应用服务网格

产品介绍

文档版本 02

发布日期 2023-07-30

华为技术有限公司

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

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

安全声明

漏洞处理流程

华为公司对产品漏洞管理的规定以“漏洞处理流程”为准，该流程的详细内容请参见如下网址：
https://www.huawei.com/cn/psirt/vul-response-process
如企业客户须获取漏洞信息，请参见如下网址：
https://securitybulletin.huawei.com/enterprise/cn/security-advisory

目 录

1 什么是应用服务网格

什么是应用服务网格

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

什么是 Istio

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

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

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

服务网格

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

为什么要使用 Istio

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

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

图 1-1 Istio 和 Kubernetes 的关系

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

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

Istio旨在实现可扩展性，满足各种部署需求。

产品架构

图 1-2 产品架构

产品功能

灰度发布

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

流量治理

- 七层连接池管理：支持配置HTTP最大请求数、最大重试次数、最大等待请求数、每连接最大请求数以及连接最大空闲时间。
- 四层连接池管理：支持配置TCP最大连接数、连接超时时间、最大无响应次数、最短空闲时间以及健康检查间隔。
- 熔断：支持配置服务熔断规则，包括实例被驱逐前的连续错误数、检查周期、基础隔离时间以及最大隔离实例比例。
- 重试：支持配置HTTP重试次数、重试超时时间以及重试条件。
- 超时：支持配置HTTP请求超时时间。
- 负载均衡：支持配置随机调度、轮询调度、最少连接和一致性哈希多种负载均衡算法。

- HTTP头域：可以灵活添加、修改和删除指定HTTP头域，包括将HTTP请求转发到目标服务之前对Headers的操作，以及将HTTP响应回复给客户端前，对Headers的操作。
- 故障注入：支持配置延时故障和中断故障。

安全

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

可观察性

- 应用访问拓扑：支持网格应用访问拓扑，体现服务间依赖。
- 服务运行监控：支持服务访问信息，包括服务和服务各个版本的QPS和延时等指标。
- 访问日志：支持收集和检索服务的访问日志。
- 调用链：支持非侵入调用链埋点，并可以通过检索调用链数据进行问题定界定位。

网格数据面服务框架

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

兼容性和扩展

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

2 产品优势

简单易用

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

内置金丝雀、蓝绿灰度发布流程

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

策略化的智能路由与弹性流量管理

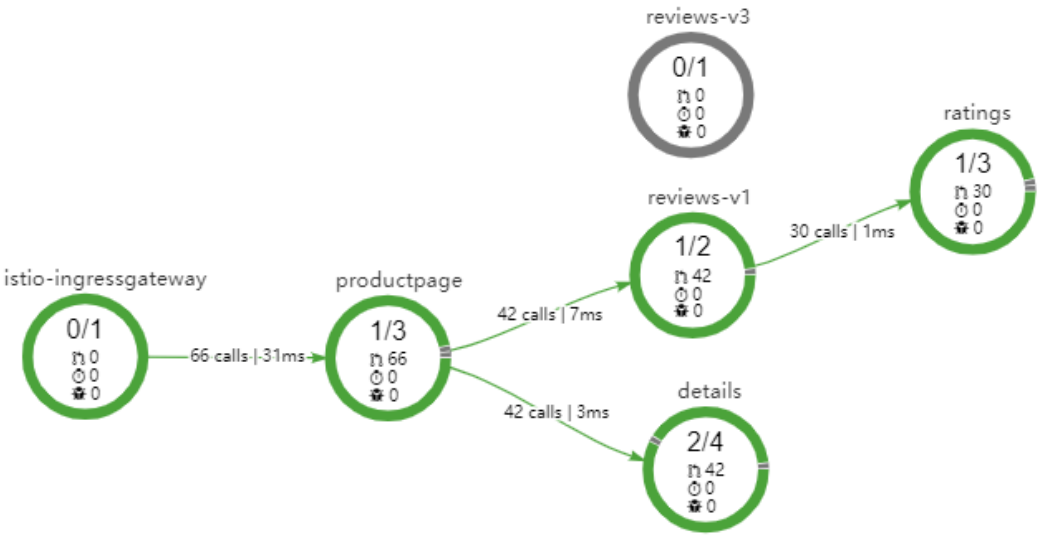
支持对服务配置负载均衡、服务路由、故障注入、熔断等治理规则，并结合一站式治理系统，提供实时的、可视化的微服务流量管理；无侵入智能流量治理，应用无需任何改造，即可进行动态的智能路由和弹性流量管理。

