应用服务网格(ASM)

用户指南

文档版本 07

发布日期 2025-07-07

华为云计算技术有限公司

商标声明


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云计算技术有限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，华为云计算技术有限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。

华为云计算技术有限公司

地址： 贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址： https://www.huaweicloud.com/

目 录

1 用户指南（新版）

1.1 欢迎使用应用服务网格

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。

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，其功能包括负载均衡、熔断、限流等多种治理能力。ASM内置金丝雀、蓝绿等多种灰度发布流程，提供一站式自动化的发布管理。ASM基于无侵入的监控数据采集，提供实时流量拓扑、调用链等服务性能监控和运行诊断，构建全景的服务运行视图。

更多应用服务网格ASM介绍请参见**产品介绍**。

1.2 购买网格

1.2.1 网格类型概述

ASM支持创建两种类型的网格，分别为基础版网格和企业版网格，两者主要区别在于网格控制面相关组件的部署模式。基础版网格的控制面组件安装在用户集群，企业版网格将控制面组件从用户集群中分离，由华为云管理和维护，简化了用户的运维负担和资源消耗，二者在具体功能上的差异请参见**基础版、社区开源版本对比**。推荐您使用企业版网格进行服务管理，本文也将着重介绍企业版网格的部署架构和工作原理。

⚠ 注意

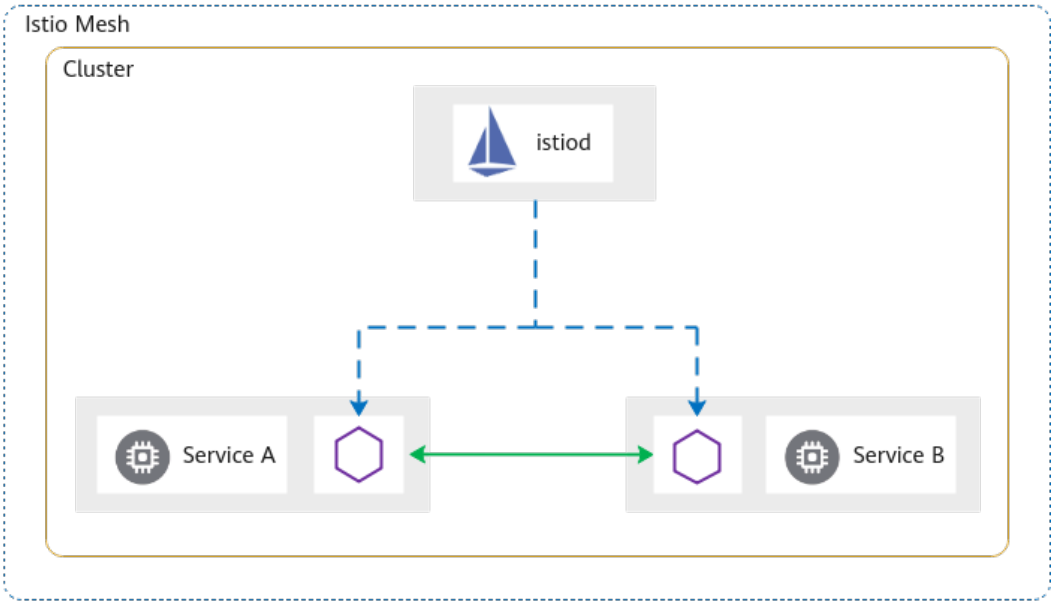
企业版网格已下线，不再支持创建企业版网格，推荐使用基础版网格。存量的企业版网格仍可继续使用。

基础版网格

在用户集群中安装网格的控制面组件，对集群内的服务进行非侵入式的治理、遥测和安全等管理。支持Istio 1.8、1.13、1.15和1.18，只能够对一个集群进行管理，且最大管理实例数为200。

如图1-1所示，istiod是网格的控制面组件，它被部署在用户集群中，管理其他服务的Envoy组件。这是Istio传统的部署形式。

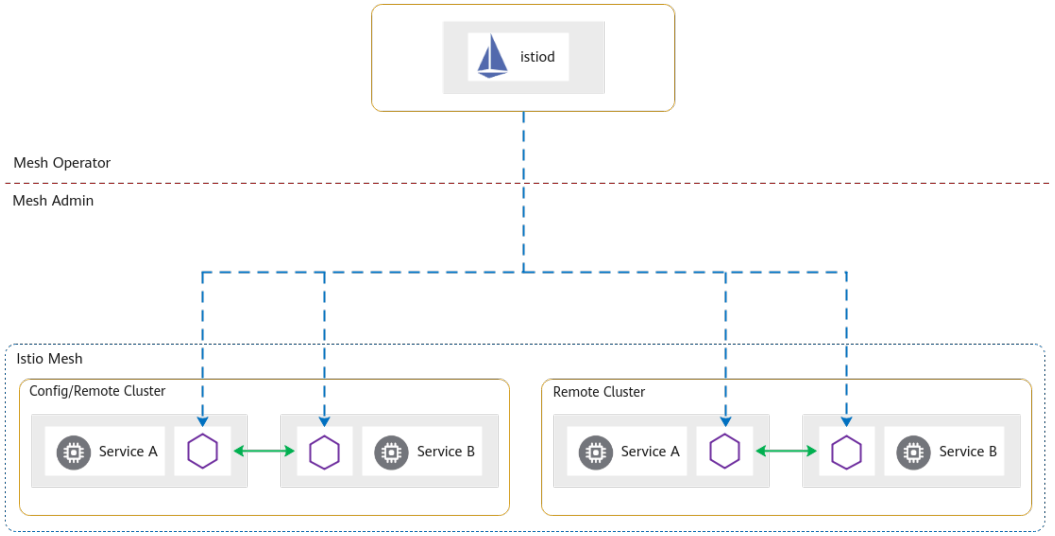
图 1-1 基础版网格



企业版网格

企业版将网格的控制面组件从用户集群中分离，部署到一个独立的控制面集群中，简化了用户运维负担和资源消耗，用户只需要基于网格进行服务管理。企业版还能够对N（N>=1）个集群进行管理，支持服务跨集群通信。

图 1-2 企业版网格



如图1-2所示，istiod是网格的控制面组件，它被部署到一个独立的控制面集群中，能够同时管理多个集群的Envoy组件。

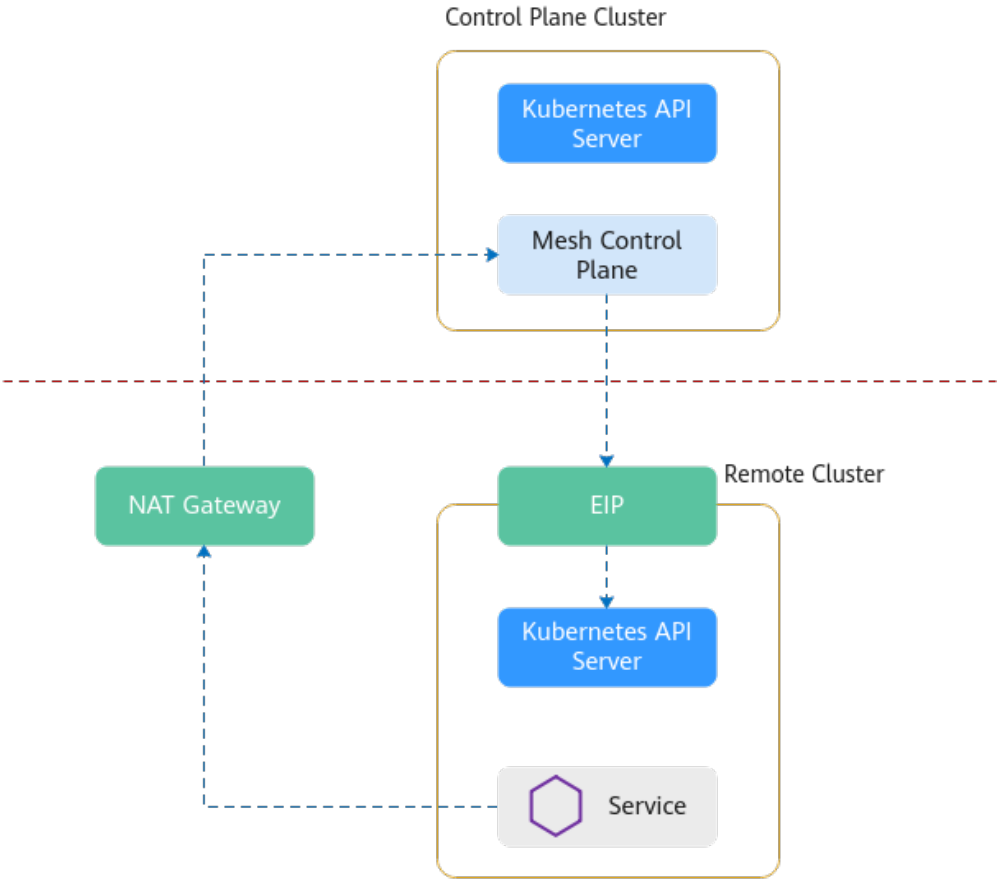
企业版网格工作原理

企业版网格是如何管理用户集群的，这要从用户集群和网格控制面的连接方式说起。企业版网格支持用户集群通过公网连接和私网连接两种方式连接网格控制面。

- **公网连接主要用于跨Region添加集群或对接其他云厂商的集群**

  通过公网连接网格控制面，用户需要为集群绑定弹性公网IP（EIP），因为网格的控制面需要访问用户集群的Kubernetes API Server服务。还需要为集群所在的VPC创建公网NAT网关，因为服务的Envoy组件需要通过NAT网关连接网格的控制面。

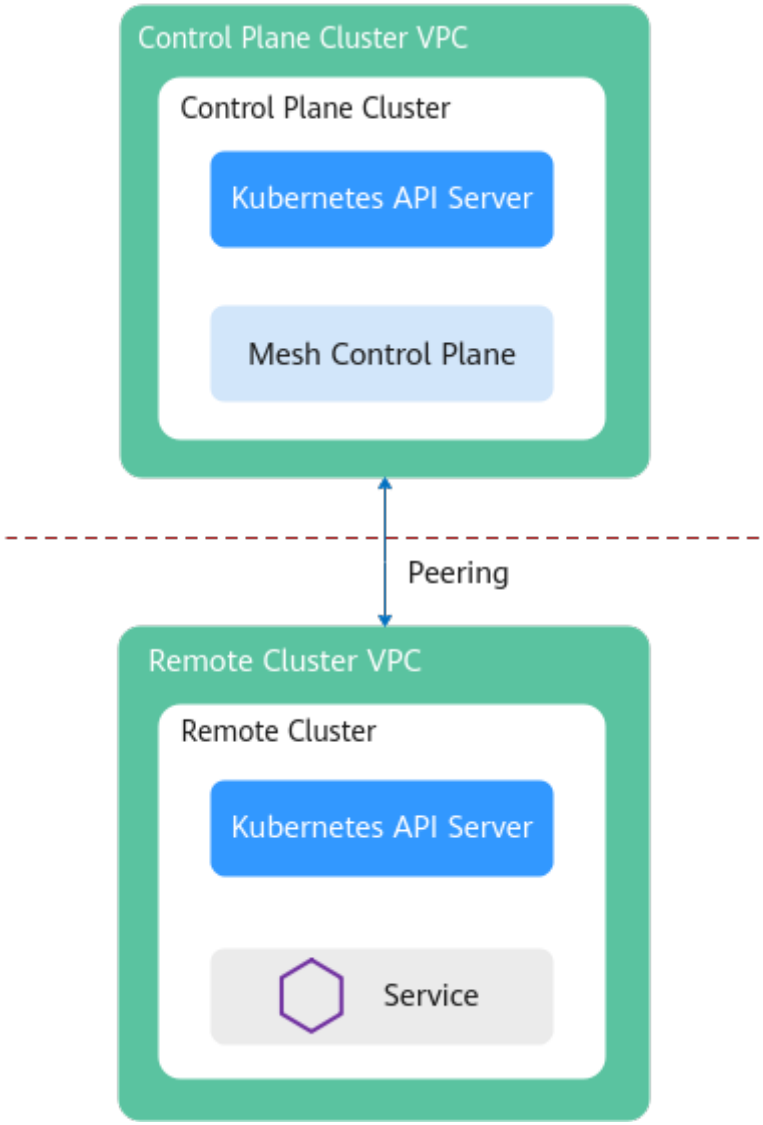
  图 1-3 公网连接网格控制面



- **私网连接用于对接华为云Region内部的集群**

  私网连接网格控制面则是利用VPC间的对等连接功能，打通了不同VPC之间的网络隔离。在创建网格时，要提前规划网格控制面的网段。

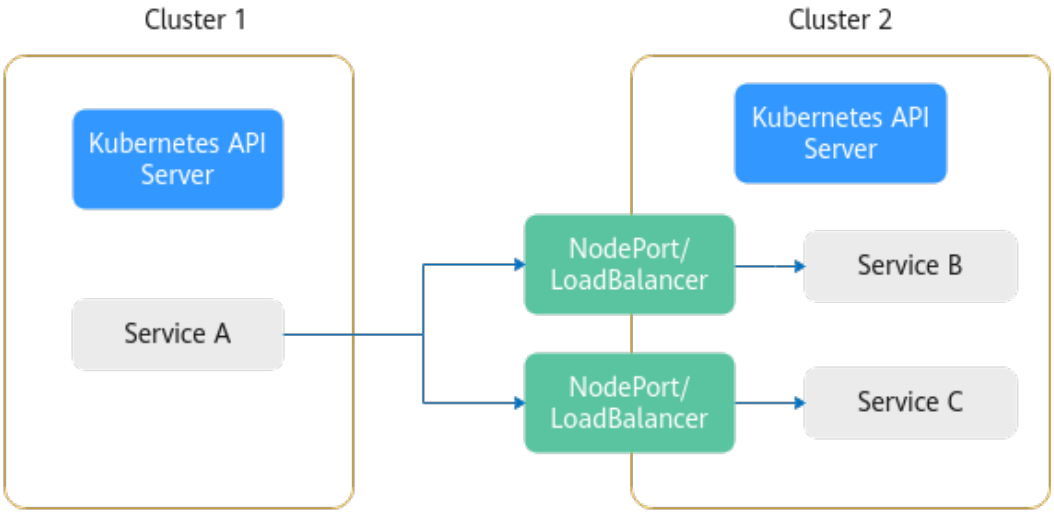
图 1-4 私网连接网格控制面



下面要讲一下企业版网格的多集群治理能力。

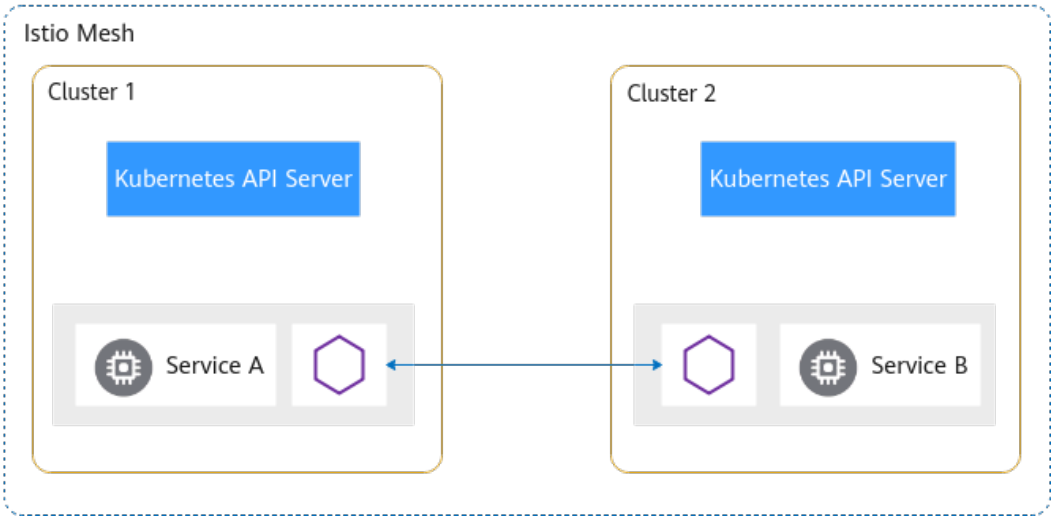
传统的服务跨集群访问，用户需要创建NodePort或LoadBalancer服务，绑定ELB，配置转发端口，而且对每一个需要跨集群访问的服务都要执行此操作。除此之外，用户还需要维护各个服务的对外访问配置。

图 1-5 传统跨集群访问



ASM企业版网格提供的服务跨集群访问能力，不需要用户进行复杂的配置，只需要将集群添加到网格，网格管理的所有集群的所有服务之间都是能够相互访问的，不管服务属于哪个集群。

图 1-6 ASM 跨集群访问



1.2.2 购买基础版网格

基础版网格的控制面组件安装在用户集群，用户需要自行管理和维护集群内的控制面组件。

前提条件

已创建CCE集群，如果未创建，请参照**购买CCE集群**创建。

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（控制面节点/工作节点）所在安全组的入方向15017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15017端口，以确保Istio自动注入功能正常。
- 1.13和1.15版本的istio组件不支持在CentOS和EulerOS2.5操作系统的节点上运行，在创建网格时，请不要指定这些类型的节点为master节点。
- 仅支持华为云CCE集群启用ASM服务网格，暂不支持CCE Autopilot集群。
- 不支持安全容器类型的CCE Turbo集群启用网格。
- 不支持已经启用了开源Istio网格的集群，包括自行部署的Istio和通过精选开源方式部署的Istio。
- 不支持用于出站流量控制的egressgateway。
- 不支持无 sidecar 的 Ambient 模式。
- 不支持通过以下自定义资源进行网格自定义： ProxyConfig, Envoyfilter, WorkloadEntry, WorkloadGroup, IstioOperator, WasmPlugin 。
- 不支持使用Kubernetes Gateway API配置Istio入口网关和管理网格流量。
- 请根据下表ASM网格版本与集群版本适配规则匹配合适的网格版本和集群版本。

表 1-1 ASM 网格版本与集群版本适配规则

ASM网格版本	集群版本
1.3	v1.13、v1.15、v1.17、v1.19
1.6	v1.15、v1.17
1.8	v1.15、v1.17、v1.19、v1.21
1.13	v1.21、v1.23
1.15	v1.21、v1.23、v1.25、v1.27、v1.28
1.18	v1.25、v1.27、v1.28、v1.29、v1.30、v1.31、v1.32

操作步骤

步骤1 登录应用服务网格控制台，按以下说明进入购买网格页面。

- 如果还未创建过网格，请单击ASM基础版中的“创建网格”。
- 如果当前已有网格，请在网格列表页面右上角单击“购买网格”。

步骤2 网格类型选择“基础版”。

步骤3 设置基础版网格参数。

图 1-7 基础版网格参数

- **网格名称**

  基础版网格的名称，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。

- **企业项目**（可选）

  企业项目是一种云资源管理方式，**企业项目管理**提供统一的云资源按项目管理，以及项目内的资源管理、成员管理。

  如需使用企业项目，账号必须为企业实名认证，且已开通企业项目。更多信息请参见**如何开通企业项目**。

- **Istio版本**

  基础版网格支持的Istio版本。

- **启用IPv6**

  启用双栈网格需满足以下条件：

网格类型	Istio版本	支持启用的集群类型	集群网络类型	其他说明
基础版	1.18及以上	CCE Turbo集群	云原生网络2.0	集群需要已启用IPv6，请参考**通过CCE搭建IPv4/IPv6双栈集群**

  说明

  - 是否启用IPv6功能，仅istio1.18及以上版本的基础版网格支持。
  - 低版本的网格升级到1.18及以上时，不支持升级后开启IPv4/IPv6双栈。
  - 网格开启IPv4/IPv6双栈后不支持关闭， 未开启双栈的网格也不支持创建完成之后开启双栈。

- **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，但可能会和用户的业务容器运行在一起。

- **设置可观测性配置**
  - 应用指标

    是否启用应用指标，应用指标开启后，可以在网格中构建服务访问指标、应用拓扑、服务健康和服务SLO定义。

    说明

    - Istio1.18之前版本的应用指标仅支持对接华为云APM，将为您自动开启基础版免费套餐，如需切换至高级套餐，请参考**使用指南**。1.18以及后续版本网格仅支持对接到华为云AOM，同时低版本网格升级到1.18后也可以切换到AOM，为了获取更优的使用体检，建议切换到AOM，如何使用AOM查看应用指标请参考**AOM页面如何查询网格指标**。开启AOM后，AOM将按用量进行收费。收费标准请参考**华为云定价**。
    - 1.18以及后续版本网格若要对接AOM启用应用指标，需提前在CCE集群插件中心中安装云原生监控插件（kube-prometheus-stack）。

  - 访问日志

    是否启用访问日志，访问日志开启后，可以在网格中查询到详细的服务键访问记录，定位每次访问异常。选择”访问日志”后，选择需要对接到的LTS日志组和LTS日志流，访问日志将传输到对应的LTS日志流，并可在“监控中心-访问日志”中查看访问日志。

    说明

    访问日志目前仅Istio 1.18及以上版本支持对接到华为云云日志服务（LTS）。若要对接到华为云LTS，请提前在CCE集群插件中心中安装云原生日志采集插件（CCE Log-Agent）。

  - 调用链

    采样率：用于调用链生成的请求占全部请求的采样百分比。

    选择使用服务：即选择调用链上报的服务后端。可选择“华为云APM”和“第三方Jaeger/Zipkin服务”，当选择“第三方Jaeger/Zipkin服务”时，需要配置“服务地址”和“服务端口”两个参数，即第三方调用链服务接收请求信息的地址和端口。

    说明

    - 仅Istio 1.15及以上版本支持第三方调用链。
    - 如果您要使用“第三方Jaeger/Zipkin服务”调用链，请先自行完成调用链服务的安装，也可参考**如何对接Jaeger/Zipkin查看调用链**进行安装。之后获取服务地址。
    - Jaeger和Zipkin的默认服务端口均为9411，如果安装的时候自定义了服务端口在配置“服务端口”时请填写实际的值。

步骤4 （可选）高级配置。

- **命名空间注入配置**

  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。

  如果不进行命名空间注入配置，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。

  须知

  如果所选集群注入过sidecar，建议您选择重启已有服务，如果不重启，服务所在实例中已存在istio-proxy容器的情况下，网关会访问失败。

- **流量拦截配置**

  说明

  - 仅Istio 1.13及以上版本支持流量拦截功能。
  - 默认情况下，ASM所注入的sidecar会拦截所有入方向和出方向流量，可通过流量拦截配置修改默认的流量拦截规则。

  入方向端口：可用于配置端口级别拦截规则，生效于网格服务的内部端口，以逗号分隔入站端口。

  - 仅拦截指定端口表示访问网格服务指定端口范围内的请求将被重定向到sidecar中。
  - 仅排除指定端口表示访问网格服务的请求，除端口范围外的请求将被重定向到sidecar中。

  出方向端口：可用于配置端口级别拦截规则，生效于网格服务对外访问的流量方向上，以逗号分隔出站端口。

  - 仅拦截指定端口表示网格服务访问指定端口范围内的请求将被重定向到sidecar中。
  - 仅排除指定端口表示网格服务对外访问的请求，除指定端口范围外的流量都将被重定向到sidecar中。

  出方向网段：可用于配置网段级别拦截规则，生效于网格服务对外访问的流量方向上，以逗号分隔 IP ，以 CIDR 形式表示。

  - 仅拦截指定网段表示指定网段范围内的流量将被重定向到sidecar中。
  - 仅排除指定网段表示除指定网段范围外的流量将被重定向到sidecar中。

- **资源标签**

  填写标签键和标签值，最多添加20个标签。

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”。

创建基础版网格预计需要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

创建网格会自动创建一个otel-collector工作负载。详情请参考**otel-collector工作负载作用**。

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。
- ASM从1.8.6-r4、1.13.9-r5、1.15.5-r3开始支持金丝雀升级，部署的istio资源均会带上版本后缀，如istiod-1-13-9-r5。

----结束

1.2.3 基础版、社区开源版本对比

大类	功能项	功能点	社区开源版本	基础版
规格	管理规模	支持最大管理实例数	-	200
基础功能	服务发现和服务注册	通过服务中心集群获取服务列表、服务实例状态自动刷新、容器服务自动服务注册，业务无需实现注册逻辑、容器服务自动服务发现，业务无需实现发现订阅逻辑、服务实例副本数动态管理	√	√
	服务多版本	服务创建分版本管理、支持分版本进行监控	-	√
		支持分版本进行服务负载管理	√	√
	服务多端口	支持管理多个端口的服务、支持管理多个端口多协议的服务	√	√
		灰度发布支持多端口	-	√
	服务多形态	支持容器类型的服务后端	√	√
	协议及语言支持	HTTP协议灰度、治理、监控（根据协议特征不同细节不同，参照协议功能矩阵）、gRPC协议灰度、治理、监控（根据协议特征不同细节不同，参照协议功能矩阵）、开发语言无关、开发框架不限定、业务代码无侵入	√	√
	应用网关	支持四层协议对外访问、支持七层协议对外访问、支持入口路径映射、支持网关处TLS终止，支持配置对外证书和密钥	√	√
	负载均衡	支持轮询、随机、最小连接数以及一致性哈希的LB算法，可以基于特定的HTTP Header，或者基于Cookie值	√	√

大类	功能项	功能点	社区开源版本	基础版
	故障注入	支持注入指定时延或特定错误的故障，支持配置故障百分比	√	√
	熔断	支持配置最大请求数、每连接最大请求数、最大等待请求数、最大重试次数等七层请求管理；支持配置最大连接数、连接超时时间等四层连接管理；支持异常点检查、故障实例的自动隔离和自动恢复	√	√
	治理流量类型	支持对服务间内部通信流量进行治理、支持对服务外网访问流量（即Ingress流量）进行治理	√	√
	运行环境支持	支持容器应用治理	√	√
	认证	非侵入的双向TLS认证和通道加密	√	√
	授权	服务访问授权管理	√	√
	灰度发布	支持基于浏览器、操作系统、自定义HTTP Header、Cookie内容等配置灰度分流策略，支持基于URL配置灰度分流策略，支持基于请求参数、流量权重的灰度发布	√	√
		支持金丝雀灰度发布模板	-	√
		支持蓝绿灰度发布模板	-	√
		支持灰度发布过程中服务运行情况的监控以辅助灰度发布决策	-	√
		支持灰度发布过程中服务请求情况的监控以辅助灰度发布决策	-	√
		灰度发布时动态配置服务实例数	-	√
		支持灰度发布过程中动态的流量比例监控	-	√
	应用拓扑	提供应用下服务调用关系的全局拓扑	-	√
		提供拓扑图上各个服务间请求数、异常请求数等重要指标	-	√
		实时应用拓扑查看	-	√

大类	功能项	功能点	社区开源版本	基础版
	链路跟踪/调用链	支持非侵入调用链埋点	√	√
	指标监控	提供服务实例CPU、内存、磁盘等运行数据监控，提供服务访问RPS、时延等访问指标的监控，提供对访问指标、异常指标的统计分析，支持对接Prometheus等开源Metric组件，支持通过配置对接不同的Metric后端	√	√
	访问日志	访问日志非侵入采集	√	√
	安装	支持现有、新建Kubernetes集群按需一键安装启用Istio能力	-	√
	升级	支持控制面平滑升级，不中断应用业务	√	√
		支持数据面平滑升级，不中断应用业务	√	√
	插件管理	支持社区插件按需一键安装，支持Grafana、Prometheus	-	√
		支持社区插件按需一键安装，支持Tracing插件	-	√
		支持社区插件按需一键安装，支持Kiali插件	-	√
		支持社区插件按需一键安装，支持ELK插件	-	√
	代理管理	透明流量拦截、基于Iptables流量拦截、支持代理自动注入、支持Namespace级别和工作负载级别的注入管理	√	√
	代理形态	支持每Pod的Sidecar模式	√	√
	命令行工具	支持使用命令行进行流量策略管理（如istioctl、kubectl）	-	√

1.2.4 大规格实例优化

当网格内实例规模持续增大时，会引起Istio控制面组件Istiod和数据面组件Envoy的内存飙升问题。在购买网格时，可以为网格启用Mantis插件来解决此问题。

