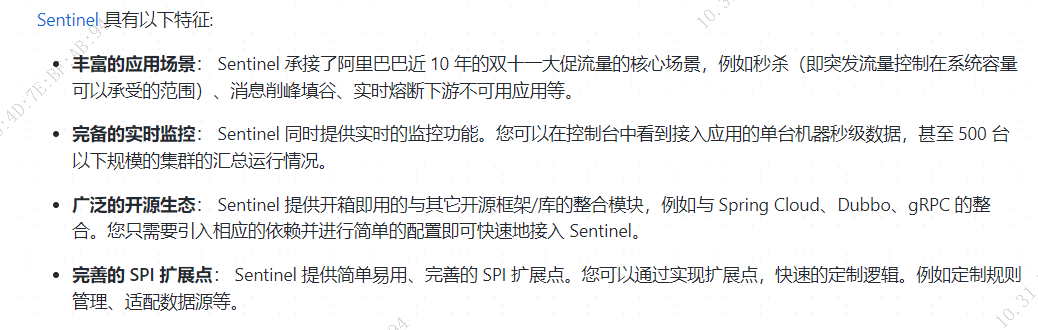
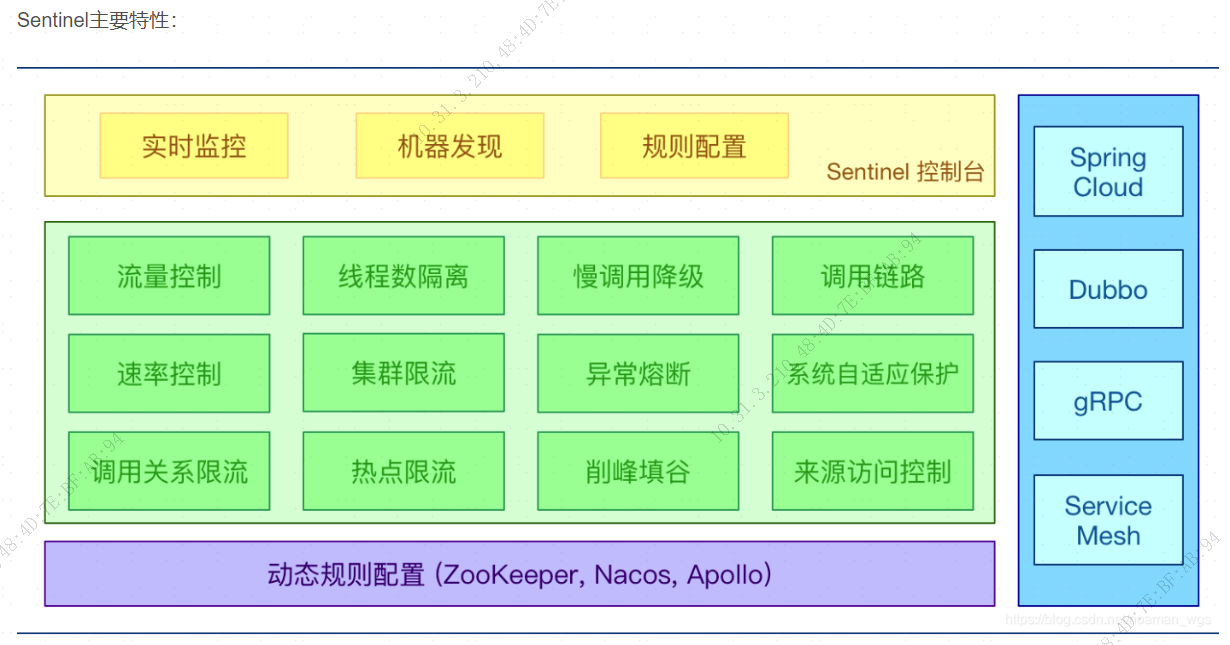
# Sentinel

## 1.Sentinel简介

### 1.1什么是Sentinel?





### 1.2 Sentinel的官网？

<https://github.com/alibaba/spring-cloud-alibaba/wiki/Sentinel>

### 1.3 Sentinel对比



## 2.安装启动

### 2.1安装

1. Pom文件引入如下依赖

<dependency>

<groupId>com.alibaba.cloud</groupId>

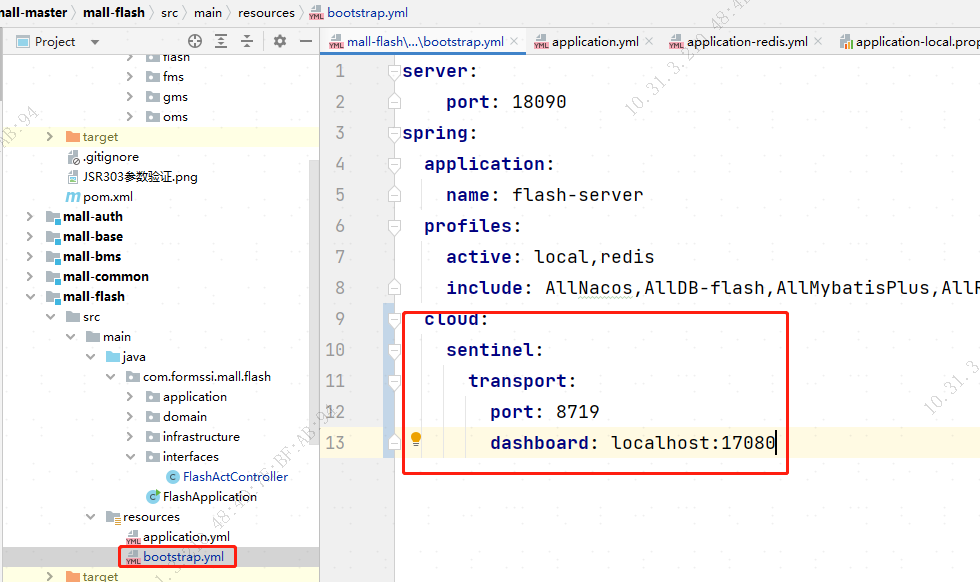
<artifactId>spring-cloud-starter-alibaba-sentinel</artifactId>

