|  |  |  |  |
| --- | --- | --- | --- |
| **修订历史** | | | |
| 日期 | 版本 | 说明 | 作者 |
| 7/11/2018 | 1.0 | 第一次迭代测试 | 金瑞洋、王见思、宋逸凡、李翌珺 |
| 7/26/2018 | 1.1 | 第二次迭代测试 |
|  |  |  |
|  |  |  |

**“慧眼识踪“——基于深度学习的人员即时搜寻系统**

**测试报告**

版本1.1

目录

1. 前言 2

1.1编写目的 2

1.2项目背景 2

1.3系统简介 2

1.4术语和缩略语 2

1.5参考资料 3

1. 测试概要 3

2.1测试方法和工具 3

2.2测试环境与配置 4

1. 第一次迭代测试情况 4

3.1测试执行情况与记录 4

3.2覆盖分析 5

3.3测试结论与建议 5

4. 第二次迭代测试情况 5

4.1 单元测试 5

4.3集成测试 10

4.3.1执行情况与记录 10

4.3.2覆盖分析 10

3.3缺陷统计与分析 10

3.4测试结论与建议 10

# 前言

## 1.1编写目的

本测试报告为GETS项目的各项测试总结报告，目的是对该项目开发中的各项成果按照测试用例进行测试，并按照需求规约等标准进行评估，以找出潜在的问题并加以改正，同时验证系统是否符合各项需求。

预期参考人员包括用户、测试人员、开发人员、项目管理者、需要阅读本报告的指导老师等人员。

## 1.2项目背景

项目名称：“慧眼识踪“——基于深度学习的人员即时搜寻系统

项目开发组人员：金瑞洋 项目经理，技术助理

王见思 全栈工程师，架构师

宋逸凡 开发工程师，技术助理

李翌珺 技术总监

项目指导：沈备军老师，陈锴嘉助教

## 1.3系统简介

本项目针对寻找走失人员、跟踪犯罪嫌疑人和定位传染病人等众多现实场景下都必须要解决的搜寻指定人员的问题，设计并开发了一个“基于深度学习的人员即时搜寻系统”。

## 1.4术语和缩略语

搜索目标：需要搜寻系统根据监控画面锁定位置的人物

GETS：God Eye Tracking System， “慧眼识踪”人员搜寻系统

MTBF：Mean Time Between Failure，平均失效间隔时间

MTTR：Mean Time To Restoration，平均恢复时间

## 1.5参考资料

《“慧眼识踪“——基于深度学习的人员即时搜寻系统 需求规约》

《“慧眼识踪“——基于深度学习的人员即时搜寻系统 测试用例》

《“慧眼识踪“——基于深度学习的人员即时搜寻系统 立项建议书》

# 测试概要

本测试报告根据项目需求规约列出的各项功能，测试项目的各个模块是否满足各项需求。每次迭代中分别针对当前开发的模块进行单元测试和集成测试，测试当前模块的功能；在第二次迭代完成后系统具备所有基本功能，进行系统测试，测试系统的功能性需求和非功能性需求，并在完成后继续进行优化，在最后一次迭代完成后再次进行系统测试。

## 2.1测试方法和工具

项目测试时主要采用探索式测试。

## 2.2测试环境与配置

|  |  |  |
| --- | --- | --- |
| 设备 | 参数 | 用途 |
| 阿里云服务器 | 操作系统：Ubuntu  单核CPU  内存：2GB  硬盘：40GB SSD云盘 | 静态资源服务，动态资源服务，视频直播转发，集群管理，CI/CD |
| 笔记本电脑1  （MSI GT72） | 操作系统：Windows10  CPU：intel Core i7  内存：16GB  硬盘：128GB SSD+1TB HDD | 进行各项测试 |
| 笔记本电脑2  （MacBook Pro） | 操作系统：MacOS 10.13.1  CPU：Intel Core i5  内存：4GB  硬盘：128GB SSD | 运行摄像头1 |
| 笔记本电脑3  （MacBook Air） | 操作系统：MacOS 10.13.6  CPU：Intel Core i5  内存：8GB  硬盘：256GB SSD | 运行摄像头2 与性能测试 |

