**接口性能测试**

项目名称：聚票网

小组成员：潘子奕，徐天强，裘鹏泽，应邦豪

时间：2018年7月25日

目录

[1. 概述 3](#_Toc520413473)

[1.1 目的 3](#_Toc520413474)

[1.2 术语 3](#_Toc520413475)

[2. 测试需求 4](#_Toc520413476)

[2.1被测系统分析 4](#_Toc520413477)

[2.2 测试通过标准 4](#_Toc520413478)

[3. 测试前置操作 5](#_Toc520413479)

[3.1 测试环境 5](#_Toc520413481)

[3.2 基础数据 5](#_Toc520413482)

[3.3 测试监控 5](#_Toc520413483)

[4. 测试用例设计 6](#_Toc520413484)

[对于单个pod服务的登录功能进行压力测试，用户并发量逐渐从100加到200最后加到500并持续一定时间，记录服务器响应时间以及错误率 6](#_Toc520413485)

[在用户并发量为500的情况下，通过增加登录服务的pod数1个，2个再到4个，来验证服务器的可伸缩性以及对应的性能，记录在不同pod数情况下服务器响应时间以及错误率 6](#_Toc520413486)

[在用户并发量为1000的情况下，分别对于单节点的票品功能进行一个有cache一个无cache的测试，来验证cache对于服务器性能的影响，分别记录在cache存在和不存在的不同情况下服务器响应时间以及错误率 6](#_Toc520413487)

[在用户并发量为1000的情况，对于双节点的有cache票品功能进行测试，来验证大量数据对于服务器性能的影响，记录服务器响应时间以及错误率 6](#_Toc520413488)

[5. 测试过程 7](#_Toc520413489)

[5.1 登录压力测试 7](#_Toc520413490)

[5.2 500并发多pod性能测试 8](#_Toc520413491)

[5.3 有无cache对性能的影响 9](#_Toc520413502)

[5.4 pod伸缩对于大数据量的影响 9](#_Toc520413503)

[6. 测试结论 10](#_Toc520413508)

# 概述

## 1.1 目的

该文档详细描述压力测试过程、性能测试过程、监控测试数据以及测试数据分析结论。

## 1.2 术语

压力测试： 通过测试工具不断增大压力，查看系统性能最大承受界限。

性能测试： 通过测试工具产生并发用户，查看系统性能支持的并发量以及响应时间。

负载机：发送请求，生产测试压力的机器。

# 测试需求

2.1被测系统分析

聚票网是一个类似于电商平台的项目，其目的是让大家可以在该平台上看到各种场次类型的展览，舞台剧，歌剧等的票品，并支持用户对于票品进行收藏下单购买，因而预计在未来会有很大的访问量而且随着一些秒杀活动的开展，必然会对系统造成很大的考验。

当前项目通过微服务架构，部署在6台阿里云服务器上，采用k8s部署环境，其中一台2核2G的服务器作为master调度pod，剩余5台作为node工作节点，带宽为1mb/s由于服务器的性能较差，带宽较低，处理很大的并发是不可能的。因为目前系统注册和使用用户非常少，并不会对系统造成威胁。但是系统的处理效率、容量和稳定性未经过验证，还不确定系统在服务器的效率、容量和稳定性。

2.2 测试通过标准

|  |  |
| --- | --- |
| 通过指标 | |
| 错误率 | <5% |
| 响应时间 | <5s |
| CPU | <75% |
| 内存 | <75% |

# 测试前置操作



## 3.1 测试环境

首先由于测试服务器有限，没有独立的服务器供压测使用，所以采用自己的笔记本进行远程压测，因为服务器在华南深圳地区，网络本身就有一定的延时，其次目前线上用户量非常少，压测非订单业务接口不影响生产环境的运行，所以选择合适的时间在生产环境下直接压测。

|  |  |
| --- | --- |
| **性能测试环境** | |
| jdk版本 | jdk1.8 |
| 部署容器 | apache-tomcat-8 |
| 测试工具 | Jmeter3.2 |
| Jmeter负载服务器 | 4核16GWindows 64位 1台 |
| mysql数据库服务器 | 1核2G CentOS 64位 1台 |
| Web应用服务器 | 1核2GCentOs 64位 5台 |