</dependency>

## 3搭建控制台

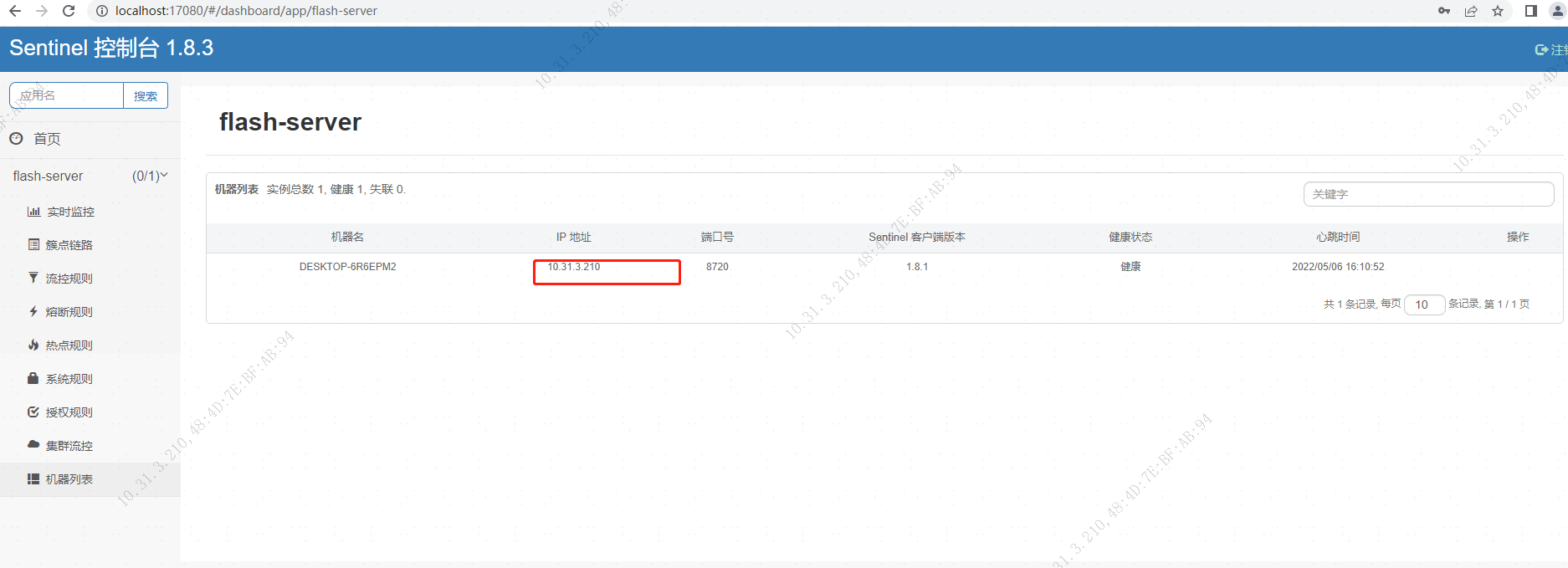
### 3.1 搭建步骤

* 下载jar包。地址：https://github.com/alibaba/Sentinel/releases
* 启动控制台。在jar包对应路径执行命令：java -Dserver.port=17080 -Dcsp.sentinel.dashboard.server=localhost:17080 -Dproject.name=sentinel-dashboard -jar sentinel-dashboard.jar
* 项目新增配置。spring.cloud.sentinel.transport.port 端口配置会在应用对应的机器上启动一个 Http Server，该 Server 会与 Sentinel 控制台做交互。比如 Sentinel 控制台添加了一个限流规则，会把规则数据 push 给这个 Http Server 接收，Http Server 再将规则注册到 Sentinel 中



### 3.2 验证

启动项目,机器列表出现了微服务的ip，表示微服务和控制台链接成功。



## 4.如何使用

### 4.1 流控

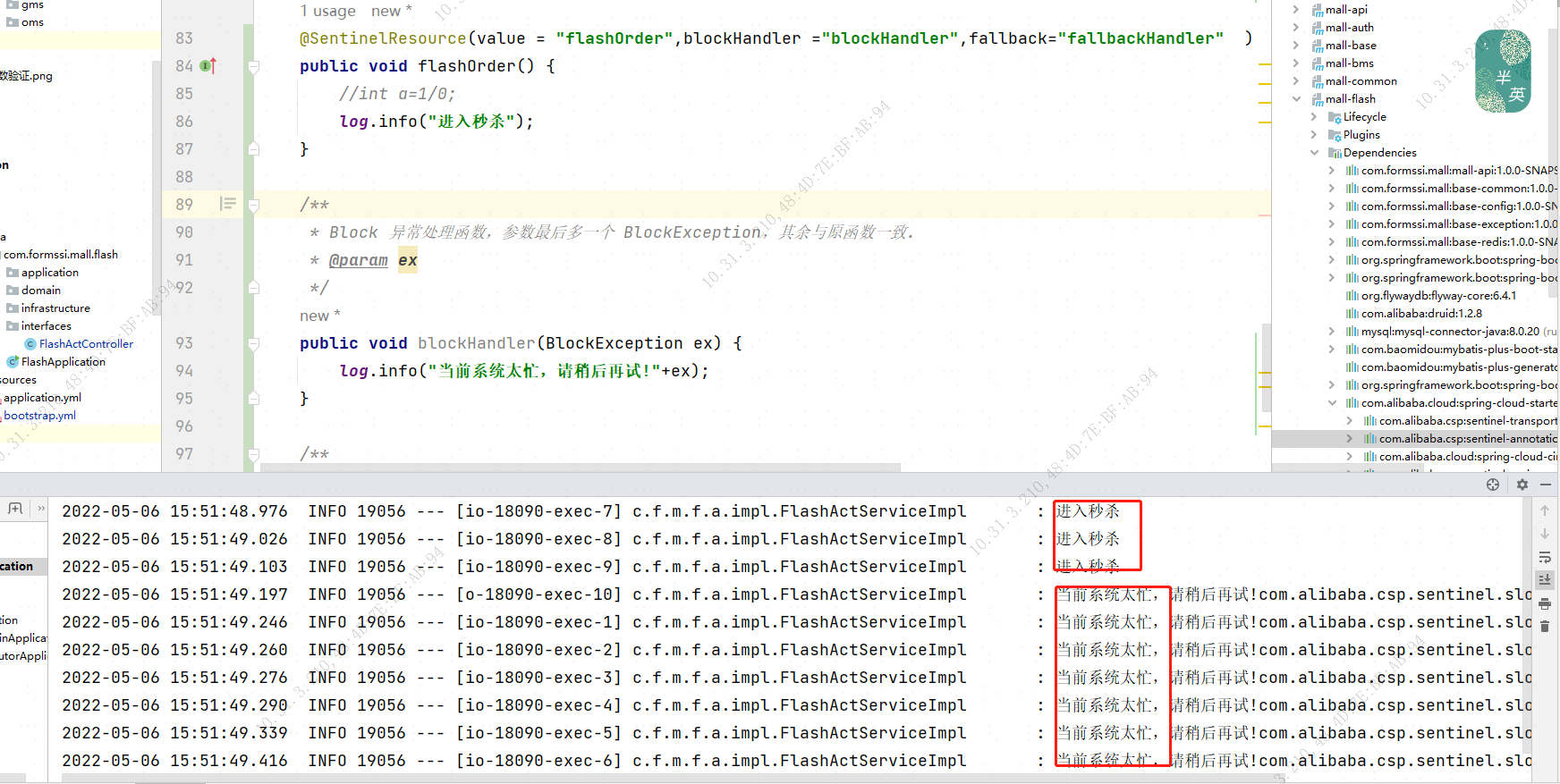


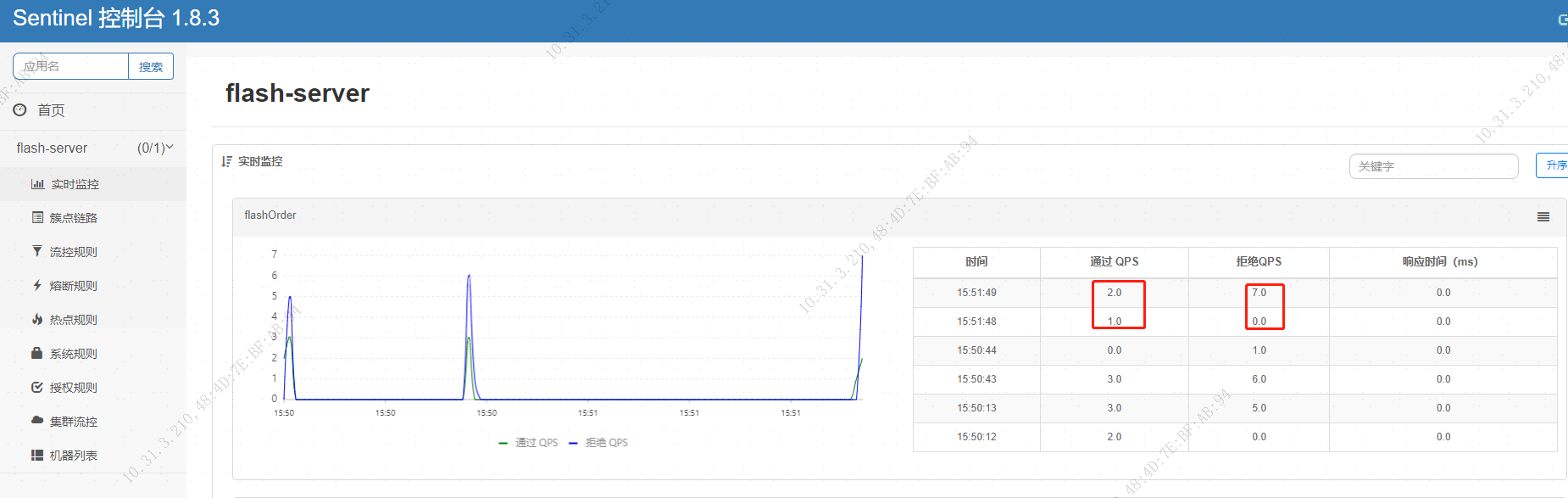
配置说明：

|  |  |
| --- | --- |
|  | 说明 |
| 1 | 必填的注解只有一个，就是value |
| 2 | 2个异常处理方法都配置了，限流会走blockhandler。其他异常会走fallback |
| 3 | 若配置了fallback，没有配置blockhandler，限流后，会走fallback。 |
| 4 | Block 异常处理函数，参数最后多一个 BlockException，其余与原函数一致. |

