9.1.1 微服务的限流

限流、熔断、重试、超时等是微服务治理中很重要的一组概念。通过这些手段，我们保证微服务的整体可用性，微服务的限流和熔断通常是配套使用。通过限流设置，设定在某一时间窗口内对某一个或者多个微服务的请求数进行限制，保持系统的可用性和稳定性，防止因流量暴增而导致的系统运行缓慢或宕机。通过熔断设置，当一个微服务出现故障或者由于流量冲击造成微服务无法响应时，打开断路器，微服务对外停止服务。

- 熔断时：本微服务对外停止服务。
- 限流时：超过阈值的请求将不会被响应，但本微服务本身不对外停止服务。

本小节我们先介绍 Istio 中的限流，熔断我们将在下一小节中介绍。

在 Istio 中的限流可以根据客户端信息或一些条件进行限流，Istio 的限流功能基于 Mixer 架构，目前支持 Memoryquota 适配器和 Redisquota 适配器。Memquota 和 Redisquota 适配器都支持 quota template，因此，在两个适配器上启用速率限制的配置是相同的。

Memoryquota 虽然功能齐全，但该适配器不适合生产使用，仅适用于本地测试。这种限制的原因是此适配器只能用于运行一个 Mixer 的环境中，不支持 HA 配置。如果该 Mixer 单点故障，则所有重要的配额值都将丢失。企业生产里如果要使用限流功能，需要安装 Redis，然后配置 Redisquota 适配器。

为了方便，我们使用 Memoryquota 方式进行演示。在 Istio 中和限流相关的配置分为两部分：客户端和 Mixer 端。

客户端：

- QuotaSpec：用于定义客户端请求的配额实例的名称以及每个请求消费的配额实例数。
- QuotaSpecBinding ：绑定限流器影响的应用，可以将 QuotaSpec 与一个或多个 OpenShift 中的 Service 绑定。

Mixer 端：

- 配额实例 ：定义了 Mixer 如何确定配额的大小。
- Memquota：定义了 memquota 适配器配置。memquota 绑定在 Mixer 进程上且未实现持久化，只能运行在开发测试环境。

- Quota rule ：定义何时将配额实例分派给 memquota 适配器。

接下来，我们通过定义 Isito 限流的 5 个配置，以 bookinfo 为例展示 Istio 限流功能。

- QuotaSpec 的配置

首先定义 QuotaSpec 对象，内容如下：

```
apiVersion: config.istio.io/v1alpha2
kind: QuotaSpec
metadata:
  name: request-count
  namespace: istio-system
spec:
  rules:
  - quotas:
    - charge: 1
      quota: requestcountquota
```

在上述配置中，定义了名为 request-count 的 QuotaSpec 对象，并设置请求的配额实例为 requestcountquota，每个请求消费 1 个配额实例数。

- QuotaSpecBinding 的配置

定义了 QuotaSpec 之后，需要定义 QuotaSpecBinding 设定配额作用于哪个服务，内容如下：

```
apiVersion: config.istio.io/v1alpha2
kind: QuotaSpecBinding
metadata:
  name: request-count
  namespace: istio-system
spec:
  quotaSpecs:
  - name: request-count
    namespace: istio-system
  services:
```

```
    - name: productpage
      namespace: myproject
```

可以看到，将之前创建的 QuotaSpec 绑定到 myproject 下的 productpage 服务上。

- memquota 的配置

有了客户端配置，还需要配置 memquota 适配器，定义适配器 handler，内容如下：

```
   apiVersion: config.istio.io/v1alpha2
   kind: handler
   metadata:
     name: quotahandler
     namespace: istio-system
   spec:
     compiledAdapter: memquota
     params:
       quotas:
       - name: requestcountquota.instance.istio-system
         maxAmount: 500
         validDuration: 1s
         # The first matching override is applied.
         # A requestcount instance is checked against override
dimensions.
         overrides:
         # The following override applies to 'reviews' regardless
         # of the source.
         - dimensions:
             destination: reviews
           maxAmount: 1
           validDuration: 1s
         # The following override applies to 'productpage' when
         # the source is a specific ip address.
```

```
      - dimensions:
          destination: productpage
          source: "192.168.137.1"
        maxAmount: 500
        validDuration: 1s
```

