# Qing Gateway—高性能响应式云原生网关的设计与实现

**设计文档**

**所在赛道与赛项：A**

## 目标问题与意义价值

随着互联网与云计算的蓬勃发展，高流量、高并发的挑战与日俱增，极大促进了微服务架构的演变。分布式架构和微服务可以使后台服务变得高可用、加快开发周期并降低成本，但云原生应用程序在提供可靠连接方面带来了新挑战，对流量入口的并发性、无损性要求也越来越高。

因此，网关诞生了，现在市面上主流的网关有spring cloud gateway、zuul、openrestry，对三者进行对比如下图1-1所示：



图1-1 三种网关性能对比

可见，spring cloud gateway要比其它两款网关综合性价比高。但是spring cloud gateway的缺点如下：需要引入第三方组件才能实现负载均衡和限流，如使用Sentinel组件进行熔断限流配置；只对开发人员友好，没有管理台界面，需要引入单独的监控组件查看网关负载等实时运行信息。Qing Gateway正是为了解决这一系列问题。

Qing Gateway是一个高性能、多协议、易扩展、分布式、响应式的API网关，拥有交互友好的管理平台界面。单机网关可承受10w Qps流量，网关处理损耗仅为1-3ms，性能不输Spring Cloud Gateway。兼容主流的微服务工具：Nacos、Redis、Mysql。基于SPI机制，支持热插拔，用户可以定制化开发，满足用户各种场景的现状和未来需求。