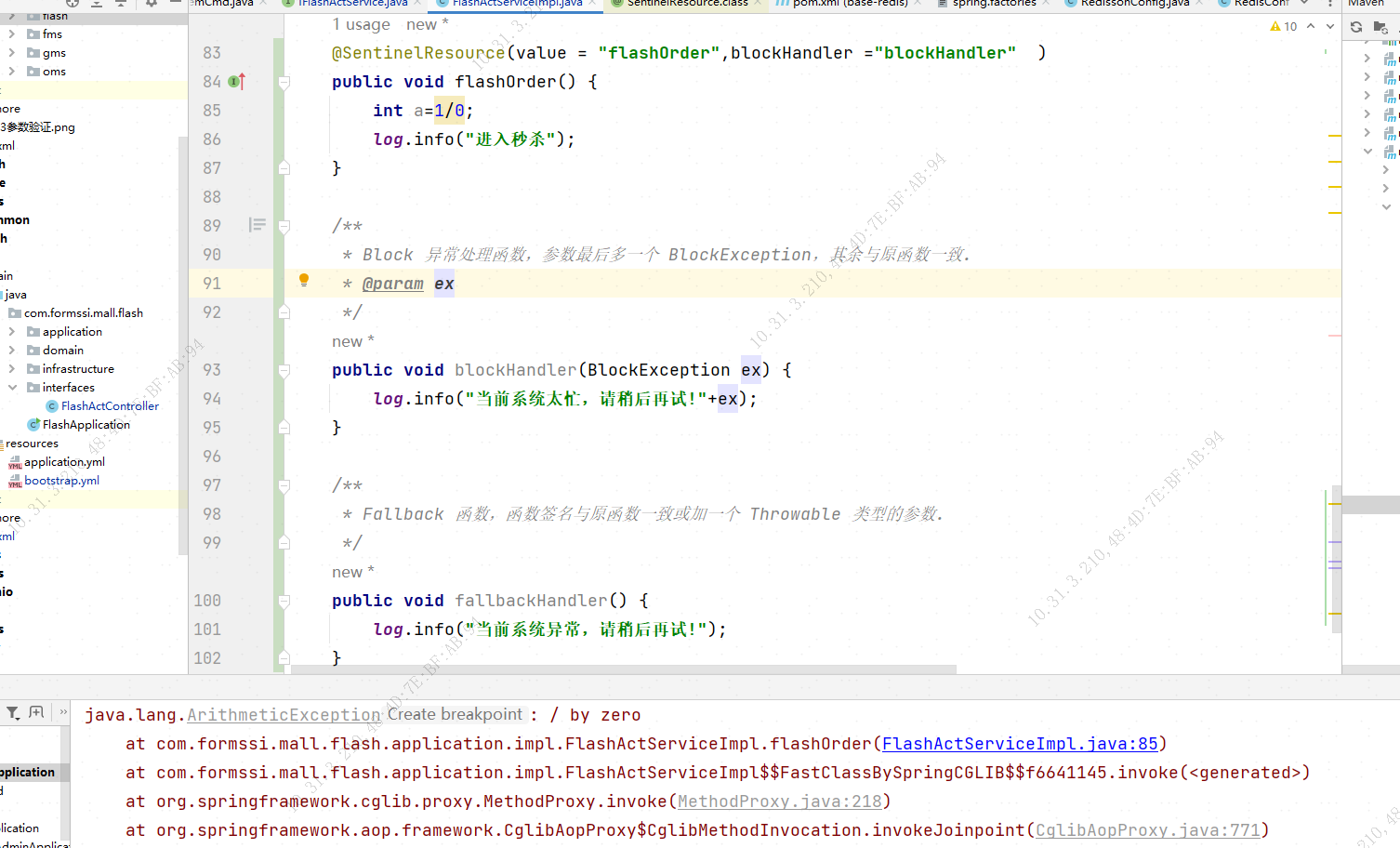


使用postman请求10次。结果如下：





若没配置了fallback，只配置blockhandler，报错会直接抛出。

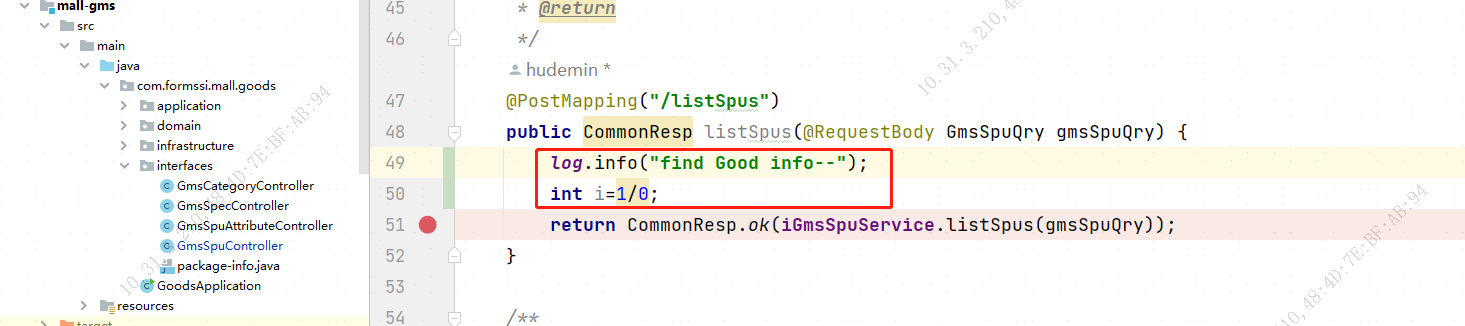


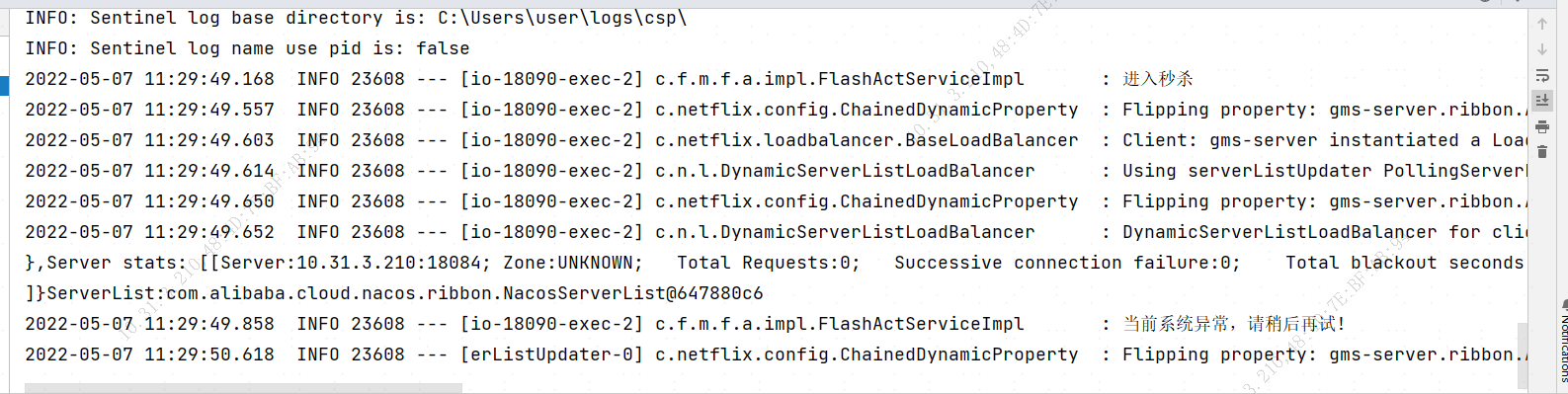
### 4.2 降级

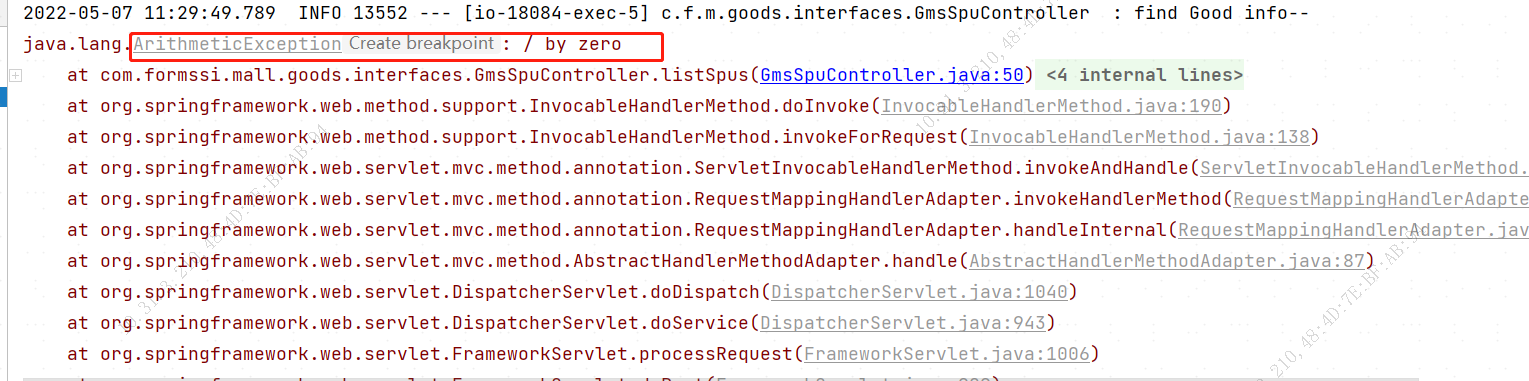
* 没有使用sentinal降级。