说明

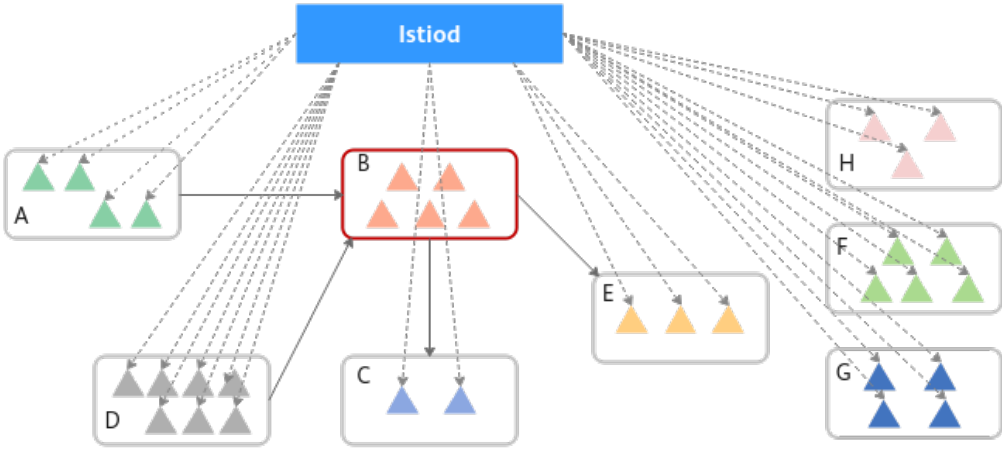
大规格实例优化特性仅在“华东-上海一”区域开放。

Mantis 插件介绍

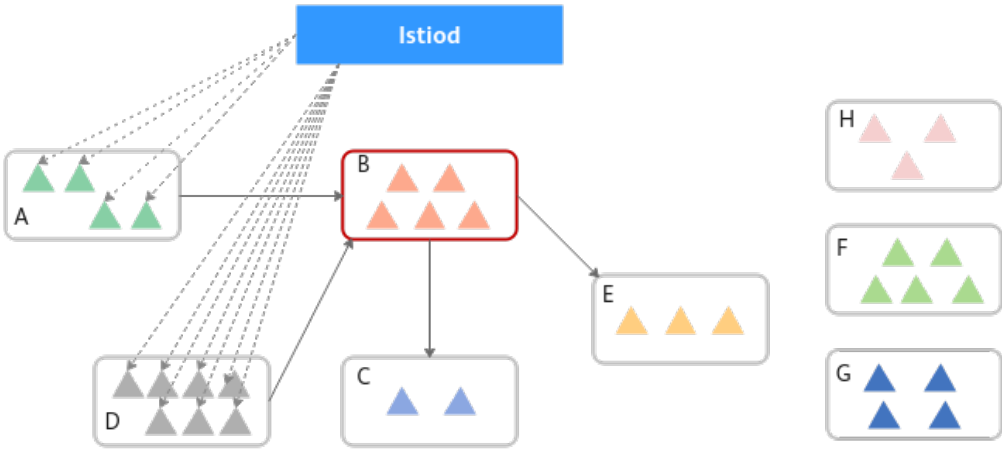
Mantis致力于解决由网格内实例规模增大而引起的Istio控制面组件Istiod和数据面组件Envoy的内存飙升问题。同时，从多个维度输出了Istio相关性能模型，填补了Istio在大规模实例场景下的性能模型空白，为Istio性能提升提供了有力支撑。

- **提高控制面Istiod稳定性**

  Istio采用的是全局更新配置信息策略，如下图所示，当服务B的相关信息发生变化时，Istiod会生成全量xds并推送给网格内所有Proxy，这会导致Istiod的瞬时CPU、内存占用过高，可能导致Istiod重启。

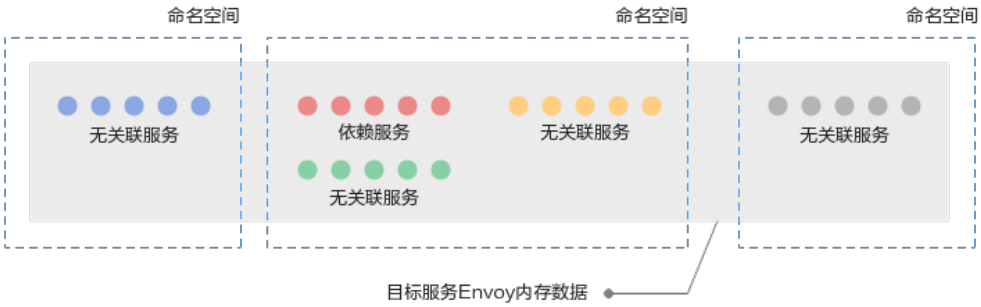


  Mantis根据服务依赖关系按需更新配置，当服务B的相关信息发生变化时，Istiod会将服务B相关的更新推送给调用服务B的相关服务（服务A和服务D）的所有Endpoint中，包括生成配置和下发配置，可极大缓解Istiod CPU、内存占用过高的问题，提高控制面的稳定性。

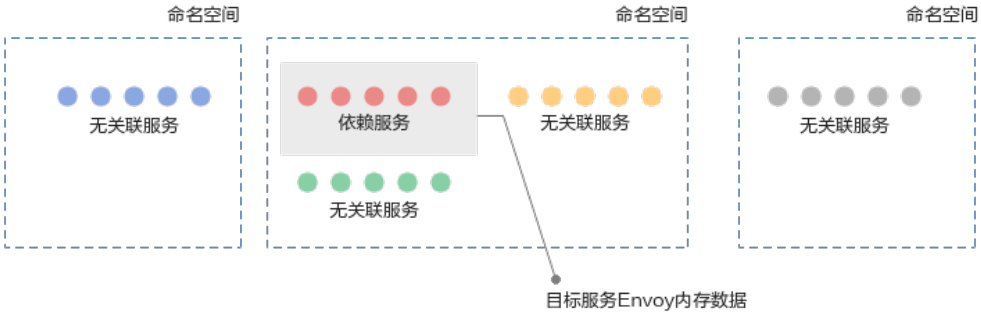


- **减小数据面sidecar资源消耗**

  Istio使用Envoy作为网格数据面的sidecar。目前Istio采用的是全局更新配置信息策略，即网格内每一个实例中的Envoy都储存了网格内其他所有服务的相关信息，包括listener、route、cluster和endpoint。这在实例规模不断增大的情况下，将会发生内存爆炸，在实际生产环境中是不可接受的。

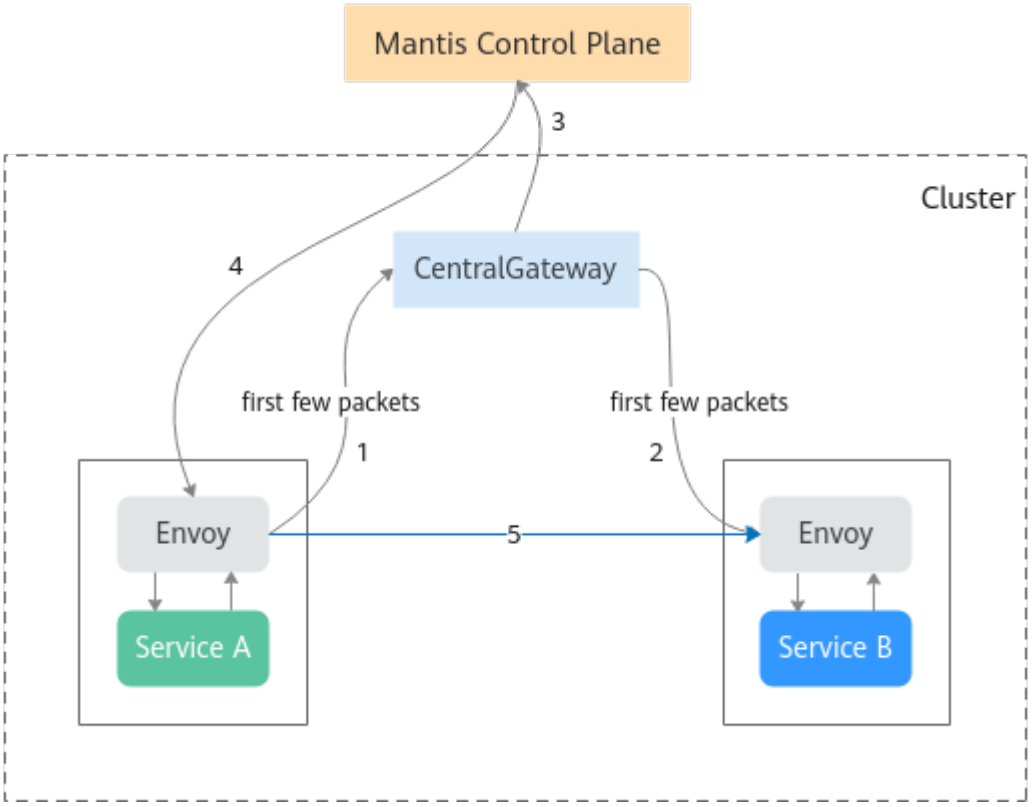


Mantis根据服务依赖关系按需更新配置，即每一个实例的Envoy只存储和本服务需要调用的服务的相关信息，可以将Envoy的内存占用最小化。



Mantis 工作原理

Mantis的架构和工作流程如下图所示：



网格安装Mantis后，会在用户集群安装CentralGateway（CGW）组件。CGW包含网格全局的服务路由信息，用于转发首包流量；网格控制面会根据服务调用关系按需更新服务路由信息。

网格安装初始化完成后，所有服务实例（Pod）的Envoy中只存储了CGW组件的路由信息。

1. 当某个服务Service A的实例（Pod）第一次访问另一个服务Service B时，由于Service A实例的Envoy中没有Service B的路由信息，Envoy会将请求发送到CGW上。
2. CGW保存了网格全局的服务路由信息，所以CGW可以将请求转发到Service B的实例。
3. 同时，CGW会将Service B的路由信息上报到Mantis控制面。
4. 然后再由Mantis控制面下发给Service A的所有实例。
5. 在后续Service A访问Service B时，由于Service A所有实例的Envoy中已经存在Service B的路由信息，请求将会被直接转发给Service B的实例。

而对于Service A未访问过的服务，Service A所有实例的Envoy中不会存储服务的路由信息，可以有效减小Envoy内存消耗。

Mantis 使用约束

- 启用Mantis插件对网格有如下限制：
  企业版网格，1.8.4-r4及以上网格版本。

  ⚠ 注意

  企业版网格已下线，不再支持创建企业版网格，推荐使用基础版网格。存量的企业版网格仍可继续使用。
- Mantis插件仅针对HTTP协议的服务做优化。
- 安装Mantis插件后，暂时不支持卸载。
- 安装Mantis插件的网格升级时，会同步升级Mantis插件。
- 启用Mantis插件后，如有访问外部服务的需求，需要为集群所在VPC子网配置NAT网关。

1.3 网格管理

1.3.1 续费网格

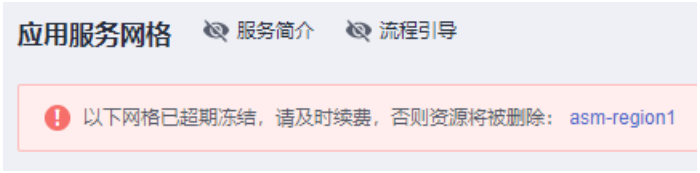
操作场景

包年/包月的网格如果即将到期或已经超期，需要及时续费避免网格被删除而影响业务。建议您开通自动续费功能。

操作步骤

步骤1 登录应用服务网格控制台，确认网格是否需要续费。判断方法如下：

- 列表上方是否提示已超期冻结需及时续费的网格。



- 网格计费模式中是否存在已超期或即将到期提示。

图 1-8 已超期提示

图 1-9 即将到期提示

若存在已超期或即将到期的网格，单击网格右上角的图标，进入续费页面。

步骤2 选择续费时长，单击“去支付”，选择支付方式后，单击“确认付款”即可成功续费。

更多信息（自动续费、导出续费清单等）请参考**续费管理**。

----结束

1.3.2 按需转包周期

操作场景

- 包年/包月是预付费模式，按订单的购买周期计费，适用于可预估资源使用周期的场景，价格比按需计费模式更优惠。
- 按需计费是后付费模式，按资源的实际使用时长计费，可以随时开通/删除资源。

说明

“包年/包月”计费模式的网格也称“包周期”网格。

如果您需要长期使用当前网格，可以将按需购买的网格转为包周期计费模式，节省开支。

约束与限制

由于基础版网格免费，按需转包周期仅适用于企业版或专有版网格。

操作步骤

步骤1 登录应用服务网格控制台，在对应的按需计费网格下单击“转包周期”。

步骤2 选择网格规模和计费周期，单击“提交”。

说明

可选的网格规模受限于当前实例数。例如，企业版按需计费网格的规格为5000实例，转包周期时，如果当前实例数为550，则只能选择超过550实例的网格规模，如1000、2000、5000。

步骤3 选择支付方式后，单击“确认付款”。

变更完成后，约一分钟左右生效，请耐心等待。

----结束

1.3.3 变更规格

操作场景

当您购买的服务网格规模无法满足业务需要时，可参考本章节变更规格。

约束与限制

- 基础版网格规模固定为200实例，不支持变更。
- 企业版按需计费网格规模固定为5000实例，不支持变更。

操作步骤

步骤1 登录应用服务网格控制台，在对应的网格下单击“规格变更”。

步骤2 选择网格规模，单击“提交”。

步骤3 选择支付方式后，单击“确认付款”。

变更完成后，约一分钟左右生效，请耐心等待。

----结束

1.3.4 网格事件

操作场景

应用服务网格支持事件中心，即通过界面提供网格创建、删除；网关创建、删除等重要操作的详细信息。

说明

在ASM 1.0中购买的基础版1.8以及以上版本的网格可以查看网格内事件。

操作步骤

步骤1 登录应用服务网格控制台，按照网格类型搜索基础版网格。

步骤2 单击网格右上角的图标，在右侧弹出的页面查看网格事件。

----结束

1.3.5 专有版转基础版

操作场景

在ASM 1.0中购买的按需或包周期计费专有版网格，可以转为免费基础版网格，以节省开支。转为基础版网格后，网格规格统一为200实例。

说明

如果已有专有版网格实例超过200个，则转为基础版网格以实际个数为准。仅新建网格场景受限最多可创建200个网格实例。

操作步骤

步骤1 登录应用服务网格控制台，按照网格类型搜索专有版网格。

步骤2 根据专有版网格的计费模式，转换方法如下：

- 按需计费

  单击按需计费网格右上角的图标，在弹出的提示页面单击“确认”，即可完成迁移。

- 包周期计费

  单击包年/包月网格右上角的图标，在弹出的提示页面单击“确认”。专有版包周期网格退订成功后将自动转为基础版免费网格。

----结束

1.3.6 退订与卸载

操作场景

当网格不再需要时，可以将其删除，避免继续产生费用。

- 包年/包月网格采用退订方式删除，请参考**退订网格**。
- 按需计费或免费网格采用卸载方式删除，请参考**卸载网格**。

对于包年/包月网格，即使集群已被删除，服务网格仍然会继续计费，如果网格不再使用，请将其退订。

对于按需计费网格，即使集群已被删除，服务网格仍然会继续计费，如果网格不再使用，请将其卸载。

约束与限制

- 企业版网格需移除全部集群后，才可支持卸载或退订。
- 专有版包周期网格如需删除，需要先通过退订转为基础版网格，再执行卸载操作。
- 如果网格有正在运行的灰度发布任务，请先完成灰度发布，再卸载或退订网格。
- 卸载基础版或专有版网格时，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。

退订网格

步骤1 登录**应用服务网格控制台**，在对应的包年/包月网格下单击 图标，跳转至“云服务退订”页面。

步骤2 勾选待退订的网格，单击操作列的“退订资源”。

步骤3 选择或填写一个退订原因，单击“退订”。

步骤4 在弹出的确认页面单击“退订”。

----结束

卸载网格

步骤1 登录**应用服务网格控制台**，在对应的按需计费或免费网格下单击 图标。

- 如果是企业版网格，请执行**步骤2**。
- 如果是专有版或基础版网格，请执行**步骤3**。

步骤2 在“卸载服务网格”页面，单击“确定”，卸载后将直接删除网格。

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-poxy 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 > 服务发现”页面创建服务暴露对外访问方式。
- 对于专有版网格，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 > 节点管理 > 节点”页面中查看。

图 1-10 卸载专有版或基础版网格

----结束

1.4 服务管理

1.4.1 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、流量监控及灰度发布等操作。

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。

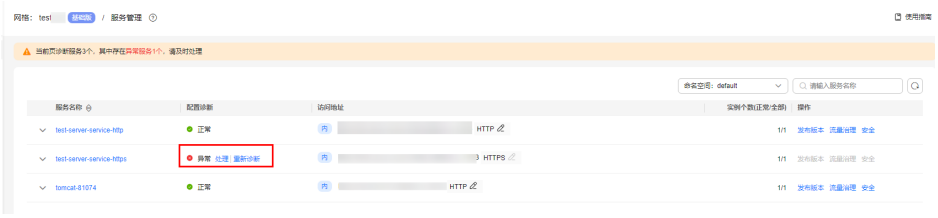
服务诊断

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 1-11 服务诊断



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 1-12 手动处理项

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 1-13 自动处理项

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

1.4.2 手动修复项

1.4.2.1 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}

1.4.2.2 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

1.4.2.3 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。

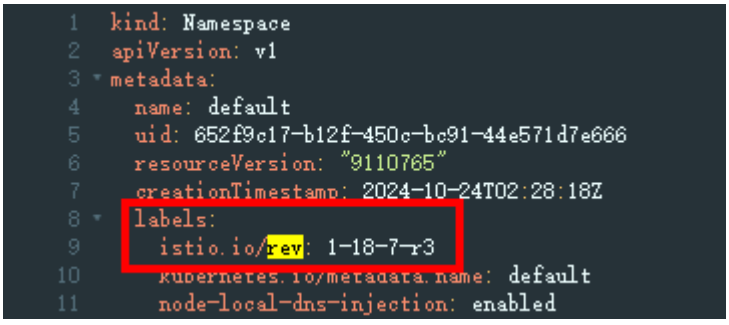
修复指导

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。

- 未注入，执行**步骤2**。
- 已注入，执行**步骤3**。
  检查方法如下：
  在CCE控制台，进入集群详情的命名空间页面，单击您的命名空间操作列的“编辑YAML”，如果存在istio.io/rev=<revision>或者istio-injection=enabled标签内容说明已注入sidecar。

  说明

  - 1.13.9-r3及以下istio版本、1.15.5-r2及以下istio版本，需存在istio-injection=enabled标签。请注意版本号使用中划线连接。
  - 1.13.9-r3以上istio版本、1.15.5-r2以上istio版本、1.18所有istio版本，需存在istio.io/rev=<revision> 标签。请注意版本号使用中划线连接，如下图：



步骤2 可参考**sidecar注入**为您的某个负载或者命名空间下的所有工作负载关联的Pod注入sidecar。

这两种场景注入的方法如下：

- 为命名空间下所有工作负载关联的Pod注入sidecar ，即打上标签。请根据下面说明中的不同istio版本打不同的标签。打标签命令如下：
  ```
  kubectl label ns <namespace> istio-injection=enabled
  ```
  或者
  ```
  kubectl label ns <namespace> istio.io/rev=<revision>
  ```

  说明

  系统将根据不同的istio版本为命名空间设置标签：
  - 设置istio-injection=enabled标签，适用于1.13.9-r3及以下版本、1.15.5-r2及以下版本。
  - 设置istio.io/rev=<revision> 标签，适用于1.13.9-r3以上版本、1.15.5-r2以上版本、1.18所有版本。
- 只为某个工作负载注入sidecar

  在CCE控制台，在工作负载所在行，单击您的工作负载操作列的“更多 > 编辑YAML”，请根据您的istio版本手动添加annotations字段或者labels字段。
  - 1.13.9-r3以上版本、1.15.5-r2以上版本、1.18所有版本，设置如下：
    ```
    labels:
      istio.io/rev=<revision>
    ```
  - 1.13.9-r3及以下版本、1.15.5-r2及以下版本，设置如下：
    ```
    annotations:
      istio-injection: enabled
    ```

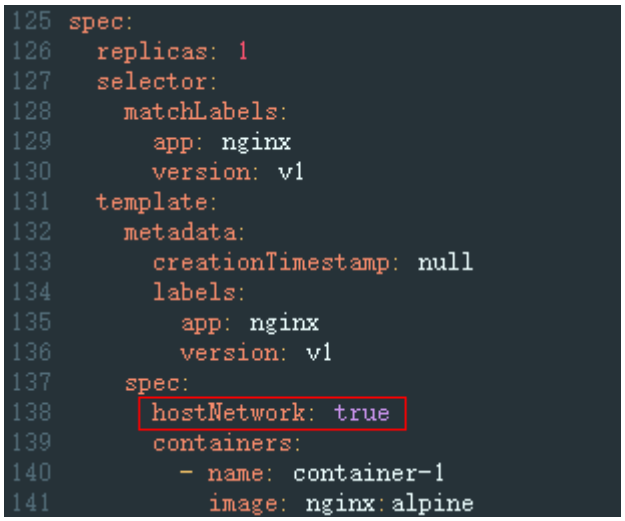
您可以单击**Installing the Sidecar**了解更多sidecar注入的知识。

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。



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

1.4.2.4 Service 在所有集群的配置是否相同

问题描述

如果网格管理了多个集群，其他集群的同名Service和当前集群的Service被视为同一个服务，spec.selector配置、spec.ports配置必须和当前Service相同。

说明

仅在网格管理的集群数量大于1时才可能出现此异常。

修复指导

修改多集群下同名Service的配置：

步骤1 登录CCE控制台，单击集群名称进入详情页面。

步骤2 在左侧导航栏选择“资源 > 服务发现”，找到与本服务同名的服务，单击“更多 > 编辑YAML”，修改spec.selector配置、spec.ports配置与本服务一致。

```
spec:
  ports:
    - name: http-cce-service-0
      protocol: TCP
      port: 8443
      targetPort: 80
  selector:
    app: test
```

步骤3 重复步骤1~步骤2，修改其他集群下同名Service的配置。

----结束

1.4.3 自动修复项

1.4.3.1 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，单击集群名称进入详情页面。

步骤2 在左侧导航栏单击kubernetes 资源下的"服务"，进入"服务"页签，单击对应服务后的“更多 > 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

1.4.3.2 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，单击集群名称进入详情页面。

步骤2 在左侧导航栏选择“资源 > 服务发现”，单击对应服务后的“更多 > 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

1.4.3.3 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 1-14 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 1-15 Service 的端口协议类型

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

----结束

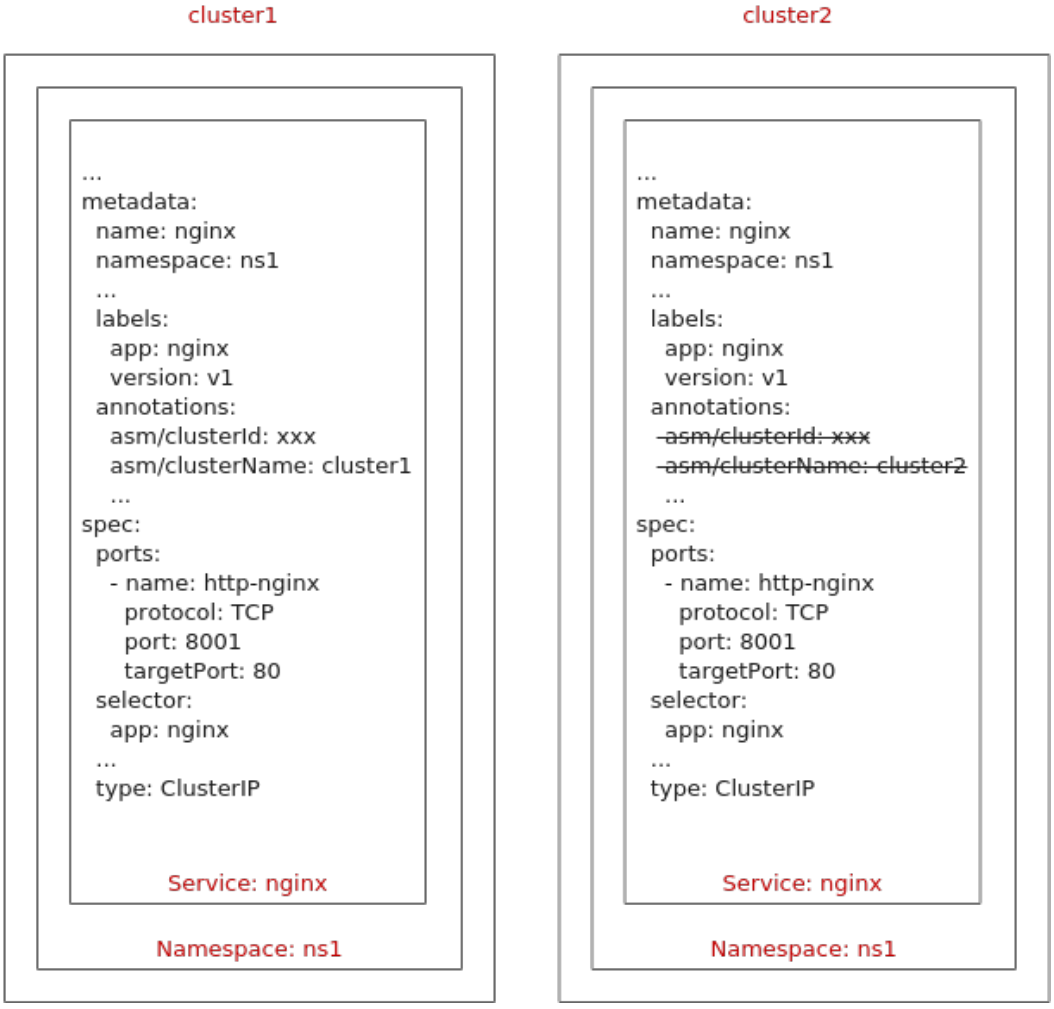
1.4.3.4 Service 是否支持跨集群访问

问题描述

服务支持跨集群访问，需要满足以下规则：

- 每个集群中必须有和此服务所在命名空间同名的命名空间。
- 每个集群中，同名命名空间下有同名同类型的服务。
- 每个同名服务中的labels、端口信息，以及selector中的内容完全相同。
- 除服务本身所在集群中的YAML文件annotations下带有asm/clusterId、asm/clusterName字段外，其他集群的同名服务都不包含上述字段。

通过以下示例，可以直观地看出服务支持跨集群访问需要满足的规则：



修复指导

步骤1 登录CCE控制台，选择服务所在集群，在“资源 > 服务发现”页面查看服务所在的命名空间，然后选择网格内其他集群，创建同名命名空间。

如果已有同名命名空间则跳过此步骤。

步骤2 选择服务所在集群，在“资源 > 服务发现”页面选择对应的命名空间和服务，单击操作列的“更多 > 编辑YAML”，复制文本框中的内容。

步骤3 选择网格内其他集群，在“资源 > 服务发现”页面选择对应的命名空间，单击右上角“YAML创建”。

步骤4 将步骤2中复制的内容粘贴到文本框中，删除其中的metadata.uid、metadata.resourceVersion、spec.clusterIP、spec.clusterIPs，以及metadata.annotations中的asm/clusterId、asm/clusterName字段。

步骤5 单击“确定”，确保服务创建成功。

----结束

1.5 网关管理

1.5.1 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，详情请参见**创建独享型负载均衡器**或**创建共享型负载均衡器**。如果是独享型，实例规格需要选择“网络型（TCP/UDP）”，且网络类型需要勾选“IPv4私网”，确保负载均衡实例有私网IP地址。

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。

- **访问方式**
  - 协议版本：支持选择IPV4和双栈两种，已开启ipv6的网格方可配置该参数。
  - 服务网关使用弹性负载均衡服务（ELB）的负载均衡实例提供网络访问，支持共享型和独享型规格，且支持公网和私网。如果是独享型，实例规格需要选择“网络型（TCP/UDP）”，且网络类型需要勾选“IPv4私网”，确保负载均衡实例有私网IP地址。