# 第一次迭代测试情况

## 3.1测试执行情况与记录

测试人员：金瑞洋

测试时间：2018年7月11日

测试内容：“查看摄像头参数”，“查看实时监控”两项用例

测试版本：1.0

## 3.2覆盖分析

|  |  |  |
| --- | --- | --- |
| 需求/功能 | 测试内容 | 是否通过 |
| 查看摄像头参数 | 手动查看参数是否正确显示 | 是 |
| 查看实时监控 | 手动打开实时监控 | 是 |

需求覆盖率：100%

## 3.3测试结论与建议

由于本次迭代实现功能有限，测试采用探索式测试，形式较简单，完成较快。测试一定程度上验证了模块的功能性需求且较为充分，基本达到预定目标，可以交付第一次迭代，进入第二次迭代。

测试中发现，前端页面的外观有待改进，且实时视频播放响应时间略长，这两点需要在第三次迭代中加以优化。

# 4. 第二次迭代测试情况

## 4.1 单元测试

### 4.1.1 执行情况与记录

测试人员：王见思

测试时间：2018年7月24日

测试内容：前端单元测试

测试版本：1.1

测试工具：Jest + Enzyme

### 4.1.2 覆盖分析

利用Jest+Enzyme对于每个React Component进行正确性测试。覆盖率如下表所示。

|  |  |  |  |
| --- | --- | --- | --- |
| % Stmts | % Branch | % Funcs | % Lines |
| 67.48 | 65.63 | 65.96 | 67.48 |

之后额外做了关于前端数据的管理后台页面，覆盖率如下表所示

|  |  |  |  |
| --- | --- | --- | --- |
| % Stmts | % Branch | % Funcs | % Lines |
| 47.55 | 50 | 47.48 | 48.43 |