## 3.2 基础数据

通过java随机生成100名用户，数据库中票品通过爬虫服务在大麦网上爬得，约有3000条左右。

3.3 测试监控

1. 应用服务器监控：使用k8s的dashboard对于内存以及cpu进行监控。
2. 数据库监控：没有做监控。后续可以增加慢查询的跟踪。

# 测试用例设计

对于单个pod服务的登录功能进行压力测试，用户并发量逐渐从100加到200最后加到500并持续一定时间，记录服务器响应时间以及错误率

在用户并发量为500的情况下，通过增加登录服务的pod数1个，2个再到4个，来验证服务器的可伸缩性以及对应的性能，记录在不同pod数情况下服务器响应时间以及错误率

在用户并发量为1000的情况下，分别对于单节点的票品功能进行一个有cache一个无cache的测试，来验证cache对于服务器性能的影响，分别记录在cache存在和不存在的不同情况下服务器响应时间以及错误率

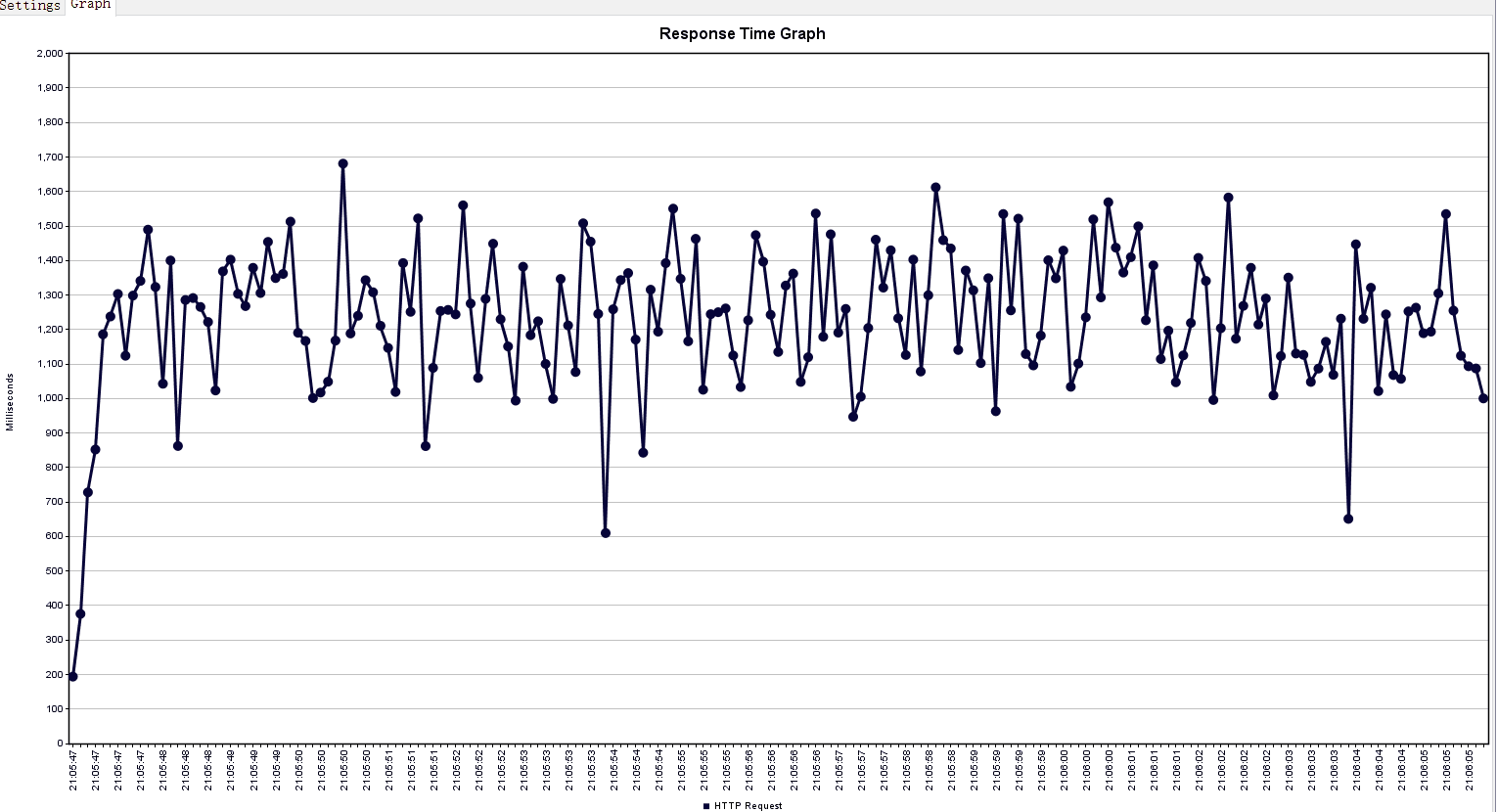
在用户并发量为1000的情况，对于双节点的有cache票品功能进行测试，来验证大量数据对于服务器性能的影响，记录服务器响应时间以及错误率