- **访问入口**
  - 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。

- 外部访问地址

  系统自动填充负载均衡实例的IP地址，作为服务访问入口地址，您也可以将其修改为负载均衡实例关联的域名。

- 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。

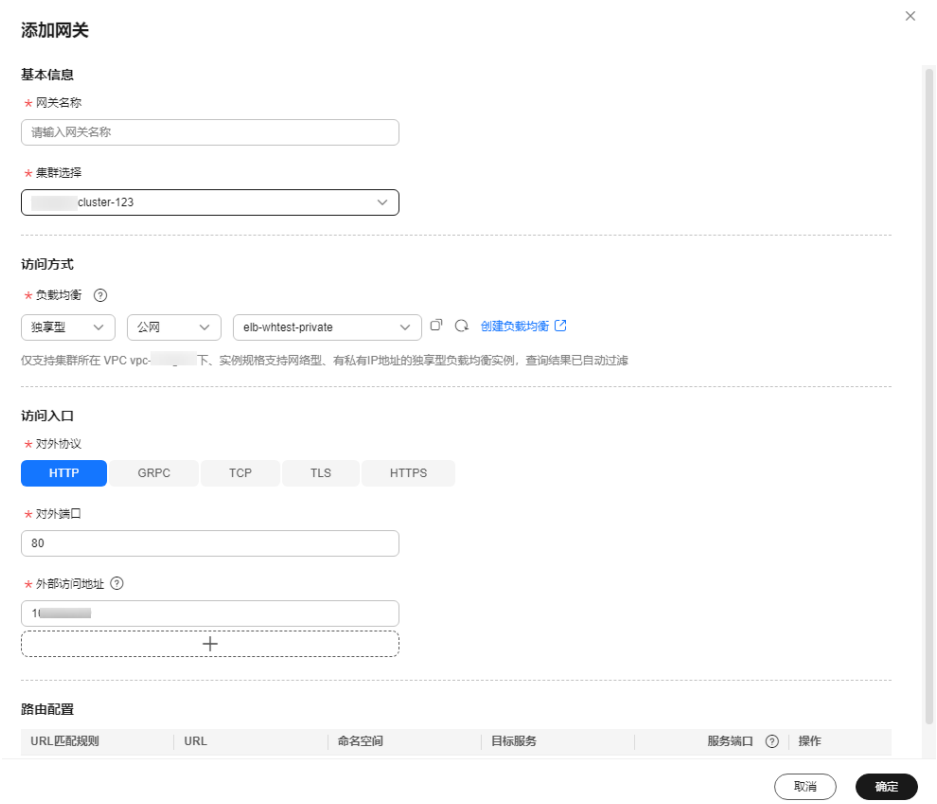
- 密钥证书

  - 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。

图 1-16 添加网关



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（转发规则由外部访问地址+URL组成）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。

- **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。

  说明

  支持通过YAML方式配置正则匹配规则，修改VirtualService配置文件即可：

  ```
  ...
    http:
      - delegate:
          name: nginx-80
          namespace: default
        match:
          - uri:
              regex: /do[a-z]*/
  ...
  ```

  如上所示，“regex”表示按正则表达式方式匹配URL，注意URL必须以“/”开头。
- **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/HTTPS时可配置）

  重写HTTP/HTTPS 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/HTTPS的Host/Authority头。
- **域名单独配置路由**

  为网关中某个域名单独配置路由规则。

图 1-17 添加路由

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 1-18 服务外网访问地址

----结束

1.5.2 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。

- **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。

  说明

  支持通过YAML方式配置正则匹配规则，修改VirtualService配置文件即可：

  ```
  ...
   http:
     - delegate:
         name: nginx-80
         namespace: default
       match:
         - uri:
             regex: /do[a-z]*/
  ...
  ```

  如上所示，“regex”表示按正则表达式方式匹配URL，注意URL必须以“/”开头。

- **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/HTTPS时可配置）

  重写HTTP/HTTPS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/HTTPS的Host/Authority头。

- **域名单独配置路由**

  为网关中某个域名单独配置路由规则。

图 1-19 添加路由

步骤3 配置完成后，单击“确定”。

----结束

1.6 灰度发布

1.6.1 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

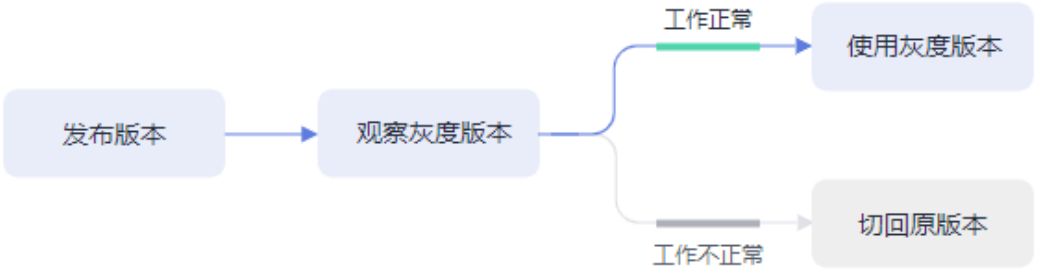
又称灰度发布，是版本升级平滑过渡的一种方式，当版本升级时，使部分用户使用新版本，其他用户继续使用老版本，待新版本稳定后，逐步扩大范围把所有用户流量都迁移到新版本上面来。这样可以最大限度地控制新版本发布带来的业务风险，降低故障带来的影响面，同时支持快速回滚。

图 1-20 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 1-21 蓝绿发布流程



1.6.2 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

  当您需要在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 1-22 灰度发布基本信息

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。
- **自定义镜像**
  用户可通过自定义镜像自行配置本地或第三方镜像地址，灰度发布镜像将采用所配置镜像。注意需确保自定义镜像地址有效，镜像可拉取。

图 1-23 灰度版本信息

图 1-24 灰度版本信息

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。

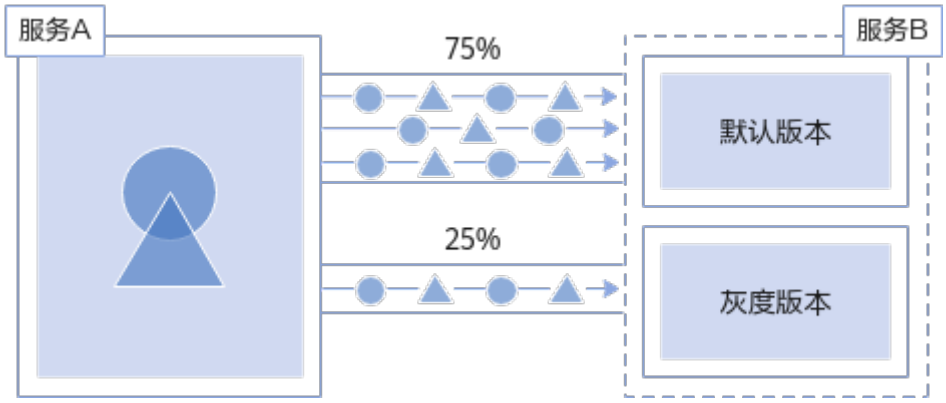
步骤5 （仅金丝雀发布涉及）单击“配置流量策略”，进行流量策略配置。

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 1-25 基于流量比例

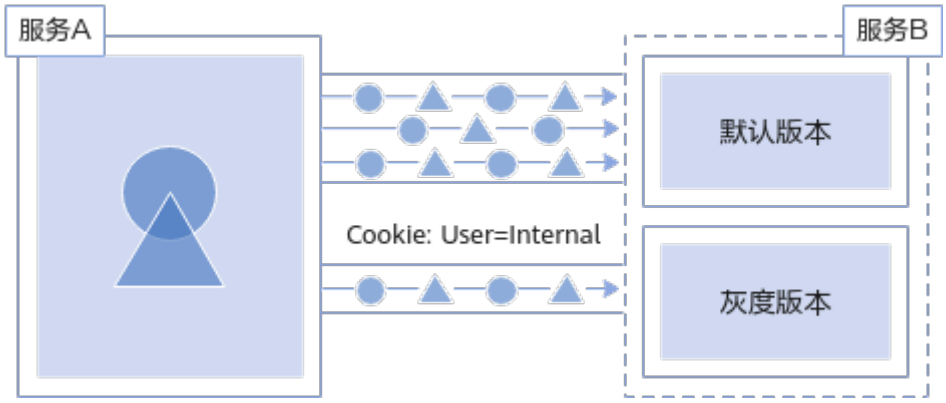
  

  流量配比：可以为默认版本与灰度版本设置流量配比，系统将根据输入的流量配比来确定流量在两个版本间分发的比重。

- **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 1-26 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（详情可参考**如何使用Istio调用链埋点**），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

1.6.3 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 1-27 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

1.7 网格配置

1.7.1 网格配置概述

网格配置提供了集群管理、系统组件管理、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、网格规格、当前版本、计费模式、网格控制面网段、可观测性以及已启用网格的集群。
  企业版网格还支持添加、移除集群，详情请参见**添加集群**。
- “系统组件管理”页签：（仅企业版网格包含）展示所有控制面组件对应的工作负载，包括istio-egressgateway、istio-ingressgateway、istio-eastwestgateway（非扁平网络包含）和istiod。如果启用了大规格实例优化（即安装Mantis插件），还会显示mantis-centralgateway组件的信息。
- “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**。
- “网格扩展”页签：配置可观测性配置。详情请参见**网格扩展**。

1.7.2 添加集群

企业版网格支持对多个集群进行管理，且支持服务跨集群通信。

⚠ 注意

企业版网格已下线，不再支持创建企业版网格，推荐使用基础版网格。存量的企业版网格仍可继续使用。

约束与限制

- 目前支持v1.15、v1.17、v1.19和v1.21版本的集群加入企业版网格。
- 不支持安全容器类型的CCE Turbo集群添加至网格。
- 同一网格最多只能添加五个集群。
- 同一虚拟私有云的集群只能加入同一个网格。
- 为了满足高可用的要求，集群需要至少包含两个可用节点，每个节点至少保证有2U4G的可用资源。
- 集群的服务网段、容器网段不能和网格内已有集群的服务网段、容器网段冲突。如果集群和网格内的已有集群处于不同的VPC，集群的子网网段也不能冲突。
- 如果实例（Pod）需要跨集群通信，集群需要使用ENI网络模型，且集群之间网络互通，可以处于同一VPC内，也可以将多个集群的VPC通过其他方式（对等连接、云连接等）连通。

- CCE集群和CCE Turbo集群混合多集群场景，CCE集群服务访问Turbo集群服务时，需要为Turbo集群的ENI安全组入方向放通CCE集群的容器网段，否则会访问不通。

操作步骤

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添加集群页面。

- （快捷方式）在企业版网格右上方，单击图标。
- 在企业版网格详情页，单击左侧导航栏的“网格配置”，在“基本信息”页签单击“添加集群”。

步骤2 设置集群信息。

- **集群配置**

  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。勾选后，系统自动校验集群是否符合添加要求，若校验不通过，会以图标标识，鼠标放上去可以查看校验不通过的原因以及解决方案。

  说明

  如果当前没有可用集群，需要先创建集群后，再进行添加，详情可参考**购买CCE集群**。

  选择好集群后，按照集群的网络模型或实际的通信需求选择集群的网络类型（要求网格版本为1.8.4-r3及以上）。

- **命名空间注入配置**
  - 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。
  - 是否重启已有服务：如果开启（即重启已有服务），会自动注入istio-proxy sidecar，业务将会暂时中断。
- **可观测性配置**

  继承自网格配置，不可修改。

步骤3 设置完成后，单击“确定”。

添加集群大约需要一分钟，请耐心等待。添加完成后，返回网格详情页面可查看到添加的集群信息。

----结束

相关操作

- 移除集群

  在集群所在行单击“移除”，选择重启已有服务，服务重启后会去除注入的istio-proxy sidecar。

图 1-28 移除集群

1.7.3 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 1-29 注入 sidecar

参数释意：

- 选择命名空间：选择一个或多个命名空间，系统将根据不同的istio版本为命名空间设置标签：
  - 设置istio-injection=enabled标签，适用于1.13.9-r3及以下版本、1.15.5-r2及以下版本。
  - 设置istio.io/rev=<revision> 标签，适用于1.13.9-r3以上版本、1.15.5-r2以上版本、1.18所有版本。
- 是否重启已有服务：

   ：会重启命名空间下已有服务关联的Pod，新生成的Pod自动注入istio-proxy sidecar。重启将会暂时中断业务。
  - 新勾选的命名空间，会添加自动注入标签，重启命名空间下所有Deployment类型的Pod，新生成的Pod自动注入istio-proxy sidecar。
  - 去勾选的命名空间，会删除自动注入标签，重启命名空间下所有Deployment类型的Pod，新生成的Pod去除istio-proxy sidecar。
  - 已勾选的命名空间，若存在未注入的Pod，将会重启该工作负载的所有Pod以注入sidecar。若不存在未注入的Pod，将不会重启。

   ：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

  sidecar注入失败常见原因请参考**sidecar注入失败可能的原因**。
- **流量拦截配置**

  说明

  默认情况下，ASM所注入的sidecar会拦截所有入方向和出方向流量，可通过流量拦截配置修改默认的流量拦截规则。

  入方向端口：可用于配置端口级别拦截规则，生效于网格服务的内部端口，以逗号分隔入站端口。
  - 仅拦截指定端口表示访问网格服务指定端口范围内的请求将被重定向到sidecar中。
  - 仅排除指定端口表示访问网格服务的请求，除端口范围外的请求将被重定向到sidecar中。

  出方向端口：可用于配置端口级别拦截规则，生效于网格服务对外访问的流量方向上，以逗号分隔出站端口。
  - 仅拦截指定端口表示网格服务访问指定端口范围内的请求将被重定向到sidecar中。
  - 仅排除指定端口表示网格服务对外访问的请求，除指定端口范围外的流量都将被重定向到sidecar中。

  出方向网段：可用于配置网段级别拦截规则，生效于网格服务对外访问的流量方向上，以逗号分隔 IP ，以 CIDR 形式表示。
  - 仅拦截指定网段表示指定网段范围内的流量将被重定向到sidecar中。
  - 仅排除指定网段表示除指定网段范围外的流量将被重定向到sidecar中。

说明

- 若界面提示“以下集群未开放命名空间注入修改操作”，可能因为当前网格是通过1.0控制台创建的，默认不开放命名空间注入，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。

----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。

图 1-30 sidecar 资源限制

- 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 ≥ 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 ≥ 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

为 sidecar 资源配置内存告警

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。因此，建议用户为所有sidecar资源配置内存告警，当告警被触发后，及时扩容对应sidecar的内存最大值，可降低业务影响范围。

步骤1 登录应用运维管理 AOM控制台。

步骤2 在左侧导航栏选择“告警 > 告警规则”，单击右上角“添加规则”。

步骤3 设置告警规则。

- 规则名称：输入规则名称，例如：istio-proxy。
- 规则类型：选择“阈值规则”。
- 监控对象：单击“选择资源对象”，选择“按指标维度添加”，指标名称选择“云服务指标 > CCE > 容器 > 物理内存使用率”。

  指标维度选择“集群ID”，选择开启告警的集群ID；选择“容器名称”，容器名称选择“istio-proxy”，单击“确定”。
- 告警条件：设置统计周期、连续周期、阈值条件等触发条件参数，如下图所示，请根据具体需求设置。
- 告警方式：选择“直接告警”。

步骤4 单击“立即创建”。

创建后在规则列表中可以看到如下一行，表示创建成功。

步骤5 在告警列表中可以查看最新的活动告警信息及对应负载的详情。

当告警被触发后，活动告警列表会新增一条或多条记录。单击告警名称，在“告警对象”页签查看deploymentName、nameSpace等参数，以确定告警源自哪个sidecar资源。业务面sidecar资源限制请参考**配置sidecar资源限制**修改，istio-ingressgateway、istio-egressgateway、istio-eastwestgateway等控制面sidecar资源限制需要在CCE控制台“工作负载”页面通过升级方式来修改。

----结束

1.7.4 istio 资源管理

1.7.4.1 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。

⚠ 注意

使用YAML方式创建和修改Istio资源可能导致与控制台配置不兼容，控制台相关功能将不再开放使用。如果您确定后续将只通过YAML方式维护Istio资源，则参考本章节编辑或新建Istio资源，否则请不要在此处操作。

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 1-31 筛选 istio 资源

步骤4 单击操作列的“编辑”，在右侧页面修改相关配置，默认勾选底部提示信息，即控制台相关功能将不再开放使用，单击“确定”保存。

说明

编辑Istio资源后，具体哪些控制台功能不可用，与Istio资源类型有关。详细说明请参见**YAML配置资源处理策略**。

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 1-32 创建 istio 资源

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，默认勾选底部提示信息，即控制台相关功能将不再开放使用，单击“确定”保存。

说明

创建Istio资源后，具体哪些控制台功能不可用，与Istio资源类型有关，详细说明请参见**YAML配置资源处理策略**。

----结束

Istio 资源说明

表 1-2 Istio 资源说明

资源类型	说明
AuthorizationPolicy	用于配置授权规则。
DestinationRule	定义路由的目标服务和流量策略。VirtualService和DestinationRule是流量控制最关键的两个资源，DestinationRule定义了网格中某个Service对外提供服务的策略及规则，包括负载均衡策略、异常点监测、熔断控制、访问连接池等。
EnvoyFilter	EnvoyFilter为服务网格控制面提供更强大的扩展能力，使Envoy中Filter Chain具备自定义配置的能力。
Gateway	Gateway定义了所有HTTP/TCP流量进入网格或者从网格中出站的统一入口和出口，它描述了一组对外公开的端口、协议、负载均衡以及SNI配置。
PeerAuthentication	Istio的认证策略包含PeerAuthentication和RequestAuthentication，PeerAuthentication策略用于配置服务通信的mTLS模式。

资源类型	说明
RequestAuthentication	Istio的认证策略包含PeerAuthentication和RequestAuthentication，RequestAuthentication策略用于配置服务的请求身份验证方法。
ServiceEntry	用于注册外部服务到网格内，并对其流量进行管理。
Sidecar	对Sidecar代理进行整体设置。
VirtualService	用于网格内路由的设置。VirtualService和DestinationRule是流量控制最关键的两个资源，在VirtualService中定义了一组路由规则，当流量进入时，逐个规则进行匹配，直到匹配成功后将流量转发给指定的路由地址。
WorkloadEntry	用来将虚拟机（VM）或者裸金属（Bare Metal）进行抽象，使其在网格化后作为与Kubernetes中的Pod同等重要的负载，具备流量管理、安全管理、可视化等能力。

1.7.4.2 YAML 配置资源处理策略

通过控制台和“Istio资源管理”中的YAML方式均可以创建Istio资源，为了避免两个入口的配置相冲突，建议您：

- 控制台创建的资源，在控制台维护
- YAML创建的资源，YAML方式维护

如果控制台创建的资源通过YAML方式修改，将会导致控制台对应功能不可用（例如，YAML配置VirtualService资源后，控制台对应服务的流量治理、灰度发布将不可用）。

处理策略

在“网格配置 > Istio资源管理”页面编辑或创建服务关联的Istio资源时，页面底部会默认勾选“控制台相关功能不开放使用”，当保持勾选状态并单击“确定”后，配置生效，但该服务的控制台部分功能将不再开放使用，后续将只能通过YAML方式维护。

⚠ 注意

如果不勾选“控制台相关功能不开放使用”，可能会导致修改的配置与部分控制台功能冲突，请谨慎选择。

表 1-3 勾选“控制台相关功能不开放使用”对服务的控制台功能的影响

现象	可能原因
在“服务管理”页面，目标服务的“安全”按钮置灰，不可选择	通过YAML方式修改了AuthorizationPolicy资源
在“服务管理”页面，目标服务的“流量治理”按钮置灰，不可选择	通过YAML方式修改了DestinationRule或VirtualService资源

现象	可能原因
在“服务管理”页面，目标服务的“发布版本”按钮置灰，不可选择	通过YAML方式修改了DestinationRule或VirtualService资源
在“灰度发布”页面创建灰度发布任务时，目标服务置灰，不可选择	通过YAML方式修改了DestinationRule或VirtualService资源
在“网关管理”页面，目标网关的添加路由按钮置灰，不可选择	通过YAML方式修改了Gateway资源
在“网关管理”页面添加路由时，目标服务置灰，不可选择	通过YAML方式修改了DestinationRule或VirtualService资源

1.7.4.3 IstioOperator 配置资源处理策略

Istio采用Istio Operator安装的场景下，有时需要更新被Istio Operator管理的组件（包括istiod、istio-ingressgateway、istio-egressgateway）的工作负载，例如：升级网格版本、扩容istio-ingressgateway实例数等。更新这些工作负载可在CCE控制台“工作负载”页面修改。

处理策略

为了避免多个入口的配置相冲突，以及确保Istio各工作负载持续稳定运行，ASM 1.8.6及以上版本采取如下策略：

- 定义工作负载的关键运行配置和非关键运行配置

  表 1-4 各资源类型下的关键运行配置

工作负载	资源类型	配置项	配置项描述	适用版本
istiod istio-ingressgateway istio-egressgateway	Deployment	spec.replicas	实例数	1.8.6及以上版本
		spec.strategy	升级策略	
		spec.template.spec.nodeSelector	调度策略	
		spec.template.spec.affinity	调度策略	
		spec.template.spec.tolerations	调度策略	
		spec.template.spec.containers.resources	资源请求和限制	

工作负载	资源类型	配置项	配置项描述	适用版本
istiod istio-ingressgateway istio-egressgateway	Deployment	spec.template.spec.containers.env	容器环境变量	1.13.9-r10、1.15.7-r3及以上小版本 1.18.7-r3及以上版本
istio-cni-node	DaemonSet	spec.updateStrategy	升级策略	1.18.5-r1及以上版本
		spec.template.spec.nodeSelector	调度策略	
		spec.template.spec.affinity	调度策略	
		spec.template.spec.tolerations	调度策略	
		spec.template.spec.containers.resources	资源请求和限制	
		spec.template.spec.containers.env	容器环境变量	1.18.7-r3及以上版本

- Istio Operator默认保持当前集群中工作负载的关键运行配置不做更新，仅支持非关键运行配置更新。
- 若需要对关键运行配置进行修改，建议用户通过CCE控制台“工作负载”页面修改，若用户有特定需求，可通过工单进行咨询。

1.7.5 升级

1.7.5.1 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。

⚠ 注意

企业版网格已下线，不再支持创建企业版网格，推荐使用基础版网格。存量的企业版网格仍可继续使用。

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
企业版	1.8.4-r1	1.8.6-r3	补丁更新（原地升级）
	1.8.4-r2	1.8.6-r3	补丁更新（原地升级）
	1.8.4-r3	1.8.6-r3	补丁更新（原地升级）
	1.8.4-r4	1.8.6-r3	补丁更新（原地升级）
	1.8.4-r5	1.8.6-r3	补丁更新（原地升级）
	1.8.6-r1	1.8.6-r3	补丁更新（原地升级）
	1.8.6-r2	1.8.6-r3	补丁更新（原地升级）
	1.6.9-r4	1.6.9-r5	补丁更新（原地升级）
基础版	1.8.x	1.18.x	先版本升级（金丝雀升级）到1.15最新版本，再升级（金丝雀升级）到1.18.x
	1.13.x	1.18.x	先版本升级（金丝雀升级）到1.15.x-rx，再升级（金丝雀升级）到1.18.x
	1.15.x-rx	1.18.x	版本升级（金丝雀升级）到1.18.x
专有版	1.3.0-rx	1.18.x	先迁移至基础版本网格，再版本升级至1.8.6-r4（金丝雀升级），再版本升级（金丝雀升级）到1.15.x最新版本，再版本升级（金丝雀升级）到1.18.x
	1.6.9-rx	1.18.x	先迁移至基础版本网格，再版本升级至1.8.6-r4（金丝雀升级），再版本升级（金丝雀升级）到1.15.x最新版本，再版本升级（金丝雀升级）到1.18.x
	1.8.4-r1	1.18.x	先迁移至基础版本网格，再版本升级（金丝雀升级）到1.15.x最新版本，再版本升级（金丝雀升级）到1.18.x

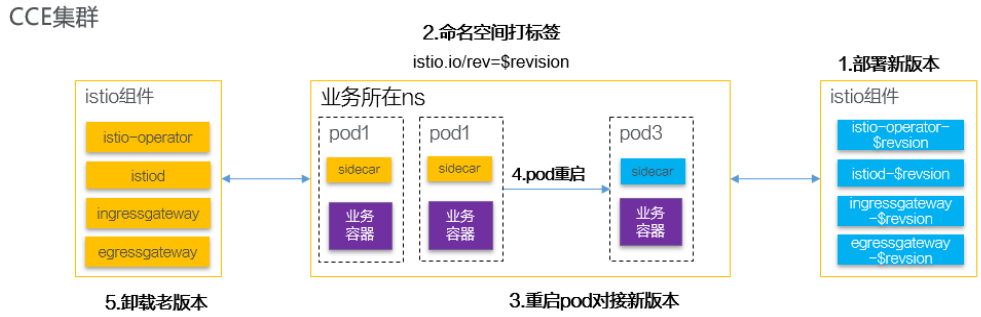
说明

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**、**1.13版本特性**、**1.15版本特性**和**1.18版本特性**。

金丝雀升级原理

ASM基础版网格支持金丝雀升级，金丝雀升级过程中将允许新老网格控制面同时存在，通过在命名空间打上版本对应的标签，可以选择一部分sidecar重启，并连接上新版本的控制面，待所有sidecar都正常连接到新控制面之后下面老版本网格的控制面。

需要注意的是，在部署新网格版本控制面时，istio-ingressgateway和istio-egressgatway的新版本也会同时部署，新老网格版本的网关将同时工作。



金丝雀升级流程

金丝雀升级流程包括升级前检查、控制面升级、数据面升级、升级后处理几个步骤，下面介绍金丝雀升级过程中的相关流程。

1. **升级前检查**

   升级前，会对集群资源、集群版本、集群状态等多方面进行检查，尽可能避免升级失败，详情请查看**升级前检查说明**。
2. **控制面升级**

   详情请查看**控制面升级说明**。
3. **数据面升级**

   详情请查看**数据面升级说明**。
4. **升级后处理**

   详情请查看**升级后处理说明**。

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。

- **基础版本升级**
  升级详情请查看**金丝雀升级**。
- **企业版补丁更新**
  单击“补丁更新”，在弹出的提示框中单击“确定”。

----结束

1.7.5.2 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
- 华为私有版本Virtual Service Delegation上线，提前为大客户提供解耦Gateway流量配置
- 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/**

1.7.5.3 1.6 版本特性

- 控制面组件合一，简化控制面安装和运维
- 社区正式版本Virtual Service Delegation更新（华为贡献特性，API和华为1.3先发版本完全相同）
- 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

- 支持控制面托管和非托管形态
- 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/**

1.7.5.4 1.8 版本特性

- 托管版本支持扁平网络多集群
- 托管版本支持非扁平网络多集群
- 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
- 支持VM类型服务治理
- 启用基于AuthorizationPolicy的新的授权策略
- 支持Istio 1.8.4、1.8.6版本
- 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/**

1.7.5.5 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/**

1.7.5.6 1.15 版本特性

- 支持Istio 1.15.7版本
- 支持v1.21、v1.23、v1.25、v1.27、v1.28 CCE Turbo集群版本
- 支持v1.21、v1.23、v1.25、v1.27、v1.28 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/**

1.7.5.7 1.18 版本特性

- 支持Istio 1.18.7版本
- 支持v1.25、v1.27 、v1.28、v1.29 、v1.30、v1.31 、v1.32 CCE Turbo集群版本