## 4.2 压力测试

### 4.2.1 执行情况与记录

测试人员：王见思

测试时间：2018年7月24日

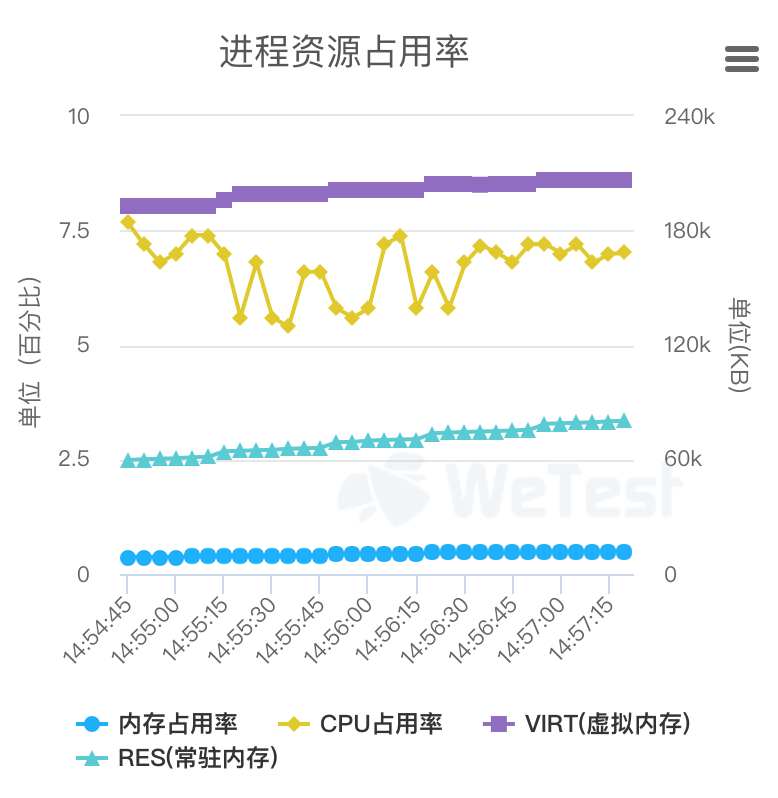