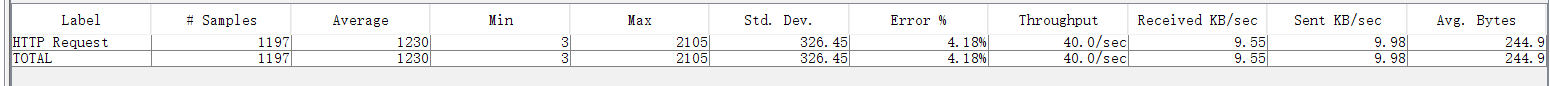
# 测试过程

整个测试过程中

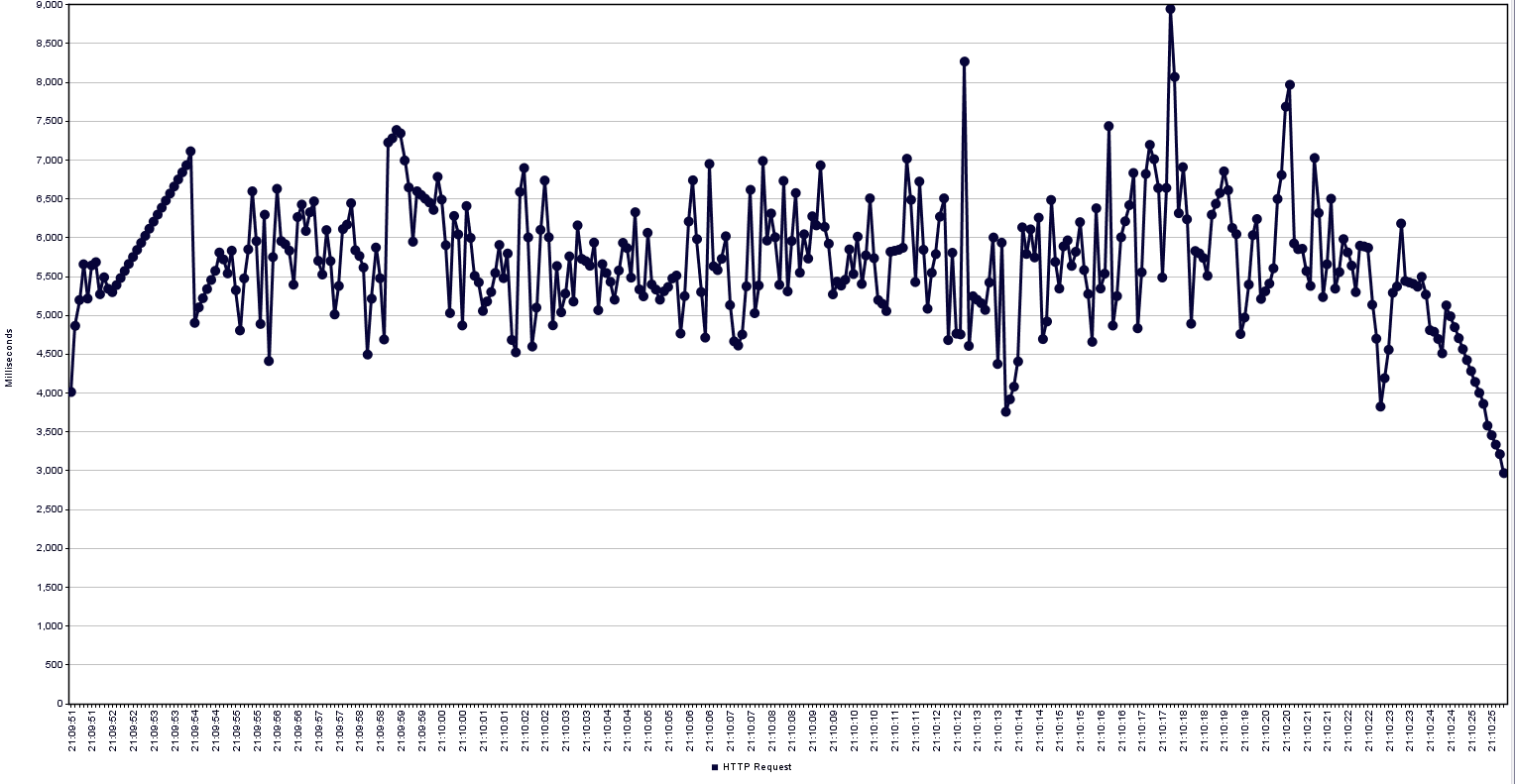
## 5.1 登录压力测试

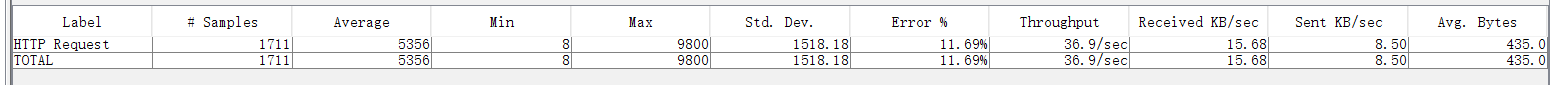
100并发



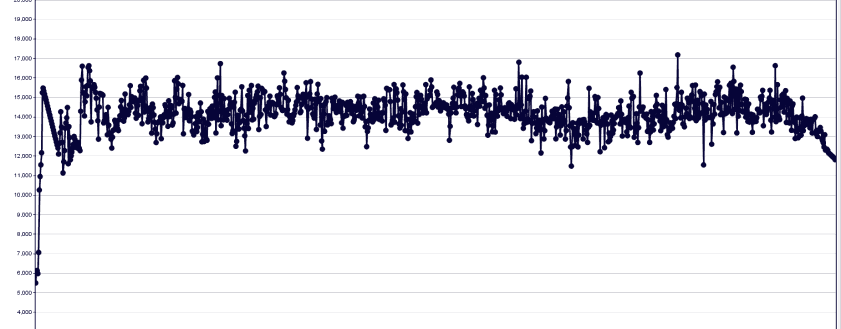


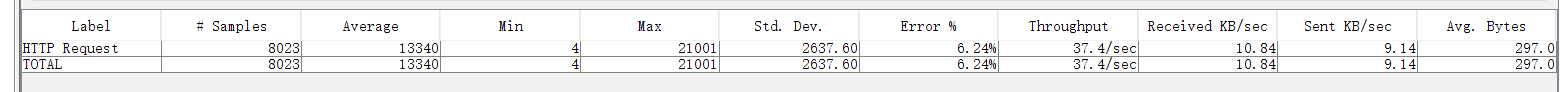
200并发





500并发

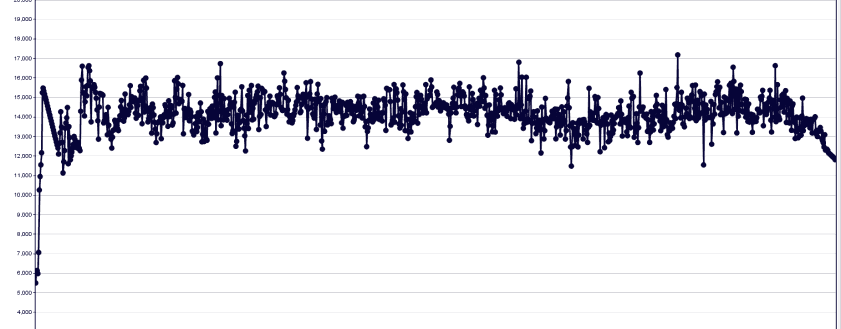


