单击“已结束灰度任务”页签。

您可以查看：发布任务名称、发布结果、服务、发布时间，还可以删除已结束的灰度任务。

----结束

## 3.6 网格配置

### 3.6.1 网格配置概述

网格配置提供了集群管理、sidecar管理、istio资源管理以及升级能力。

Istio控制面组件负责向数据面组件注入sidecar，管理数据面sidecar行为，下发策略配置，搜集监控数据等。其中，sidecar是指运行在业务Pod中，与业务容器协同工作，负责业务Pod的路由转发，监控数据采集，流量规则配置等功能。

“网格配置”中各个页签的功能如下：

- “基本信息”页签：可查看网格名称、网格ID、网格状态、网格类型、当前版本、创建时间以及已启用网格的集群。
- “sidecar管理”页签：支持查看所有注入了sidecar的工作负载信息，还可以进行sidecar注入、配置sidecar资源限制等操作。详情请参见[sidecar管理](#)。
- “istio资源管理”页签：展示所有istio资源（如VirtualService、DestinationRule），还支持以YAML或JSON格式创建新的istio资源，或对现有istio资源进行修改。详情请参见[istio资源管理](#)。
- “升级”页签：提供网格版本升级能力。详情请参见[升级网格](#)。

### 3.6.2 sidecar 管理

sidecar管理中支持查看所有注入了sidecar的工作负载信息，还可以进行sidecar注入、配置sidecar资源限制等操作。



## sidecar 注入

可展示当前已注入sidecar的命名空间及所属集群。如果还未做过注入操作，或者需要为更多命名空间注入sidecar，请参考以下操作：

- 步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。
- 步骤2** 在左侧导航栏选择“网格配置”，单击“sidecar管理”页签。
- 步骤3** 单击“sidecar管理”，选择命名空间，判断是否重启已有服务，单击“确定”。


图 3-14 注入 sidecar

请选择命名空间 为命名空间设置标签 istio-injection=enabled，其中的 Pod 在重启后会自动注入 istio-proxy sidecar



- 选择命名空间：选择一个或多个命名空间，系统将为命名空间设置标签istio-injection=enabled。
- 是否重启已有服务：

：会重启命名空间下已有服务关联的Pod，将会暂时中断业务。只有在重启后，已有服务关联的Pod才会自动注入istio-proxy sidecar。

：已有服务关联的Pod不会自动注入istio-proxy sidecar，需要在CCE控制台，手动重启工作负载才会注入sidecar。是否重启已有服务只会影响已有服务，只要为命名空间设置了istio-injection=enabled标签，后面新建的服务实例都会自动注入sidecar。

### 📖 说明

- 若界面提示“以下集群未开放命名空间注入修改操作”，需要通过kubectl命令行开放，具体操作请参见[如何为集群开放命名空间注入？](#)。
- 为集群的命名空间开启sidecar注入后，该命名空间下所有工作负载关联的Pod将自动注入sidecar。如果某些工作负载不希望注入sidecar，可参考[某些工作负载不注入Sidecar，该如何配置？](#)进行配置。
- 对于1.8.4-r1及之前（所有1.3、1.6）版本的网格，建议确认工作负载是否包含注解：sidecar.istio.io/inject: 'true'。如未包含，请在spec.template.metadata.annotations字段下添加：

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

----结束

## 查看工作负载详情

列表中展示了该网格所管理的集群下所有已创建服务的工作负载，支持查看负载的名称、所属集群、服务，以及负载的sidecar信息，包括sidecar名称、sidecar版本、状态、CPU使用率、内存使用率等。操作方法如下：

**步骤1** 在列表右上角搜索框，选择集群、命名空间，并输入工作负载名称搜索指定工作负载，查看负载的相关信息。

**步骤2** 单击工作负载前的  图标，查看负载的sidecar信息。

如果提示工作负载中无sidecar，是因为该负载所属命名空间还未注入sidecar，参考 [sidecar注入](#) 进行注入。

----结束

## 配置 sidecar 资源限制

支持为sidecar（即istio-proxy容器）配置CPU和内存的资源上下限。同一个节点上部署的工作负载，对于未设置资源上下限的工作负载，如果其异常资源泄露会导致其他工作负载分配不到资源而异常。未设置资源上下限的工作负载，工作负载监控信息也会不准确。

默认的sidecar资源上下限为：

- CPU（Core）：最小 0.1，最大 2
- MEM（MiB）：最小 128，最大 1024

如需更改，请参考以下操作：

**步骤1** 单击工作负载操作列的“sidecar资源限制”，也可以勾选多个工作负载，在列表左上角单击“sidecar资源限制”进行批量配置。

- CPU最小值：也称CPU请求，表示容器使用的最小CPU需求，作为容器调度时资源分配的判断依赖。只有当节点上可分配CPU总量  $\geq$  容器CPU请求数时，才允许将容器调度到该节点。
- CPU最大值：也称CPU限制，表示容器能使用的CPU最大值。
- MEM最小值：也称内存请求，表示容器使用的最小内存需求，作为容器调度时资源分配的判断依赖。只有当节点上可分配内存总量  $\geq$  容器内存请求数时，才允许将容器调度到该节点。
- MEM最大值：也称内存限制，表示容器能使用的内存最大值。当内存使用率超出设置的内存限制值时，该实例可能会被重启进而影响工作负载的正常使用。

----结束

## 3.6.3 istio 资源管理

### 3.6.3.1 YAML 方式配置 Istio 资源

网格中服务关联的Istio资源（如VirtualService、DestinationRule）如需修改，可以在“istio资源管理”中以YAML或JSON格式进行编辑。同时，还支持创建新的istio资源。