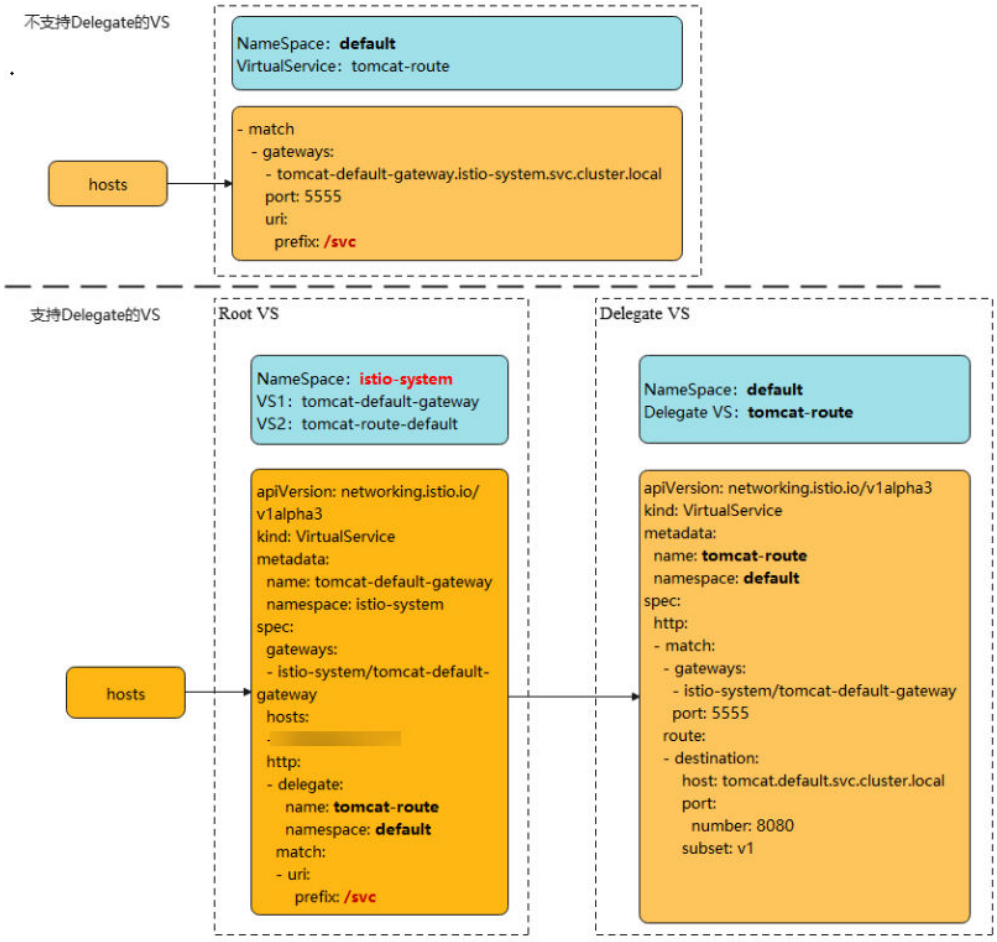
- 支持v1.25、v1.27、v1.28 、v1.29、v1.30、v1.31 、v1.32 CCE集群版本
- 支持 Kubernetes Gateway API

详细内容请参阅：**https://istio.io/latest/news/releases/1.18.x/**

1.7.5.8 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。

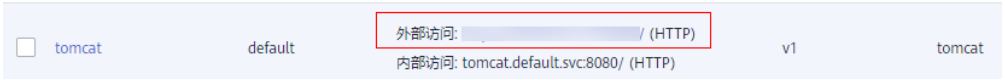
kubectl create -f tomcat-route-default.yaml

kubectl create -f tomcat-default-gateway.yaml

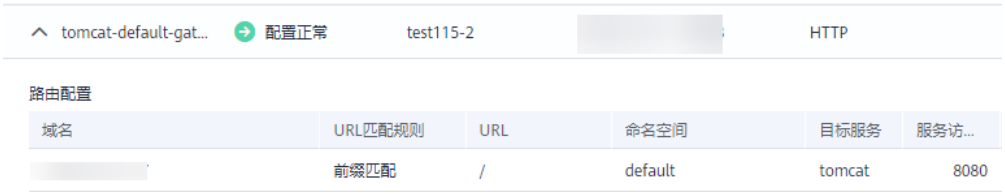
3. **kubectl -n{namespace} get vs**获取到服务的VirtualService，执行**kubectl -n{namespace} edit vs tomcat-route**修改如下：
   a. 删除spec.gateways、spec.hosts和spec.http.match.uri
   b. tomcat-default-gateway.istio-system.svc.cluster.local替换成istio-system/tomcat-default-gateway
   c. 修改apiVersion: networking.istio.io/v1alpha3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。

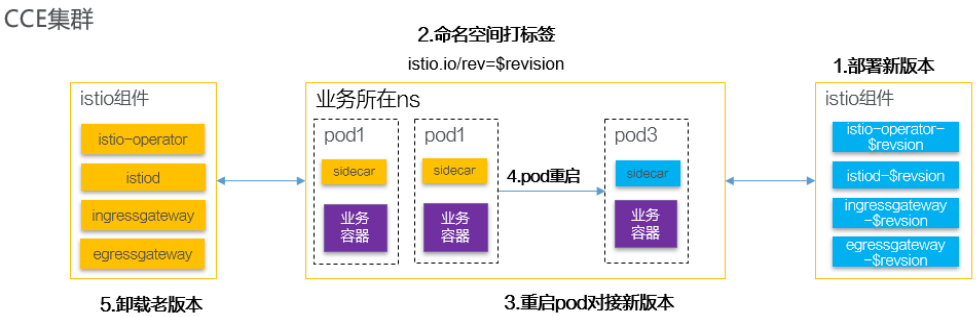


1.7.5.9 金丝雀升级

1.7.5.9.1 金丝雀升级步骤

ASM基础版网格支持金丝雀升级，金丝雀升级过程中将允许新老网格控制面同时存在，通过在命名空间打上版本对应的标签，可以选择一部分sidecar重启，并连接上新版本的控制面，待所有sidecar都正常连接到新控制面之后下线老版本网格的控制面。

需要注意的是，在部署新网格版本控制面时，istio-ingressgateway和istio-egressgateway的新版本也会同时部署，新老网格版本的网关将同时工作。



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“网格配置”，单击“升级”页签。

步骤3 选择可升级版本，单击”版本升级”。

步骤4 在弹出的提示信息后，单击“确定”。

步骤5 进入升级前检查页面，单击“执行”，进行升级前检查，详情请查看**升级前检查说明**。

步骤6 升级前检查结束后，单击“下一步”，进入控制面升级。

步骤7 单击“升级”，进行控制面升级，详情请查看**控制面升级说明**。

步骤8 控制面升级结束后，单击“下一步”，进行数据面升级。

步骤9 单击“升级”，进行数据面升级，详情请查看**数据面升级说明**。

步骤10 升级后处理，详情请查看**升级后处理说明**。

----结束

1.7.5.9.2 升级前检查说明

- 集群资源检查，检查规则如下：

  CPU：istiod+istio-ingressgateway+istio-egressgateway实例数 * cpu申请值。

  Memory: istiod+istio-ingressgateway+istio-egressgateway实例数 * 内存申请值。
- 集群版本检查。
- 集群状态检查，集群状态不可用不能进行升级。
- 网格状态检查，网格状态不可用不能进行升级。
- 废弃CRD检查（1.8及以上的版本不涉及）。

  1.3和1.6版本使用的部分CRD在1.8版本中会废弃，升级到1.8版本后将不可用，请在升级前进行处理，包括如下资源

  clusterrbacconfigs、servicerole、servicerolebinding、policy。
- Istio 网关标签检查（1.8及以上的版本不涉及）。

  Istio 网关标签（matchLabels）必须为 {app: istio-ingressgateway, istio: ingressgateway}。
- 升级前vs格式检查（1.8及以上的版本不涉及）。

  1.8及以上版本网格界面仅支持显示 delegate 格式的 VirtualService，请用户根据指南**1.3升级1.8 VirtualService支持Delegate切换**进行修改，否则升级后将看不到创建的网关路由。注：该检查项并不会导致升级失败。
- 升级前网关配置检查（1.8及以上的版本不涉及）。

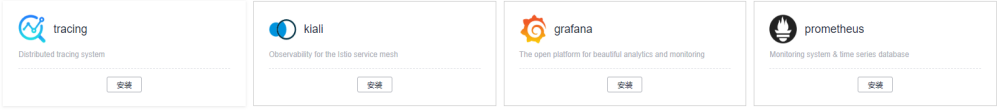
  1.8及以上版本网关监听端口需要大于1024，若您的网关使用了小于1024以及以下的端口需要进行整改。
- 升级前组件亲和性检查。

  金丝雀升级将在集群里新部署一套网格控制面和网关实例，升级前需要至少确保如下一个条件。

  a. 具有istio:master label的节点数量是 istiod实例数的两倍、所有可调度节点是 istio-ingressgateway、istio-egressgateway 实例个数最大值的两倍。

  b. istiod、istio-ingressgateway、istio-egressgateway pod反亲和性从【必须满足】改为【尽量满足】。
- 升级前插件检查（1.8及以上的版本不涉及）。

  1.3和1.6网格支持的如下插件在1.8及以上版本不再支持，升级前请确保将如下插件卸载。

- 升级前命名空间自动注入检查。

  基础版网格注入方式为命名空间注入，若您在当前已经注入pod所在的命名空间未打上注入label，则需要在升级前打上注入label。
- 升级前service selector检查。

升级前若业务service的参数selector中包含sidecarVersion的key，则需要在升级前去除该selector。

1.7.5.9.3 控制面升级说明

- 此步骤将部署灰度版本网格控制面，所有资源都将带上版本revision标签，如负载istiod-1-13-9-r5，现有业务实例sidecar不会对接到该灰度版本，升级和回退不会对原网格业务造成影响，但是需要注意若您的版本是从非金丝雀版本升级到金丝雀版本时，升级过程中会将老版本istio-operator实例数置为0。
- 控制面部署完成后会自动创建实例数为1的istio-ingressgateway-{revision} 和 istio-egressgateway-{revision} deployment，labels和原有网关实例相同。
- 若您在其他命名空间自行创建了istio-ingressgateway网关，升级过程中会自动将网关按照金丝雀的方式升级到新版本：istio-ingressgateway-{revision}-{namespace}（其中{revision} 为版本号，如1-13-9-r5，namespace为命名空间）。

注： 金丝雀版本从1.8.6-r4,1.13.9-r4, 1.15.7-r1开始支持。

1.7.5.9.4 数据面升级说明

- 单击“升级”会将所有服务实例对接到新版本控制面，您可以通过修改命名空间的标签（去除istio-injection:enabled ，加上istio.io/rev: {revision} ，其中{revision} 为版本号，如1-13-9-r5 ），并重启部分pod进行测试，若无问题，则可以全部切换。
- 1.3和1.6版本升级到1.8版本，因为证书切换，sidecar需要同时全部重启。
- 数据面升级结束后，请等待所有升级中的工作负载滚动升级完成。

1.7.5.9.5 升级后处理说明

- 请确认数据面已成功升级完成并充分观察业务，业务正常运行后再执行本步骤。
- 此步骤将卸载老版本网格控制面并释放相关资源。

⚠ 注意

一旦执行此步骤，网格将无法回退，请谨慎操作。

1.7.6 网格扩展

可观测性配置展示了当前应用服务网格的应用指标、访问日志、调用链配置；并可启用应用指标和访问日志功能。

说明

调用链只能在创建网格的时候设置启用，网格扩展页面不支持设置启用。

约束与限制

- Istio1.18之前版本的应用指标仅支持对接APM。1.18以及后续版本网格仅支持对接到AOM，同时低版本网格升级到1.18后也可以切换到AOM，为了获取更优的使用体检，建议切换到AOM。

- 目前访问日志仅Istio 1.18及以上版本支持对接到云日志服务（LTS）。若要启用访问日志，请提前在CCE集群插件中心中安装云原生日志采集插件（CCE Log-Agent）。

启用应用指标

⚠ 注意

对接AOM场景下，若要启用应用指标，请提前在CCE集群插件中心中安装云原生监控插件（kube-prometheus-stack）。

步骤1 登录应用服务网格ASM控制台。

步骤2 单击应用服务网格名称，进入应用服务网格详情页。

步骤3 单击左侧导航栏“网格配置”，选择“网格扩展”页签。

步骤4 单击应用指标 的“立即开启”按钮，选择AOM实例，单击“确定”即可。

说明

如果需要查看Istio的应用指标，可参考**AOM页面查询应用服务网格详细指标**。

----结束

启用访问日志

步骤1 登录应用服务网格ASM控制台。

步骤2 单击应用服务网格名称，进入应用服务网格详情页。

步骤3 单击左侧导航栏“网格配置”，选择“网格扩展”页签。

步骤4 单击访问日志 的“立即开启”按钮，选择需要使用的日志组，日志流，单击“确定”即可。

----结束

1.8 流量治理

1.8.1 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 1-5 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。

1.8.2 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 1-6 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	1-4294967295
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**retry policies**和**gRPC retry policies**。	-

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 1-7 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 1-8 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	1-4294967295
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）	1-4294967295
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 1-9 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	1-4294967295
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	1-4294967295
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	1-4294967295
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 1-10 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	1-4294967295

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>=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>=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1-11所述。

表 1-11 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 1-12 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 1-13 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 1-14 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。 • 延时故障：对通往组件的请求有延迟。 • 中断故障：会中断该组件的服务并返回预设状态码。	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

1.8.3 查看流量监控

操作场景

服务的流量治理页面支持查看近1小时内流量的监控数据，包括RPS、成功率和请求时延。

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”，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”，在右侧页面查看近1小时内流量的监控数据。

图 1-33 流量监控

步骤4 开启实时监控后，数据将进行每分钟动态刷新。

----结束

1.8.4 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。

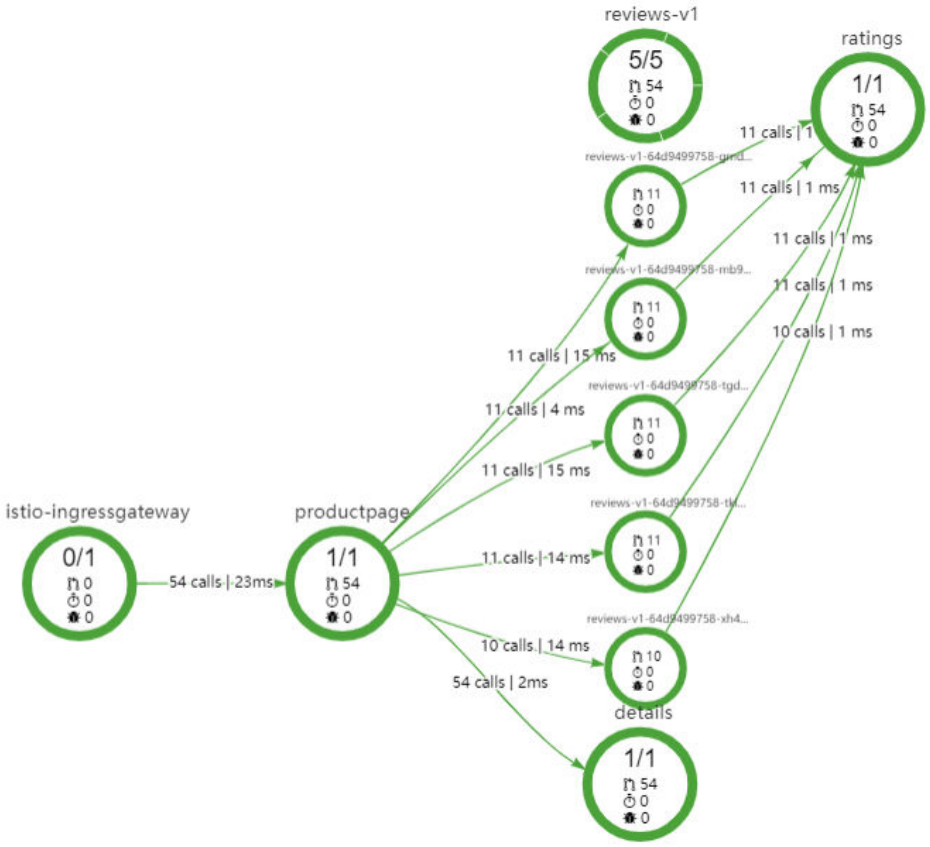
步骤2 在左侧导航栏选择“服务管理”，选择需要更改流量策略的服务，单击操作列的“流量治理”，在右侧页面进行流量策略更改。

接下来，以**Bookinfo应用**的reviews服务为例，结合APM（应用性能管理）的监控拓扑图来描述负载均衡算法更改前后的配置效果。

1. 负载均衡算法为“轮询调度”时，不断访问productpage页面，观察流量如何分发。

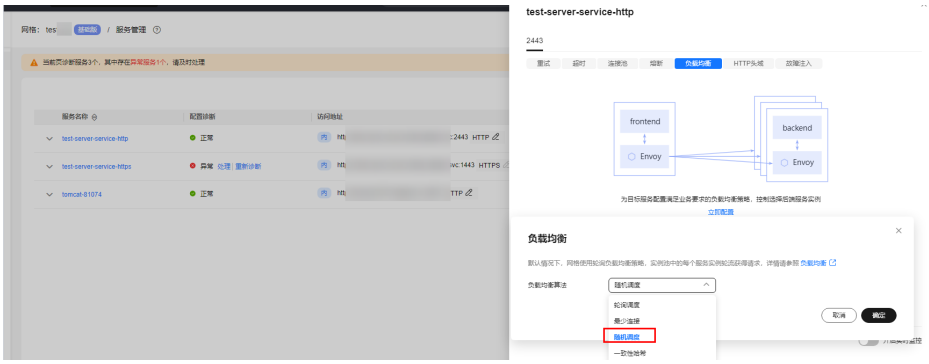
   进入“流量监控”页面，鼠标右键单击reviews服务，选择“展开”选项，这时可以看到所有实例的被分发情况。

从下图可以看出，请求依次分发给reviews的5个实例。这与负载均衡“轮询调度”算法相吻合。

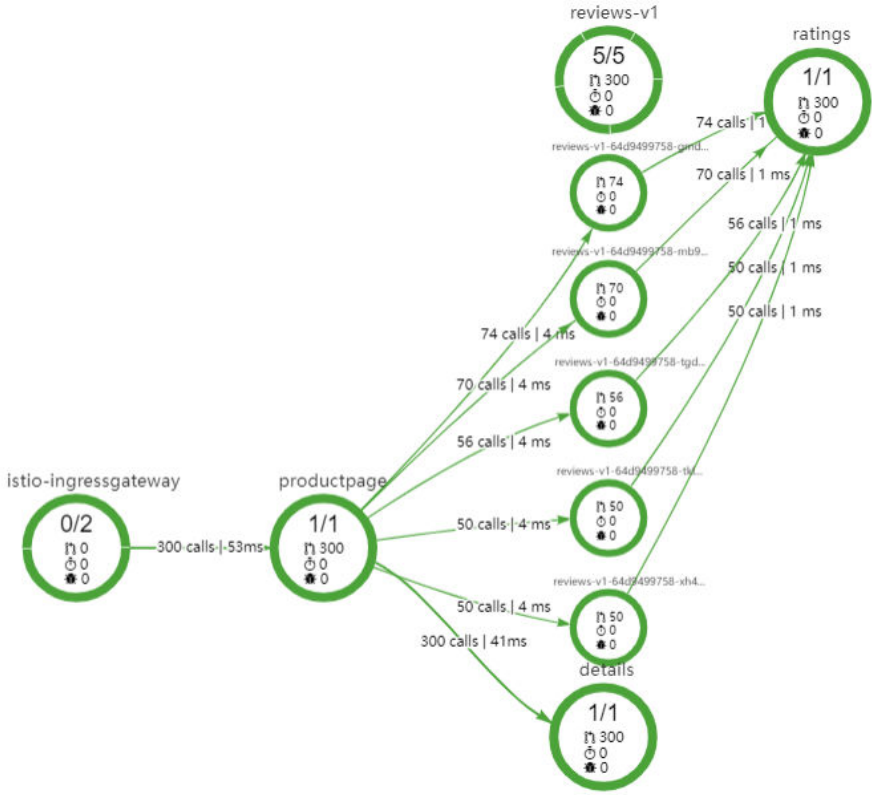


2. 在“负载均衡”页签中，单击“立即配置”，将负载均衡算法修改为“随机调度”。

图 1-34 修改负载均衡算法



3. 不断访问productpage页面，观察流量分发情况。



可以发现流量分发没有什么固定规律，各个实例的访问次数参差不齐，说明随机算法已经生效。

----结束

1.9 安全

1.9.1 配置安全策略

ASM提供的安全功能主要分为访问授权、对端认证和JWT认证，以确保服务间通信的可靠性。

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”，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“安全”，在右侧页面进行访问授权和对端认证配置。

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。

选择“访问授权”，单击“立即配置”，在弹出对话框中，单击 

，选择指定命名空间下的一个或多个服务进行访问授权设置。

对端认证

Istio通过客户端和服务端的PEP（Policy Enforcement Points）隧道实现服务实例之间的通信，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。已经注入sidecar的服务实例之间，默认通过隧道进行通信，流量会自动进行TLS加密。

选择“对端认证”页签，单击“立即配置”，在弹出对话框中选择认证策略类型。

表 1-15 参数说明

参数名称	参数说明
默认模式（UNSET）	如果父作用域配置了对端认证策略，服务将继承父作用域的配置。
宽容模式（PERMISSIVE）	请求可以不通过隧道进行传输，既可以是明文，也可以是TLS加密的密文。默认情况下，网格配置了宽容模式（PERMISSIVE）的对端认证策略。
严格模式（STRICT）	流量只能通过隧道进行传输，因为请求必须是TLS加密的密文，且必须带有客户端证书。

JWT认证

在服务网格中配置JWT（JSON Web Token）请求授权，可以实现来源认证。在接收用户请求时，该配置用于认证请求头信息中的Access Token是否可信，并授权给来源合法的请求。

说明

仅支持为HTTP协议的服务配置JWT认证。

选择“JWT认证”页签，单击“立即配置”，在弹出对话框中配置如下参数：

- 发行者：JWT的颁发者。
- 令牌受众：设置哪些服务可以使用JWT Token访问目标服务，多个受众用“,”隔开，若为空表示对访问的服务不受限制。
- jwks：JWT规则集。

JWT认证的原理及应用示例请参见**JWT认证原理**和**在ASM中对入口网关进行JWT请求认证**。

----结束

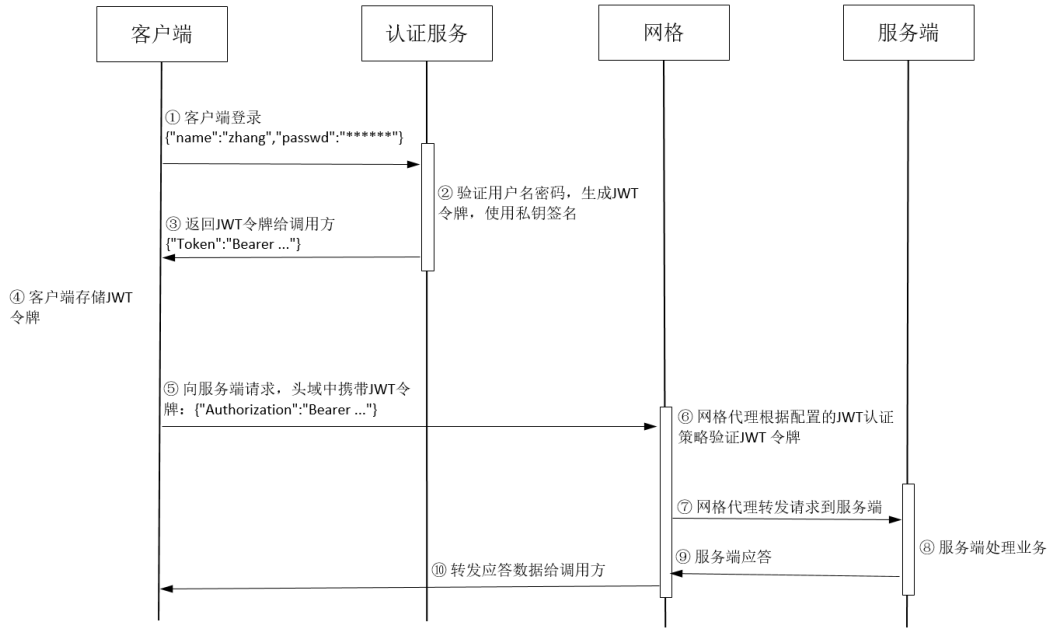
1.9.2 JWT 认证原理

JWT 认证原理

JWT（Json Web Token）是一种服务端向客户端发放令牌的认证方式。客户端用户名密码登录时，服务端会生成一个令牌返回给客户端；客户端随后在向服务端请求时只需携带这个令牌，服务端通过校验令牌来验证是否是来自合法的客户端，进而决定是否向客户端返回应答。从机制可以看到，这种基于请求中携带令牌来维护认证的客户端连接的方式解决了早期服务端存储会话的各种有状态问题。

在Istio使用中，JWT令牌生成由特定的认证服务提供，令牌验证由网格执行，彻底解耦用户业务中的认证逻辑，使应用程序专注于自身业务。基于Istio的JWT完整机制如图1-35所示。

图 1-35 Istio JWT 认证流程



① 客户端连接认证服务，提供用户名和密码；

② 认证服务验证用户名和密码，生成JWT令牌，包括用户标识和过期时间等信息，并使用认证服务的私钥签名；

③ 认证服务向客户端返回生成的JWT令牌；

④ 客户端将收到的JWT令牌存储在本端，供后续请求时使用；

⑤ 客户端在向其他服务发起请求时携带JWT令牌，无需再提供用户名、密码等信息；

⑥ 网格数据面代理拦截到流量，使用配置的公钥验证JWT令牌；

⑦ 验证通过后，网格代理将请求转发给服务端；

⑧ 服务端处理请求；

⑨ 服务端返回应答数据给网格代理；

⑩ 网格数据面代理转发应答数据给调用方。

在这个过程中，重点是第六步，原来服务端的JWT认证功能迁移到了网格代理上。网格数据面从控制面配置的认证策略中获取验证JWT令牌的公钥，可以是jwks（JSON Web Key Set）上配置的公钥，也可以是从jwksUri配置的公钥地址获取到的公钥。获得公钥后，网格代理使用该公钥对认证服务私钥签名的令牌进行验证，并解开令牌中的iss，验证是否匹配认证策略中的签发者信息。验证通过的请求发送给应用程序，验证不通过则直接拒绝，不会发送给应用程序。

JWT 结构

JWT是一个包含了特定声明的Json结构。从前面介绍的JWT认证流程第六步知道，只要验证这个Json结构本身，即可以确认请求身份，不需要查询后端服务。下面解析JWT结构从而了解如何携带这些认证信息。

JWT包含三部分：头部Header、负载Payload和签名Signature。

- 头部Header

  头部描述JWT的元数据，包括算法alg和类别typ等信息。alg描述签名算法，这样接收者可以根据对应的算法来验证签名，默认是如下所示的HS256，表示 HMAC-SHA256；typ表示令牌类型，设置为JWT，表示这是一个JWT类型的令牌。

  ```
  {
    "alg": "HS256",
    "typ": "JWT"
  }
  ```

- 负载Payload

  存放令牌的主体内容，由认证服务AuthN生成相关信息并放到令牌的负载中。重要属性包括：

  - iss：令牌发行者 issuer
  - aud：令牌受众 audience

  在JWT验证时，会校验发行者、受众信息和令牌负载中的发行者iss、受众audience是否匹配。JWT的内容本身不是加密的，所有拿到令牌的服务都可以看到令牌负载Payload中的内容，因此建议Payload里不要存放私密的信息。

- 签名Signature

  签名字段是对头部和负载的签名，确保只有特定合法的认证服务才可以发行令牌。实际使用中一般是把头部和负载分别执行Base64转换成字符串，然后使用认证服务的密钥对拼接的字符串进行签名，签名算法正是前面介绍的头域中定义的算法。

一个完整的JWT示例如下，对于头部Header和负载Payload进行签名得到Signature。

```
# Header:
{
  "alg": "RS512",
  "typ": "JWT"
}

# Payload
{
  "iss": "weather@cloudnative-istio",
  "audience": "weather@cloudnative-istio"
}

# Signature
RSASHA512(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload)
  )
```