可以看到，上述配置定义的限流规则是：默认所有请求，每秒钟被调用最多 500 次；如果目标服务是 reviews 的请求，被限制为每秒最多 1 次调用；如果目标服务是 productpage 且源地址是 192.168.137.1 的请求，被限制为每秒最多 1 次调用。

- 配额实例的配置

配额实例定义了流量处理的规则，它包含了对服务源和目标的定义，配置内容如下：

```
  apiVersion: config.istio.io/v1alpha2
  kind: instance
  metadata:
    name: requestcountquota
    namespace: istio-system
  spec:
    compiledTemplate: quota
    params:
      dimensions:
        source: request.headers["x-forwarded-for"] | "unknown"
        destination: destination.labels["app"] |
destination.service.name | "unknown"
        destinationVersion: destination.labels["version"] | "unknown"
```

上面配置文件逻辑如下：通过请求中的 x-forwarded-for header 获取请求源地址。判断请求目标服务是否有 app 标签，如果有，则将目的地址定义为目标服务 Service 名称；如果没有，则按照 unkonwn 处理。判断请求目标服务是否有 version 标签，如果没有，按照 unkonwn 处理。

- quota rule 的配置

定义 quota rule 将配额实例和 memquota 适配器 handler 进行绑定。内容如下：

```
apiVersion: config.istio.io/v1alpha2

kind: rule

metadata:

  name: quota

  namespace: istio-system

spec:

  # quota only applies if you are not logged in.

  # match: match(request.headers["cookie"], "user=*") == false

  actions:

  - handler: quotahandler

    instances:

    - requestcountquota
```