商品服务，加一会代码，模拟报错。







* 集成sentinal降级处理







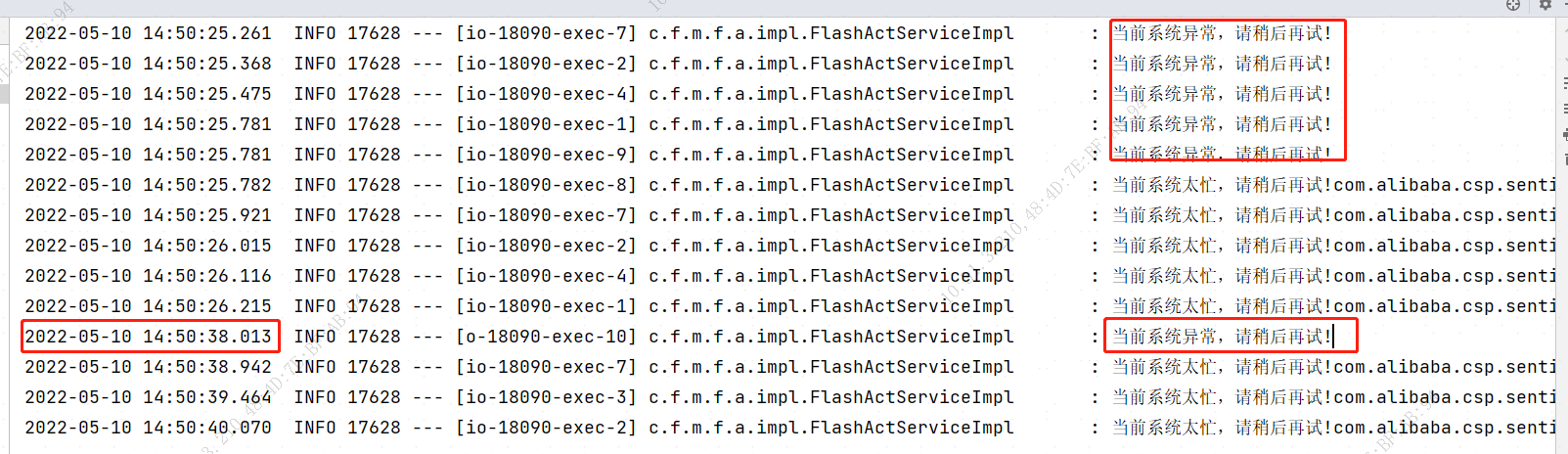
再次请求



### 4.3 熔断

设置熔断规则

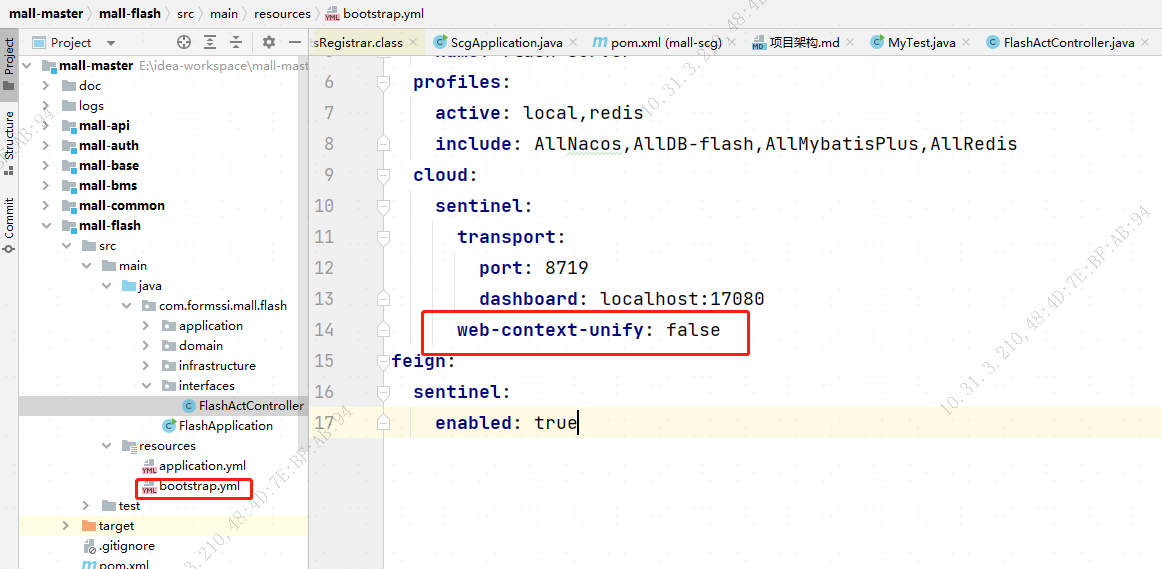




如图，请求5次，都是异常，触发熔断。熔断期间的请求都是“提示太忙“。

熔断时长是10s，如图红框时间，结束熔断，进入半开状态。此时请求再次异常，继续进入熔断状态。