以上结构最终输出的令牌如下，可以看到“.”分割的三个字符串分别对应JWT结构的头部、负载和签名三部分。

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjQ2ODU5ODk3MDAsInZlciI6IjIuMCIsImlhdCI6MTUzMjM4OTcwMCwiaXNzIjoid2VhdGhlckBjbG91ZG5hdGl2ZS1pc3Rpby5ib29rIiwic3ViIjoid2VhdGhlckBjbG91ZG5hdGl2ZS1pc3Rpby5ib29rIn0.SEp-8qiMwI45BuBgQPH-wTHvOYxcE_jPI0wqOxEpauw
```

1.9.3 在 ASM 中对入口网关进行 JWT 请求认证

本文介绍如何在ASM中对入口网关进行JWT请求认证，确保用户在通过入口网关访问服务时，必须带有可信赖的Access Token。

准备工作

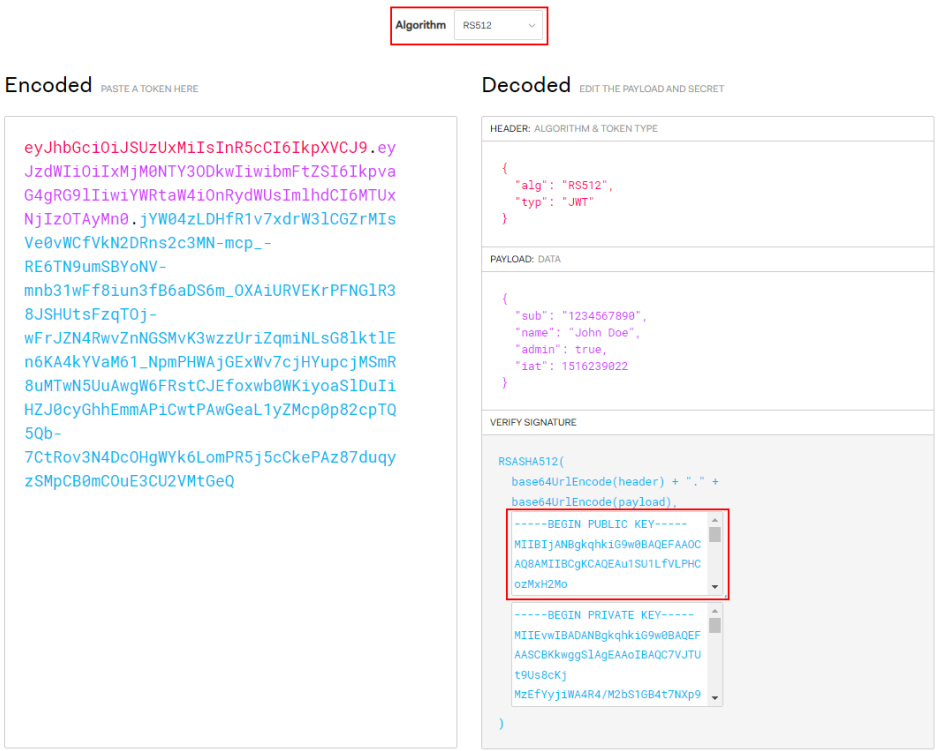
1. 已创建一个1.13或1.15或1.18版本网格。
2. 网格中有诊断通过的httpbin服务，镜像为httpbin，端口协议为http，端口号为80。
3. 网格中已为httpbin服务创建可访问的网关。

创建 JWT 认证

步骤1 创建JWK。

1. 访问**JWT工具网站**，选择“Algorithm”为“RS512”，获取公钥（PUBLIC KEY）。

图 1-36 生成公钥



2. 在**JWK to PEM Convertor online**工具中选中“PEM-to-JWK (RSA Only)”，输入上一步获取的公钥，单击“submit”，将公钥转换为JWK。

图 1-37 将公钥转换为 JWK

{"kty":"RSA","e":"AQAB","kid":"a78641b9-d81e-4241-
b35a-71726c3fa053","n":"u1SU1LfVLPHCozMxH2Mo4lgOEePzNm0tRgeLezV6ffAt0gunVTLw7onLRnrq0
_IzW7yWR7QkrmBL7jTKEn5u-qKhbwKfBstIs-
bMY2Zkp18gnTxKLxoS2tFczGkPLPgizskuemMghRniWaoLcyehkd3qqGElvW_VDL5AaWTg0nLVkjRo9z-40
RQzuVaE8AkAFmxZzow3x-
VJYKdjykkJ0iT9wCS0DRTXu269V264Vf_3jvredZiKRkgwlL9xNAwxXFg0x_XFw005UWVRIkdgcKWTjpBP2d
PwVZ4WWC-9aGVd-Gyn1o0CLelf4rEjGoXbAAEgAqeGUxrcIlbjXfbcmw"}

步骤2 创建JWT认证。

1. 登录**应用服务网格控制台**，单击网格名称，进入网格详情页面。
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。
3. 选择httpbin服务，单击操作列的“安全”，在右侧页面选择“JWT认证”页签，单击“立即配置”，在弹出的“JWT认证”页面填写如下信息：
   - 发行者：JWT的颁发者，本文设置为“test”。
   - 令牌受众：JWT受众列表，设置哪些服务可以使用JWT Token访问目标服务，本文设置为“ASM”。
   - jwks：设置JWT信息，本文设置为{"keys": [**步骤1**创建的JWK]}，例如**步骤1**创建的JWK为{"kty":"RSA","e":"AQAB","kid":"a78641b9-d81e-4241-b35a-71726c3****"}，则jwks为{"keys": [{"kty":"RSA","e":"AQAB","kid":"a78641b9-d81e-4241-b35a-71726c3****"}]}。

图 1-38 创建 JWT 认证

4. 单击“确定”完成创建。

----结束

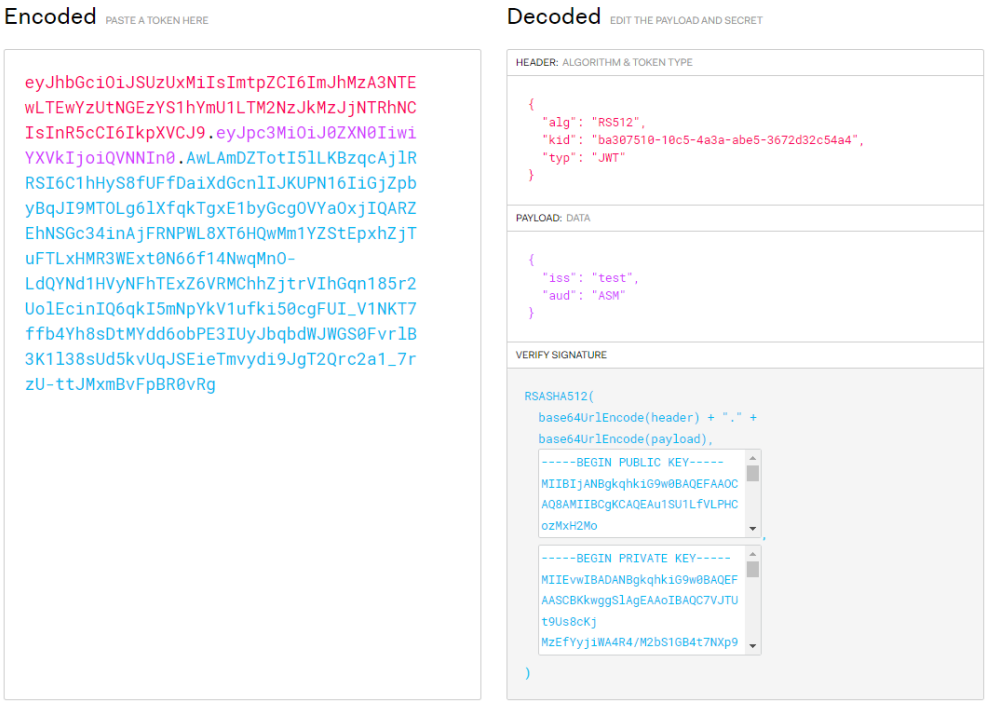
验证 JWT 认证是否生效

步骤1 使用**JWT工具**将JWT请求信息编码成JWT Token。

在“Decoded”区域输入以下JWT请求信息，在“Encode”区域将看到自动转换后的JWT Token。

- HEADER：设置alg为“RS512”，输入**步骤1**创建的JWK中的kid，设置type为“JWT”。
- PAYLOAD：设置iss为“test”，aud为“ASM”，确保与**步骤2**中配置的发行者、令牌受众保持一致。
- VERIFY SIGNATURE：与**步骤1.1**中的公钥保持一致。

图 1-39 创建 JWT Token



步骤2 通过入口网关访问httpbin服务。

1. 执行以下命令，带**步骤1**创建的JWT Token访问服务。

   TOKEN=*步骤1创建的JWT Token*

   curl -I -H "Authorization: Bearer $TOKEN" http://*{httpbin服务的外部访问地址}/*

   预期输出：

   ```
   HTTP/1.1 200 OK
   server: istio-envoy
   date: Wed, 21 Sep 2022 03:11:48 GMT
   ```

2. 执行以下命令，带无效的JWT Token访问服务。

   curl -I -H "Authorization: Bearer invalidToken" http://*{httpbin服务的外部访问地址}/*

   预期输出：

   ```
   HTTP/1.1 401 Unauthorized
   www-authenticate: Bearer realm="http://***.***.***.***:***/", error="invalid_token"
   content-length: 145
   content-type: text/plain
   date: Wed, 21 Sep 2022 03:12:54 GMT
   server: istio-envoy
   x-envoy-upstream-service-time: 19
   ```

3. 修改**步骤2**中创建的JWT认证，将令牌受众置空（表示对访问的服务不受限制），然后执行以下命令，带**步骤1**创建的JWT Token访问服务。

   curl -I -H "Authorization: Bearer $TOKEN" http://*{httpbin服务的外部访问地址}/*

   预期输出：

   ```
   HTTP/1.1 200 OK
   server: istio-envoy
   date: Wed, 21 Sep 2022 03:20:07 GMT
   ```

4. 执行以下命令，不带JWT Token访问服务。

   curl -I http://*{httpbin服务的外部访问地址}/*

   预期输出：

   ```
   HTTP/1.1 403 Forbidden
   content-length: 85
   content-type: text/plain
   date: Wed, 21 Sep 2022 03:29:31 GMT
   server: istio-envoy
   x-envoy-upstream-service-time: 6
   ```

根据以上结果，可以看到带有正确的JWT Token的请求访问服务成功，带有错误的JWT Token或者不带JWT Token的请求访问服务失败，说明请求身份认证生效。

----结束

1.10 监控中心

1.10.1 流量监控

购买网格时，基础配置 中的“ 可观测性配置”选择对接APM服务，则可以通过流量监控来监测流量概况、组件运行状态、调用链等信息，并在系统业务异常时快速定位到问题点。

适用场景

适用于1.15之前版本的网格，且APM服务在需要监控网格的region已开通。

约束与限制

- 流量监控记录的是组件过去24小时内的数据，所以当部分组件删除后并不能及时的在拓扑图上消失，而是会仍然存在一段时间。
- 企业版网格多集群场景下，只支持集群内部服务之间的流量监控，暂时不支持跨集群服务之间的流量监控。

查看流量监控情况

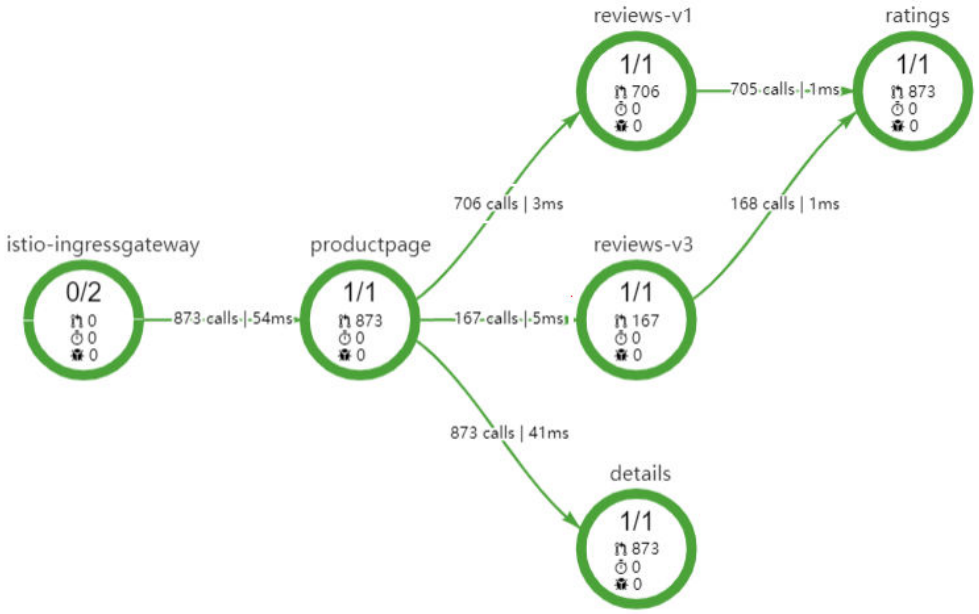
步骤1 登录**应用服务网格控制台**，单击已对接APM的服务网格名称进入详情页面。

步骤2 在左侧导航栏中选择“监控中心”-“流量监控”。

步骤3 查看整个系统的监控情况。

拓扑图中显示了系统处于运行中和未就绪的应用数量。

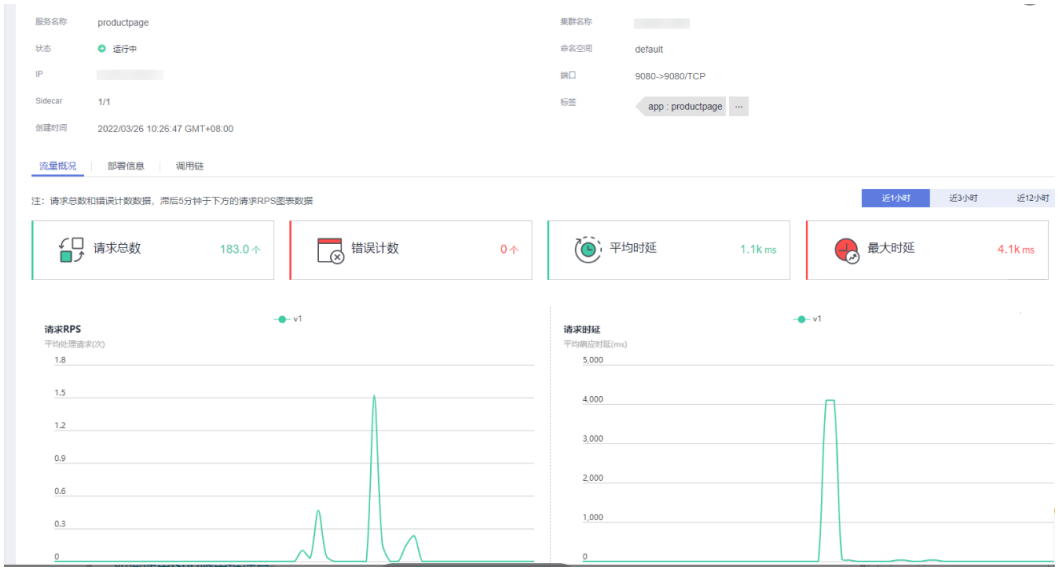
图 1-40 流量拓扑



步骤4 查看某个组件的监控情况。

选择服务网格、集群及命名空间，单击拓扑图上的组件，进入组件监控详情页面，如图1-41所示。

图 1-41 流量监控详情页



其中，

- 流量概况：展示了应用详细的流量信息，包含请求总数、错误计数、平均时延、最大时延等。
- 部署信息：显示了组件内的所有实例状态及信息。
- 调用链：通过时间维度和组件版本来查看调用情况，支持使用高级搜索中的选项进行精准搜索。使用istio做服务治理时，无需在微服务代码中进行调用链埋点。

但微服务代码在接收和发送请求时需要传递调用链相关的Header信息，才能构造成完整的调用链路，详情请参见**如何使用Istio调用链埋点**。

----结束

如何使用 Istio 调用链埋点

Header信息包括如下内容，更多关于调用链的信息请参见**https://istio.io/docs/tasks/telemetry/distributed-tracing/**。

- x-request-id
- x-b3-traceid
- x-b3-spanid
- x-b3-parentspanid
- x-b3-sampled
- x-b3-flags
- x-ot-span-context

下面以一个典型的例子来说明如何在接收和发送请求时，通过修改代码来传递调用链相关Header信息。

- **Python代码示例**

  服务接收端在处理请求时，从请求端解析相关Header。在调用后端请求时，传递Trace相关的header。

```
def getForwardHeaders(request):
    headers = {}

if 'user' in session:
    headers['end-user'] = session['user']

incoming_headers = ['x-request-id',
    'x-b3-traceid',
    'x-b3-spanid',
    'x-b3-parentspanid',
    'x-b3-sampled',
    'x-b3-flags',
    'x-ot-span-context'
]

return headers


@ app.route('/productpage')
def front():
    product_id = 0# TODO: replace
default value
headers = getForwardHeaders(request)
user = session.get('user', '')
product = getProduct(product_id)
detailsStatus, details = getProductDetails(product_id, headers)
reviewsStatus, reviews = getProductReviews(product_id, headers)
return render_template(
    'productpage.html',
    detailsStatus = detailsStatus,
    reviewsStatus = reviewsStatus,
    product = product,
    details = details,
    reviews = reviews,
    user = user)
```

- **Java代码示例**

  在java Rest接口上除了解析一般业务参数外，需要从header中解析trace相关信息。同样在调用下一个服务时传递该header信息。

```
@GET
@Path("/reviews/{productId}")
public Response bookReviewsById(@PathParam("productId") int productId,
@HeaderParam("end-user") String user,
@HeaderParam("x-request-id") String xreq,
@HeaderParam("x-b3-traceid") String xtraceid,
@HeaderParam("x-b3-spanid") String xspanid,
@HeaderParam("x-b3-parentspanid") String xparentspanid,
@HeaderParam("x-b3-sampled") String xsampled,
@HeaderParam("x-b3-flags") String xflags,
@HeaderParam("x-ot-span-context") String xotspan) {
    int starsReviewer1 = -1;
    int starsReviewer2 = -1;
    if (ratings_enabled) {
        JsonObject ratingsResponse = getRatings(Integer.toString(productId), user, xreq, xtraceid,
xspanid, xparentspanid, xsampled, xflags, xotspan);
        if (ratingsResponse != null) {
            if (ratingsResponse.containsKey("ratings")) {
                JsonObject ratings = ratingsResponse.getJsonObject("ratings");
                if (ratings.containsKey("Reviewer1")){
                    starsReviewer1 = ratings.getint("Reviewer1");
                }
                if (ratings.containsKey("Reviewer2")){
                    starsReviewer2 = ratings.getint("Reviewer2");
                }
            }
        }
    }
    String jsonResStr = getJsonResponse(Integer.toString(productId), starsReviewer1, starsReviewer2);
    return Response.ok().type(MediaType.APPLICATION_JSON).entity(jsonResStr).build();
}
}
```

1.10.2 访问日志

1.10.2.1 访问日志

服务网格提供了访问日志查询能力，可对网格中数据面所有Proxy的AccessLog进行采集。本文介绍如何查看采集的访问日志。

须知

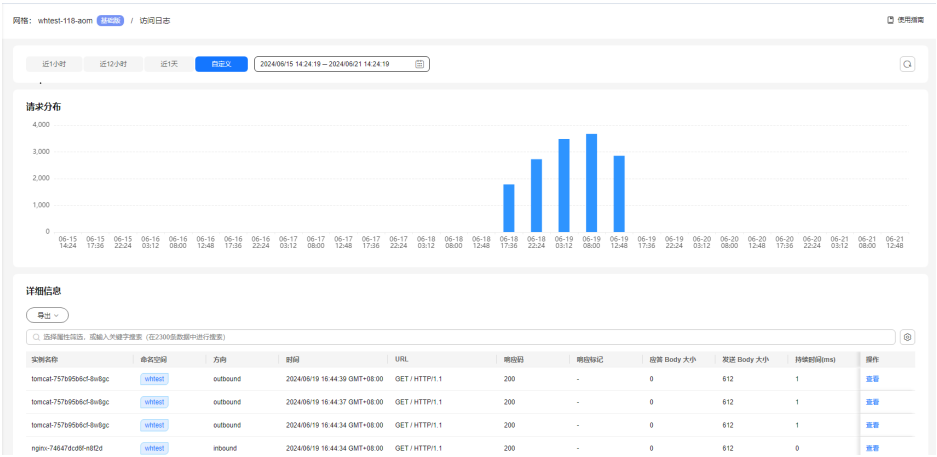
网格需已“启用访问日志”能力，如何开启请参考**设置可观测性配置**。

操作步骤

步骤1 登录应用服务网格ASM控制台。

步骤2 单击服务网格名称，进入服务网格详情页。

步骤3 在左侧导航栏，单击“监控中心>访问日志”，进入访问日志详情页。



步骤4 单击“请求分布”页签上，分布时间按钮，可以观测到不同时间的请求数量详情。

注意

自定义查询时间范围不能大于31天。

步骤5 单击“详细信息”页签的输入框，根据属性类型可以通过指定属性的关键字搜索、过滤，查看详情。例如：实例名称。

步骤6 单击“详细信息”页签的输入框后的设置按钮，添加自定义显示列，单击“确定”，完成自定义显示列添加。

步骤7 单击“详细信息”页签的中的“导出”按钮，可将访问日志的详情导出到本地进行查看。

----结束

1.10.2.2 访问日志各字段解读

sidercar会在标准输出中打印访问日志，istio日志中每个字段的含义解读如下。由于不同istio版本的访问日志格式及其字段的内容存在差异，下面分1.15及以下版本和1.18及以上版本两大类进行说明。

1.15 及以下版本

1.15版本及以下采用Istio的默认格式。以下面的istio日志为例，字段含义见下表。

```
[%START_TIME%] \"%REQ(:METHOD)% %REQ(X-ENVOY-ORIGINAL-PATH?:PATH)% %PROTOCOL%\"
%RESPONSE_CODE% %RESPONSE_FLAGS% %RESPONSE_CODE_DETAILS%
%CONNECTION_TERMINATION_DETAILS% \"%UPSTREAM_TRANSPORT_FAILURE_REASON%\"
%BYTES_RECEIVED% %BYTES_SENT% %DURATION% %RESP(X-ENVOY-UPSTREAM-SERVICE-TIME)%
\"%REQ(X-FORWARDED-FOR)%\" \"%REQ(USER-AGENT)%\" \"%REQ(X-REQUEST-ID)%\"
\"%REQ(:AUTHORITY)%\" \"%UPSTREAM_HOST%\" %UPSTREAM_CLUSTER%
%UPSTREAM_LOCAL_ADDRESS% %DOWNSTREAM_LOCAL_ADDRESS%
%DOWNSTREAM_REMOTE_ADDRESS% %REQUESTED_SERVER_NAME% %ROUTE_NAME%\n
```

表 1-16 1.15 及以下版本 istio 日志字段解读

日志标记	HTTP场景含义	TCP场景含义	UDP场景含义	样例	备注
[%START_TIME%]	请求开始时间，毫秒	Downstream发起连接的时间	UDP proxy会话开始的时间	[2020-11-25T21:26:18.409Z]	-
\"%REQ(:METHOD)% %REQ(X-ENVOY-ORIGINAL-PATH?:PATH)% %PROTOCOL%\"	请求方法 请求PATH 请求协议	\	\	"GET /status/418 HTTP/1.1"	-
%RESPONSE_CODE%	响应码	\	\	418	-
%RESPONSE_FLAGS%	响应或连接的其他信息	响应或连接的其他信息	\	UH	参考**响应标记解读**。
%RESPONSE_CODE_DETAILS%	响应码详情：返回对象和原因	\	\	via_upstream	-
%CONNECTION_TERMINATION_DETAILS%	请求被Envoy中止的L4层原因	同HTTP	\	xxx	-
\"%UPSTREAM_TRANSPORT_FAILURE_REASON%\"	传输层失败原因（TLS等）	\	\	"TLSV1_ALERT_UNKNOWN_CA"	-
%BYTES_RECEIVED%	收到的Body体大小	收到的数据包大小	\	0	-
%BYTES_SENT%	发送的Body体大小	发送的数据包大小	\	135	-
%DURATION%	从开始到发送最后1个Bytes的时间（毫秒）	整个TCP连接的时间（毫秒）	\	4	-

日志标记	HTTP场景含义	TCP场景含义	UDP场景含义	样例	备注
%RESP(X-ENVOY-UPSTREAM-SERVICE-TIME)%	响应头X-ENVOY-UPSTREAM-SERVICE-TIME的内容	\	\	4	该头代表UPSTREAM处理请求和Envoy与UPSTREAM之间的网络延迟
\"%REQ(X-FORWARDED-FOR)%\"	请求头X-FORWARDED-FOR的内容	\	\	"10.44.x.x"	-
\"%REQ(USER-AGENT)%\"	请求头USER-AGEN的内容	\	\	"curl/7.73.0-DEV"	-
\"%REQ(X-REQUEST-ID)%\"	请求头X-REQUEST-ID的内容	\	\	"84961386-6d84-929d-98bd-c5aee93b5c88"	-
\"%REQ(:AUTHORITY)%\"	请求头AUTHORITY的内容	\	\	"httpbin:8000"	-
\"%UPSTREAM_HOST%\"	UPSTREAM_HOST的主要地址	同HTTP	同HTTP	"10.44.x.x:80"	-
%UPSTREAM_CLUSTER%	UPSTREAM_HOST所属的Cluster	同HTTP	同HTTP	outbound|8000||httpbin.foo.svc.cluster.local	-
%UPSTREAM_LOCAL_ADDRESS%	连接UPSTREAM_HOST所使用的本地地址	同HTTP	同HTTP	10.44.x.x:37652	-
%DOWNSTREAM_LOCAL_ADDRESS%	DOWNSTREAM连接的本地地址	同HTTP	同HTTP	10.0.x.x:8000	-

日志标记	HTTP场景含义	TCP场景含义	UDP场景含义	样例	备注
%DOWNSTREAM_REMOTE_ADDRESS%	DOWNSTREAM连接的对端地址	同HTTP	同HTTP	10.44.x.x:46520	-
%REQUESTED_SERVER_NAME%	SSL连接的SNI	同HTTP	\	xxx	-
%ROUTE_NAME%	路由的名称	同HTTP	\	default	-

1.18 及以后版本

1.18的访问日志提供JSON格式的内容，以如下JSON格式为例。

```
{
   "start_time": "%START_TIME%",
   "route_name": "%ROUTE_NAME%",
   "method": "%REQ(:METHOD)%",
   "path": "%REQ(X-ENVOY-ORIGINAL-PATH?:PATH)%",
   "protocol": "%PROTOCOL%",
   "response_code": "%RESPONSE_CODE%",
   "response_flags": "%RESPONSE_FLAGS%",
   "response_code_details": "%RESPONSE_CODE_DETAILS%",
   "connection_termination_details": "%CONNECTION_TERMINATION_DETAILS%",
   "bytes_received": "%BYTES_RECEIVED%",
   "bytes_sent": "%BYTES_SENT%",
   "duration": "%DURATION%",
   "upstream_service_time": "%RESP(X-ENVOY-UPSTREAM-SERVICE-TIME)%",
   "x_forwarded_for": "%REQ(X-FORWARDED-FOR)%",
   "user_agent": "%REQ(USER-AGENT)%",
   "request_id": "%REQ(X-REQUEST-ID)%",
   "authority": "%REQ(:AUTHORITY)%",
   "upstream_host": "%UPSTREAM_HOST%",
   "upstream_cluster": "%UPSTREAM_CLUSTER%",
   "upstream_local_address": "%UPSTREAM_LOCAL_ADDRESS%",
   "downstream_local_address": "%DOWNSTREAM_LOCAL_ADDRESS%",
   "downstream_remote_address": "%DOWNSTREAM_REMOTE_ADDRESS%",
   "requested_server_name": "%REQUESTED_SERVER_NAME%",
   "upstream_transport_failure_reason": "%UPSTREAM_TRANSPORT_FAILURE_REASON%",
   "pod_name": "%ENVIRONMENT(POD_NAME)%",
   "pod_namespace": "%ENVIRONMENT(POD_NAMESPACE)%",
   "mesh_id": "%ENVIRONMENT(ISTIO_META_ASM_MESH_ID)%",
   "cluster_id": "%ENVIRONMENT(ISTIO_META_ASM_CLUSTER_ID)%",
}
```

其中日志标记与表1含义相同，新增字段解释见表2。

表 1-17 1.18 及以上版本 istio 日志新增字段解读

Log operator	解释	样例
pod_name	产生日志的pod名称	istio-ingressgateway-1-15-7-r2-599d4cf747-ngcfd

Log operator	解释	样例
pod_namespace	产生日志的pod的namespace	istio-system
mesh_id	网格ID	84961386-6d84-929d-98bd-c5aee93b5c88
cluster_id	集群ID	84961386-6d84-929d-98bd-c5aee93b5c88

了解更多

Istio官方日志介绍：*Istio / Envoy Access Logs*

Envoy官方日志字段含义：*Access logging — envoy 1.31.0-dev-3d906a documentation (envoyproxy.io)*

1.10.2.3 访问日志的响应标记解读

UH（没有健康后端）

含义

UH（**NoHealthyUpstream**）表示上游服务没有健康的后端实例。

典型现象

目标服务的后端实例都不可用，如构造将目标服务的实例数设置为0。

典型日志

客户端日志。