## 设计思路与方案

### 2.1 总体设计

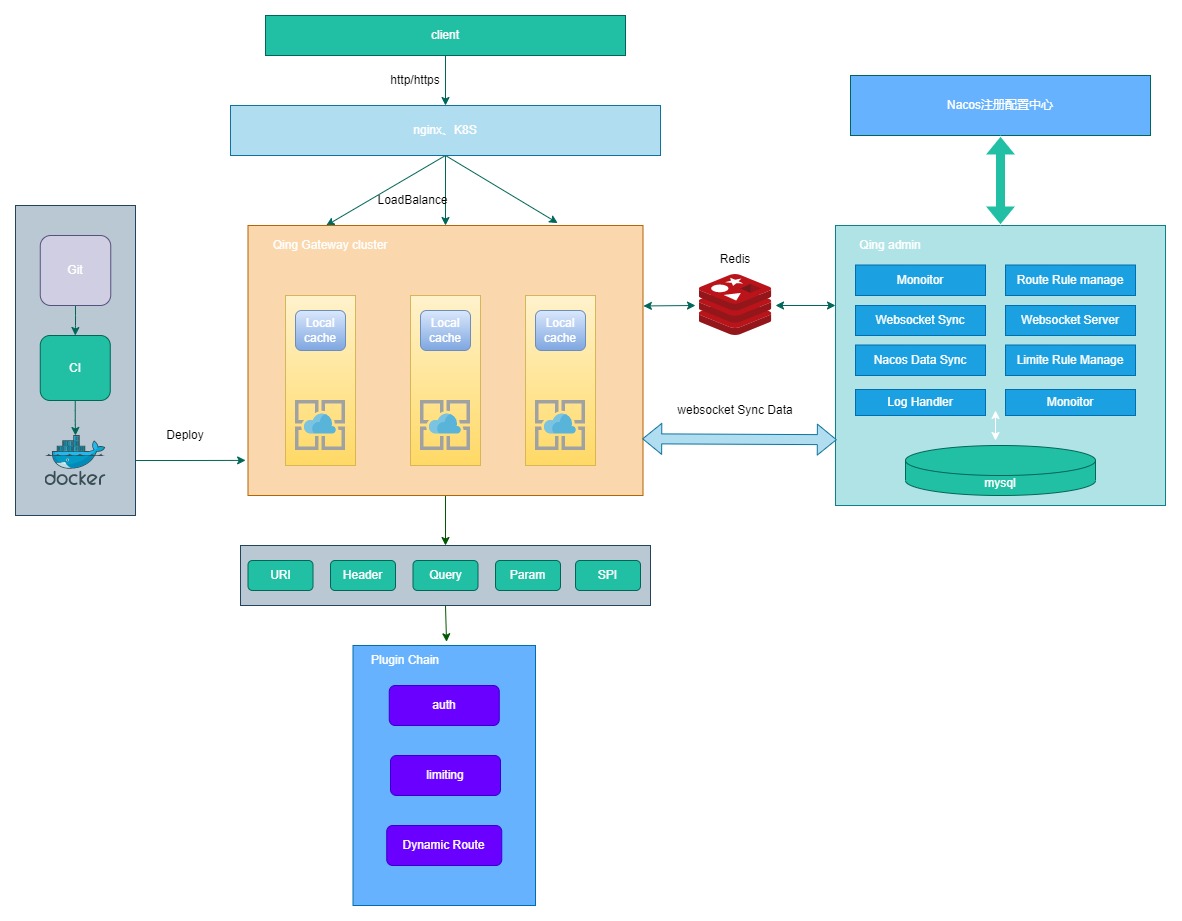


图2-1 网关整体架构

网关是一个典型的IO密集型应用，因此传统的阻塞IO（BIO）不适合高性能网关的构建，Qing Gateway Server采用同步非阻塞的I/O模型（NIO）来处理高并发、大量连接的问题，对应的框架是Spring Webflux；使用Redis作为消息队列、缓存；管理后台由Springboot+React+Mysql搭建；管理后台与网关Server之间采用WebSocket通信，管理后台统一接入多网关结点；选用Nacos作为微服务配置中心。

### 2.2 模块设计

项目主要分为两大模块：Qing Gateway server、Qing admin。

Qing Gateway server，为网关单节点的服务端，是网关逻辑的核心。分为以下几个模块介绍：

* 基于责任链设计模式、插件化设计思想，设计一条插件链，默认注册鉴权插件、限流插件、动态路由插件，除此之外，用户通过继承AbstractQingPlugin接口可自定义插件。
* 基于SPI机制，热插拔负载均衡规则。
* 缓存模块，将限流规则、路由规则、服务及实例规则等缓存至本地，使用时直接走缓存，减少网络IO次数。
* 数据同步模块是基于Websocket，与admin中心建立长连接，实时接收admin的数据，并同步至本地缓存。

Qing admin，为集成所有网关结点的数据同步中枢、消息发送中枢。分为以下几个模块介绍：

* 注册中心与网关节点配置，可在管理台配置Nacos注册中心的地址、各网关节点的Websocket通信的地址。



图2-2 系统配置

* 动态路由配置，可根据路由断言+转发的目标服务+优先级动态添加路由匹配规则，热更新机制，无需重启。当请求来到网关时，Qing Gateway将根据配置的负载均衡策略选择目标服务中的一个实例进行转发。



图2-3 动态路由

* 服务实例配置，如图2-4所示，admin可以对服务实例的协议、版本号、集群名、权重作热更新，同步至Nacos。



图2-4 服务实例配置

* 限流规则配置，支持对单账号、单接口进行分钟级、秒级限流熔断策略配置，如图2-5所示。



图2-5 限流规则配置

* 实时监控中心，如图2-6所示，系统实时监控网关节点的QPS、内存使用情况、JVM使用情况，以便开发运维人员快速定位问题。



图2-6 实时监控中心

* 数据同步模块，开启一个后台子线程去轮询拉取Naocs中心的在线服务及实例数据，开启一个线程池去维护各网关节点的Websocket通道。
* 事件侦听器，当admin有要通知网关节点的时刻，触发异步通知事件：限流事件、规则事件、服务及实例事件。

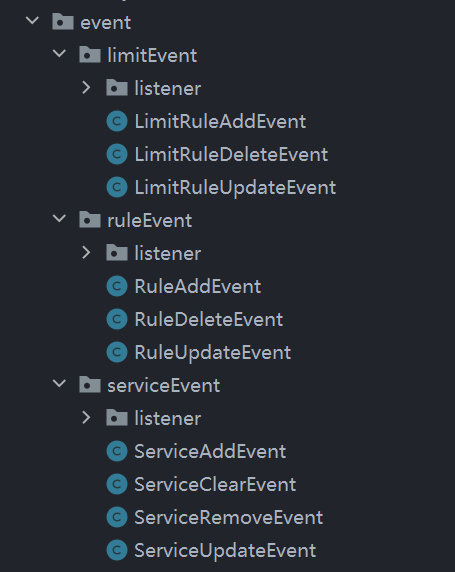


图2-7 事件侦听

* 鉴权控制器，admin项目采用jwt token作为身份校验工具，对密码采用MD5加密算法。
* 调用行为日志分析，如图2-8所示，可查看访问者源IP、源请求路径、代理URI、目标服务实例等信息。