所有资源对象定义完成后，使用 oc create -f 创建上面的 5 个配置。

使用 curl 对 bookinfo 发起大量请求。当压力较小的时候，页面访问正常。当访问并发增多以后，页面访问出现问题，调用关系如下图 9-9 所示：

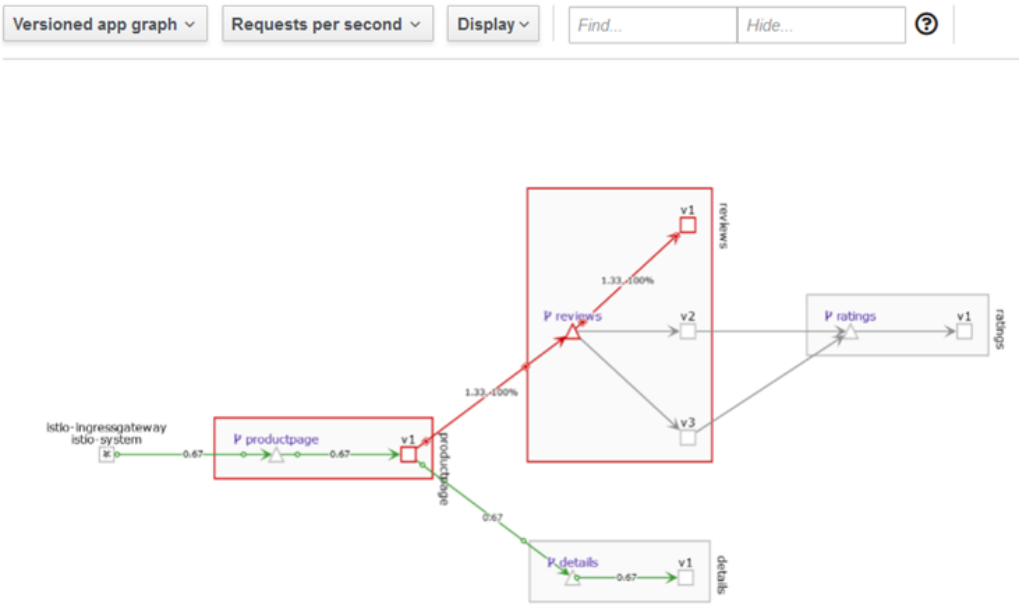

图 9-9 Kiali 展示

在 Kiali 中查看 review 服务的入口流量统计，可以看到流量被进行了限制（3OPS 代表

每秒三次调用），如下图 9-10 所示：

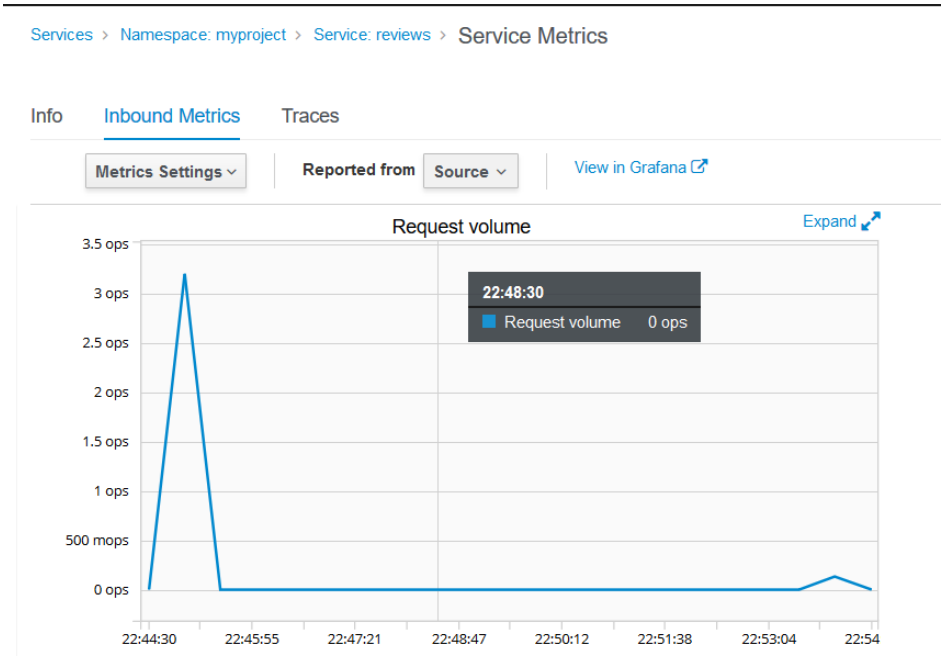

图 9-10 Kiali 展示

此时查看 Grafana，可以看出，对 productpage 的访问量增加时，会由于限流原因，出现迅速下降，如下图 9-11 所示：

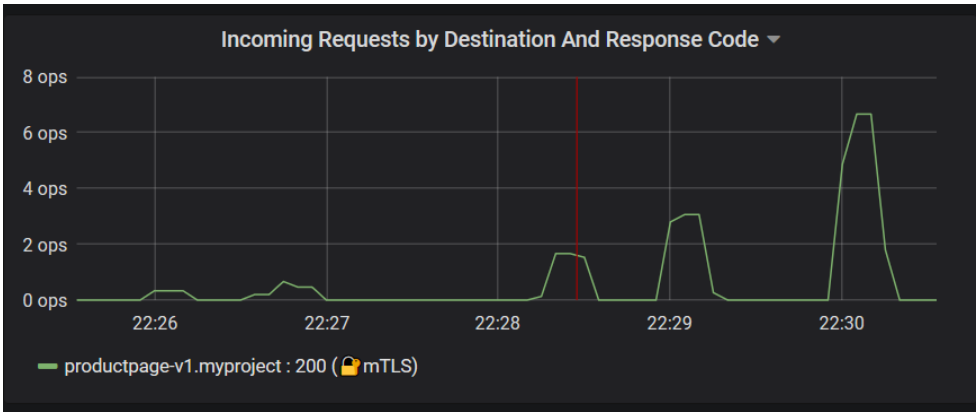

图 9-11 Grafana 展示