```
[2023-08-1910750:46.616Z]"GET/HP/1.1" 503 UH no_healty_upstream-"0190-"5 -"curl/7.52.1" "25e82276
 -6d3e 481d-9007-c1a3404hf5a9""nginx.accesseg" outbound|80 v1 ngin.accesdog.svc-chster.local[redacted]:8010.66.0.2450552--
```

应对建议

检查目标服务的负载配置，确认服务的实例均正常运行。

DC（下游连接终止）

含义

DC(**DownstreamConnectionTermination**)表示下游连接终止。

若服务客户端因为各种原因在收到应答前中断连接，则在访问日志中一般会记录本次请求的结果为DC，大部分时候应答体的大小是0。

典型现象

客户端在访问目标服务时，在得到应答前，在客户端断开连接。或者客户端因为设置了请求超时，从客户端断开了连接。

典型日志

客户端日志。

```
[2023-08-18T11:31:40.069Z]"GET/HTTP/1.1" 0 DC downstream_remote_disconnect -"-" 0 0
1999 - "-" "curl/7.52.1" "afe165f1-27ab-447e-823d-b5t50103t197" "ngiw_external"
9090" outbound|9999||nginx_external "58540 9999
41660--
```

应对建议

一般无需特殊处理。大部分情况下因为服务端耗时较长导致客户端在一定时间后断开了连接。这时候一般考虑优化目标服务，快速返回应答。

UF（上游连接失败）

含义

UF（UpstreamConnectionFailure)表示上游连接失败。

典型场景

目标服务的服务端口不通。如客户端通过错误的端口访问目标服务时，会导致服务访问失败，客户端代理的Outbound日志会记录503UF。

目标服务的服务实例端口不通，会导致服务访问失败，同时服务端Inbound日志会记录503UF。

典型日志

服务端口错误的客户端出流量日志。

```
[2023-08-19T08:10:17.447Z]"GET/HTTP/1.1" 503 UF
upstream_reset_before_response_started{connection_failure} - "-" 09110001 - "-"
"curl/7.52.1" "819fb0d7-1ae1-4689-b10a-394a0a53546" "nginx_accesslog"
":80 PassthroughCluster - 80 60926 -allow any
```

服务实例端口错误的服务端入流量日志。

```
[2023-08-19T08:38:56.207Z]"GET/HTIP/1.1" 503 UF
upstream_reset_before_response_started{connection_failure, delayed_connect_error:_111) -
"delayed_connect_error:_111" 0 145 0 - "-" "curl/7.52.1" "1bd7e9c8-0881-41b2-a964-
525cad938d00" "nginx_accesslog" "388" inbound |888|| - :888
57376 outbound.80.v1.nginx_accesslog_svc_cluster.local default
```

应对建议

检查服务定义，确保服务端口和服务端口定义正确。客户端通过正确的端口访问目标服务。

URX（超过重试次数）

含义：

URX (UpstreamRetryLimitExceeded)表示超过了HTTP的请求重试阈值或者TCP的重连阈值的访问被拒绝。

典型场景

网格中对目标服务配置了重试次数N（或者没有配置Istio默认2次失败重试），在调用服务端失败次数超过了阈值，客户端的出流量日志中会记录URX。

典型日志

客户端出流量日志。

```
[2023-08-19T08:38:56.203Z]"GET /HTTP/1.1" 503 URX via_upstream  -"-" 01455151"  -"
"curl/7.52.1" "1bd7e9c8-088141b2-a964-525cad938dX0" "nginx_accesslog" "          :888"
outbound|80|v1 |nginx_accesslog_svc_cluster.local          :57388          :80
          :37236 -
```

应对建议

提高服务的成功率，保证服务在有限重试后能正常的被访问。

UPE（上游协议错误）

含义：

UPE (UpstreamProtocolError)表示上游服务协议错误。

典型场景

在网格中定义的服务的协议和实际服务的协议不一致时，当服务访问时，客户端会得到502协议错误的响应。同时服务端的入流量日志会记录502 UPE。

典型日志

客户端出流量日志。

```
[2023-08-19T09:42:38.460Z]"GET /HTTP/1.1" 502 - via_upstream - "-"087 15 15"    -"
"curl/7.52.1" "56f62238-c1064a75-a848-429a36489142" "nginx_accesslog" "          :80"
outbound |80|v1|nginx_accesslog_svc_cluster.local          :43276          :80
          :51868 -
```

服务端入流量日志。

```
[2023-08-19T09:42:38.464Z]"GET/HTTP/2" 502 UPE
upstream_reset_before_response_started{protocol_error} - "_" 0 87 0 - "-" "curl/7.52.1"
"56f62238-c106-4a75-a848-42a36489142" nginx_accesslog" "          :80" inbound|80||
          :45649          270
outbound.80.v1.nginx_accesslog_svc_cluster.local default
```

应对建议

在服务中正确的定义服务端口的应用协议。Istio中读取Service中端口名中的协议信息，或AppProtocol 来判定服务的应用协议，确保字段上协议配置正确。

DPE（下游协议错误）

含义：

DPE (DownstreamProtocolError) 表示下游协议错误。

典型场景

下游客户端通过一个错误的协议访问目标服务。

典型日志

客户端出流量日志。

```
[2023-08-21T1356:45.757Z]"---" 0---"-" 25 162 53038-" "_" "_" "_" "_"          :80"
PassthroughCluster          :45964          :80          43958--
```

服务端入流量日志。

```
[2023-08-21T13:57:37.7927]"--HTTP/1.1" 400 DPE http1_codec_error-"-" 0110-_" "_" "_" "_
" "_"--          10.66.0.1:19532--
```

应对建议

客户端使用正确的协议访问目标服务。

NC（没有上游集群）

含义：

NC(NoClusterFound) 表示没有上游集群，即在网格流量路由中定义的目标服务后端不存在。

典型场景

流量策略中定义的路由后端在网格中没有定义。如分流策略中流量发送给V2标识的服务子集，但是DestinationRule中并没有定义该版本标识的服务子集。

典型日志

客户端日志。

```
[2023-08-19T10:09:13.864Z]"GET/HTTP/1.1* 503 NC cluster_not _foumnd - "-"000    - "-"
"cur/7.52.1" "1f8f95a5-5139-4790-9d7e-80d28fda24b7" "nginx_accesslog""  _"  _  _
          :80          :51674--
```

应对建议

检查VirtualService和DestinationRule中定义的后端一致，保证VirtualService中引用的服务子集在DestinationRule中正确定义。

NR（没有匹配路由）

含义：

NR(NoRouteFound)表示没有匹配的路由来处理请求的流量，一般伴随“404”状态码。

典型场景

实际的访问流量不匹配VirtualService中定义的路由匹配条件，因而没有找到匹配的路由处理流量。

典型日志

客户端出流量日志。


```
[2023-08-19T11:13:06.484Z]"GET/HTTP/1.1" 404 NR route_not_found -"-" 000    -"-"
"curl/7.52.1" "d9e4d779-ee2f4dab-8546-46d61e789d35" "nginx_accesslog" "     _"   -  -
            :80          35016--
```

应对建议

客户端的流量满足路由中定义的流量特征，保证所有请求都有服务端定义的路由处理。

DI（延时故障注入）

含义：

DI(DelayInjected)表示请求中注入了一个延时故障。

典型场景

在VirtualService中配置了延时故障注入时，会发生服务请求延时，在访问日志中会记录DI的应答标记。

典型日志

客户端出流量日志。

```
[2023-08-20T03:21:03.863Z]"GET/HTTP/1.1" 200 DI via_upstream-"-" 0615100000"
"cur/7.52.1" "3a52e6e3-df38-47af-912d a21d65de4d79* "nginx_accesslog" "          80"
outbound |80|v1 |nginx_accesslog-svc_cluster.local          60450          :80
          :42038
```

服务端入流量日志。

```
[2023-08-20T03:21:13.863Z]"GET/HTTP/1.1" 200    - via_upstream - "-061500"  -"
"curl/7.52.1" "3a52e6e3-d3847af-912d a21d65de4d79 "nginx_accesslog" "          :80"
inbound|80|| :49911          :80          :60450
outbound_.80_.v1_.nginx_accesslog.svc_cluster.local default
```

应对建议

访问日志中记录了人为注入的故障，无需特殊处理。

FI（故障注入错误）

含义：

FI(FaultInjected) 表示故障注入错误。

典型场景

通过VirtualService给目标服务注入了一个特定状态码的故障。

典型日志

客户端出流量日志。

```
[2023-08-20T14:38:38.690Z]"GET/HTIP/1.1" 418 Fl fault_filter_abort  -"-"0180    -"-"
"curl/7.52.1- "1f085510-06de-439a-8600-8oc2c1e3sss8ssccd" "nginx_accesslog""     -"
outbound |80 |v1 |nginx_accesslog-svc_cluster.local -          :80          3616--
```

应对建议

访问日志中记录了人为注入的故障，无需特殊处理。

UT（上游请求超时）

含义：

UT(UpstreamRequestTimeout)表示上游请求超时，一般伴随返回“504”的HTTP状态码。

典型场景

VirtualService中给目标服务配置了超时时间，在配置的超时时间后，客户端代理自动超时，取消请求。

典型日志

客户端出流量日志。

```
[2023-08-20T15:00:52.250Z]"GET/HTTP/1.1" 504 UT reponse_timeout -"-" 0 24 3000 "-"
"curl/7.52.1" "c27e6f6d-9ad8-dcke-ddee-0ddasd9sds9dsd"  "nginx_external"
"          5:9090" outbound|9999||nginw_external          :40924          :9999
          :40444--
```

应对建议

用户为了保证目标服务快速失败，配置了服务访问超时，超时会返回该应答标记，无需特殊处理。

RL（触发服务限流）

含义：

RL (RateLimited)表示触发服务限流。一般会伴随返回“429”的HTTP状态码。

典型场景

在网格中配置了本地限流或者全局限流策略，若在单位时间内请求数超过配置的阈值，则触发限流。

典型日志

限流策略作用在服务端入流量的场景：

客户端出流量日志。

```
2023-08-21T03:01:28.024Z]"GET/HTTP/1.1" 429 - via_upstream - ""018 11 10" -   _"
"curl/7.52.1" "38fO4fb8-269d-4bf8-89c46e18461d38fS" "nginx_accesslog" "          80"
outbound |80 |v1 |nginx_accesslog-svc_cluster.local          :33554          :80
          :44902 --
```

服务端入流量日志。

```
[2023-08-21T03:01:28.025Z]"GET/HTTP/1.1" 429 - local_rate_limited - "-" 0180 - "-"
"curl/7.52.1" "38fO4fb8-269d-4bf8-89c46e18461d38f5" "nginx_accesslog" " -"inbound|80||-
          :80          boumd_.80_.v1_.nginx_accesslog.cluster.local-
```

限流策略作用在客户端出流量的场景：