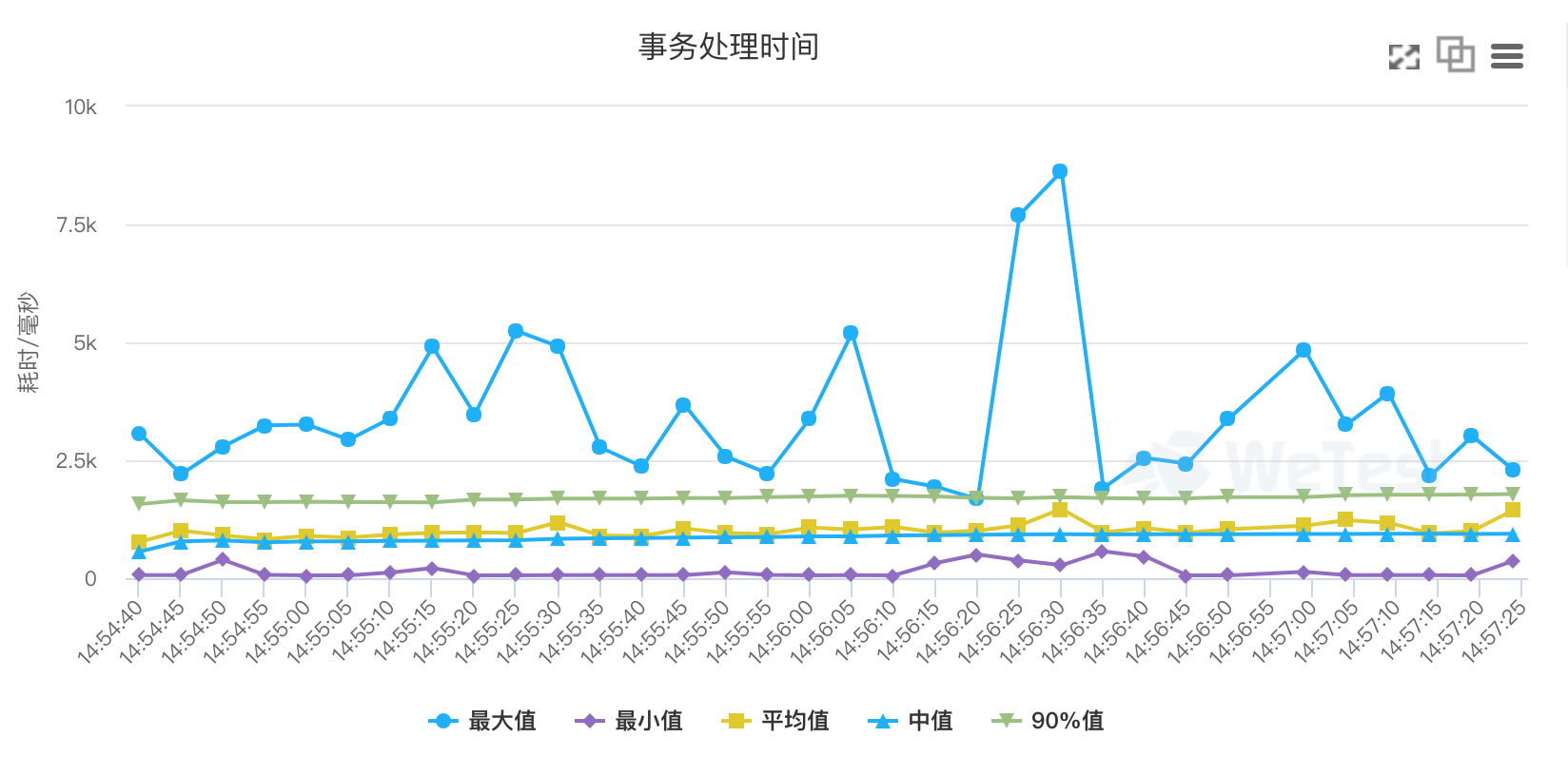
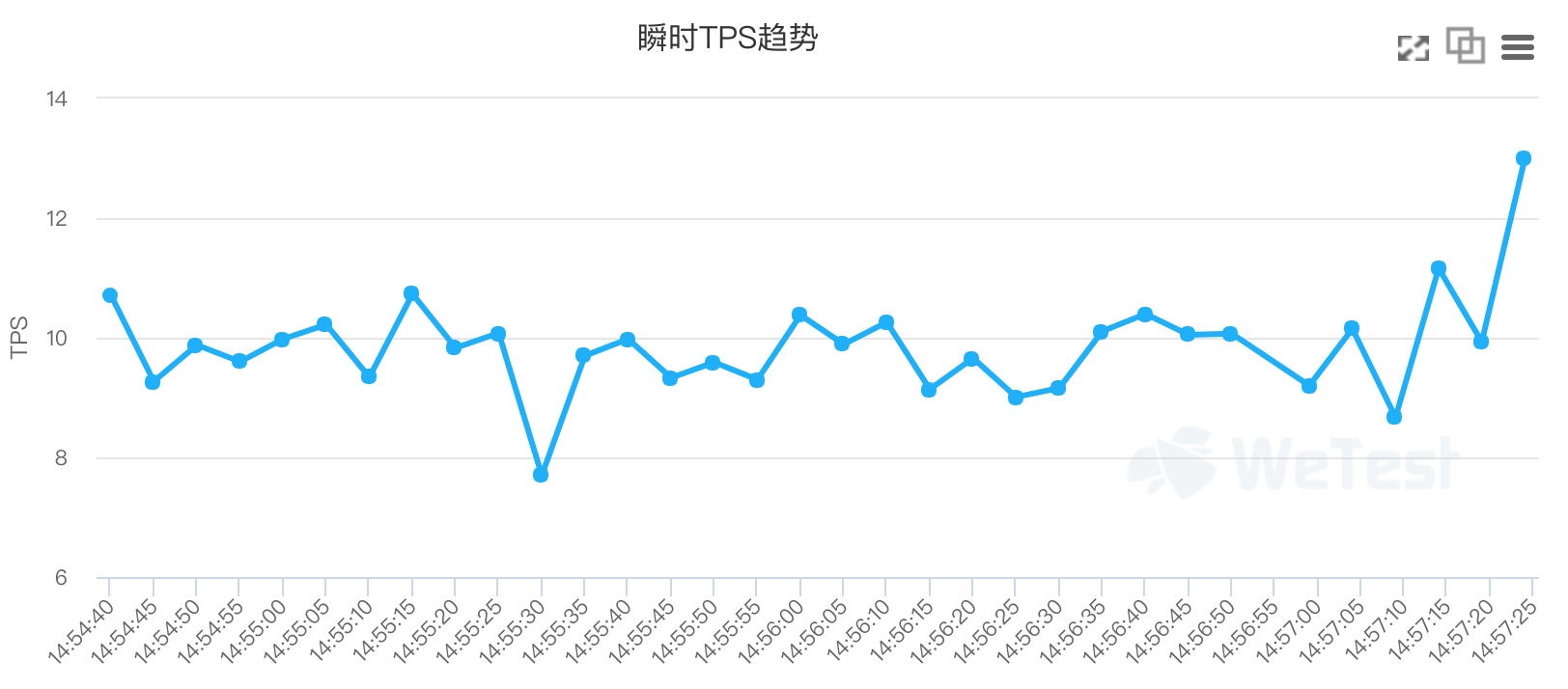
测试内容：将服务部署于阿里云服务器上进行压力测试