图2-8 调取行为日志

* 负载均衡策略配置，通过SPI机制，负载均衡策略可热插拔配置。

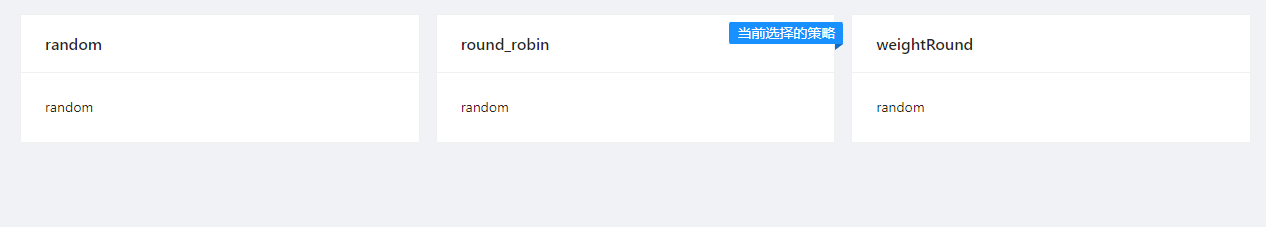


图2-9 负载均衡配置

* 访问者IP热力图，通过调用百度地图开放API，可以对日志中的调用者进行定位，最终处理的数据由前端展示出热力图，方便用户画像。



图2-10 访问IP热力图

## 方案实现

### 3.1 插件链设计

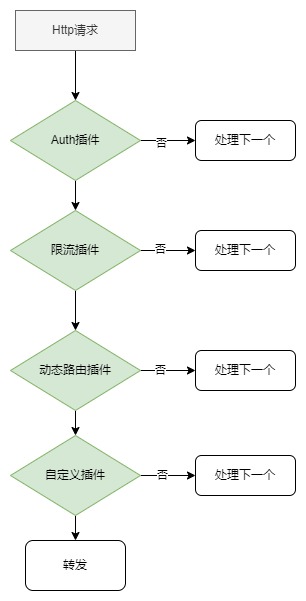


图3-1 插件链

如图3-1所示，基于责任链设计模式和插件化思想，将网关的处理逻辑抽象成多个插件，按顺序去处理推到下一层或拒绝服务。

### 3.2 动态路由设计

网关使用Nacos作为微服务统一注册中心，因此路由规则主要有路由断言、服务名、优先级组成。网关会将匹配路由断言的请求，在目标服务的所有实例中根据配置的负载均衡策略选择一个实例转发。如下图3-2的规则，可以将/example1/medical/getMedical的服务转发至medical服务下。



图3-2 路由规则

### 3.3 熔断限流设计

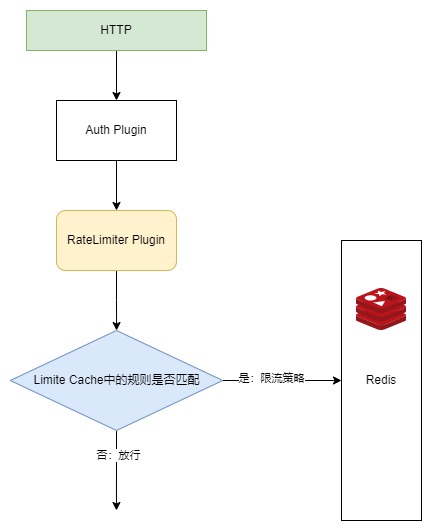
****

图3-3 限流流程

网关目前基于Redis，支持计数器限流、滑动窗口限流、漏桶限流、令牌桶限流四种策略。同样使用SPI热插拔技术，可动态配置。

### 3.4 SPI扩展设计

SPI 全称为 Service Provider Interface, 是 JDK 内置的一种服务提供发现功能,一种动态替换发现的机制。

目前支持负载均衡扩展、RateLimiter扩展，用户分别实现LoadBalance与RateLimiterAlgorithm接口即可。

### 3.5 平滑发布设计

针对每个服务实例，可设置其版本号来完成平滑发布。如下图3-4，当124.222.224.173:8901的服务版本升级为2.0后，网关将服务无缝衔接转发到该服务下的最高版本实例，随后将1.0版本的实例逐步升级至2.0，完成平滑发布。