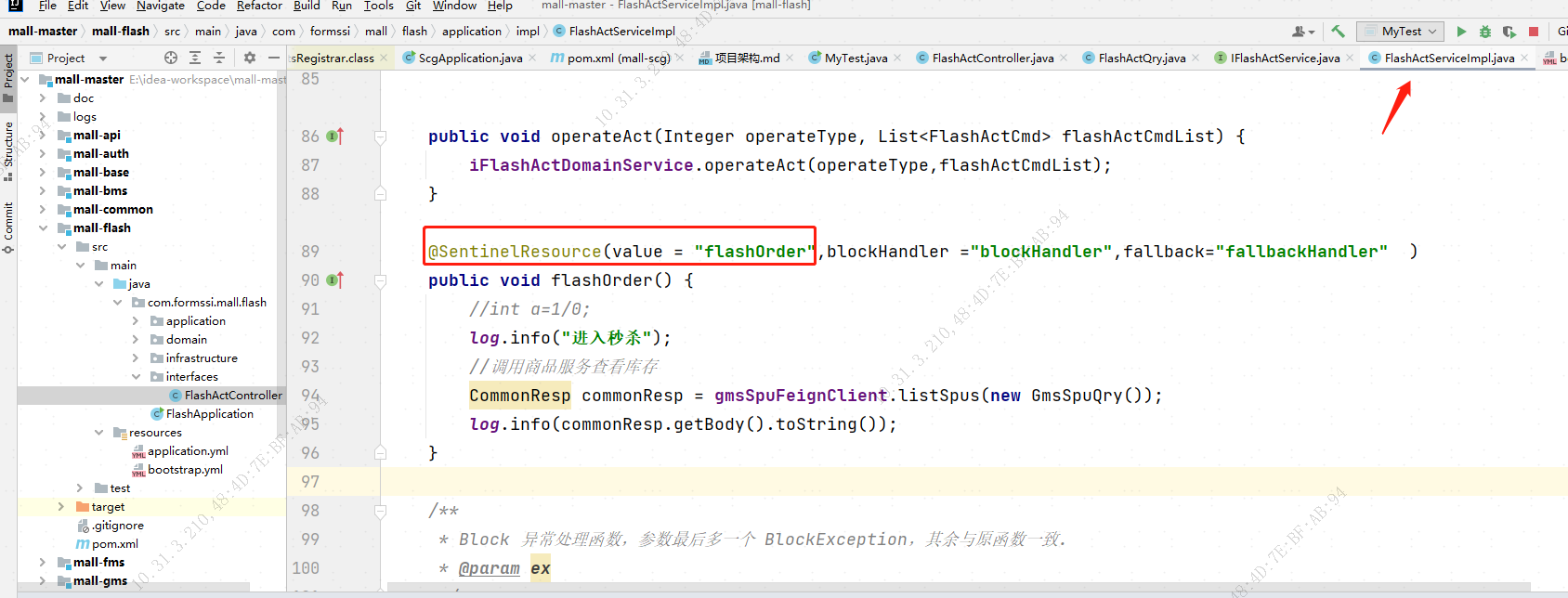
### 44链路



新增配置web-context-unify: false，打开链路，默认是关闭的。

启动服务后，发现链路多了一级



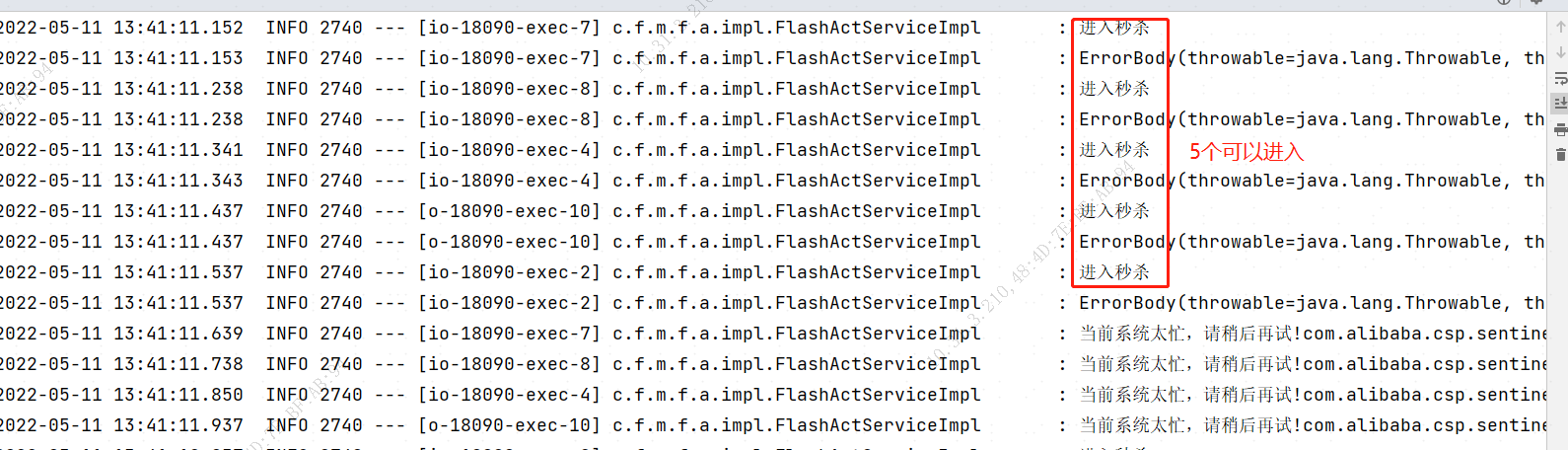


如图资源是service

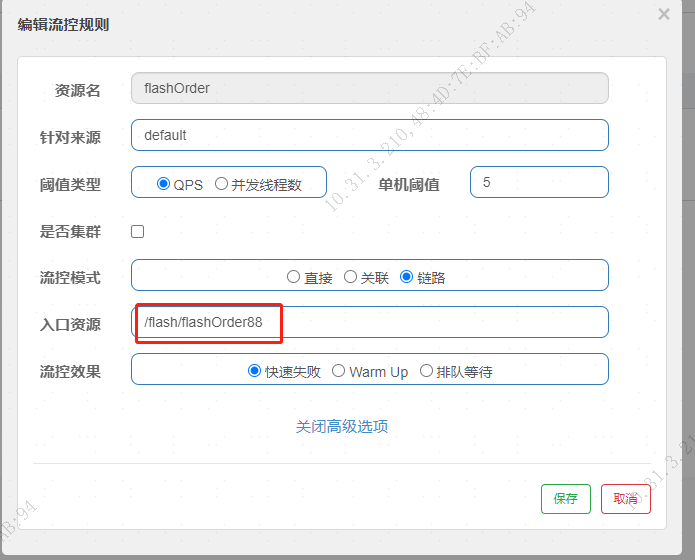


配置如上。入口资源是调用的链接。

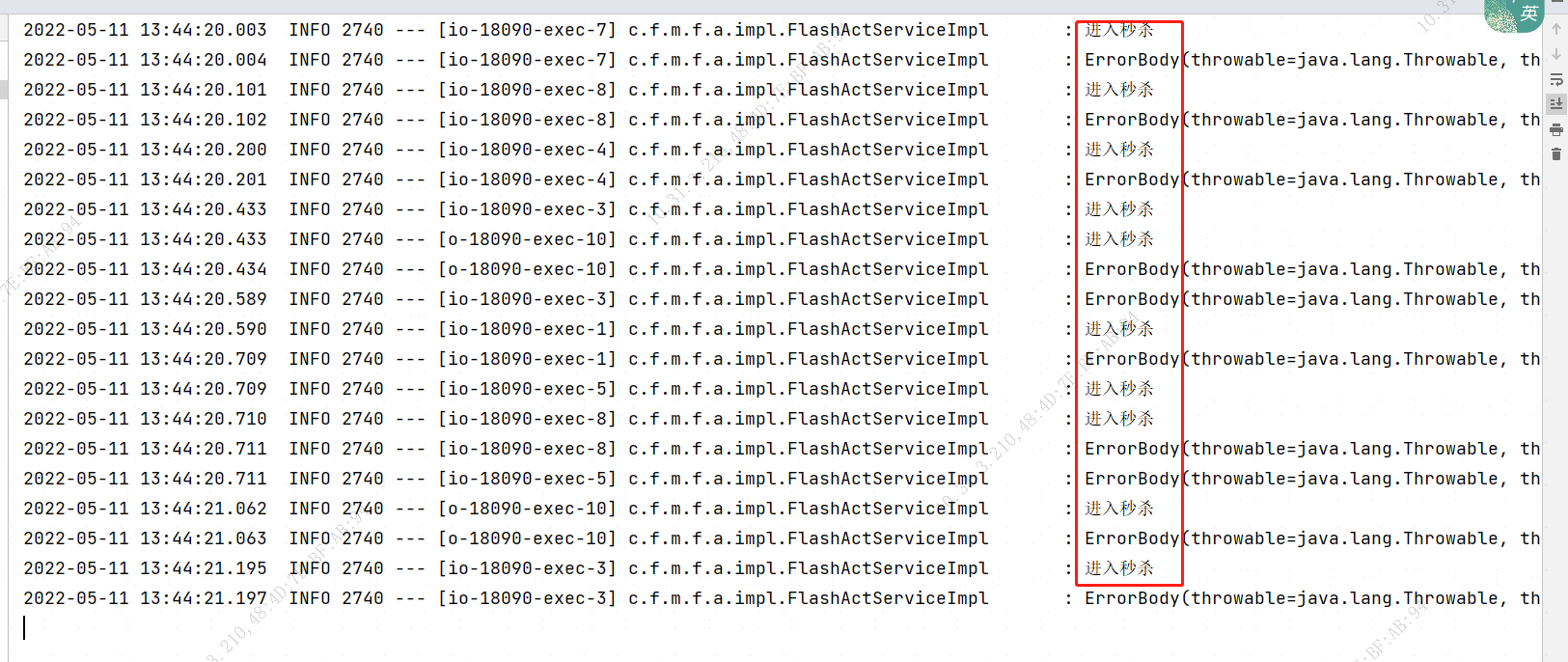