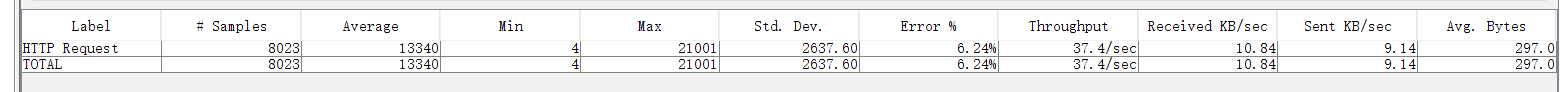


可以看到随着并发量增大，响应时间也呈几何数地增长，当达到500并发量时，服务器逐渐出现错误

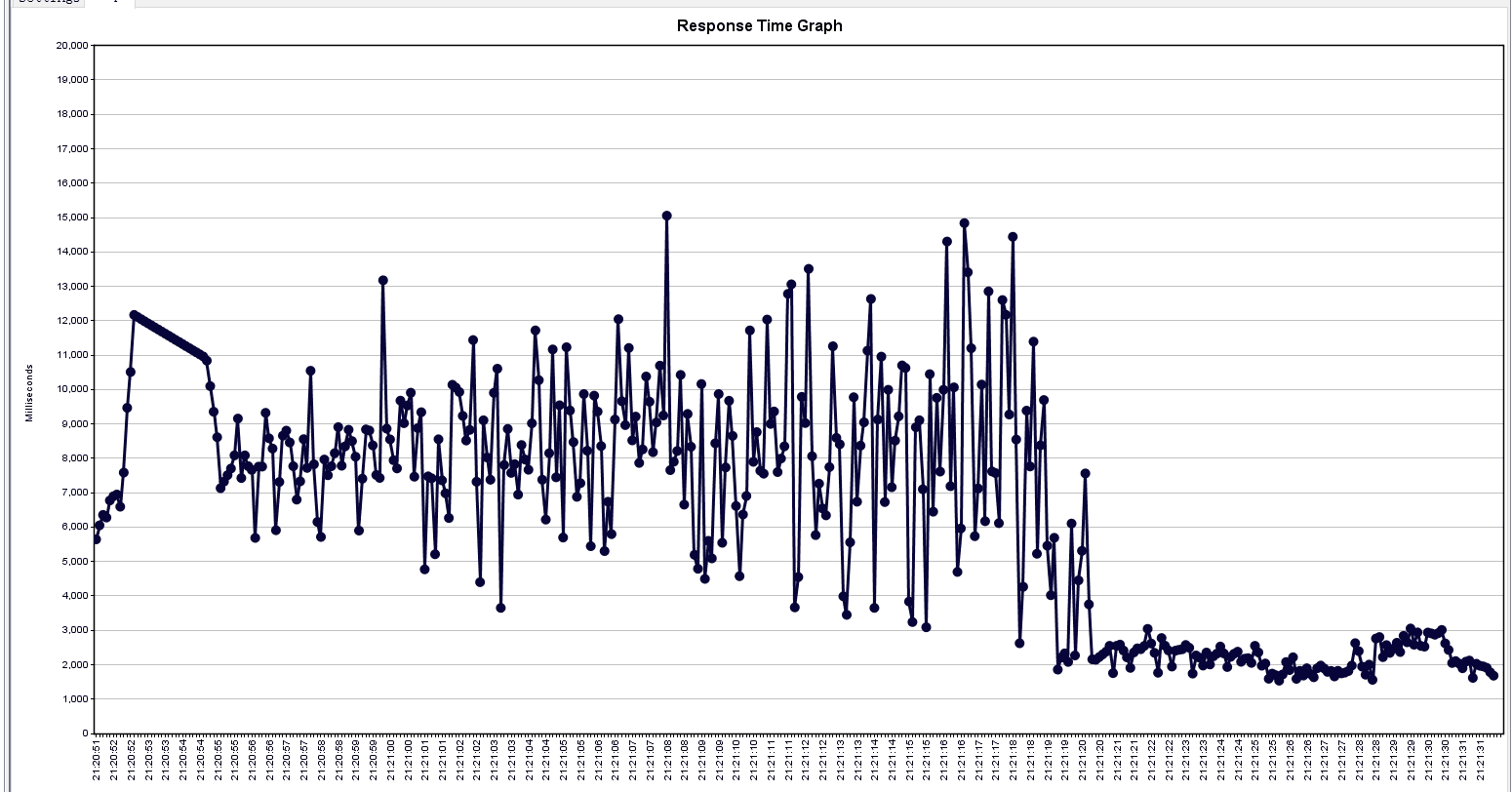
## 5.2 500并发多pod性能测试

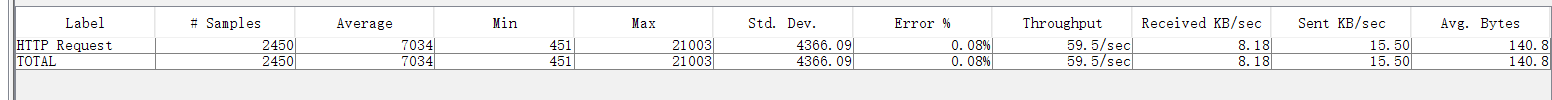
1个pod



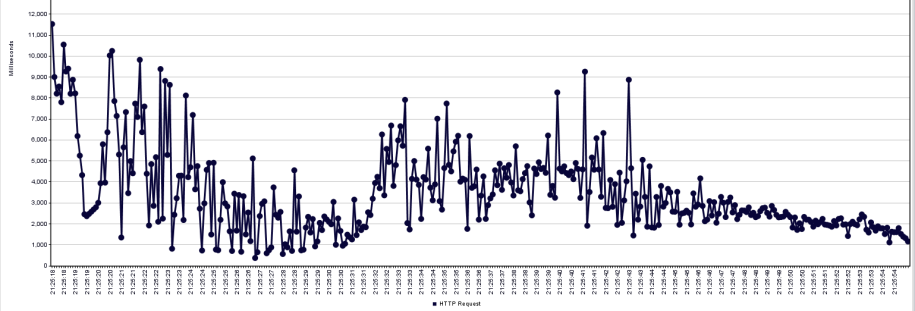


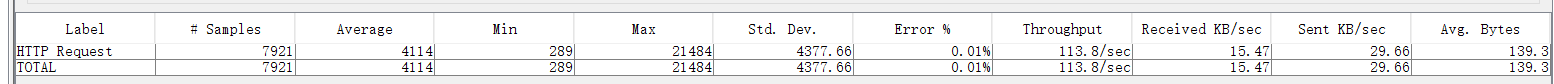