图3-4 设置版本号

### 3.6 分布式数据同步设计

为了提升网关的性能，Qing Gateway将所有的路由规则、限流规则、服务及实例数据缓存在JVM 内存里面。在集群部署/分布式场景中，Qing Gateway自主研发了一套将Admin控制台的数据，远程同步到每一个Qing Gateway网关节点JVM内存的方案，流程如图3-5所示。

Qing admin开启一个子线程轮询拉取Nacos数据，首次启动时，将Nacos的数据全量同步至网关节点，之后采用数据增量同步的策略，减少数据包的大小，加快消息传输速度。当在控制台更改路由规则、实例信息、限流规则时，会触发Spring的发布事件机制，异步将消息通知网关节点。

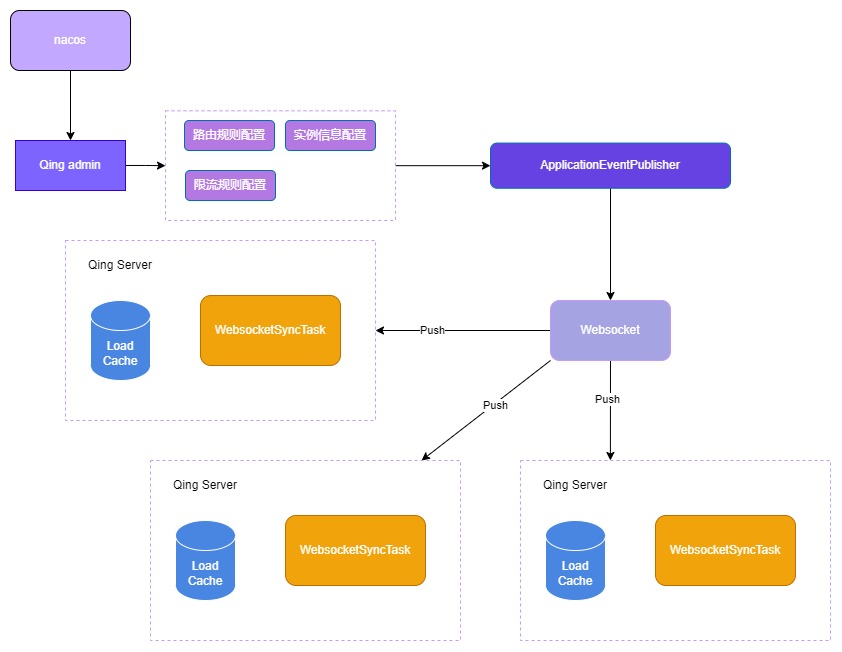
****

图3-5 分布式数据同步方案

### 3.7 实时监控设计

如下图3-6所示，各个网关节点通过一个子线程每1秒查询自身QPS、memory、JVM运行状态，写入Redis中，管理台前端每2s进行一次数据查询，admin后台将从Redis中读到的数据返回给前端。