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

图形化应用全景拓扑，流量治理可视化

应用服务网格提供了可视化的流量监控，链路健康状态、异常响应、超长响应时延、流量状态信息拓扑等一目了然。

图 2-1 流量监控



结合**应用运维管理 AOM**服务，ASM提供了详细的微服务级流量监控、异常响应流量报告以及调用链信息，实现更快速、更准确的问题定位。

性能增强，可靠性增强

控制面和数据面在社区版本基础上进行可靠性加固和性能优化。

多基础设施

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

协议扩展

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

传统 SDK 集成

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

3 应用场景

3.1 服务灰度发布

适用场景

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

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

价值

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

优势

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

3.2 服务流量管理

适用场景

流量治理是一个非常宽泛的话题，例如：

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

价值

在Istio中实现这些服务治理功能时无须修改任何应用的代码。

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

优势

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

3.3 端到端的透明安全

适用场景

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

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

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

价值

- **默认的安全性：**无需修改即可保证应用程序代码和架构的安全性。
- **纵深防御：**与现有的安全系统结合并提供多层防御。
- **零信任网络：**在不受信任的网络上构建安全解决方案。

优势

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

3.4 服务运行监控

适用场景

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

价值

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

优势

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

3.5 传统微服务 SDK 结合

适用场景

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

价值

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

优势

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

4 约束与限制

集群限制

启用应用服务网格前，您需要创建或已有一个可用集群，并确保集群版本为v1.15、v1.17、v1.19、v1.21、v1.23、v1.25或v1.27。不支持安全容器类型的CCE Turbo集群添加至网格。

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

- Ubuntu 22.04

网格功能约束

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

旧版 ASM 与新版 ASM 区别

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

旧版ASM与新版ASM的区别如下：

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

5 计费说明

计费项

ASM计费单位为实例数（CCE集群中Pod数量）。目前提供**免费**基础版网格，最大管理实例数为200。

- ASM套餐价格不包含用户使用华为云上的资源费用（弹性云服务器、CCE集群管理费、ELB费用等），相关链接如下：
  - 集群创建费用详见：**云容器引擎价格详情**。
  - 更多价格详情请参见：**产品价格详情**。

6 权限管理

ASM服务功能请使用系统管理员（包含集群的RBAC权限）操作。

7 基本概念

工作负载

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

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

实例（Pod）

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

金丝雀发布

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

蓝绿发布

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

流量治理

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

连接池管理

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

熔断

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

调用链分析

跟踪大规模复杂的分布式系统运行服务调用关系，解决分布式服务故障定位定界问题。

8 规格推荐

独享节点规格推荐

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

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

说明

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

ingressgateway 实例资源消耗参考

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

表 8-1 长连接内存消耗

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

表 8-2 短连接 CPU 和内存消耗

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

以上数据仅供参考，具体的资源消耗和实际的业务模型有关，以实际测试结果为准。

9 与其他云服务的关系

应用服务网格与周边服务的依赖关系如图9-1所示。

图 9-1 应用服务网格与其他云服务关系

应用服务网格与其他服务的关系

表 9-1 应用服务网格与其他服务的关系

服务名称	应用服务网格与其他服务的关系	主要交互功能
云容器引擎 CCE	云容器引擎（Cloud Container Engine，CCE）提供高可靠高性能的企业级容器应用管理服务，支持Kubernetes社区原生应用和工具，简化云上自动化容器运行环境搭建。 您可以为CCE集群启用服务网格功能，对集群中的服务进行治理。	购买CCE集群
弹性负载均衡 ELB	弹性负载均衡（Elastic Load Balance，ELB）将访问流量自动分发到多台云服务器，扩展应用系统对外的服务能力，实现更高水平的应用容错。 您可以通过弹性负载均衡从外部访问应用服务网格。	创建共享型负载均衡器