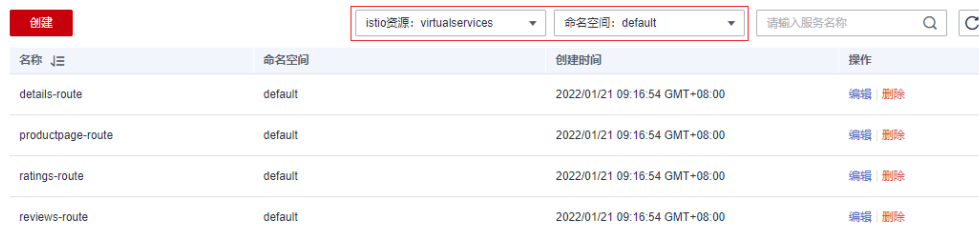
## 编辑已有 istio 资源

**步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。

**步骤2** 在左侧导航栏选择“网格配置”，单击“istio资源管理”页签。

**步骤3** 在搜索框中选择istio资源类型（如“istio资源：virtualservices”），以及资源所属命名空间。

图 3-15 筛选 istio 资源



名称	命名空间	创建时间	操作
details-route	default	2022/01/21 09:16:54 GMT+08:00	编辑 删除
productpage-route	default	2022/01/21 09:16:54 GMT+08:00	编辑 删除
ratings-route	default	2022/01/21 09:16:54 GMT+08:00	编辑 删除
reviews-route	default	2022/01/21 09:16:54 GMT+08:00	编辑 删除

**步骤4** 单击操作列的“编辑”，在右侧页面修改相关配置后单击“确定”保存。  
支持以YAML或JSON格式显示，同时可以将配置文件下载到本地。

----结束

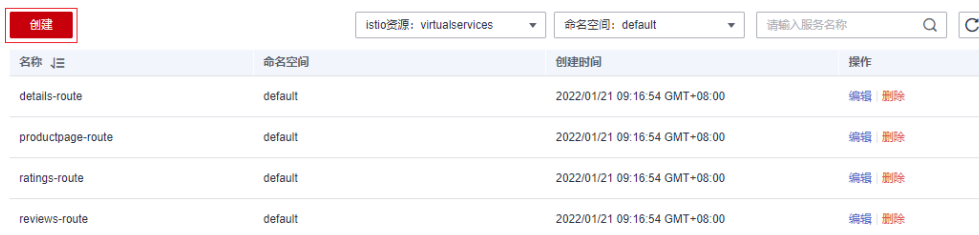
## 创建新的 istio 资源

**步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。

**步骤2** 在左侧导航栏选择“网格配置”，单击“istio资源管理”页签。

**步骤3** 单击列表左上方的“创建”。

图 3-16 创建 istio 资源



名称	命名空间	创建时间	操作
details-route	default	2022/01/21 09:16:54 GMT+08:00	编辑 删除
productpage-route	default	2022/01/21 09:16:54 GMT+08:00	编辑 删除
ratings-route	default	2022/01/21 09:16:54 GMT+08:00	编辑 删除
reviews-route	default	2022/01/21 09:16:54 GMT+08:00	编辑 删除

**步骤4** 在右侧页面直接输入内容，或者单击“导入文件”，将本地编辑好的YAML或JSON文件上传上来。

**步骤5** 确认文件内容无误后，单击“确定”。

----结束

## 3.6.4 升级

### 3.6.4.1 升级网格

#### 操作场景

用户可以将低版本的网格升级到高版本，以获取更优质的体验。基础版网格支持金丝雀升级，企业版本支持补丁原地升级。

#### 升级影响

- 网格升级将自动重新注入新版本数据面代理，过程中会滚动重启服务Pod，可能造成短暂服务实例中断。

- 升级期间请勿进行灰度发布、流量规则配置等操作。

## 升级路径

网格类型	源版本	目标版本	升级方式
基础版	1.8.4-r1	1.8.4-r4	补丁更新
	1.8.4-r2	1.8.4-r4	补丁更新
	1.8.4-r3	1.8.4-r4	补丁更新
	1.8.x	1.15.5-r4	先版本升级到1.13最新版本（金丝雀升级），再版本升级（金丝雀升级）到1.15.5-r4
	1.13.x	1.15.5-r4	版本升级（金丝雀升级）到1.15.5-r4
	1.15.5-rx	1.15.5-r4	补丁更新（金丝雀升级）到1.15.5-r4

### 说明

各大版本特性请参见[1.3版本特性](#)、[1.6版本特性](#)、[1.8版本特性](#)、[7.6.7 1.13版本特性](#)和[7.6.8 1.15版本特性](#)。

## 操作步骤

**步骤1** 登录应用服务网格控制台，确认网格是否需要升级版本。判断方法如下：

- 列表上方是否提示可升级版本的网格。
- 网格名称右侧是否存在“可升级”提示。

若存在可升级版本的网格，单击该网格名称，进入网格详情页面。

**步骤2** 在左侧导航栏选择“网格配置”，单击“升级”页签。

**步骤3** 根据[升级路径](#)选择合适的升级方式完成网格升级。

- 版本升级**
- 补丁更新**  
单击“补丁更新”，在弹出的提示框中单击“确定”。

图 3-17 补丁更新

## 补丁更新

×

升级影响 升级期间请勿进行添加服务，流量规则配置等操作

温馨提示 本次升级将自动为您重新注入新版本数据面代理，过程中会重启服务Pod，可能造成短暂服务实例中断

确定

取消

----结束

### 3.6.4.2 1.3 版本特性

- 基于Mixer的Metric、访问日志和调用链
- 支持SDS，证书轮换无需重启Pod
- 支持Sidecar API
- 控制面性能优化
- 支持v1.13、v1.15、v1.17和v1.19 CCE集群版本

详细内容请参阅：<https://istio.io/latest/news/releases/1.3.x/announcing-1.3/change-notes/>

### 3.6.4.3 1.6 版本特性

- 控制面组件合一，简化控制面安装和运维
- Workload Entry方便对非Kubernetes负载进行定义和管理
- SDS默认启用
- 支持基于数据面，通过Telemetry V2的非Mixer方式的监控数据采集
- 支持多端口服务基于端口粒度的服务治理
- 支持v1.15和v1.17 CCE集群版本

详细内容请参阅：<https://istio.io/latest/news/releases/1.6.x/announcing-1.6/change-notes/>

### 3.6.4.4 1.8 版本特性

- Mixer组件正式下线，访问日志、调用链和监控均基于数据面采集
- EnvoyFilter增强，支持更多灵活的Insert操作
- 启用基于AuthorizationPolicy的新的授权策略
- 支持v1.15、v1.17、v1.19和v1.21 CCE集群版本