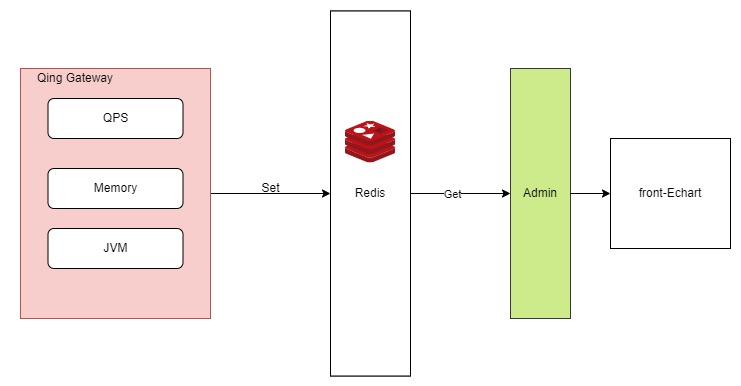


图3-6 实时监控方案

## 运行结果/应用效果

### 4.1网关插件链损耗

经实验得，如图4-1所示，网关插件链的平均损耗是1-3ms，对整个链路几乎无损。

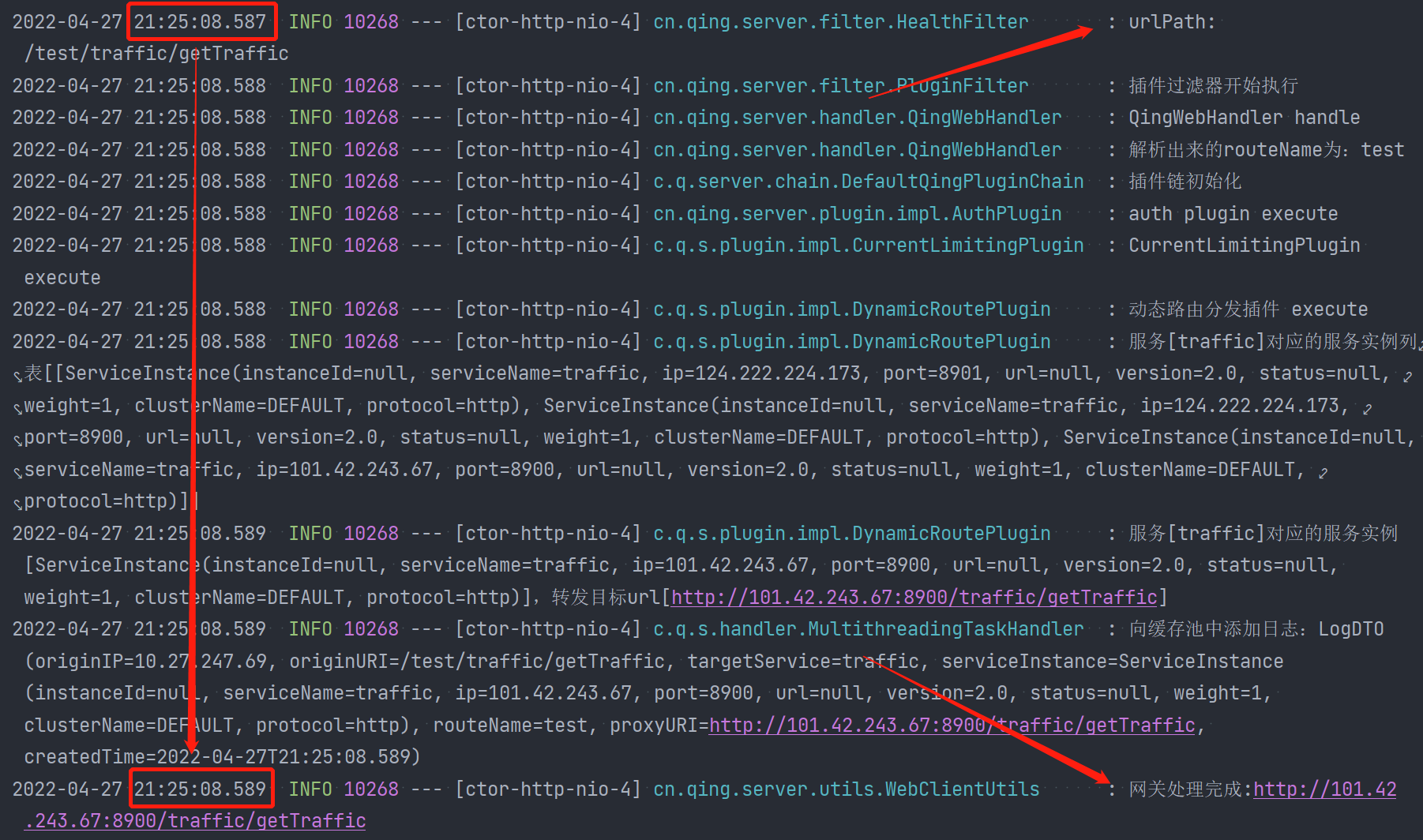


图4-1 插件链损耗

### 4.2 单机网关性能测试

如图4-2所示，在4核16G的环境对网关单机测试，设置线程数为1000，2秒内发送，循环200次，也就是10w的QPS，错误率为0.13%，因此得出结论，网关单节点能承受10w的并发量。