```
[2023-08-21T09:39:00.237Z]"GET/HTTP/1.1" 429 - local_rate_limited - "-" 0180 - "-"
"curl/7.52.1" "dc9e2396-c2c8-4f87-8994-ff4a7f8c8061" "nginx_accesslog" "      -"
outbound |80 |v1 |nginx_accesslog.svc_cluster.local -                80          :57578--
```

应对建议

用户配置限流策略保护目标服务，无需特殊处理。

1.10.3 应用拓扑

应用拓扑呈现了网格内所有服务的调用关系，增强了网格的可观测性能力，提升了网格应用管理的易用性。本节介绍如何查看网格应用拓扑。

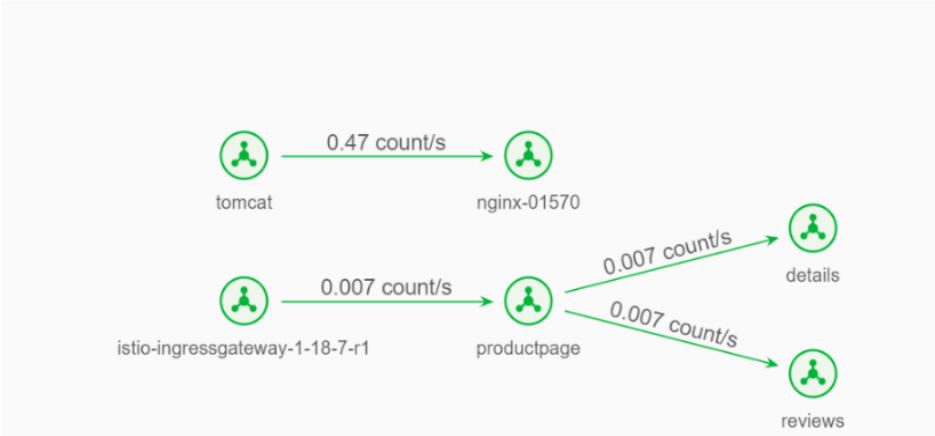
须知

网格须开通AOM服务，否则无法查看应用拓扑。

操作步骤

步骤1 登录ASM控制台，单击服务网格名称，进入服务网格详情页。

步骤2 在左侧导航栏，单击“监控中心>应用拓扑”，进入应用拓扑详情页，本页显示了整个网格内所有服务的应用拓扑。



----结束

须知

- 应用拓扑图中的连线颜色代表了当前连接的健康状况，连接展示什么颜色由错误率的值决定。当错误率小于1%时连线呈绿色；当错误率在[1%~10%]范围内连接呈黄色；当错误率大于10%连接呈红色。
- 单击拓扑中的服务节点，可以查看当前服务在所选时间内的指标数据；单击拓扑中某条连接，可以查看当前连接在所选时间内的指标数据。
- 自定义查询时间范围不能大于7天。

2 用户指南（旧版）

2.1 什么是应用服务网格

应用服务网格（Application Service Mesh，简称ASM）是华为云基于开源Istio推出的服务网格平台，它深度、无缝对接了华为云的企业级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。

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，其功能包括负载均衡、熔断、限流等多种治理能力。ASM内置金丝雀、蓝绿多种灰度发布流程，提供一站式自动化的发布管理。ASM基于无侵入的监控数据采集，提供实时流量拓扑、调用链等服务性能监控和运行诊断，构建全景的服务运行视图。

更多应用服务网格ASM介绍请参见**产品介绍**。

2.2 启用应用服务网格

启用Istio将为您的服务以无侵入的方式提供灵活的服务治理能力。您只需在CCE集群中开启应用服务网格功能，即可实现灰度发布、流量管理、熔断、监控、拓扑、调用链等丰富的服务治理能力。

前提条件

已创建CCE混合集群，如果未创建，请参照**购买CCE集群**创建。

使用须知

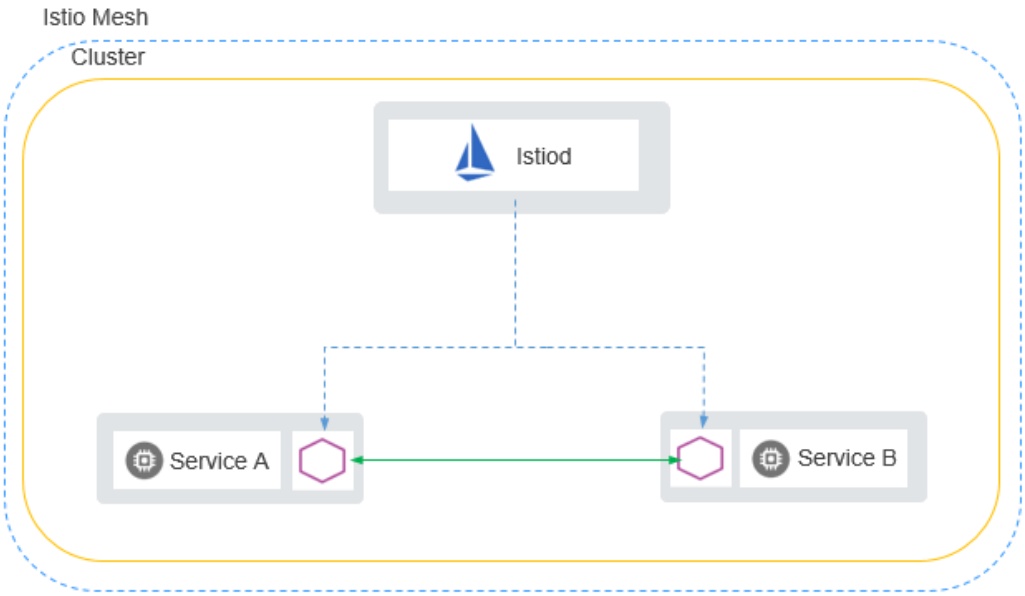
- Istio服务网格依赖集群coreDNS的域名解析能力，启用Istio服务网格前请确保集群拥有足够资源，且coreDNS组件运行正常。
- 启用Istio时，需要开通node节点所在安全组的入方向7443端口规则，用于sidecar自动注入回调。如果您使用CCE创建的默认安全组，此端口会自动开通。如果您自建安全组规则，请手动开通7443端口，以确保Istio自动注入能力正常。

背景信息

专有网格

在用户集群中安装网格的控制面组件，对集群内的服务进行非侵入式的治理、遥测和安全等管理。支持Istio1.6版本和Istio1.8版本，只能够对一个集群进行管理。

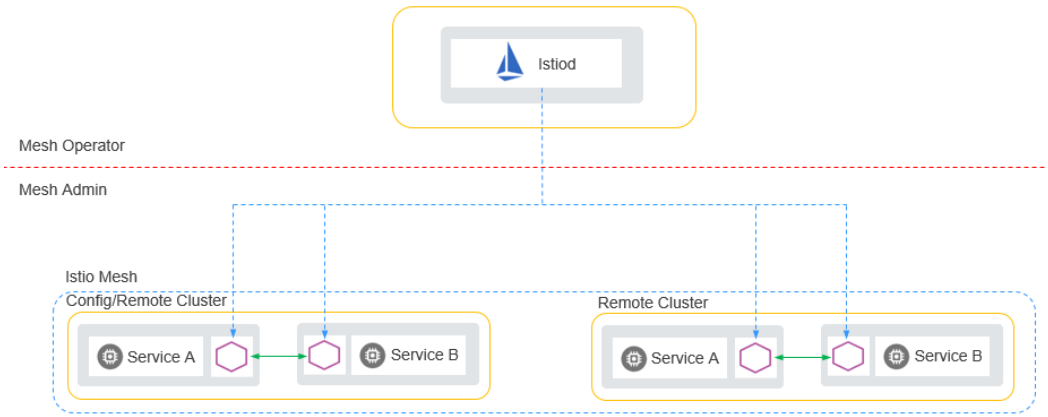
图 2-1 专有网格



托管网格

托管网格将网格的控制面组件从用户集群中分离，部署到一个独立的控制面集群中，简化了用户运维负担和资源消耗，用户只需要基于网格进行服务管理。托管网格还能够对 N（N>=1)个集群进行管理，支持服务跨集群通信。

图 2-2 托管网格

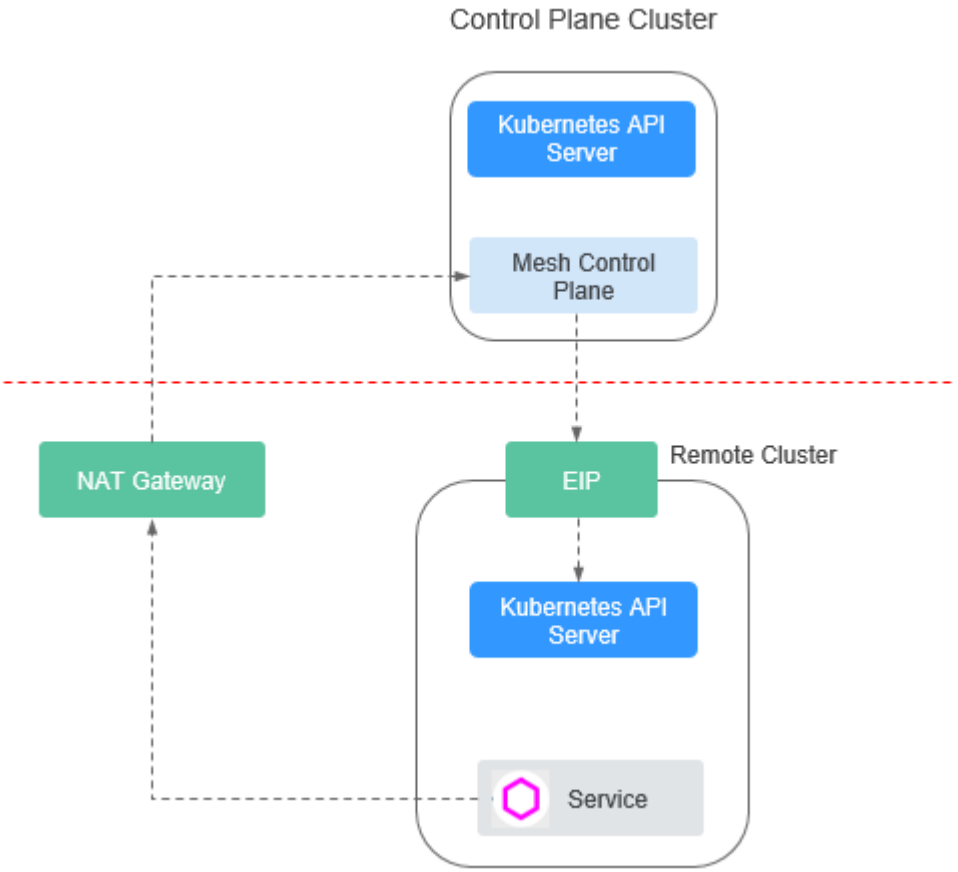


托管网格支持用户集群通过公网连接和私网连接两种方式连接网格控制面。

- 公网连接主要用于跨region添加集群。

  通过公网连接网格控制面，用户需要为集群绑定公网弹性IP（EIP），因为网格的控制面需要访问用户集群的kube-apiserver服务。还需要为集群所在的VPC创建公网NAT网关，因为服务的envoy组件需要通过NAT网关连接网格的控制面。

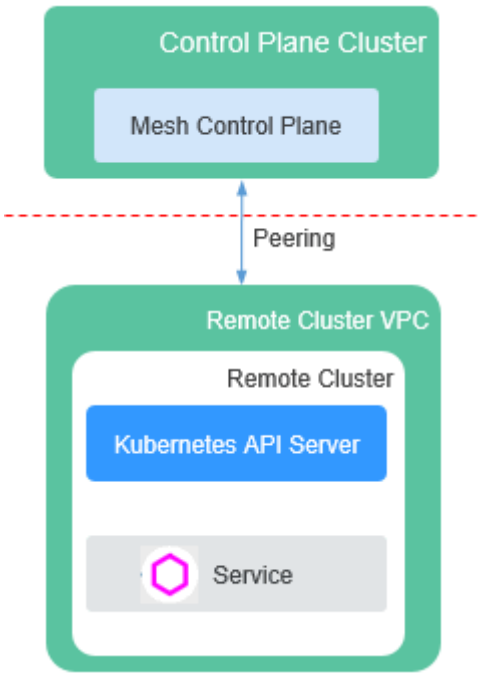
图 2-3 公网连接网格控制面



- 私网连接用于对接华为云region内部的集群。

  私网连接网格控制面则是利用VPC间的对等连接功能，打通了不同VPC之间的网络隔离。在创建网格时，要提前规划控制面的网段。

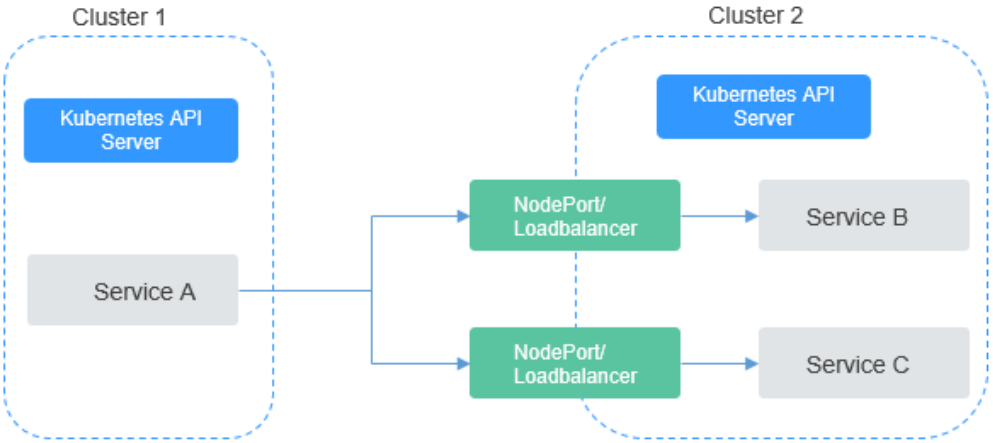
图 2-4 私网连接网格控制面



传统跨集群访问

用户需要创建NodePort或Loadbalancer服务、绑定ELB、配置转发端口，而且对每一个需要跨集群访问的服务都要执行此操作。除此之外，用户还需要维护各个服务的对外访问配置。

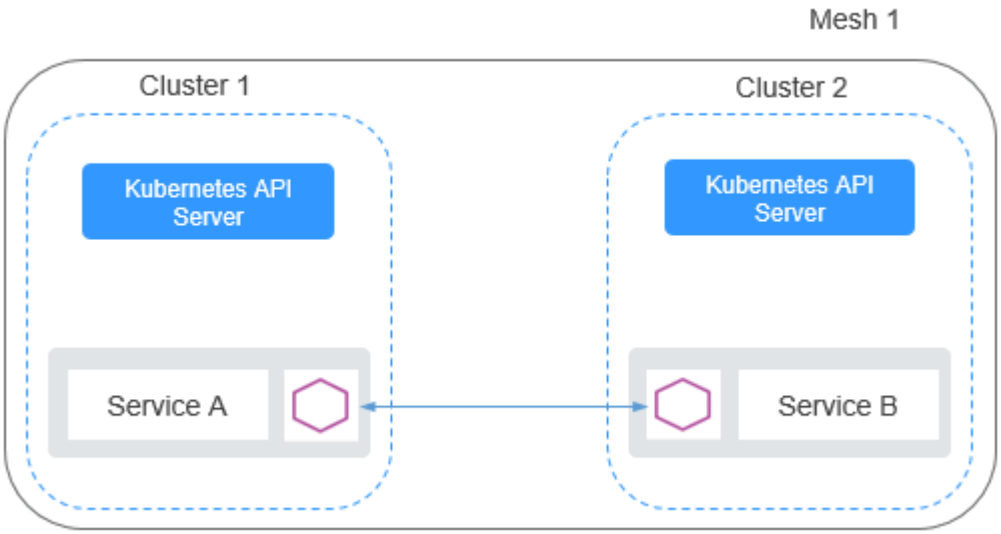
图 2-5 传统跨集群访问



ASM跨集群访问

ASM托管网格提供的服务跨集群访问能力，不需要用户进行复杂的配置，只需要将集群和服务加入网格，加入网格的任何服务之间都是能够相互访问的，不管服务属于哪个集群。

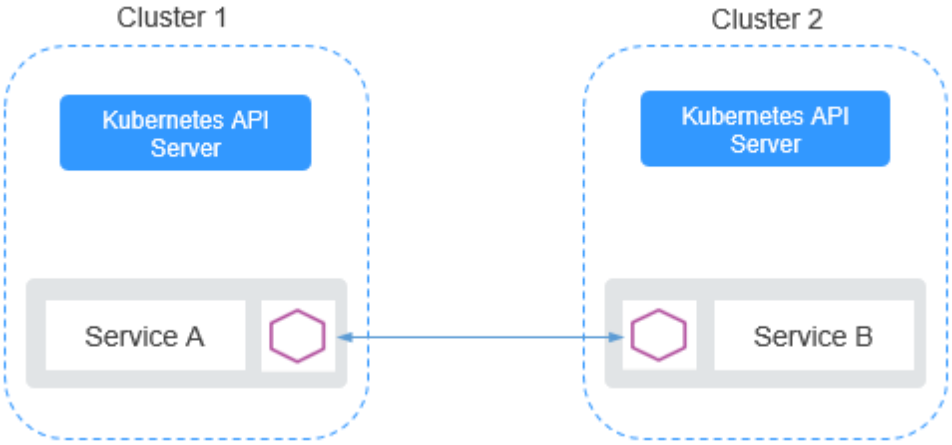
图 2-6 ASM 跨集群访问



扁平网络

是指两个集群的实例（Pod）能够通过Pod IP互相访问。这对集群的网络模型有比较高的要求，使用的限制也比较多。首先，集群必须使用“容器对接ENI”的网络模型，才能够实现实例（Pod）之间通过IP访问。其次，多个集群要处于同一VPC，或多个集群的VPC通过其他方式（对等连接等）打通。而且，集群的网络要有统一的规划，多个集群的子网网段、容器网段、服务网段都不能冲突。

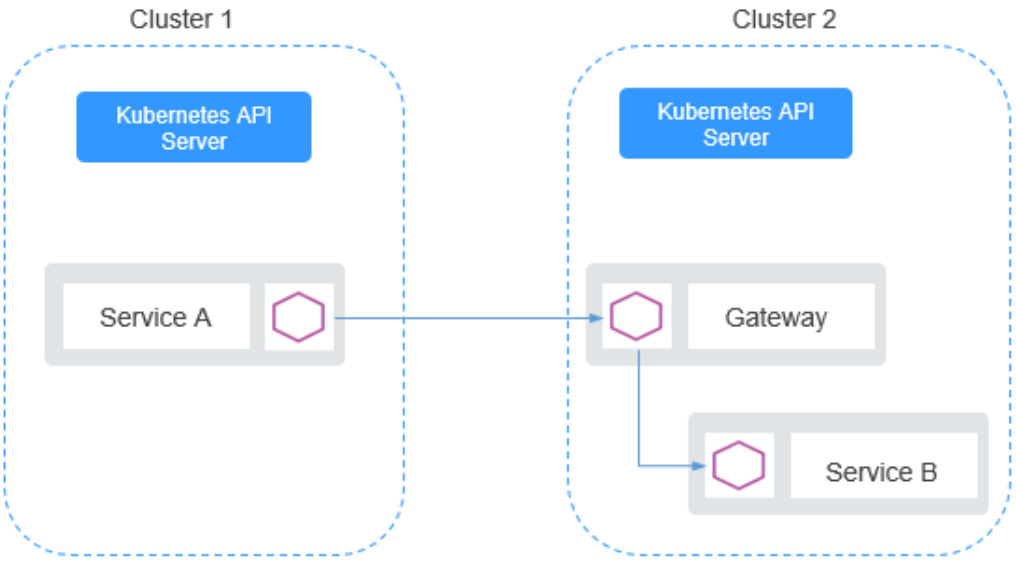
图 2-7 扁平网络



非扁平网络

是指两个集群内部实例（Pod）之间不能互相访问，必须通过一个对外网关转发。相比于扁平网络，非扁平网络具有更好的适用性。它对用户集群的网络模型没有特别的要求，用户集群只需要将Gateway地址暴露出来，供其他集群访问。但是因为使用集中的流量入口，性能瓶颈主要集中在网关上，而且多一次的转发，会对性能产生一定的影响。

图 2-8 非扁平网络



购买托管网格

网格控制面完全托管，全方位简化用户运维负担和资源消耗。用户只需基于网格进行服务管理即可。可以管理多云混合云场景的多集群，支持容器、虚拟机、物理机等多种基础设施的服务。

步骤1 登录**应用服务网格控制台**，在“总览”页单击“购买网格”进入购买Istio服务网格页面。

步骤2 网格类型选择“托管网格”。

步骤3 设置托管网格参数。

- 计费模式

  当前可支持包年/包月和按需计费模式。包年/包月创建后不能删除，如需停止使用，请到费用中心执行退订操作。

- 区域

  指在同一区域下，电力、网络隔离的物理区域，可用区之间内网互通，不同可用区之间物理隔离。如果您需要提高工作负载的高可靠性，建议您将云服务器创建在不同的可用区。

- 版本

  指定购买的Istio版本。

- 网格名称

  新建服务网格的名称，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。

- 网格管理规模

  服务网格管理的实例规模。

  说明

  当前服务网格支持管理的最大pod实例数量，请根据业务需求选择，如需支持5000实例以上，请联系管理员。

- 网格控制面网段

  说明

  此参数在网格创建后不可更改，请谨慎选择。

  网格控制面网段的掩码固定为22位，推荐使用子网10.253.0.0/22、172.31.0.0/22。建议从10.2.0.0/16~10.246.0.0/16，10.248.0.0/16~10.255.0.0/16，172.17.0.0/16~172.31.0.0/16覆盖范围中选取一个掩码为22的子网，不推荐使用192.168.0.0/16、172.16.0.0/16、 10.0.0.0/16、10.1.0.0/16、10.247.0.0/16覆盖范围内的22位掩码子网，因为很容易与用户集群环境中网段冲突。

- 购买时长

  网格购买时长，购买的时间越长越优惠。

  说明

  - 仅包年/包月的计费模式可设置购买时长，按需计费模式不支持。
  - 按需计费的托管网格购买成功后即开始计费。
  - 按需计费的托管网格治理不足20实例时，按20实例数收取费用，超过20实例时，按实际实例数收取费用。

步骤4 设置完成后，单击“下一步”。确认订单无误后，单击“提交”。

启用时间预计需要5分钟，您可以单击请求提交页面的“返回Istio管理”或“前往CCE集群管理”查看Istio运行状态。

说明

购买后，会进行启用Istio的操作，期间会操作如下资源：

- 创建一个Helm应用编排release对象，作为Istio控制面的资源 。
- 开通ECS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。
- 为该集群的default的命名空间打上inject=true标签。

步骤5 查看Istio服务网格是否启用成功。

在应用服务网格控制台，单击左侧导航栏中的“网格管理”，如果网格状态为“运行中”表明启用Istio成功。

图 2-9 Istio 服务网格启用成功

----结束

添加集群到托管网格

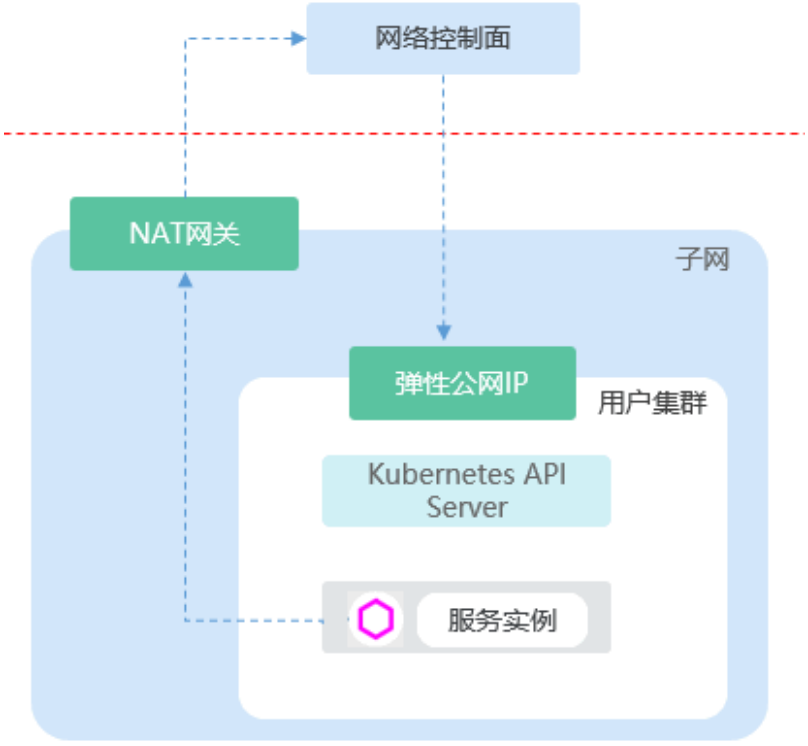
托管网格支持对多个集群进行管理，且支持服务跨集群通信。

步骤1 登录**应用服务网格控制台**，在左侧导航栏中选择“网格管理”。

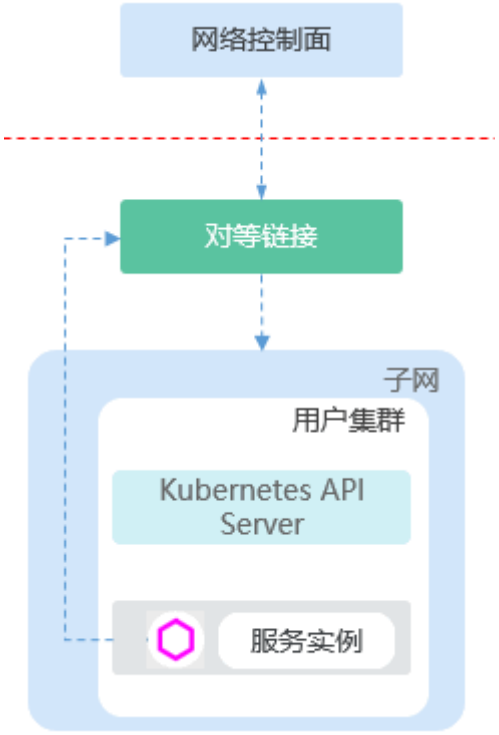
步骤2 在右侧页面，单击对应网格中的“操作 > 添加集群”。

步骤3 选择集群，设置集群信息。

- **选择集群**
  - 目前支持v1.15.x及以上版本的集群加入托管网格。
  - 同一虚拟私有云的集群只能加入同一个网格。
  - 为了满足高可用的要求，集群需要至少包含两个可用节点，每个节点至少保证有2U4G的可用资源。
  - 如果实例（Pod）需要跨集群通信，集群需要使用ENI网络模型，且集群之间网络互通（建议集群处于同一VPC内）。
  - 集群的服务网段、容器网段不能和网格内已有集群的服务网段、容器网段冲突。
  - 如果集群和网格内的已有集群处于不同的VPC，集群的子网网段也不能冲突。
- **选择连接方式**
  - 公网访问：需要为集群绑定公网弹性IP（EIP），因为网格的控制面需要访问用户集群的kube-apiserver服务。同时还需要为集群所在的VPC创建公网NAT网关，因为服务的envoy组件需要通过NAT网关连接网格的控制面。



- 私网访问：利用VPC间的对等连接功能，打通了不同VPC之间的网络隔离。



步骤4 设置完成后，单击“确定”。

添加集群大约需要一分钟，请耐心等待。添加完成后，单击“返回网格详情”页面可查看到添加的集群信息。

----结束

购买专有网格

在集群中一键启用，自动在集群中安装网格控制面，对集群内服务进行非侵入的治理、遥测和安全等管理。兼容Kubernetes和Istio生态。

步骤1 登录**应用服务网格控制台**，在“总览”页单击“购买网格”进入购买Istio服务网格页面。

步骤2 网格类型选择“专有网格”。

步骤3 设置服务选择参数。

- **计费模式**

  支持“按需计费”和“包年/包月”模式。包年/包月模式的节点创建后不能删除，如需停止使用，请到**费用中心**执行退订操作。

- **当前区域**

  指节点实例所在的物理位置。请就近选择靠近您业务的区域，可减少网络时延，提高访问速度；不同区域的云服务产品之间内网互不相通。

- **集群**

  选择要启用Istio服务网格功能的集群。

- **版本**

  指定购买的Istio版本。

- **网格管理规模**

  说明

  - 当前服务网格支持管理的最大pod实例数量，请根据业务需求选择，如需支持5000实例以上，请联系管理员。
  - 按需计费模式，当前可享受20实例的体验。

- **购买时长**

  说明

  - 仅包年/包月的计费模式可设置购买时长，按需计费模式不支持。
  - 按需计费的托管网格当前可体验20实例。
  - 按需计费的托管网格治理不足20实例时，按20实例数收取费用，超过20实例时，按实际实例数收取费用。

步骤4 设置完成后，单击“下一步”，配置“创建节点”参数。

- **高可用**

  Istio启用后，高可用模式与普通模式不可切换，请根据实际使用情况选择。

  - 是：高可用模式要求独享节点个数不少于2，用于Istio控制面多实例部署，多实例能大大的提高Istio控制面可靠性，推荐使用。
  - 否：非高可用模式为单点模式，Istio控制面组件均为单实例，无法提供更高的可靠性保证。

- **节点配置**

  此处为节点配额提示，如需申请更多配额，请单击申请扩大配额。

  - **可用区&规格**

    指在同一区域下，电力、网络隔离的物理区域，可用区之间内网互通，不同可用区之间物理隔离。如果您需要提高工作负载的高可靠性，建议您将云服务器创建在不同的可用区。

    选择所需的规格。为确保Istio控制面运行稳定，推荐您使用 8U 16G 或以上规格控制面节点，详情请参见**规格推荐**。

- **登录方式**

  支持密码和密钥两种方式。

  - 选择“密码”：用户名默认为“root”，请输入登录节点的密码，并确认密码。

    登录节点时需要使用该密码，请妥善管理密码，系统无法获取您设置的密码内容。

  - 选择“密钥对”：在选项框中选择用于登录本节点的密钥对，并单击勾选确认信息。

    密钥对用于远程登录节点时的身份认证。如果没有密钥对，可单击选项框右侧的“创建密钥对”来新建，创建密钥对操作步骤请参见**创建密钥对**。

步骤5 设置完成后，单击“下一步”。确认订单无误后，单击“提交”。

启用时间预计需要5分钟，您可以单击请求提交页面的“返回Istio管理”或“前往CCE集群管理”查看Istio运行状态。

说明

购买后，会进行启用Istio的操作，期间会操作如下资源：

- 如果为高可用，则创建需要至少2个ECS节点，作为Istio控制面运行节点。
- 创建一个Helm应用编排release对象，作为Istio控制面的资源 。
- 开通ECS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。
- 为该集群的default的命名空间打上istio-injection=enabled标签。

步骤6 查看Istio服务网格是否启用成功。

在应用服务网格控制台，单击左侧导航栏中的“网格管理”，如果网格状态为“运行中”表明启用Istio成功。

图 2-10 Istio 服务网格启用成功

----结束

访问控制面集群

步骤1 登录应用服务网格控制台，单击左侧导航栏的“网格管理”。

步骤2 单击托管网格的名称，进入网格详情页面。

步骤3 在“访问控制面集群”页签，参照界面提示完成对应配置。

----结束

卸载服务网格

包周期的网格不支持直接卸载，请单击“更多 > 退订”到订单中心处理。卸载服务网格前，请确保已移除全部集群。如果集群启用了包周期的网格，集群被删除后，服务网格仍然会继续计费，请前往订单处理中心手动退订。

步骤1 登录应用服务网格控制台，单击左侧导航栏的“网格管理”，在对应的服务网格下单击“操作 > 卸载”。

步骤2 在卸载Istio服务网格页面，单击“确定”，执行卸载操作。

- 卸载Istio服务网格将会卸载Istio控制面组件及数据面sidecar，卸载期间将自动为您重启业务pod，期间可能造成断服。
- 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。
- 如需查看节点信息，请至“CCE界面资源管理 > 节点管理 > 节点列表”中查看节点。
- 如需更新对外访问方式，请至“CCE界面资源管理 > 网络管理 > 添加service”暴露对外访问方式。
- 卸载Istio时，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。

须知

退订或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，需要用户在CCE界面手动删除或退订，避免资源浪费。

----结束

网格扩容

添加节点不会增加istio控制面组件实例数，如有需要，请在节点添加完成后至控制面管理页扩容。

步骤1 登录应用服务网格控制台，单击左侧导航栏的“网格管理”，在对应的服务网格中单击“网络扩容”。

步骤2 在“运行节点”页签，单击“添加节点”，设置“创建节点”参数。

- **计费模式**
  - 包年/包月：包年包月是预付费模式，按订单的购买周期计费，适用于可预估资源使用周期的场景，价格比按需计费模式更优惠。
  - 按需计费：按需计费是后付费模式，按资源的实际使用时长计费，可以随时开通/删除资源。

  包年/包月节点创建后不能删除，如需停止使用，请到费用中心执行退订操作。
- **当前区域**

  指节点实例所在的物理位置。请就近选择靠近您业务的区域，可减少网络时延，提高访问速度；不同区域的云服务产品之间内网互不相通。
- **节点配置**

  此处为节点配额提示，如需申请更多配额，请单击申请扩大配额。
  - **可用区&规格**

    指在同一区域下，电力、网络隔离的物理区域，可用区之间内网互通，不同可用区之间物理隔离。如果您需要提高工作负载的高可靠性，建议您将云服务器创建在不同的可用区。

    选择所需的规格。为确保Istio控制面运行稳定，推荐您使用 8U 16G 或以上规格控制面节点，详情请参见**规格推荐**。
- **登录方式**

  支持密码和密钥两种方式。
  - 选择“密码”：用户名默认为“root”，请输入登录节点的密码，并确认密码。

    登录节点时需要使用该密码，请妥善管理密码，系统无法获取您设置的密码内容。
  - 选择“密钥对”：在选项框中选择用于登录本节点的密钥对，并单击勾选确认信息。

    密钥对用于远程登录节点时的身份认证。如果没有密钥对，可单击选项框右侧的“创建密钥对”来新建，创建密钥对操作步骤请参见**创建密钥对**。

步骤3 设置完成后，单击“下一步”。确认信息无误后，单击“提交”。

----结束

2.3 添加服务

Istio服务网格支持对运行在Kubernetes集群上的无状态工作负载进行流量治理和流量全方位监控，将服务加入网格后，可以实现服务的灰度发布、限流、熔断、会话保持等流量治理能力以及一站式、图形化拓扑的流量健康与性能、调用链监控。

前提条件

- 您需要创建或已有一个可用集群，并确保集群版本为1.9及以上，且已启用Istio服务网格，如果没有请参照**启用Istio服务网格**中内容创建。
- 添加服务需要先创建无状态工作负载（Deployment），并且为每个Deployment添加一个Service，详情请参见**创建无状态负载（Deployment）**和**添加Service**。

添加服务

步骤1 登录**应用服务网格控制台**，在左侧导航栏中选择“服务列表”。

步骤2 单击“添加服务”，在“添加服务”中选择要操作的命名空间，并配置以下参数。

- **服务名称**

  待添加到网格中的服务。
- **所属集群**

  服务所属集群。
- **关联工作负载**

  该服务所属的工作负载，界面自动关联。
- **访问端口-->容器端口**
  - 容器端口：工作负载程序实际监听的端口，需用户确定。
  - 访问端口：容器端口映射到集群虚拟IP上的端口，用虚拟IP访问工作负载时使用，端口范围为1-65535，可任意指定。
- **访问协议**

  请根据业务的协议类型选择，目前支持http、tls、tcp、grpc和dubbo协议，默认选中http协议。
- **版本号**

  为服务设定版本号。
- **状态**

  如果服务不可添加时，此处会显示不可添加的具体原因。
- **边车资源配置**
  - CPU申请：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 ≥ 容器CPU申请数时，才允许将容器调度到该节点。
  - CPU限制：容器能使用的CPU最大值。
  - MEM申请：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 ≥ 容器内存申请数时，才允许将容器调度到该节点。
  - MEM限制：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。

说明

- 服务加入网格后，会修改服务和负载的labels：
  - 修改Service和Deployment的app标签为Service的名字。
  - 在Deployment中增加标志版本的version标签。
  - 如果Service的spec.selector中存在version标签，会被删除。
- 加入网格后，会重启负载进行sidecar注入以实现网格流量治理能力，期间服务访问会短暂终止。
- 加入网格后，会在网格管理下的所有集群中创建相应的namespace和service，请确保多个集群中不能存在名称相同的不同服务。
- 加入网格后，会修改集群安全组规则，以保证服务能被其他集群访问。

步骤3 配置完成后，勾选“我已阅读以下内容”，单击“确定”。

----结束

配置服务网关

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。

创建服务网关前，请提前创建好弹性负载均衡。为了更好的与服务网格配合，推荐创建并使用增强型弹性负载均衡。创建增强型弹性负载均衡时，需要注意确保所选VPC和子网与创建集群时设置的VPC和子网保持一致，详情请参见**创建负载均衡器** 。

步骤1 登录**应用服务网格控制台**，在左侧导航栏中选择“服务网关”。

步骤2 在右侧页面，单击“添加服务网关”，配置对外访问网络参数。

- **网关名称**

  新建服务网关的名称，由小写字母、数字、横线（-）和点（.）组成，且必须以字母或数字开头结尾，长度不能超过63个字符。

- **集群名称**

  服务网关所属的集群。

- **负载均衡配置**

  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，支持公网和私网两种类型。

  负载均衡实例需与当前集群处于相同VPC 。

- **监听器配置**
  - **对外协议**

    请根据业务的协议类型选择。
  - **对外端口**

    开放在负载均衡服务地址的端口，可任意指定。
- **转发策略配置**
  - **域名**

    请填写组件对外发布域名。不配置访问地址默认为负载均衡实例IP地址。如果您开启了TLS认证，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。

– **服务访问端口**

仅显示匹配对外协议的端口。

步骤3 配置完成后，单击“创建”。

网关添加完成后，在左侧导航栏中单击“服务列表”，获取服务外网访问地址。

图 2-11 访问服务

----结束

2.4 灰度发布

2.4.1 灰度发布概述

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。

应用服务网格基于Istio提供的服务治理能力，对服务提供多版本支持和灵活的流量策略，从而支持多种灰度发布场景。

当前版本支持金丝雀发布和蓝绿发布。

金丝雀发布

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。

金丝雀发布的特点：

通过在线上运行的服务中，新加入少量的新版本的服务，然后从这少量的新版本中快速获得反馈，根据反馈决定最后的交付形态。

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OK，将流量切到新版本，然后老版本同时也升级到新版本。始终有两个版本同时在线，有问题可以快速切换。

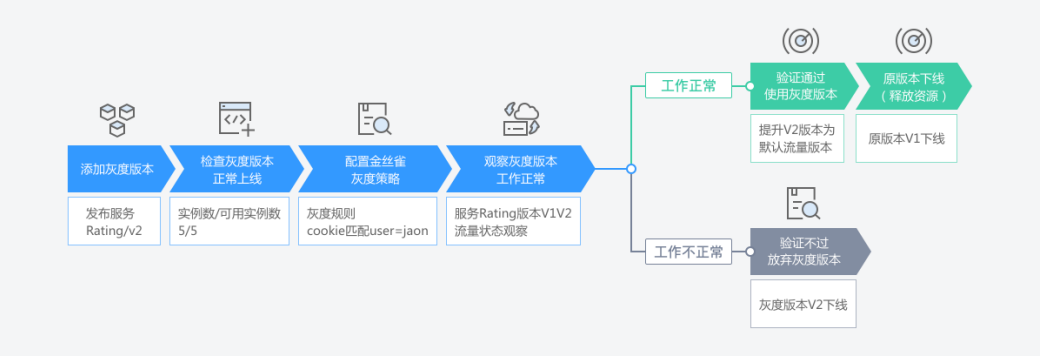
蓝绿发布的特点：

在部署应用的过程中，应用始终在线。并且新版本上线过程中，不会修改老版本的任何内容，在部署期间老版本状态不受影响。只要老版本的资源不被删除，可以在任何时间回滚到老版本。

灰度发布流程

在ASM中，用户无需使用繁琐的命令行配置，只需通过清晰友好的图形界面，就可轻松直观的完成灰度发布整个过程，如图2-12。

图 2-12 灰度发布过程



2.4.2 为服务添加灰度版本

一个服务仅支持发布一个灰度版本，并支持为版本添加灰度策略。

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

  当您需要在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此。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。

创建金丝雀发布

步骤1 登录应用服务网格控制台，在左侧导航栏中选择“灰度发布”。在金丝雀发布下，单击“创建”。

步骤2 配置灰度版本基本信息。

- **灰度发布服务**

  单击“选择服务”，选择添加灰度版本的服务。

- **部署集群**

  灰度发布服务所属的集群。

- **发布任务名称**

  系统自动生成，可根据实际需求进行修改。

- **版本号**

  新增服务中灰度版本号。

- **版本描述**

  灰度版本描述信息。

步骤3 根据界面提示的流程图，熟悉灰度发布版本流程，单击“创建”。

步骤4 部署灰度版本。

- **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**

  请选择需要添加灰度的镜像版本。

步骤5 单击“部署灰度版本”，灰度版本开始创建。

请确保灰度版本的实例状态为正常时，再开始下一步进行灰度策略的配置。

步骤6 单击“配置灰度策略”，进行灰度策略配置（可选）。

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

- **基于请求内容发布**

  对当前版本配置相应的请求内容规则，服务流量在满足此规则的情况下，会走此版本。例如http请求，请求头cookie必须满足“访问条件”走版本A。

  - 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：当且仅当表达式完全符合此情况时，流量才会走到这个版本。
    - 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。

  - 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。
  - 灰度策略规则描述：当前服务的流量转发的规则描述信息及Yaml的查看。

- **基于流量比例发布**

  对当前版本配置相应的流量权重，服务流量将会按照权重比率以对应的概率分发当前版本。例如10%的流量走版本A，90%的流量走版本B。

  - 版本流量配比：根据输入的流量配比来确定流量分发的比重。

    范围限制为[0,默认版本权重w]。例如，当您配置为10，则10%的服务流量会走向此版本，(w-10)%的流量会走向默认版本，即从默认版本分走一部分流量。
  - 灰度策略规则描述：当前服务的流量转发的规则描述信息及Yaml的查看。

说明

- 如果当前服务已有灰度版本并配置了灰度策略，则策略类型不可选（灰度版本只能配置一种策略类型），默认为已经配置的策略类型。
- 基于请求内容发布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review组件，配置了基于请求内容的灰度发布策略，通过访问productpage组件的界面，是无法看到效果的。

  原因为，您的客户端访问productpage组件携带了HTTP请求的header信息，而productpage组件请求reviews组件时，将这些header信息丢失了（详情可参考**如何使用Istio调用链埋点**），从而失去了基于请求内容的灰度发布效果。

步骤7 设置完成后，单击“策略下发”。

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

----结束

创建蓝绿发布

步骤1 登录**应用服务网格控制台**，在左侧导航栏中选择“灰度发布”。在蓝绿发布下，单击“创建”。

步骤2 配置灰度版本基本信息。

- **蓝绿发布服务**

  单击“选择服务”，选择待发布服务。
- **部署集群**

  灰度发布服务所属的集群。
- **发布任务名称**

  系统自动生成，可根据实际需求进行修改。
- **版本号**

  新增服务中灰度版本号。
- **版本描述**

  灰度版本描述信息。

步骤3 根据界面提示的流程图，熟悉灰度发布版本流程，单击“创建”。

步骤4 部署灰度版本。

- **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**

  请选择需要添加灰度的镜像版本。

步骤5 单击“部署新版本”，新版本开始创建。

请确保灰度版本的实例状态为正常时，再开始下一步进行灰度策略的配置。

步骤6 单击“蓝绿策略配置与监控”，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

2.4.3 灰度版本基本操作

您可以对现有的服务进行多个版本的管理，针对不同版本配置不同的策略，从而达到流量管理的目的。

使用说明

对灰度版本的相关基本操作，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。

为灰度版本添加灰度策略

如果您在添加灰度版本时，未配置灰度策略，可以基于以下步骤创建策略。

步骤1 登录应用服务网格控制台，在左侧导航栏中选择“灰度发布”。

步骤2 单击已创建灰度任务的名称，进入查看灰度版本状态页面。

步骤3 单击“查看/配置灰度策略”。在灰度策略配置中，单击策略类型下的“基于请求内容”策略或“基于流量比例”策略。

- **基于请求内容发布**

  对当前版本配置相应的请求内容规则，服务流量在满足此规则的情况下，会走此版本。例如http请求，请求头cookie必须满足“访问条件”走版本A。

  - Cookie内容：
    - 完全匹配：当且仅当表达式完全符合此情况时，流量才会走到这个版本
    - 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：当且仅当表达式完全符合此情况时，流量才会走到这个版本
    - 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。
  - 允许访问的操作系统：请选择允许访问的操作系统。
  - 允许访问的浏览器：请选择允许访问的浏览器。
  - 灰度策略规则描述：当前服务的流量转发的规则描述信息及Yaml的查看。

- **基于流量比例发布**

对当前版本配置相应的流量权重，服务流量将会按照权重比率以对应的概率分发当前版本。例如10%的流量走版本A，90%的流量走版本B。

- 流量配比：根据输入的流量配比来确定流量分发的比重。

  范围限制为[0,默认版本权重w]，例如，当您配置为10，则10%的服务流量会走向此版本，(w-10)%的流量会走向默认版本，即从默认版本分走一部分流量。
- 灰度策略规则描述：当前服务的流量转发的规则描述信息及Yaml的查看。

步骤4 单击“策略下发”，灰度策略会自动生效。

----结束

修改版本的灰度策略

- 修改基于流量比例的策略：

  基于流量比例的策略（又称为AB TEST策略）所有版本的权重之和为100，即：假设默认版本的权重为x，当前待修改版本的权重为y，则x+y=100。

  当您此时将待修改版本的权重从y改成y1，则默认版本的权重会自动改为x-y1+y，即待修改版本的权重会影响到默认版本的权重。
- 修改基于请求内容的策略：

  基于请求内容的策略（又称为金丝雀策略）会遍历除默认版本外的全部金丝雀规则，如果满足某个版本的规则，则流量走向此版本，如果全部不满足，则流量会走到默认版本，即：假设当您此时将待修改版本的匹配策略为x。

  当您此时将待修改版本的匹配策略从x改成x1，则原本满足x规则的流量将会走到默认版本，而满足新规则x1的流量将会走到当前的待修改版本。

切换灰度策略类型

您可以在“基于请求内容”的策略和“基于流量比例”的策略之间切换。策略完成切换后，原本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**发布中**的任务才支持灰度策略变更，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），将不支持重新配置灰度策略。

步骤1 登录**应用服务网格控制台**，在左侧导航栏中选择“灰度发布”。

步骤2 单击已创建灰度版本的名称。

步骤3 在监测灰度运行状态页面，单击“查看/配置灰度策略”。

步骤4 参照**配置灰度策略**重新配置灰度策略。

步骤5 配置完成后，单击“策略下发”。

----结束

切为默认流量版本

执行某个版本（如V2版本）后的“接管所有流量”，V2版本将获取到原默认版本的全部流量。原默认版本流量将变为0。

下线灰度版本

如果下线了原默认版本，则该版本的流量会被新创建的灰度版本完全接管。

步骤1 登录**应用服务网格控制台**，在左侧导航栏中选择“灰度发布”。

步骤2 单击已创建灰度任务的名称。

步骤3 单击待下线版本后的“版本下线”，下线该版本。

----结束

2.5 流量治理

通过修改负载均衡方式，进行故障注入测试及配置基本连接池管理。

配置流量策略

步骤1 登录**应用服务网格控制台**，在左侧导航栏中选择“流量治理”。

步骤2 选择组件所在的集群和命名空间。

步骤3 在拓扑图中，单击组件名称。在服务流量策略中，进行流量策略配置。

- **负载均衡算法**
  - ROUND_ROBIN：轮询，默认负载均衡算法。
  - LEAST_CONN：随机选取两个健康的主机，再从所选取的两个主机中选择一个链接数较少的主机。
  - RANDOM：从所有健康的主机中，随机选取一个。
- **会话保持**
  - 根据HTTP头部中的内容获取哈希：
    - 选择Cookie：将以HTTP请求中的所有Cookie计算哈希，哈希相同的请求将会转发至同一个实例进行处理。
    - 选择User-Agent：将以HTTP请求中的User_agent来计算哈希，UA哈希值相同的请求将会转发至同一个实例进行处理。
    - 流量治理也支持用户使用自定义Key来计算哈希，只需选择自定义模式并且输入键的名称。
  - 根据Cookie键中的内容获取哈希：支持用户输入Cookie键的名称，转发方式则由设定的Cookie键对应的值来计算哈希，哈希相同的请求则会转发至同一个实例中。例如我们设定Cookie中的User为键，则通过计算User对应的值的哈希来确认转发规则。
  - 根据SourceIp中的内容获取哈希：流量将会按照请求源IP地址的哈希值进行会话保持。

步骤4 连接池管理。

- **最大连接数**

  到目标主机HTTP或TCP连接的最大数量。
- **最大请求重试次数**

  在指定时间内对目标主机最大重试次数。

- **最大等待请求数**

  等待列队的长度，默认为1024。

- **每连接最大请求数**

  对后端连接中最大的请求数量如果设为1则会禁止keep alive特性。

- **连接超时时间**

  TCP连接超时时间。

- **最大请求数**

  后端服务处理的最大请求数，默认为1024。

步骤5 熔断配置。

- **连续错误数**

  在一个检查周期内，连续出现500及以上错误的个数，例502、503状态码。

- **检查周期**

  将会对检查周期内的响应码进行筛选。

- **最大隔离实例比例(%)**

  上游实例中，允许被隔离的最大比例。采用向上取整，如果10个实例，设为13%则最多会隔离2个实例。

- **最短隔离时间**

  实例第一次被隔离的时间，之后每次隔离时间为隔离次数与最短隔离时间的乘积。

步骤6 是否开启Mutual TLS。

- 开启Mutual TLS：组件仅会通过基于TLS建立的安全信道通信。
- 关闭Mutual TLS：组件之间通过明文通信。

步骤7 故障注入。

在故障类型中选择时延故障或中断故障。当前版本仅支持基于请求内容策略。

- **故障版本**

  故障所作用的版本。

- **故障类型**
  - 不启用：取消故障注入。如果不需要继续触发故障，可单击故障类型中的“不启用”来删除已配置的故障。
  - 时延故障：对通往组件的请求有延迟。
    - 故障百分比：故障按比例发生。
    - 延时：设定的时间延迟单位。
  - 中断故障：会中断该组件的服务并返回预设状态码。
    - 故障百分比：故障按比例发生。
    - Http状态码：终止故障时返回的http状态码，默认返回500。

步骤8 访问鉴权。

- 开启访问鉴权：当前服务只能被指定的服务访问，并且自动开启Mutual TLS。

- 关闭访问鉴权：当前服务能够被所有服务访问。

说明

访问鉴权会默认授权给网关实例（ingressgateway），通过网关间接访问当前服务不会受到访问授权配置的影响。

----结束

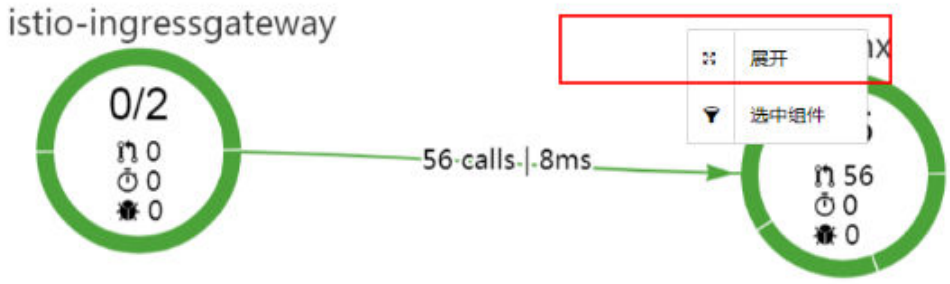
查看流量监控

轮询算法为默认负载均衡算法。即当组件有多个实例时，访问基本接近于平均分配给每一个实例。组件流量策略设置完成后，通过连续不中断的访问应用，产生访问数据。

步骤1 登录应用服务网格控制台，在左侧导航栏中选择“流量监控”。

步骤2 在上侧时间轴区域选择对应的时间范围，在所需要查看的组件区域右键单击“展开”，将其展开为版本。

图 2-13 选择时间轴并展开组件

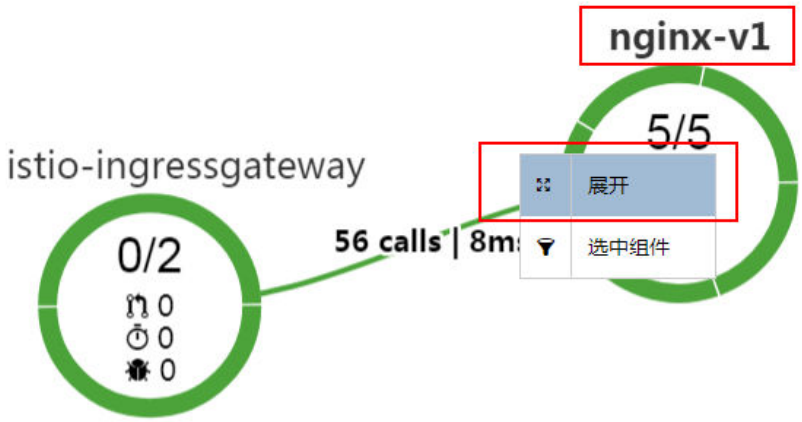


步骤3 右键单击展开的版本，选择“展开”，将其展开为实例。

说明

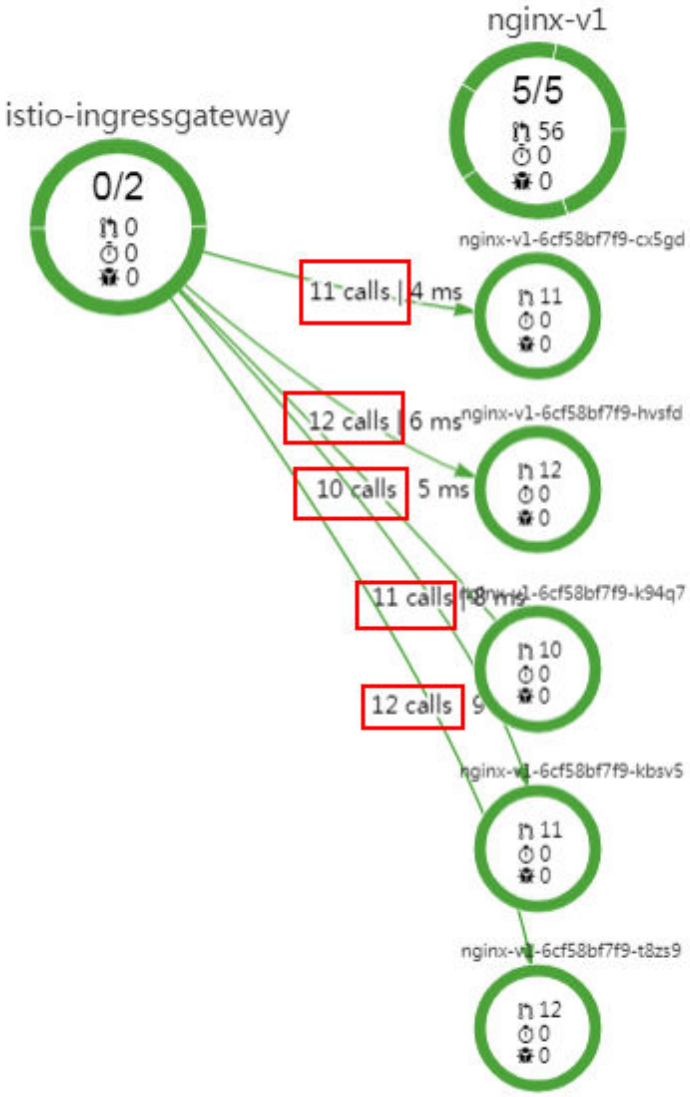
Report模式下，不提供实例展开功能。

图 2-14 展开组件为实例



步骤4 在拓扑图区域，可以看到各个流量分发到各个实例的情况。

图 2-15 轮询算法请求分发



----结束

更改流量策略算法

流量策略设置完成后，支持更改流量策略算法。如将标准负载均衡的算法轮询转为随机（RANDOM）。

步骤1 登录**应用服务网格控制台**，在左侧导航栏中选择“流量治理”。

步骤2 在“流量治理”界面中选择并单击目标组件。

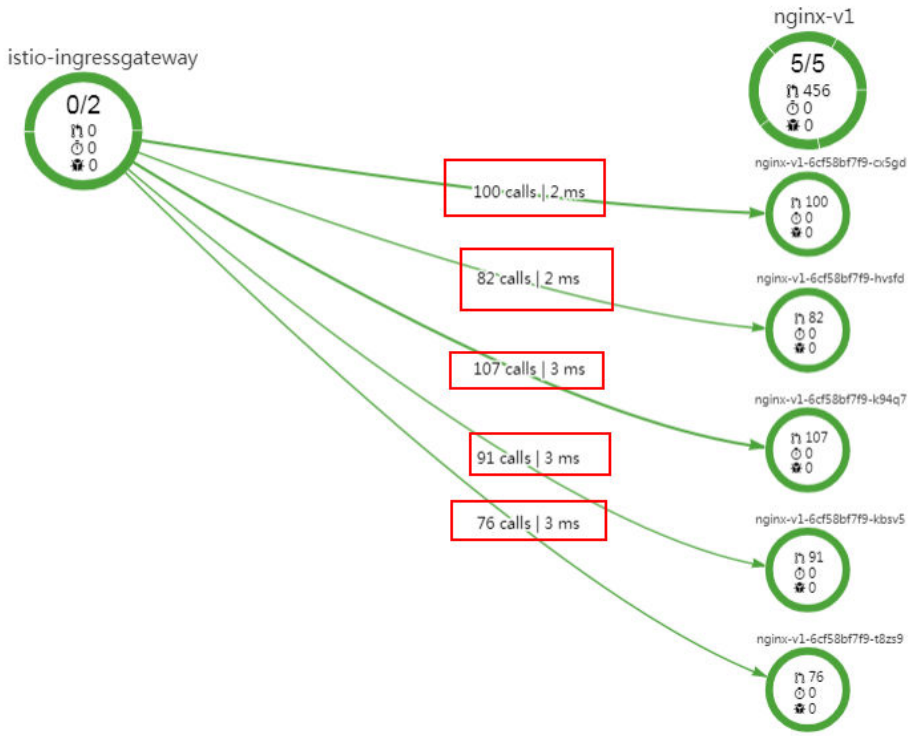
步骤3 在“标准负载均衡算法”中，选择“RANDOM”算法，单击“保存”。

图 2-16 负载均衡算法选择随机

步骤4 重复先前的操作，在流量监控里观察请求分发情况。

可以发现流量分发没有什么固定规律，各个实例差距也比较大，说明随机算法已经生效。

图 2-17 随机算法结果



----结束

2.6 流量监控

通过流量监控可以监控流量概况、组件运行状态、调用链等信息，并在系统业务异常时快速定位到问题点。

使用约束

- 流量监控记录的是组件过去24小时内的数据，所以当部分组件删除后并不能及时的在拓扑图上消失，而是会仍然存在一段时间。
- 流量治理中的故障注入功能，不能作用于基于请求比例灰度发布的场景。只接受单一版本或基于请求内容灰度发布的场景。
- 托管网格多集群场景下，只支持集群内部服务之间的流量监控，暂时不支持跨集群服务之间的流量监控。

查看流量监控情况

步骤1 登录应用服务网格控制台，在左侧导航栏中选择“流量监控”。

步骤2 查看整个系统的监控情况。

- 饼状图显示了系统处于运行中和未就绪的应用数量。
- 异常响应和超长RT时延按照租户内异常请求个数和单次请求耗时从高到低的排序。

步骤3 查看某个组件的监控情况。

选择服务网格、集群及命名空间，单击拓扑图上的组件，进入组件监控详情页面。

- 流量概况：展示了应用详细的流量信息，包含请求总数、错误数、平均时延、最大时延等。
- 部署信息：显示了组件内的所有实例状态及信息。
- 调用链：通过时间维度和组件版本来查看调用情况，支持使用高级搜索中的选项，进行精准搜索。使用Isito做服务治理时，无需在微服务代码中进行调用链埋点。但微服务代码在接收和发送请求时需要传递调用链相关的Header信息，才能构造成完整的调用链路，详情请参见**如何使用Istio调用链埋点**。

----结束

如何使用 Istio 调用链埋点

Header信息包括如下内容，更多关于调用链的信息请参见**https://istio.io/docs/tasks/telemetry/distributed-tracing/**。

- x-request-id
- x-b3-traceid
- x-b3-spanid
- x-b3-parentspanid
- x-b3-sampled
- x-b3-flags
- x-ot-span-context

下面以一个典型的例子来说明如何在接收和发送请求时，通过修改代码来传递调用链相关Header信息。

- **Python代码示例**

  服务接收端在处理请求时，从请求端解析相关Header。在调用后端请求时，传递Trace相关的header。