详细内容请参阅：<https://istio.io/latest/news/releases/1.8.x/announcing-1.8/change-notes/>

### 3.6.4.5 1.13 版本特性

- 支持Istio 1.13.9版本
- 支持v1.21、v1.23 CCE Turbo集群版本
- 支持v1.21、v1.23 CCE集群版本
- 支持通过ProxyConfig API配置Istio sidecar proxy
- 从Istio 1.10版本开始，Sidecar不再重定向流量到 loopback interface ( lo )，而是将流量重定向到应用的eth0。若您的业务pod监听到了lo，需要改成监听到eth0或者全零监听，否则升级后将导致服务无法访问。[了解更多](#)

详细内容请参阅：<https://istio.io/latest/news/releases/1.13.x/announcing-1.13/change-notes/>

### 3.6.4.6 1.15 版本特性

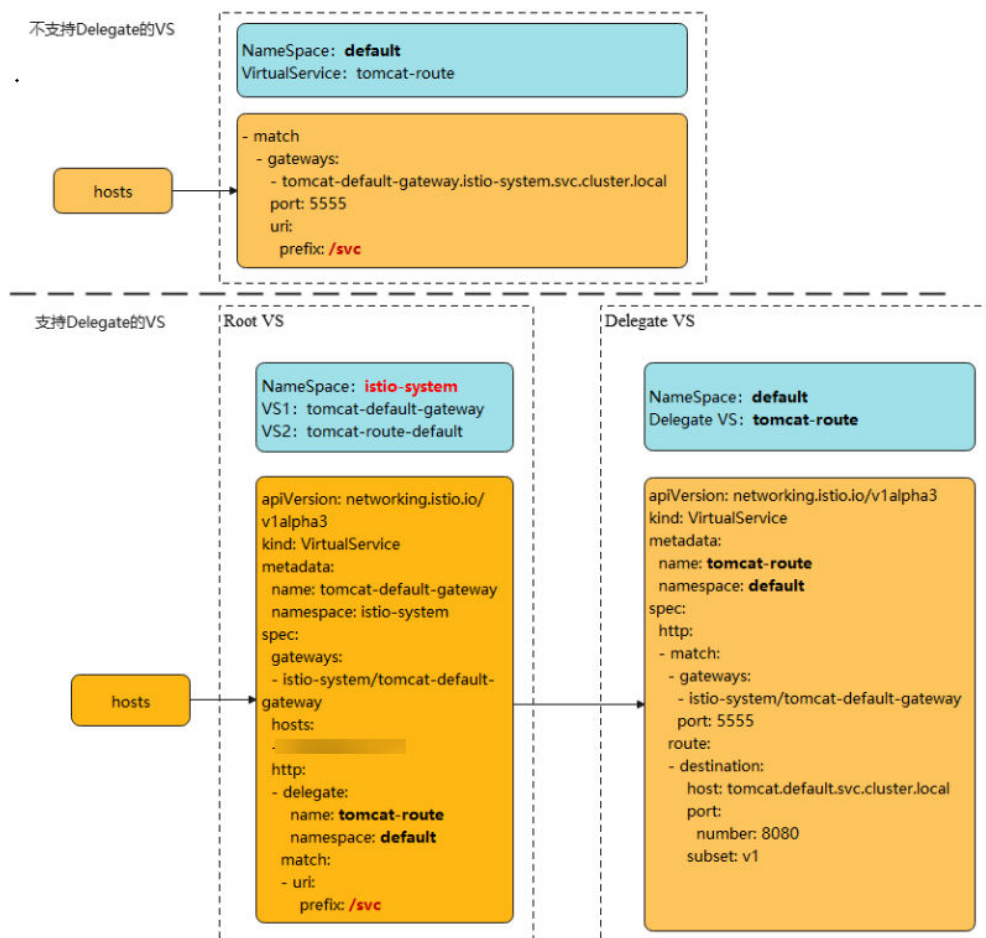
- 支持Istio 1.15.7版本
- 支持v1.21、v1.23、v1.25、v1.27 CCE Turbo集群版本
- 支持v1.21、v1.23、v1.25、v1.27 CCE集群版本
- 修复了 CVE-2023-44487、CVE-2023-39325、CVE-2023-27487 等安全漏洞

详细内容请参阅：<https://istio.io/latest/news/releases/1.15.x/announcing-1.15.7/>

### 3.6.4.7 1.3 升级 1.8 VirtualService 支持 Delegate 切换

#### 操作场景

1.8版本的网格默认支持VirtualService的Delegate功能，同时ASM控制台界面也仅支持delegate格式的VirtualService，升级版本并不会对用户的VirtualService进行修改，在升级后用户将无法在页面对路由进行维护，因此用户需要根据本文指导对应用VirtualService进行修改。



## 说明

对于Delegate的介绍可以参考istio社区的说明：

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

## 约束与限制

- 只有在route和redirect为空时才能设置Delegate。
- ASM只支持一级Delegate，多级Delegate不会生效。
- Delegate VirtualService的HTTPMatchRequest必须是root VirtualService的子集，否则会产生冲突。
- Delegate特性只对HTTP/gRPC协议有效，其他协议无需修改。

## 操作步骤

修改将分两种情况，下面以加入网格的Tomcat服务为例。

一、若升级前服务未添加网关，则升级后无需修改。

二、若升级前服务添加了网关，则升级后进行如下修改：

1. 为网格所在集群配置kubectl命令，参考CCE控制台集群详情页的指导进行配置。
2. 在istio-system命名空间下创建两个VirtualService YAML文件。  
文件名：**tomcat-default-gateway.yaml**

其中，

- tomcat: 为修改的服务名
- tomcat-default-gateway: 为该VirtualService名，格式为{服务名}-default-gateway
- tomcat-route: 为修改VirtualService的名字
- 100.85.219.117: 为ELB的IP

```
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
```

文件名：**tomcat-route-default.yaml**

其中，

- tomcat: 为修改的服务名
- tomcat-route-default: 为该VirtualService名，格式为{服务名}-route-default
- tomcat-route: 为修改VirtualService的名字

```
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: /
```