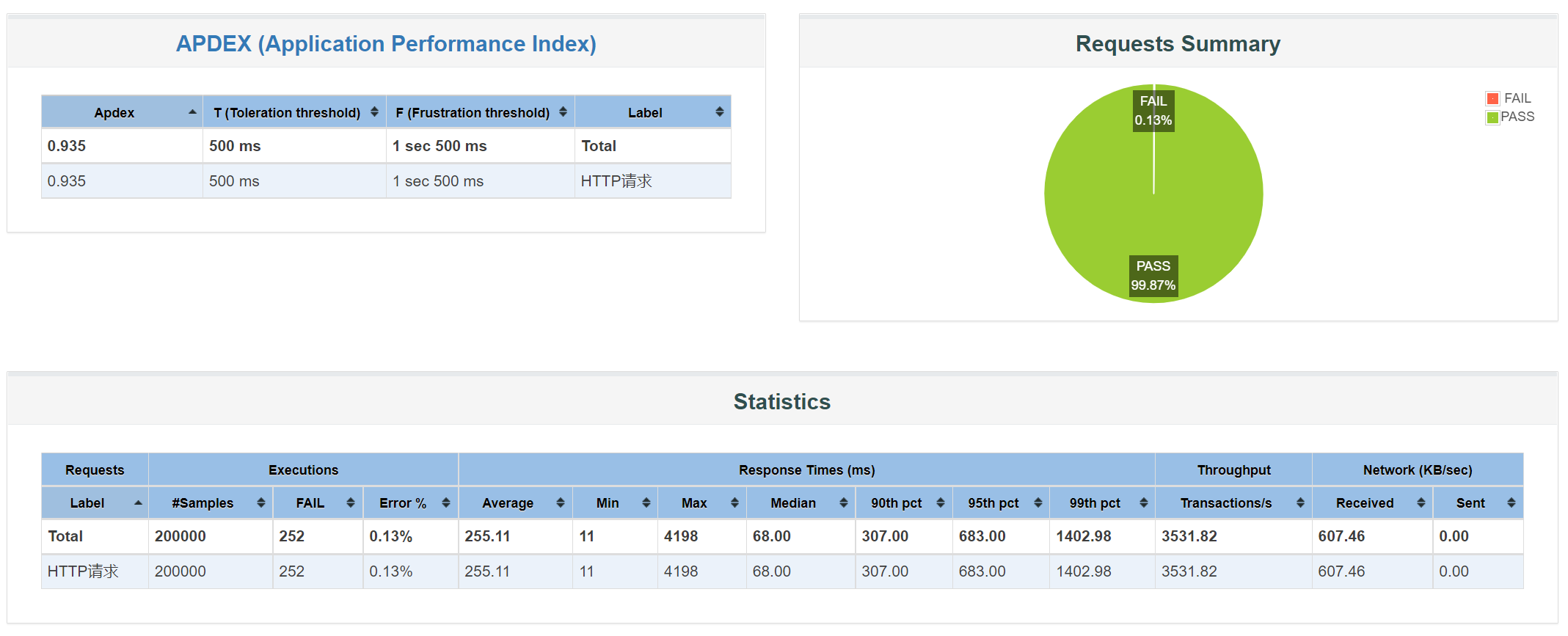


图4-2 单机网关性能

### 4.3 网关集群性能测试

分别在一台4核、16G的，三台2核4G的服务器搭建四个网关节点，使用Nginx按照4，1，1，1权重做负载均衡。如图4-3所示，设置线程数为1000，2秒内发送，循环400次，也就是20w的QPS，错误率为0.01%。因此得出结论，集群模式下，可承受至少40w的QPS。