测试版本：1.1

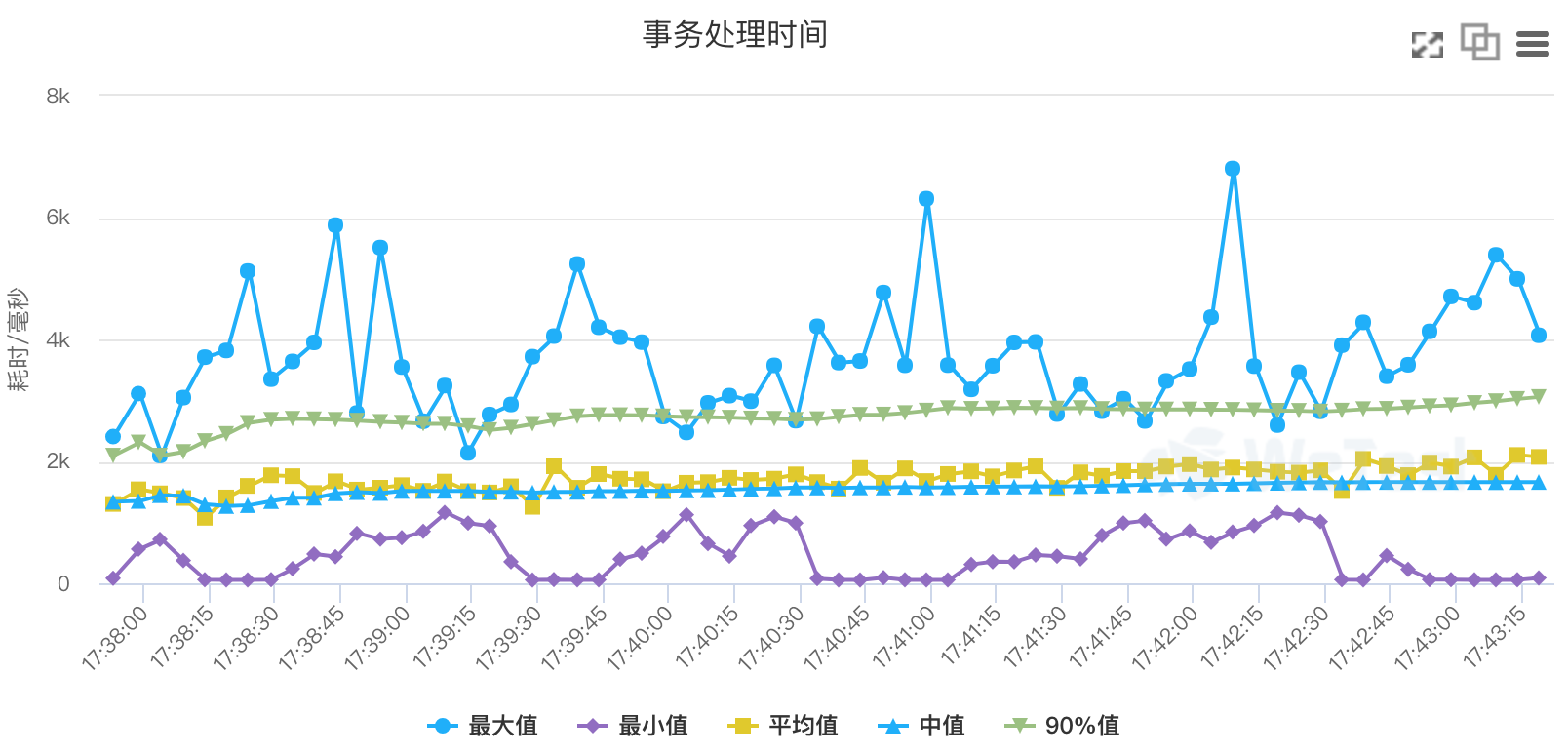
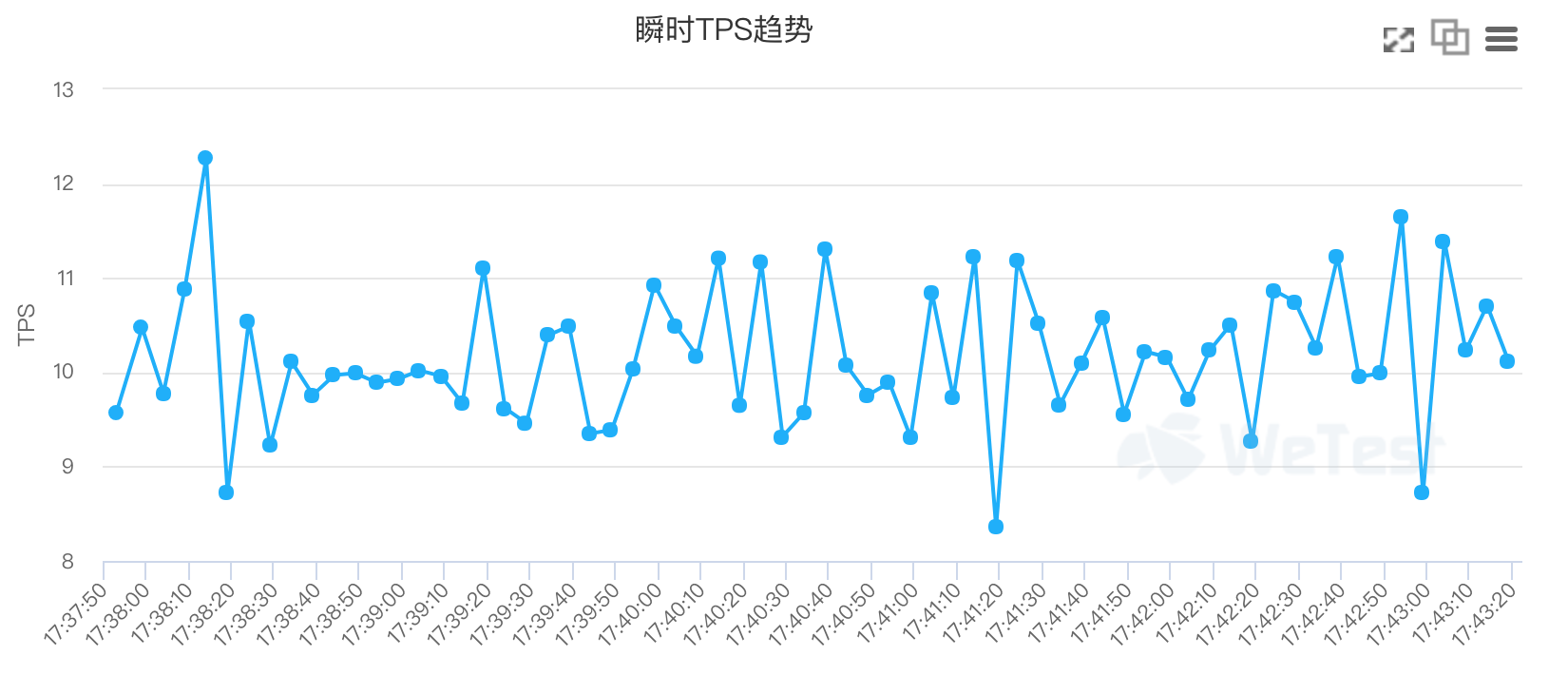