使用如下命令创建VirtualService。

**kubectl create -f tomcat-route-default.yaml**

**kubectl create -f tomcat-default-gateway.yaml**

3. **kubectl -n{namespace} get vs**获取到服务的VirtualService，执行**kubectl -n{namespace} edit vs tomcat-route**修改如下：

- 删除spec.gateways、spec.hosts和spec.http.match.uri
- tomcat-default-gateway.istio-system.svc.cluster.local替换成istio-system/tomcat-default-gateway
- 修改apiVersion: networking.istio.io/v1alpha3为apiVersion: networking.istio.io/v1beta1



## d. destination.host:格式为{服务名}.{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、spec.hosts需要删除
  http:
    - match:
        - gateways:
            - istio-system/tomcat-default-gateway
          port: 5555
          uri:
            prefix: /test # spec.http.match.uri需要删除
      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. 升级完成后在服务列表页面，单击外部访问URL，检查访问是否正常。
5. 在服务网关页面，检查服务网关路由是否显示正常。

## 3.7 流量治理

### 3.7.1 流量治理概述

流量治理是Istio的核心功能，其目标是提供非侵入的流量治理能力，用户仅仅关注自己的业务逻辑，无需关注服务访问管理。流量治理要解决的问题类似如下：

- 动态修改服务间访问的负载均衡策略，比如配置一致性哈希将流量转发到特定的服务实例上；
- 同一个服务有两个版本在线，将一部分流量切到某个版本上；
- 服务保护，如限制并发连接数、限制请求数、隔离有故障的服务实例等；
- 动态修改服务中的内容，或者模拟一个服务运行故障等。

应用服务网格ASM当前支持重试、超时、连接池、熔断、负载均衡、HTTP头域、故障注入等流量治理能力，可满足大多数业务场景的治理需求。

表 3-1 常用的网格功能和其执行位置

网格功能	治理位置	
	服务发起方	服务提供方
路由管理	Y	N
负载均衡	Y	N
调用链分析	Y	Y
服务认证	Y	Y
可观测性数据	Y	Y
重试	Y	N
重写	Y	N
重定向	Y	N
授权	N	Y
故障注入	Y	N
超时	Y	N
连接池	Y	N
熔断	Y	N
HTTP头域	Y	N

## 约束与限制

配置诊断失败的服务不能进行流量治理，请先参考[手动修复项](#)或[自动修复项](#)修复异常。

### 3.7.2 配置流量策略

- 步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。
- 步骤2** 在左侧导航栏选择“服务管理”，在列表右上方选择服务所在命名空间。
- 步骤3** 选择一个服务，单击操作列的“流量治理”，在右侧页面进行重试、超时、连接池、熔断、负载均衡、HTTP头域、故障注入策略的配置。

#### 重试

服务访问失败自动重试，提高总体访问成功率和质量。

选择“重试”页签，单击“立即配置”，在弹出对话框中，根据实际需求配置如下参数：

表 3-2 重试参数说明

参数名称	参数说明	取值范围
重试次数	单个请求允许的重试次数，间隔默认为25ms，实际的重试次数也取决于配置的超时时间和重试超时时间	1-2147483647
重试超时时间 (s)	单个请求的超时时间，包括初次请求和重试请求，默认值与配置的超时时间相同	0.001-2592000
重试条件	配置重试的条件，详情请参照 <a href="#">retry policies</a> 和 <a href="#">gRPC retry policies</a> 。	-

## 超时

服务访问超时自动处理，快速失败，避免资源锁定和请求卡顿。

选择“超时”页签，单击“立即配置”，在弹出对话框中，根据实际需求配置如下参数：

表 3-3 超时参数说明

参数名称	参数说明	取值范围
超时时间 (s)	HTTP请求超时时间	0.001-2592000

## 连接池

目的是断开超过阈值的连接和请求，保护目标服务。连接池设置应用于上游服务的每个实例，可以应用在TCP级别和HTTP级别。更多信息请参照[Circuit breaker](#)。

选择“连接池”页签，单击“立即配置”，在弹出对话框中，根据实际需求配置如下参数：

表 3-4 TCP 设置参数说明

参数名称	参数说明	取值范围
最大连接数	到目标服务的HTTP/TCP连接的最大数量，默认 $2^{32}-1$	1-2147483647
最大无响应次数	在确定连接已失效前，要发送的保活探测的最大数量。默认使用操作系统级别配置（除非被覆盖，Linux默认为9）	1-2147483647
健康检查间隔 (s)	保活探测之间的持续时间。默认使用操作系统级别配置（除非被覆盖，Linux默认为75秒）	0.001-2592000

参数名称	参数说明	取值范围
连接超时时间 (s)	TCP连接超时时间，默认值为10秒	0.001-2592000
最短空闲时间 (s)	在开始发送保活探测前，连接需要处于空闲状态的持续时间。默认是使用操作系统级别配置（除非被覆盖，Linux默认为7200秒，即2小时）	0.001-2592000

表 3-5 HTTP 设置参数说明

参数名称	参数说明	取值范围
最大请求数	转发到单个服务实例的最大请求数，默认 $2^{32}-1$	1-2147483647
最大等待请求数	转发到目标服务的最大待处理的HTTP请求数，默认 $2^{32}-1$	1-2147483647
连接最大空闲时间 (s)	上游服务连接的空闲超时。如果一定时间内没有活跃的请求，达到空闲时间后，连接将关闭，如果未设置，默认值为1小时	0.001-2592000
最大重试次数	一定时间内，服务所有实例的最大重试次数，默认为 $2^{32}-1$	1-2147483647
每连接最大请求数	每个连接到后端的最大请求数。将此参数设置为1将禁用保活，默认0，表示“无限”，最多 $2^{29}$	1-536870912

## 熔断

自动隔离不健康的实例，提高服务整体访问成功率。

熔断器跟踪服务实例的访问状态，通过跟踪一段时间内服务实例的访问请求判定其健康状况，将不健康的实例从连接池中隔离，从而提高服务总体的访问成功率，适用于HTTP和TCP服务。对于HTTP服务，连续返回5xx错误的实例认定为不健康。对于TCP服务，连接超时或者连接失败的实例认定为不健康。详情请参照[Outlier detection](#)。