2个pod





4个pod

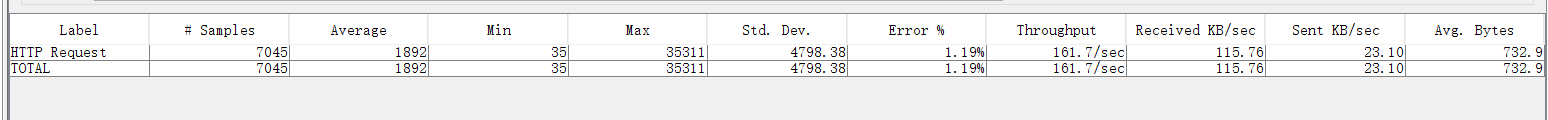




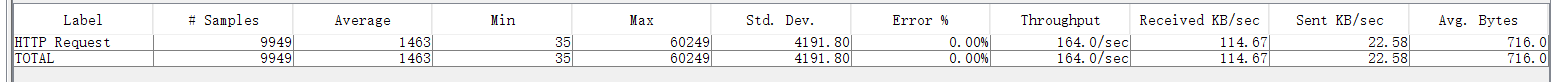
可以很明显的看出随着pod数的增加，每增加一倍，响应时间便减少一半，当数量达到4个pod时，平均响应时间在4秒左右且无错误率，符合服务器预期要求，因为使用微服务以及k8s架构，对于pod的伸缩扩展极为方便。

## 5.3 有无cache对性能的影响

无cache，1000并发量



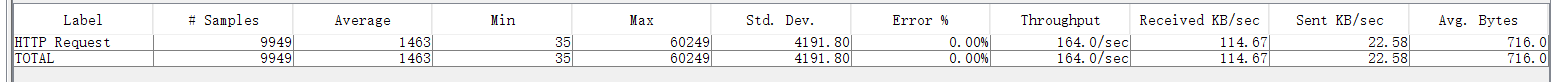