一秒发送10表请求，如下，超过5个，就触发流控。



修改规则，入口资源修改为一个不存在的，如下图



一秒发送10表请求，如下，10个全部成功，没有触发流控。



### 4.5流量变形

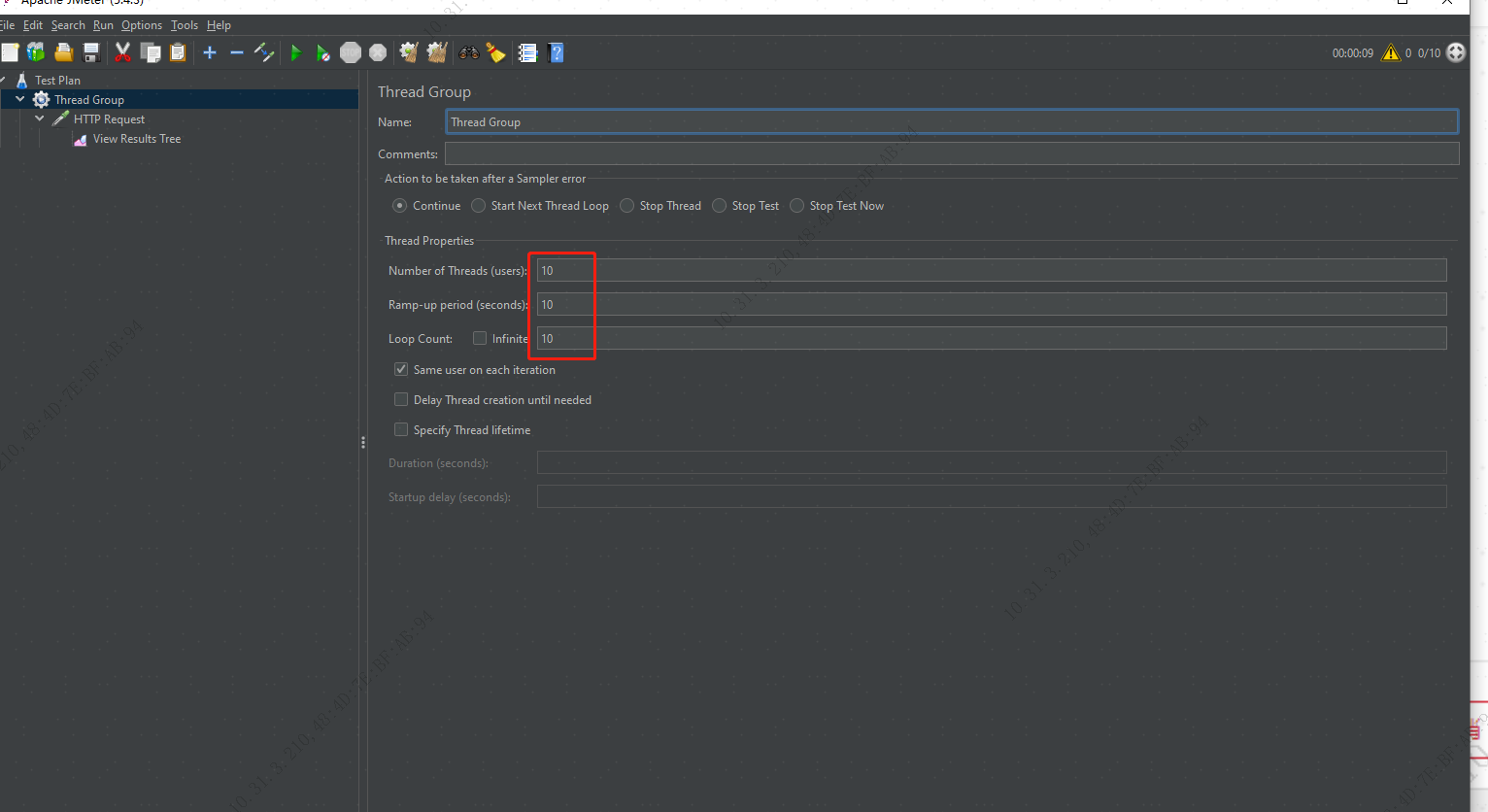
#### 快速失败

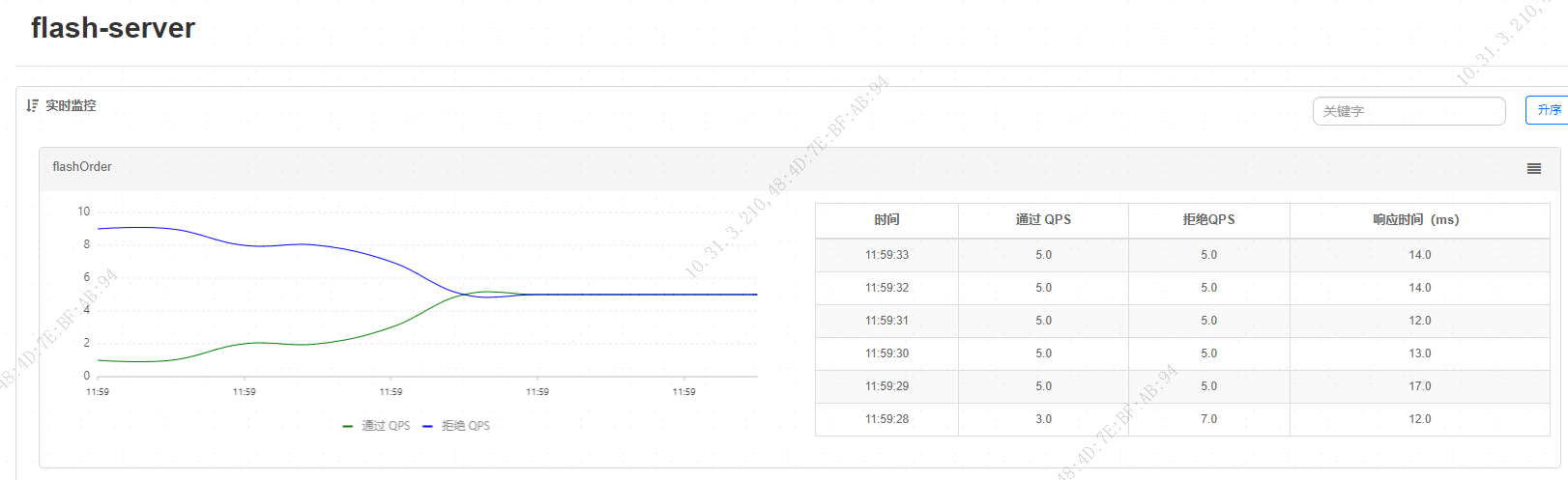
抛出Blocked by Sentinel (flow limiting)异常，前面的例子用的就是这项配置