选择“熔断”页签，单击“立即配置”，在弹出对话框中，根据实际需求配置如下参数：

表 3-6 熔断参数说明

参数名称	参数说明	取值范围
连接错误数	检查周期内连续错误的次数，连续错误次数超过该阈值将会被隔离，此功能默认为5	1-2147483647

参数名称	参数说明	取值范围
基础隔离时间 (s)	满足熔断条件的实例被隔离的基础隔离时间。服务实例实际隔离时间等于基础隔离时间 x 隔离次数，必须 $\geq 0.001s$ ，默认值为30秒	0.001-2592000
检查周期 (s)	实例异常检查的间隔，根据在该时段内异常次数是否超过阈值决定是否触发隔离，必须 $\geq 0.001s$ ，默认值为10秒	0.001-2592000
最大隔离实例比例 (%)	被隔离的服务实例最大百分比，默认为10%	1-100

## 负载均衡

为目标服务配置满足业务要求的负载均衡策略，控制选择后端服务实例。

选择“负载均衡”页签，单击“立即配置”，在弹出对话框中，根据实际需求选择负载均衡算法。

- 轮询调度：网格使用的默认负载均衡算法，实例池中的每个实例轮流获得请求。
- 最少连接：随机选取两个健康的实例，再从所选取的两个实例中选择一个连接数较少的实例。
- 随机调度：从所有健康的实例中，随机选取一个。
- 一致性哈希：包含四种类型，如表3-7所述。

表 3-7 一致性哈希算法类型

类型	说明
基于HTTP的Header	需要输入HTTP头域的键名，根据HTTP请求中的请求头名称来计算哈希值，相同的值会转发到同一实例中。
基于Cookie	需要输入Cookie的键名，根据HTTP请求中的cookie的键名来计算哈希值，相同的值会转发到同一实例中。
基于源IP	根据HTTP请求中的源IP地址来计算哈希值，相同的值会转发到同一实例中。
基于query参数	需要输入query参数的键名，根据HTTP请求中的query参数名称来计算哈希值，相同的值会转发到同一实例中。

## HTTP头域

灵活增加、修改和删除指定HTTP头域，非侵入方式管理请求内容。

选择“HTTP头域”页签，单击“立即配置”，在弹出对话框中，根据实际需求配置如下参数：

表 3-8 将 HTTP 请求转发到目标服务之前，对 Headers 的操作

参数名称	参数说明
添加请求的Headers	添加新的请求Headers参数，需要设置key和value。还可以单击⊕图标，添加更多同类型参数。
修改请求的Headers	修改已有的请求Headers参数，需要设置key和value。还可以单击⊕图标，添加更多同类型参数。
移除请求的Headers	删除已有的请求Headers参数，需要设置key。还可以单击⊕图标，添加更多同类型参数。

表 3-9 将 HTTP 响应回复给客户端前，对 Headers 的操作

参数名称	参数说明
添加响应的Headers	添加新的响应Headers参数，需要设置key和value。还可以单击⊕图标，添加更多同类型参数。
修改响应的Headers	修改已有的响应Headers参数，需要设置key和value。还可以单击⊕图标，添加更多同类型参数。
移除响应的Headers	删除已有的响应Headers参数，需要设置key。还可以单击⊕图标，添加更多同类型参数。

## 故障注入

故障注入是一种有效的测试方法，它能够将错误引入系统，以确保系统能够承受错误的并从错误中恢复。

选择“故障注入”页签，单击“立即配置”，在弹出对话框中，根据实际需求配置如下参数：

表 3-10 故障注入参数说明

参数名称	参数说明	取值范围
故障类型	包括延时故障和中断故障。 <ul style="list-style-type: none"><li>• 延时故障：对通往组件的请求有延迟。</li><li>• 中断故障：会中断该组件的服务并返回预设状态码。</li></ul>	延时故障、中断故障
延时（s）	故障类型选择“延时故障”时需要设置。 在转发请求之前添加的固定延迟。	0.001-2592000

参数名称	参数说明	取值范围
HTTP状态码	故障类型选择“中断故障”时需要设置。 终止故障时返回的HTTP状态码，默认返回500。	200-599
故障百分比 (%)	注入延时或中断故障的请求百分比。	1-100

----结束

### 3.7.3 更改流量策略

#### 操作场景

流量策略设置完成后，支持更改流量策略，如将负载均衡的算法由“轮询调度”转为“随机调度”。

#### 操作步骤

- 步骤1** 登录应用服务网格控制台，单击服务网格的名称，进入网格详情页面。
- 步骤2** 在左侧导航栏选择“服务管理”，选择需要更改流量策略的服务，单击操作列的“流量治理”，在右侧页面进行流量策略更改。

----结束

# 4 最佳实践

## 4.1 数据面 sidecar 升级不中断业务

应用服务网格作为集群网络管理的重要工具，为网格内的服务提供流量治理的能力。sidecar作为应用服务网格数据面的重要组件，需要依赖服务业务pod的更新来实现sidecar的升级和重新注入。

本章节主要介绍如何实现数据面sidecar升级过程中，不中断服务的业务流量。

### 配置服务实例数

为保证您的服务在sidecar升级的过程中不中断业务流量，首先确保您的服务实例数大于等于2，升级策略为滚动升级（rollingUpdate）。

相关滚动升级策略如下，供参考：

```
kubectl get deploy nginx -n namespace_name -oyaml | grep strategy -a10
```

```
spec:
  minReadySeconds: 2
  progressDeadlineSeconds: 600
  replicas: 2
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: nginx
      version: v1
  strategy:
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
    type: RollingUpdate
```

#### 配置项说明：

- 服务实例数：deployment.spec.replicas >= 2
- 升级策略：deployment.spec.strategy.type == RollingUpdate
- 滚动升级最小存活实例数：deployment.spec.replicas - deployment.spec.strategy.maxUnavailable > 0



## 添加 readiness 探针

添加readiness探针，可以保证您的新实例pod在真正准备就绪时，才开始接管业务流量。这就避免了在新的实例pod未启动时，接管业务流量造成的访问不通问题。

相关配置如下：

```
kubectl get deploy nginx -n namespace_name -oyaml | grep readinessProbe -a10
```

```
readinessProbe:
  failureThreshold: 3
  httpGet:
    path: /
    port: 80
    scheme: HTTP
  initialDelaySeconds: 1
  periodSeconds: 10
  successThreshold: 1
  timeoutSeconds: 1
```