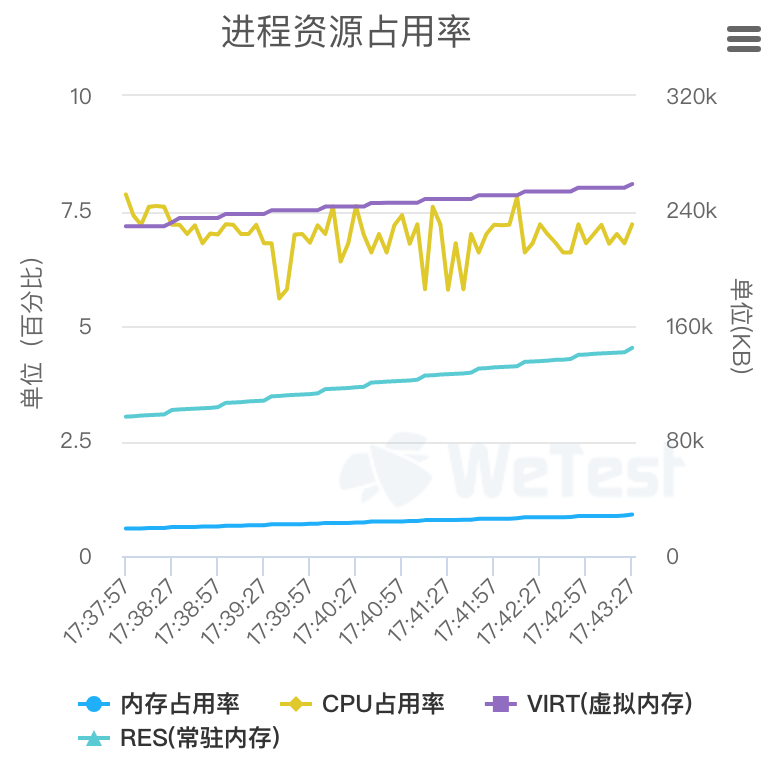
测试工具：腾讯WeTest平台