#### 预热warm UP



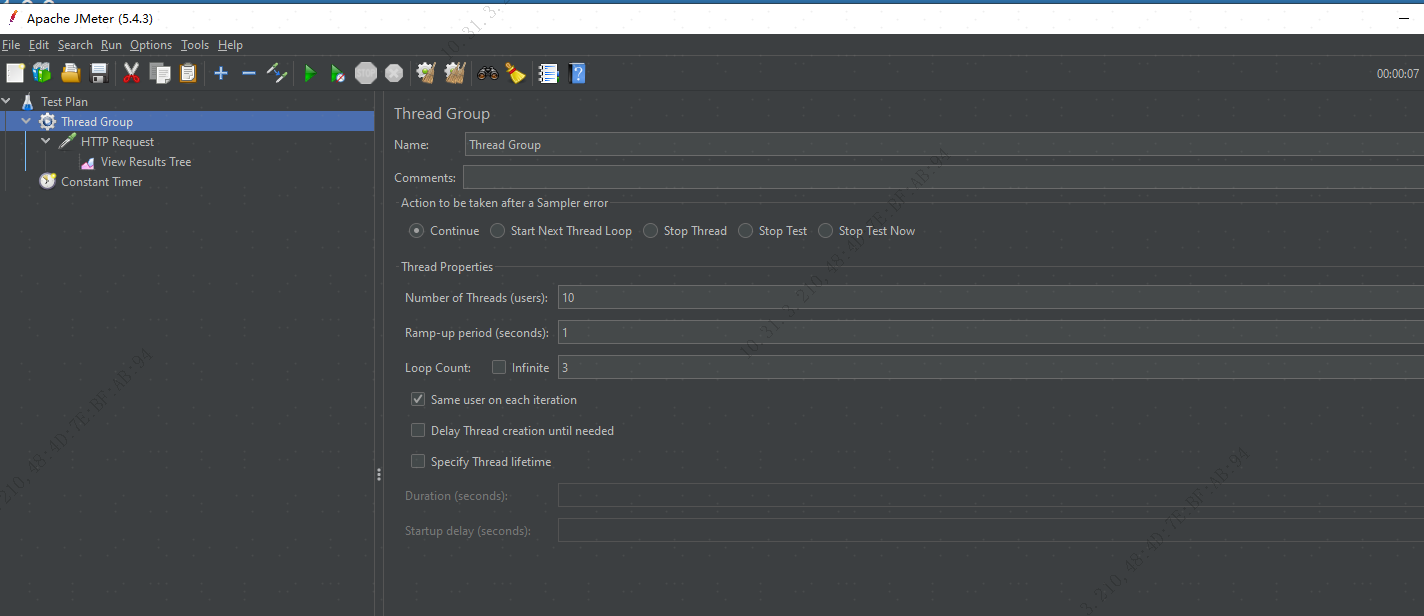
默认第一秒qps是1，3秒内增长到5



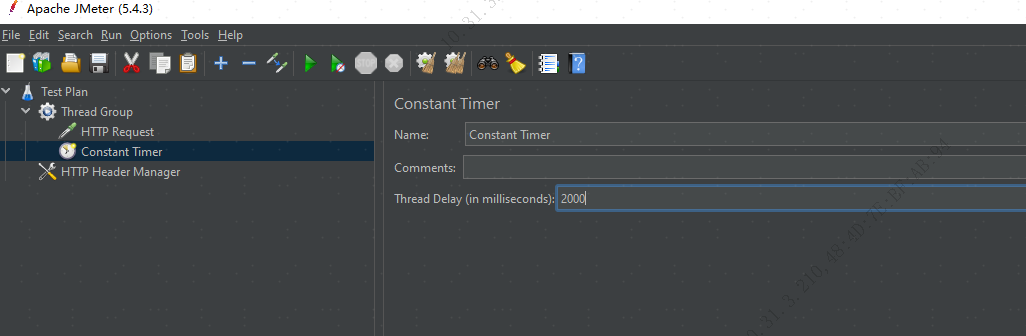


#### 排队等待

每秒10个请求，共3次，30个请求。



设置定时器，间隔2s发送一次请求



设置为快速失败的模式，



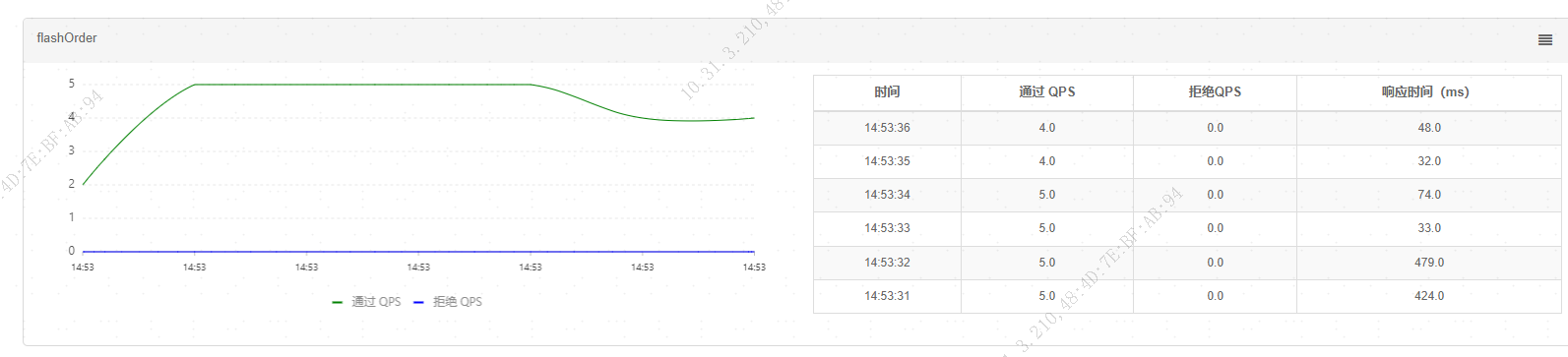
结果如下：通过一半请求，拒绝一半。且忙闲不均。



设置等待排队的模式



拒绝的QPS为0，请求分大概6次过来。每次的10个请求，分2次处理。



### 4.6 热点数据

找一个带参数的请求测试。参数类型必须是java基本数据类型或者string

如图找活动审核

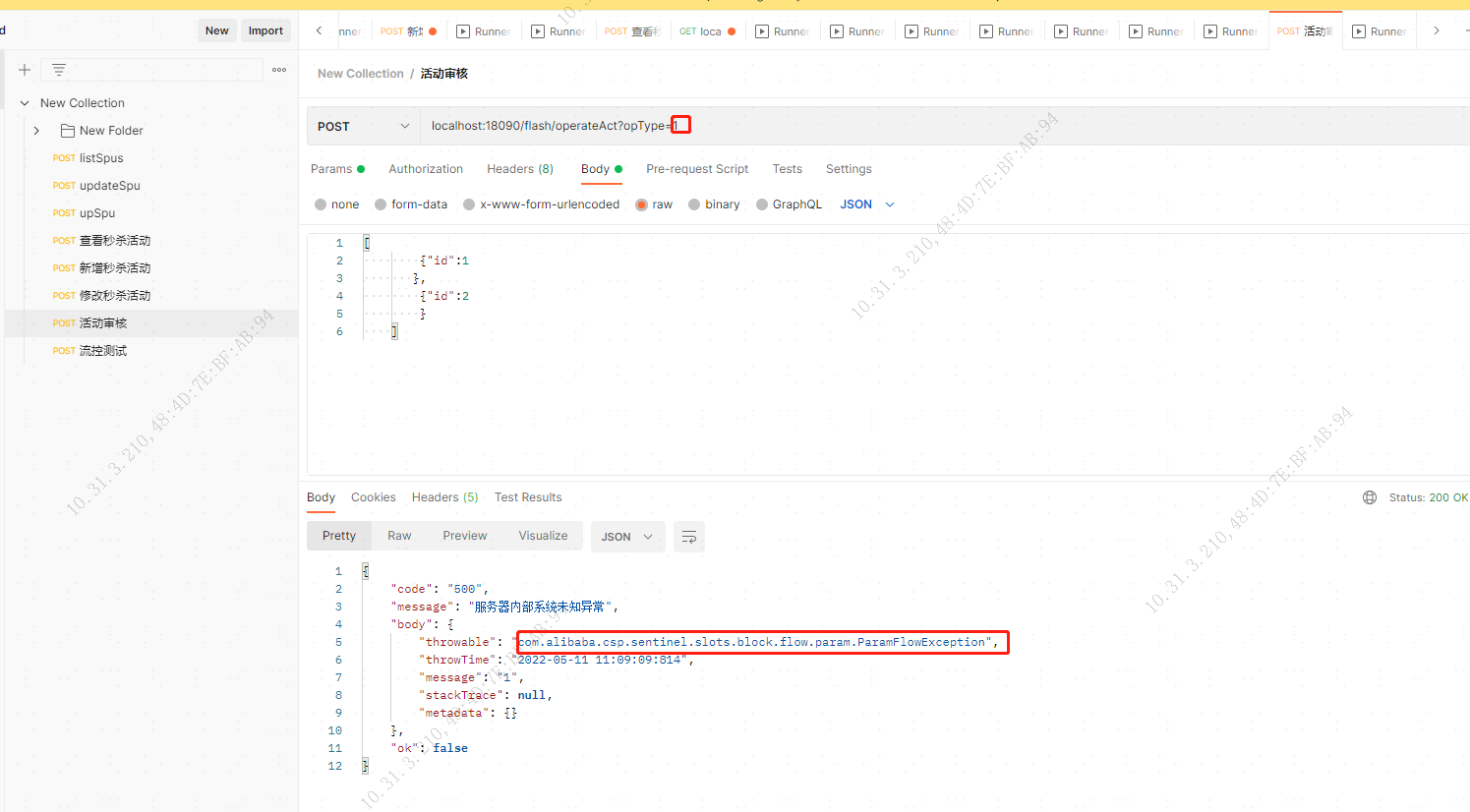