配置项说明：

readiness探针配置项：deployment.spec.template.spec.containers[i].readinessProbe

其中，包括探针检查初始时间，检查间隔，超时时间等配置。

## 设置服务就绪时间

服务就绪时间，minReadySeconds：用于标识pod的ready时间至少保持多长时间，才会认为服务是运行中。

相关配置如下：

```
kubectl get deploy nginx -n namespace_name -oyaml | grep minReadySeconds -a1
```

```
spec:
  minReadySeconds: 2
  progressDeadlineSeconds: 600
```

配置项说明：

服务就绪时间：deployment.spec.minReadySeconds，可根据您业务的实际情况确定。

## 配置优雅关闭时间

terminationGracePeriodSeconds，优雅关闭时间。在滚动升级过程中，首先会移除旧的服务实例pod的endpoint，并将实例pod的状态置为Terminating，这时K8S会发送SIGTERM信号给pod实例，并等待优雅关闭时间后，将pod强制终止。您可以利用这段时间，处理未完成的请求：

```
kubectl get deploy nginx -n namespace_name -oyaml | grep terminationGracePeriodSeconds -a1
```

```
securityContext: {}
terminationGracePeriodSeconds: 30
tolerations:
```

### 配置项说明：

优雅关闭时间：deployment.spec.template.spec.terminationGracePeriodSeconds，默认值为30s，可根据业务诉求适当调整。

## 配置 preStop

preStop会在实例pod中止前调用，可以用来实现业务pod的优雅关闭。此处需要根据业务诉求来进行相应的配置，以Nginx为例。

```
kubectl get deploy nginx -n namespace_name -oyaml | grep lifec -a10
```

```
lifecycle:
  preStop:
    exec:
      command: ["/bin/sh", "-c", "nginx -s quit; sleep 10"]
```

在“lifecycle”下的“preStop”中，定义了一个命令**nginx -s quit; sleep 10**，这个命令首先会发送一个优雅关闭信号给Nginx进程，然后暂停10秒。这样，在Pod终止之前，Nginx会有足够的时间完成正在处理的请求并优雅地关闭。

需要注意的是，sleep 10中的10秒是一个示例值，您可以根据实际需要和应用程序的性能来调整这个值，关键是为Nginx提供足够的时间来优雅地关闭。

类似地，您可以选择运行自定义命令或自定义脚本，来优雅关闭您的服务进程。

## 4.2 面向 Dubbo 协议的服务治理

### 4.2.1 简介

Dubbo作为一种特有协议，需要有如下支持：

- 网格服务数据面Envoy支持对Dubbo协议的解析和流量管理。
- 网格控制面支持对Dubbo治理规则的配置。支持灰度发布、负载均衡、访问授权等服务管理。

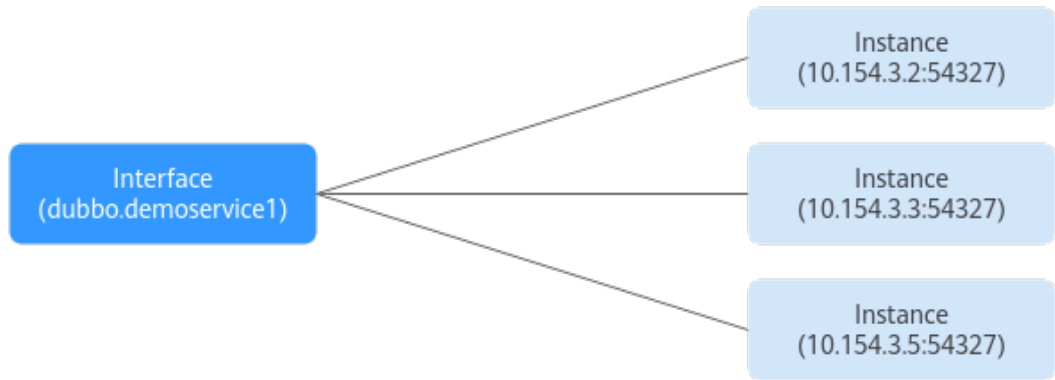
另外，Dubbo的服务发现模型和Kubernetes、Spring Cloud等服务发现模型不一致，需要额外的处理。

### 4.2.2 服务发现模型

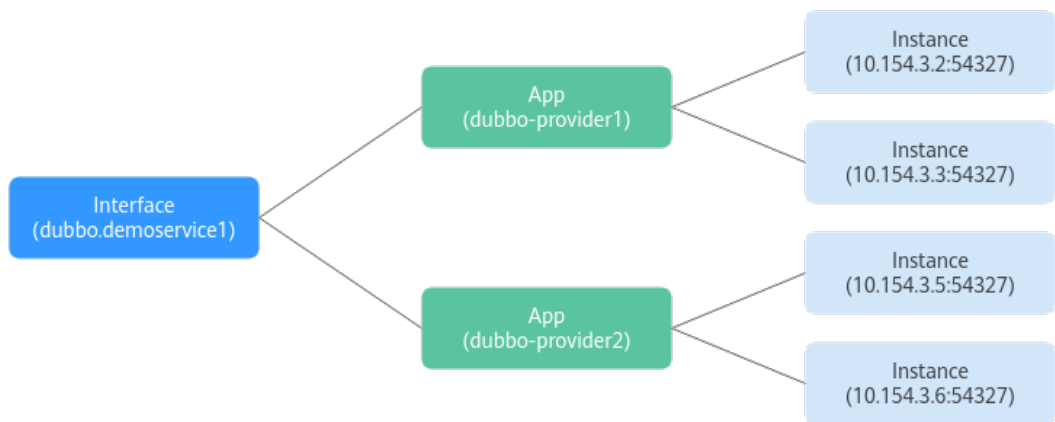
Dubbo现有模型存在的问题（来自Dubbo社区2.7.4总结）：

- 在微服务架构中，注册中心管理的对象是应用（服务），而非对外的服务接口，不过目前Dubbo的注册中心管理的对象是Dubbo服务接口，与Spring Cloud或Cloud Native注册方式背道而驰。
- 一个Dubbo应用（服务）允许注册N个Dubbo服务接口，当N越大时，注册中心的负载越重。