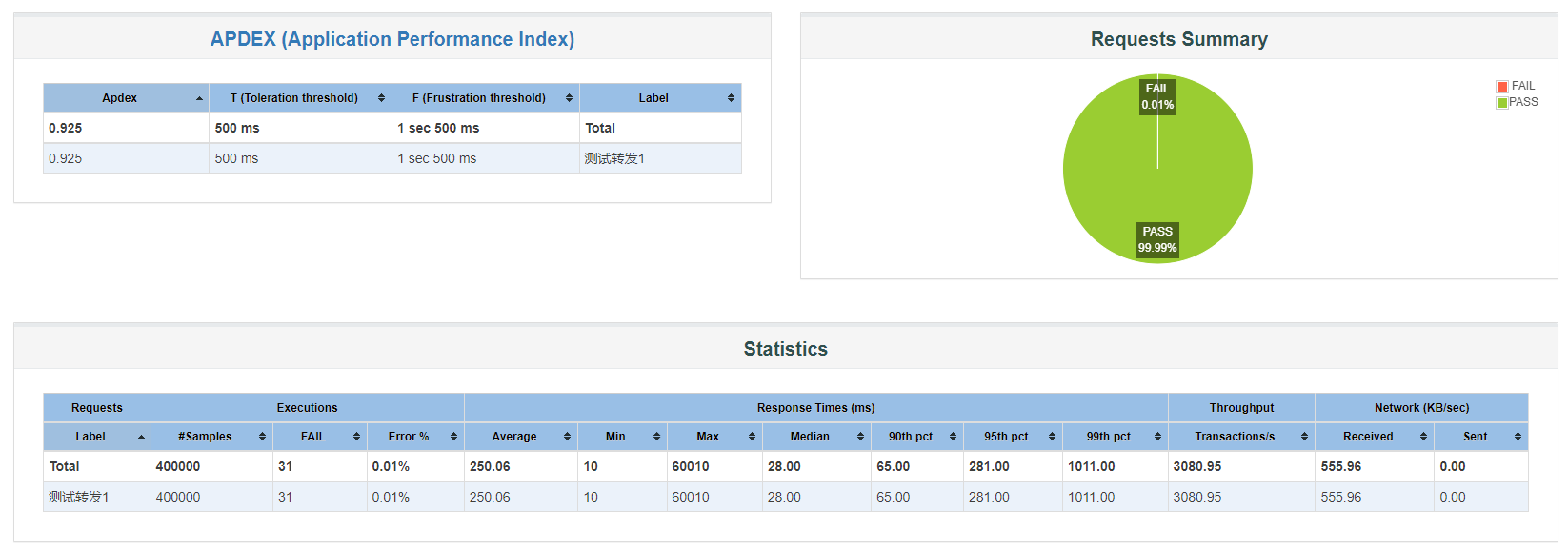


图4-3 网关集群性能

### 4.4 使用及部署

#### 4.4.1本地启动

1. 下载代码：

git clone https://github.com/conghuhu/qing-gateway.git  
cd qing-gateway  
mvn clean install

1. 在qing-admin下的application-dev文件配置Mysql和Redis连接信息。
2. 在qing-server下的application-dev文件配置Redis连接信息。
3. 在Mysql中运行qing\_gateway.sql文件。
4. 运行qing-admin下的cn.qing.admin.QingAdminApplication作为admin。
5. 运行qing-server下的cn.qing.server.QingServerApplication作为网关节点server。
6. 打开localhost: 8080/static/index.html就可以看到控制台界面，默认用户名和密码为admin、123456。

#### 4.4.2 docker部署

将qing\_admin中的admin\_dockerfile和其target下的qing\_admin.jar放入服务器一个文件夹下，运行docker build -f ./admin\_dockerfile -t qingAdmin .打包成image；然后运行docker run -it -d --name qingAdmin -p 8080:8080 qingAdmin。至此，网关管理平台已成功启动。

将qing\_admin中的server\_dockerfile和其target下的server.jar放入服务器一个文件夹下，运行docker build -f ./server\_dockerfile -t qingServer .打包成image；然后运行docker run -it -d --name qingServer -p 8101:8101 qingServer。至此，单网关节点已成功启动。如果想搭建网关集群，多台服务器重复该步骤即可，最后使用nginx对所有网关节点做反向代理。

#### 4.4.3 K8S部署

执行4.4.2后，本地应该有qingAdmin和qingServer两个镜像，使用docker push将两个镜像推送至dockerhub，这样基于K8S部署就方便了。

对于admin和server，要分开部署，各自都创建Service.yaml和Deployment.yaml，想要部署的副本数在Deployment中设置即可。



## 创新与特色

本项目总结技术点如下：

* 运用多种设计模式：责任链模式、单例模式、工厂模式、侦听器模式，增强了代码的可读性、维护性和扩展性。
* 基于NIO异步非阻塞、响应式模式，增大了网关吞吐量。
* 多协议支持：http/https、Dubbo、GRPC。
* 将路由规则、服务实例数据等信息缓存至JVM内存中，使网关前期处理请求的损耗极低，仅1-3ms。
* 针对日志处理、QPS收集采用阻塞队列，合并处理的方式，减少网络IO次数，极大提高性能。
* 插件化的思想，增强了用户的自定义扩展性。
* 对访问者IP进行地图热力图处理，方便快速定位用户画像。
* 积极向云原生靠拢，支持部署在docker、K8S环境中。