设置热点规则，索引是0，表示第一个参数opType



设置规则，参数值为1限流qps 为2.

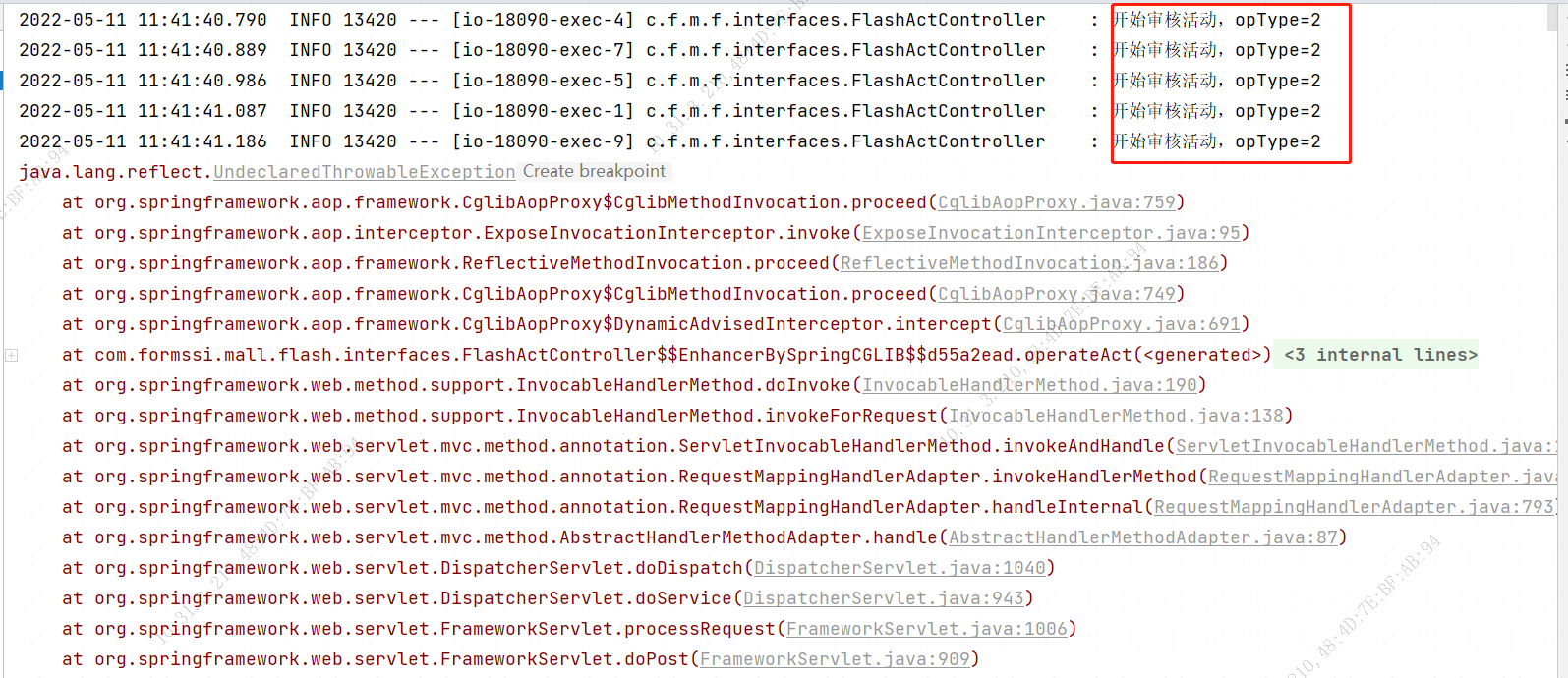
通过postman连续发送请求(opType=1)，错误提示如下：表示限流生效。



看后台，超过2次，就报错。



通过postman连续发送请求(opType=2)，超过5才会报错。

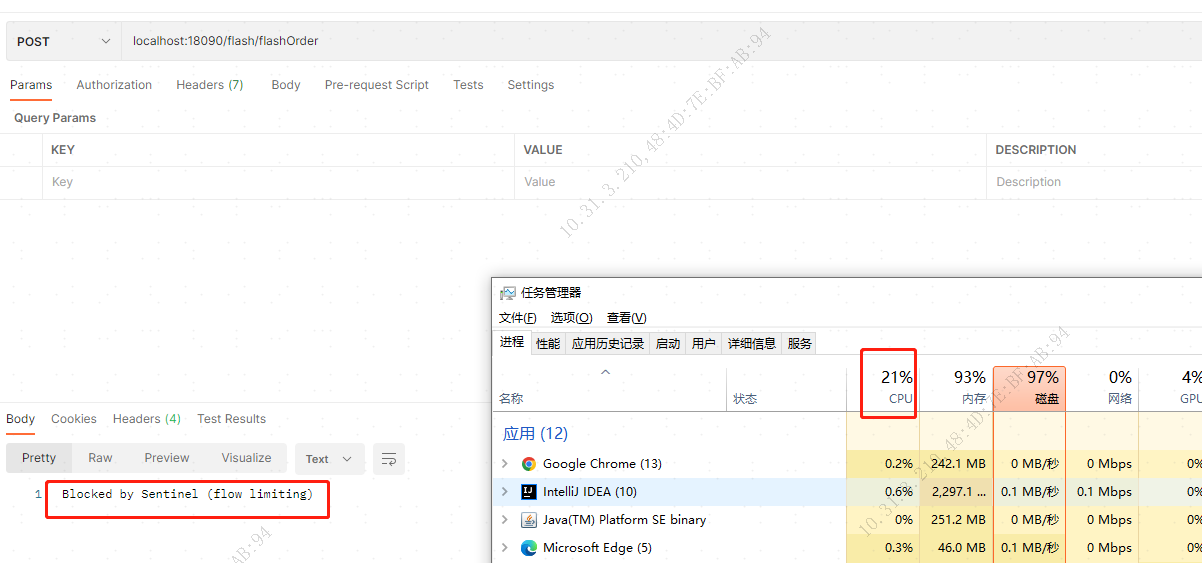


### 4.7系统规则

系统保护规则是从应用级别的入口流量进行控制，从单台机器的 load、CPU 使用率、平均 RT、入口 QPS 和并发线程数等几个维度监控应用指标，让系统尽可能跑在最大吞吐量的同时保证系统整体的稳定性



设置cpu使用超过10%，就会触发系统保护。



备注：Load 自适应（仅对 Linux/Unix-like 机器生效）