Dubbo现有服务模型：根据Dubbo接口查找服务实例。



Dubbo Cloud Native服务发现模型，将原来Interface一级的服务发现拆分成两级，基于App找实例地址。

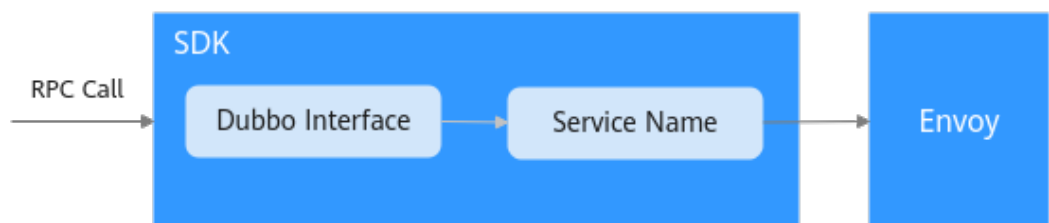


## 4.2.3 SDK 适配方式

### 4.2.3.1 PASSTHROUGH 方案

#### 方案介绍

SDK中客户端使用Interface调用目标服务时，修改原有服务发现逻辑。将原有通过Interface查找服务实例，修改为通过接口查找服务名，直接对服务名发起访问。



#### 详细说明

对于Dubbo协议的不同版本会有不同：

- 2.7.4+版本：2.7.4以上的Cloud Native版本中重构过与Kubernetes一致的服务发现模型，直接可以从Interface关联到服务信息。

- 2.7.3及之前版本：Dubbo社区版本未提供Interface到Service的二级关系，需要SDK根据自己的实际使用方式来维护Interface到服务的映射关系。如可以使用在服务注册时通过扩展信息的方式提供服务名等信息。

客户实际使用中可以根据自己的SDK使用情况选择灵活的处理方式。对于老版本的SDK可以基于现有的服务注册和服务发现流程，做如下处理：

1. 在注册信息中扩展Service的定义。在服务部署时会通过环境变量将服务的元数据信息注入到SDK中，包括appname、namespace，分别表示部署的服务名、部署的命名空间。
2. 在服务启动时向注册中心注册Dubbo接口和Kubernetes服务名和命名空间的关系。
3. 在客户端发起访问时，根据原有服务发现的流程根据Interface查询到服务的元数据，并用对应的服务信息组装RPC请求。建议在Dubbo请求中使用Attachment扩展字段存储appname、namespace信息。

### 4.2.3.2 静态目标服务

#### 方案介绍

通过 `dubbo:reference` 在Dubbo服务的服务消费者中对引用的服务提供者进行配置。使用选项 `url` 定义点对点直连服务提供者地址，绕过注册中心，直接调用目标服务。

#### 详细说明

如果原Dubbo服务中使用的是xml配置文件，则只需要修改配置文件即可。

```
<?xml version="1.0" encoding="UTF-8"?>
<beans>
  <!-- 提供哪些接口给消费者调用 -->
  <dubbo:reference id="helloService" interface="com.dubbo.service.HelloService" url = "dubbo://
helloService:20880" />
</beans>
```

如果服务中使用注解的方式来定义引用的目标服务，则需要修改代码中对目标服务的注解。

```
@Reference(url = "dubbo://helloService:20880")
HelloService helloService;
```

# 5 常见问题

## 5.1 网格集群

### 5.1.1 启用服务网格后，状态一直为安装中

#### 问题描述

为CCE集群启用服务网格（即创建网格）后，网格状态一直显示为“安装中”，鼠标放上去提示“正在启用istio服务网格：开通用户安全组规则成功”。

#### 问题定位

登录CCE控制台，在“资源管理 > 命名空间”中查看对应集群的istio-system命名空间是否存在。

#### 原因分析

存在istio-system命名空间残留。

#### 解决方法

删除已有的istio-system命名空间后即可继续安装。

### 5.1.2 卸载服务网格后，状态一直为未就绪

#### 问题描述

在ASM控制台卸载服务网格后，网格状态一直显示为“未就绪”。

#### 问题定位

**步骤1** 登录CCE控制台，在左侧导航栏选择“模板市场 > 示例模板”。

**步骤2** 单击“模板实例”页签，在搜索框中选择对应集群，查看模板实例和卸载失败最新事件。

可以看到istio-master模板实例的执行状态为“卸载失败”，并且最新事件提示如下信息：

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

----结束

## 原因分析

helm对中断状态支持不好，客户异常操作会导致istio的helm模板卡在中间状态，使卸载过程中留下残留资源，从而导致卸载失败。

## 解决方法

**步骤1** 通过kubectl连接到CCE集群。

**步骤2** 执行以下命令，清理istio相关资源。

```
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}'`
```

**步骤3** 登录ASM控制台，重新执行卸载操作。

----结束

## 5.2 网络管理

### 5.2.1 为什么我的集群不能启用网格？

#### 问题描述

集群不能启用网格。

#### 原因分析

暂不支持v1.15以下版本集群启用网格。

#### 解决方法

**步骤1** 检查您的集群版本，目前仅对v1.15、v1.17或v1.19版本的集群生效。

**步骤2** 检查您的浏览器，请尽量使用Chrome浏览器访问服务，火狐等浏览器可能因为适配的问题，导致启用网格按钮灰化。

----结束

## 5.2.2 Istio 卸载之后，为什么独享节点还在？

### 问题描述

Istio卸载后独享节点还在。

### 原因分析

Istio仅会卸载Istio相关控制面组件，不会主动卸载您的节点资源。

### 解决方法

卸载后的节点，您可以作为普通负载节点使用。如不再需要，请登录CCE控制台，在“资源管理 > 节点管理”中，删除该节点。

## 5.2.3 如何为集群开放命名空间注入？

为集群的命名空间注入sidecar时，若集群未开放命名空间注入，请参考如下指导修改集群配置：

**步骤1** 通过kubectl连接集群。

**步骤2** 执行**kubectl get iop -nistio-system**，查询iop资源。

- 若回显如下，表示存在iop资源，请执行**步骤3**。

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

- 若回显如下，表示不存在iop资源，请执行**步骤4**。

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

**步骤3** 执行**kubectl edit iop -nistio-system data-plane**，修改autoInject配置项。其中，*data-plane*为上一步查询的iop资源名称，请替换为实际值。

```
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
```