有cache，1000并发量



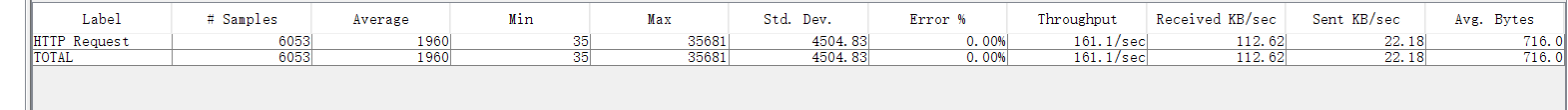
可以很明显的看到使用了redis作为cache后，相同并发量下，响应时间减少了，无cache时，有1.19%的错误率，而加上cache错误率为0

## 5.4 pod伸缩对于大数据量的影响

一个pod，1000并发量



二个pod，1000并发量



可以看到，pod增加了，响应时间却没有增加，而传输速度不变，可见是带宽影响了服务器的性能，由于经费原因，目前无法增加带宽。

# 测试结论

这次的项目充分地发挥了微服务架构的威力，具有极其良好的伸缩性以及可扩展性，在并发数量高，服务器压力大的情况下可以通过扩展pod也即对应的微服务来提升服务器性能，避免服务器挂掉，而且在用户需求不大的时候也可以缩减节点来节省成本资源。

Cache缓存的引入使服务器的性能上了一个档次，减少了服务器的错误率，加快了数据的响应时间。

当高并发量使服务器错误率增高，结束高并发之后服务器依然能正常访问，可见服务器的可恢复性良好。

当返回数据量过大时，带宽成为了服务器性能的最大限制，出于经费成本问题，目前无法增加带宽，但是测试的结果也指明了物理方面优化的方向。

RabbitMQ中间件的使用尚未纳入测试，有待于在日后的测试中验证其对于性能的优化。