目录

常见问题

1. Service对于Pod的流量负载均衡 当一个微服务有多个版本或实例的时候，如果没有为Serivce定义DestinationRule，Serivce是按照轮序（round-robin）的方式将请求发给它对应的多个Pod。我们可以为Serivce定义DestinationRule对象，以实现从Service到Pod的负载均衡策略。

2. VirtualService和DestinationRule的关系 VirtualService对象和DestinationRule对象既可以单独使用、也可以结合使用。功能上VirtualService的功能之一是在后端不同DestinationRule中选择一个转发请求，而Service是在后端不同Pod中选择一个转发请求。VirtualService和DestinationRule是通过subnet关联起来的。

3. Istio 与Kubernetes Kubernetes 提供了部署、升级和有限的运行流量管理能力；利用service的机制来做服务注册和发现，转发，通过kubeproxy有一定的转发和负载均衡能力。但并不具备上层如熔断、限流降级、调用链治理等能力. Istio 则很好的补齐了k8s在微服务治理上的这部分能力，同时是基于k8s构建的，但不是像SpringCloud Netflix等完全重新做一套。Istio是谷歌微服务治理上的非常关键的一环。

4. vs和k8s service的区别 如果没有 Istio virtual service，仅仅使用 k8s service 的话，那么只能实现最基本的流量负载均衡转发，但是就不能实现类似按百分比来分配流量等更加复杂、丰富、细粒度的流量控制了。