**步骤4** 执行**kubectl edit cm -nistio-system istio-sidecar-injector**，修改istio-sidecar-injector配置项。

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

----结束

## 5.2.4 某些工作负载不注入 Sidecar，该如何配置？

为集群的命名空间开启Sidecar注入后，该命名空间下所有工作负载关联的Pod将自动注入Sidecar。不过有些工作负载因为种种原因不能注入Sidecar，可参考如下指导进行配置：

**步骤1** 登录CCE控制台，在左侧导航栏选择“工作负载 > 无状态负载 Deployment”。

**步骤2** 在搜索框中选择对应集群，单击工作负载所在行的“编辑YAML”。

**步骤3** 找到spec.template.metadata.annotations字段，添加sidecar.istio.io/inject: 'false'。

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

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

您可以单击[Automatic Sidecar Injection](#)了解更多Sidecar注入的知识。

---结束

## 5.2.5 Sidecar 未就绪导致 Pod 启动失败

### 问题背景

加入网格的服务有时可能遇到Pod启动失败，且一直重启。排查原因发现业务容器与外部通信时流量会经过istio-proxy容器，但业务容器比istio-proxy容器先启动，在istio-proxy容器没启动成功时，业务容器已经启动，与外部通信将会失败，Pod一直重启。

### 规避方案

在Istio 1.7及以后版本，社区通过给istio-injector注入逻辑增加一个叫HoldApplicationUntilProxyStarts的开关来解决该问题，开关打开后，Proxy将会注入到第一个Container，istio-proxy容器先于业务容器启动。

开关配置分为全局和局部两种，下面介绍两种启用方法。



**须知**

需要注意的是，打开开关后，意味着业务容器需要等Sidecar完全Ready后才能启动，会让Pod启动速度变慢一些。在需要快速扩容应对突发流量场景可能会显得吃力，所以建议您自行评估业务场景，利用局部配置的方法，只给需要的业务打开此开关。

**全局配置**

- a. 执行以下命令，编辑IOP CR资源。

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

在spec.values.global.proxy字段下添加以下配置：

```
holdApplicationUntilProxyStarts: true
```

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

- b. 执行以下命令，确认最新日志无报错。

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

- c. 执行以下命令，确认IOP CR是正常状态。

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

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

- d. 执行以下命令，滚动升级已添加到网格的服务。

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

其中，nginx为示例服务，default为命名空间，请替换为实际取值。

- e. 执行以下命令，确认Pod重启成功。

```
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. 执行以下命令，确认Pod正常添加了postStart lifecycle，并且istio-proxy容器放在了第一个位置。

```
kubectl edit pod nginx-7bc96f87b9-l4dbl
```

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

- **局部配置**

如果使用Istio 1.8及其以上的版本，可以为需要打开此开关的Pod加上proxy.istio.io/config注解，将holdApplicationUntilProxyStarts置为true。以default命名空间下nginx服务为例，用户其他服务操作类似。

```
kubectl edit deploy nginx -n default
```

在spec.template.metadata.annotations字段下添加以下配置：

```
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
```

## 5.3 添加服务

### 5.3.1 添加的对外访问方式不能生效，如何排查？

出现上述问题可能是访问相关的资源配置有缺失或错误，请按照如下方法进行排查：

- 通过弹性负载均衡服务界面查看使用的ELB是否成功监听使用的外部端口和弹性云服务器。
- 登录集群，使用 `kubectl get gateway -n istio-system` 命令查看使用的gateway是否配置好使用的IP/域名和端口。使用 `kubectl get svc -n istio-system` 命令查看使用的ingressgateway是否有对应的IP和端口，且未处于pending状态。
- 核实加入服务网格的内部访问协议和添加网络配置的外部访问协议一致。
- 如果通过浏览器访问出现“ERR\_UNSAFE\_PORT”错误，是因为该端口被浏览器识别为危险端口，此时应更换为其他外部端口。

### 5.3.2 一键创建体验应用为什么启动很慢？

体验应用包含productpage、details、ratings和reviews 4个服务，需要创建所有相关的工作负载和Istio相关的资源（DestinationRule、VirtualService、Gateway）等，因此创建时间较长。

### 5.3.3 一键创建体验应用部署成功以后，为何不能访问页面？

#### 问题描述

一键创建体验应用部署成功后不能访问页面。

#### 原因分析

弹性负载均衡ELB未成功监听端口。

#### 解决方法

请在弹性负载均衡ELB中查看该端口监听器是否创建，后端服务器健康状态是否正常。

### 5.3.4 添加路由时，为什么选不到对应的服务？

添加路由时，目标服务会根据对应的网关协议进行过滤。过滤规则如下：

- HTTP协议的网关可以选择HTTP协议的服务
- TCP协议的网关可以选择TCP协议的服务
- GRPC协议的网关可以选择GRPC协议的服务
- HTTPS协议的网关可以选择HTTP、GRPC协议的服务
- TLS协议的网关如果打开了TLS终止，只能选择TCP协议的服务；关闭了TLS终止，只能选择TLS协议的服务

## 5.4 灰度发布

### 5.4.1 灰度发布部署版本为什么不能更换镜像？

#### 问题描述

灰度发布部署灰度版本时不能更换镜像类型。

#### 原因分析

灰度发布针对服务的同一镜像，只允许选择不同的版本号。

#### 解决方法

将所需镜像打包成同一镜像的不同版本并上传至镜像仓库。

### 5.4.2 基于请求内容发布策略对一些服务为什么没有生效？

#### 问题描述

基于请求内容发布策略没有生效。

## 原因分析

基于请求内容发布策略只对直接访问的入口服务有效。

## 解决方法

如果希望对所有服务有效，需要业务代码对HEAD信息传播。