```
def getForwardHeaders(request):
    headers = {}

if 'user' in session:
    headers['end-user'] = session['user']

incoming_headers = ['x-request-id',
    'x-b3-traceid',
    'x-b3-spanid',
    'x-b3-parentspanid',
    'x-b3-sampled',
    'x-b3-flags',
    'x-ot-span-context'
]

return headers


@ app.route('/productpage')
def front():
    product_id = 0# TODO: replace
default value
headers = getForwardHeaders(request)
user = session.get('user', '')
product = getProduct(product_id)
detailsStatus, details = getProductDetails(product_id, headers)
reviewsStatus, reviews = getProductReviews(product_id, headers)
```

```
return render_template(
    'productpage.html',
    detailsStatus = detailsStatus,
    reviewsStatus = reviewsStatus,
    product = product,
    details = details,
    reviews = reviews,
    user = user)
```

- **Java代码示例**

  在java Rest接口上除了解析一般业务参数外，需要从header中解析trace相关信息。同样在调用下一个服务时传递该header信息。

```
@GET
@Path("/reviews/{productId}")
public Response bookReviewsById(@PathParam("productId") int productId,
@HeaderParam("end-user") String user,
@HeaderParam("x-request-id") String xreq,
@HeaderParam("x-b3-traceid") String xtraceid,
@HeaderParam("x-b3-spanid") String xspanid,
@HeaderParam("x-b3-parentspanid") String xparentspanid,
@HeaderParam("x-b3-sampled") String xsampled,
@HeaderParam("x-b3-flags") String xflags,
@HeaderParam("x-ot-span-context") String xotspan) {
    int starsReviewer1 = -1;
    int starsReviewer2 = -1;
    if (ratings_enabled) {
        JsonObject ratingsResponse = getRatings(Integer.toString(productId), user, xreq, xtraceid,
xspanid, xparentspanid, xsampled, xflags, xotspan);
        if (ratingsResponse != null) {
            if (ratingsResponse.containsKey("ratings")) {
                JsonObject ratings = ratingsResponse.getJsonObject("ratings");
                if (ratings.containsKey("Reviewer1")){
                    starsReviewer1 = ratings.getint("Reviewer1");
                }
                if (ratings.containsKey("Reviewer2")){
                    starsReviewer2 = ratings.getint("Reviewer2");
                }
            }
        }
    }
    String jsonResStr = getJsonResponse(Integer.toString(productId), starsReviewer1, starsReviewer2);
    return Response.ok().type(MediaType.APPLICATION_JSON).entity(jsonResStr).build();
}
}
```

2.7 网格管理

网格管理提供了Istio控制面组件、数据面sidecar的健康及性能监控能力，支持运行资源的扩缩容管理，以及不同Istio版本间的升级管理。

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（边车）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。

监控已启用服务网格的集群

步骤1 登录**应用服务网格控制台**，在左侧导航栏中选择“网格管理”。

步骤2 在右侧页面，单击服务网格名称。

步骤3 单击专有网格名称，查看控制面所有组件和Sidecar的业务pod信息。

- “控制面组件管理”页签：展示Istio控制面的所有组件，以及每个组件的运行状态，CPU、内存占用率，命名空间等信息。

- “边车管理”页签：展示所有注入了边车的业务pod信息，以及每个pod的名称，CPU、内存占用率，边车版本，边车升级操作入口等信息。
- “运行节点”页签：查看节点的相关信息，包括节点名称、状态、节点模式（独享）、可用区、私有IP地址、规格、可用CPU、可用内存。
- “插件管理”页签：可以对grafana、prometheus、kiali、tracing四个插件进行安装和卸载，这些插件可以帮助用户进行流量治理与监测。
- “运行事件”页签：记录控制面组件的kubernetes事件，帮助您快速定位Istio控制面问题。
- “运行日志”页签：记录控制面组件的运行日志，帮助您快速定位Istio控制面问题。
- “配置管理”页签：支持公共配置和性能调优配置，请根据实际需求选择。详情请参见**配置管理**。
- “istioctl”页签：使用Istio命令行工具配置路由策略。详情请参见**使用Istio命令行工具配置路由策略**。

单击托管网格的名称，查看数据面所有组件和Sidecar的业务pod信息。

- “集群管理”页签：展示已添加的到网格中的集群信息，以及添加新的集群到网格，或移除已添加的集群。
- “数据面组件管理”页签：展示Istio数据面的所有组件，以及每个组件的运行状态，CPU、内存占用率，命名空间等信息。
- “边车管理”页签：展示所有注入了边车的业务pod信息，以及每个pod的名称，CPU、内存占用率，sidecar版本，sidecar升级操作入口等信息。
- “运行事件”页签：记录控制面组件的kubernetes事件，帮助您快速定位Istio控制面问题。
- “运行日志”页签：记录控制面组件的运行日志，帮助您快速定位Istio控制面问题。
- “访问控制面集群”页签：使用kubectl操作集群。详情请参见**使用kubectl操作集群**。

----结束

升级 Istio 控制面

执行升级操作需具备以下两个条件：

- Istio控制面运行版本需小于最新版本。
- 旧版本Istio控制面运行正常。

步骤1 登录**应用服务网格控制台**，在左侧导航栏中选择“网格管理”。

步骤2 在Istio管理集群列表页面，单击待升级集群后的“操作 > 升级”。

步骤3 在弹出的Isito控制面升级页面，确认升级信息无误后，单击“确认”。

图 2-18 升级 Istio 控制面

----结束

边车管理

展示所有注入了边车的业务pod信息，以及每个pod的名称，CPU、内存占用率，边车版本，边车升级操作入口等信息。

您可以在“边车管理”中进行“边车升级”，详细步骤如下：

步骤1 登录应用服务网格控制台，在左侧导航栏中选择“网格管理”。

步骤2 单击专有网格名称进入网格管理详情。

步骤3 在“边车管理”页签，单击“边车升级”，选择组件进行边车升级。

须知

- 对于单实例组件，升级会触发流量短暂中断，建议扩容后再升级。
- 仅专有网格支持边车升级，托管网格会随着网格版本的升级自动升级边车。

----结束

运行节点

“运行节点”可以帮助您查看已创建的节点。您在当前集群中创建了节点，就可以在“运行节点”页中查看节点的相关信息，包括节点名称、状态、节点模式（独享）、可用区、私有IP地址、规格、可用CPU、可用内存。

您还可以在“运行节点”中对节点进行“扩容”，详细步骤如下：

步骤1 登录应用服务网格控制台，在左侧导航栏中选择“网格管理”。

步骤2 单击专有网格名称进入网格管理详情。

步骤3 在“运行节点”页签，单击“操作”下的“扩容”链接跳转到弹性云服务器页面。

步骤4 根据节点“名称/ID”找到对应的云服务器，单击“变更规格”，即可对节点进行扩容操作。

图 2-19 变更规格

----结束

插件管理

插件管理可以对grafana、prometheus、kiali、tracing四个插件进行安装和卸载，这些插件可以帮助用户进行流量治理与监测。详细步骤如下：

步骤1 登录应用服务网格控制台，在左侧导航栏中选择“网格管理”。

步骤2 单击专有网格名称进入网格管理详情。

步骤3 在“插件管理”页签，单击待安装插件下方的“安装”，跳转到安装页面。

步骤4 在安装页面，选择插件要绑定的“弹性负载”，输入要分配给插件安装的“CPU配额”和“内存配额”。

说明

- 当前服务网格支持“经典型负载均衡”和“增强型弹性负载均衡”，为了更好地与服务网格配合，推荐创建并使用“增强型弹性负载均衡”。
- 如果您的弹性负载均衡监听器配额不足将不会显示在负载均衡器列表中。
- CPU配额和内存配额中的“申请”和“限制”值为必填，且不能超过节点剩余额度。
- 安装grafana、prometheus、kiali和tracing插件时，界面会自动过滤掉未绑定弹性IP的负载均衡器。
- 不支持在IE浏览器上使用kiali插件。

步骤5 单击“安装”，跳转到插件管理页面，在插件卡片的左上角显示插件的安装和运行状态。如果插件长时间处于安装中状态或者安装失败状态，请删除以后重新安装。

说明

如果无需使用该插件，单击插件卡片右上角的“删除”进行卸载。

----结束

使用 Istio 命令行工具配置路由策略

使用Istio命令行工具可以配置多种路由策略管理服务流量，包括流量转移、故障注入、限流熔断等。

步骤1 登录应用服务网格控制台，在左侧导航栏中选择“网格管理”。

步骤2 单击专有网格名称进入网格管理详情。

步骤3 在“istioctl”页签，参照界面提示完成对应配置。

----结束

配置管理

步骤1 登录**应用服务网格控制台**，在左侧导航栏中选择“网格管理”。

步骤2 单击专有网格名称进入网格管理详情。

步骤3 在“配置管理”页签，根据实际需求进行如下配置。

- 公共配置：配置服务网格控制面公共基础配置，对所有组件负载生效。

- 性能调优配置：根据网格下服务负载运行信息，调整参数，实现最优性能。

----结束

使用 kubectl 操作集群

您需要先下载kubectl以及配置文件，拷贝到您的客户端机器，完成配置后，即可以使用访问kubernetes集群。

步骤1 登录**应用服务网格控制台**，在左侧导航栏中选择“网格管理”。

步骤2 单击托管网格名称进入网格管理详情。

步骤3 在“访问控制面群”页签，参照界面提示完成对应配置。

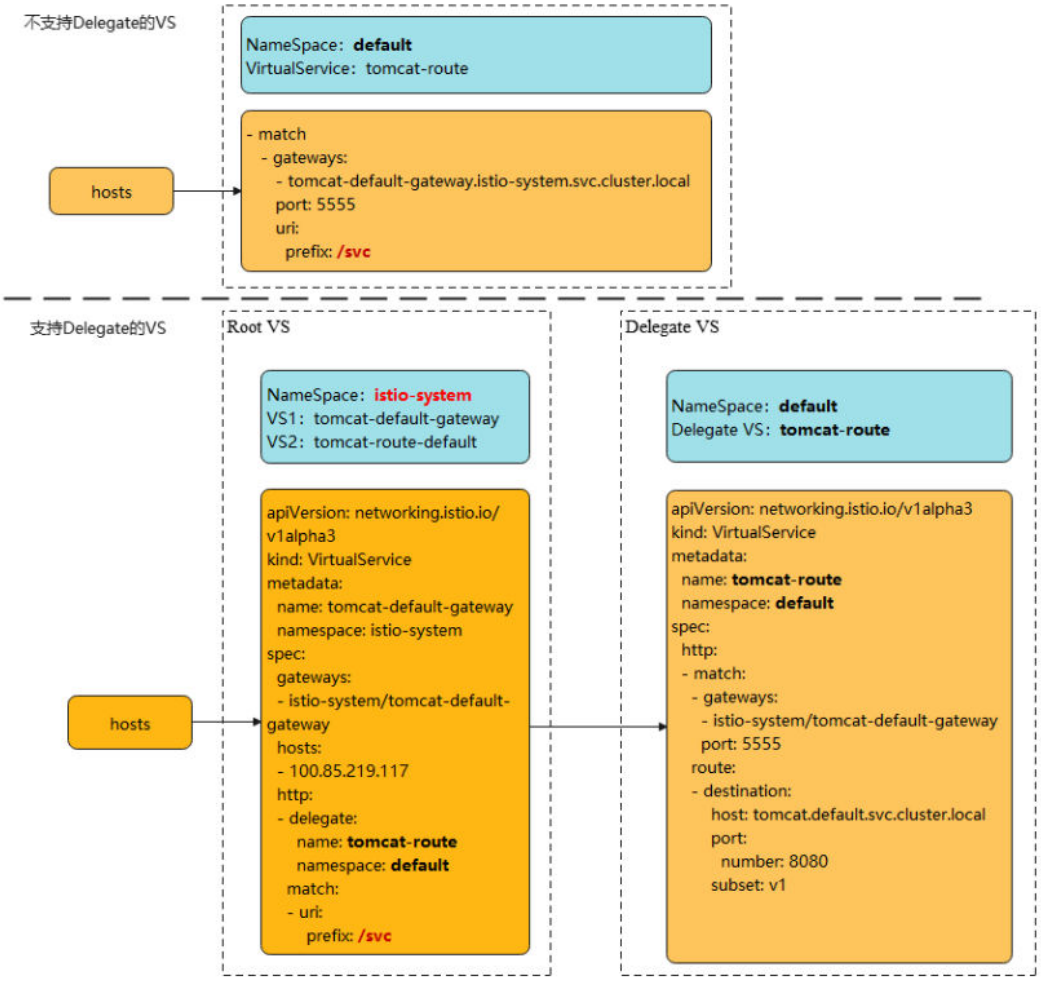
----结束

2.8 附录

2.8.1 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：

   a. 删除spec.gateways、spec.hosts和spec.http.match.uri

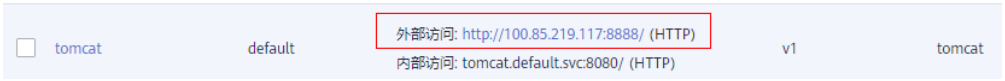
   b. tomcat-default-gateway.istio-system.svc.cluster.local替换成istio-system/tomcat-default-gateway

   c. 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。