### 4.2.2 动态资源服务

#### (1) 压力测试一

起始人数 80，每阶段增加人数 5，每阶段持续时间 30s，最大人数 100

#### (2) 压力测试二

起始人数150， 每阶段增加人数5， 每阶段持续时间30s，最大人数 200

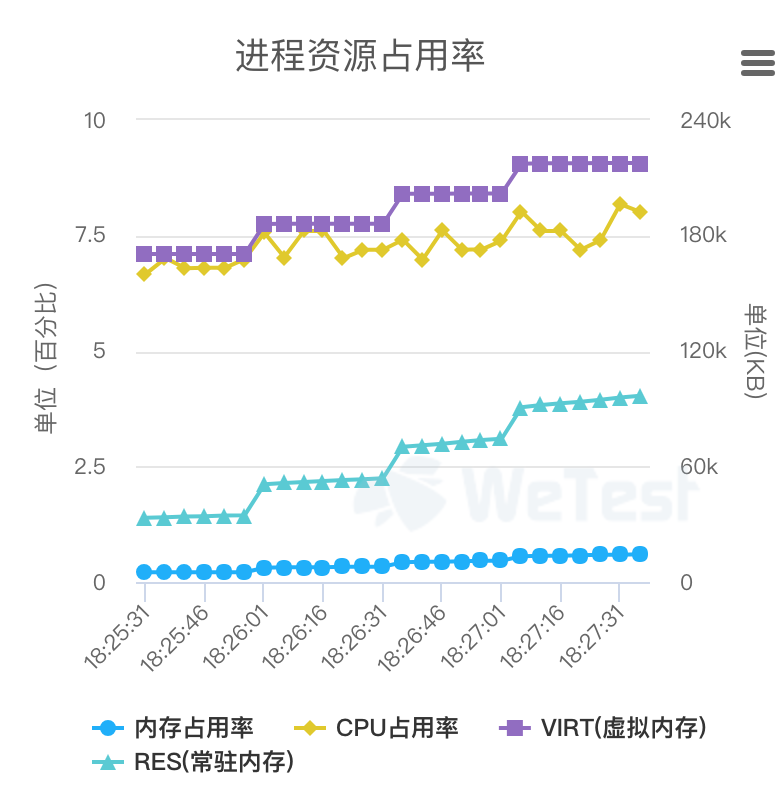
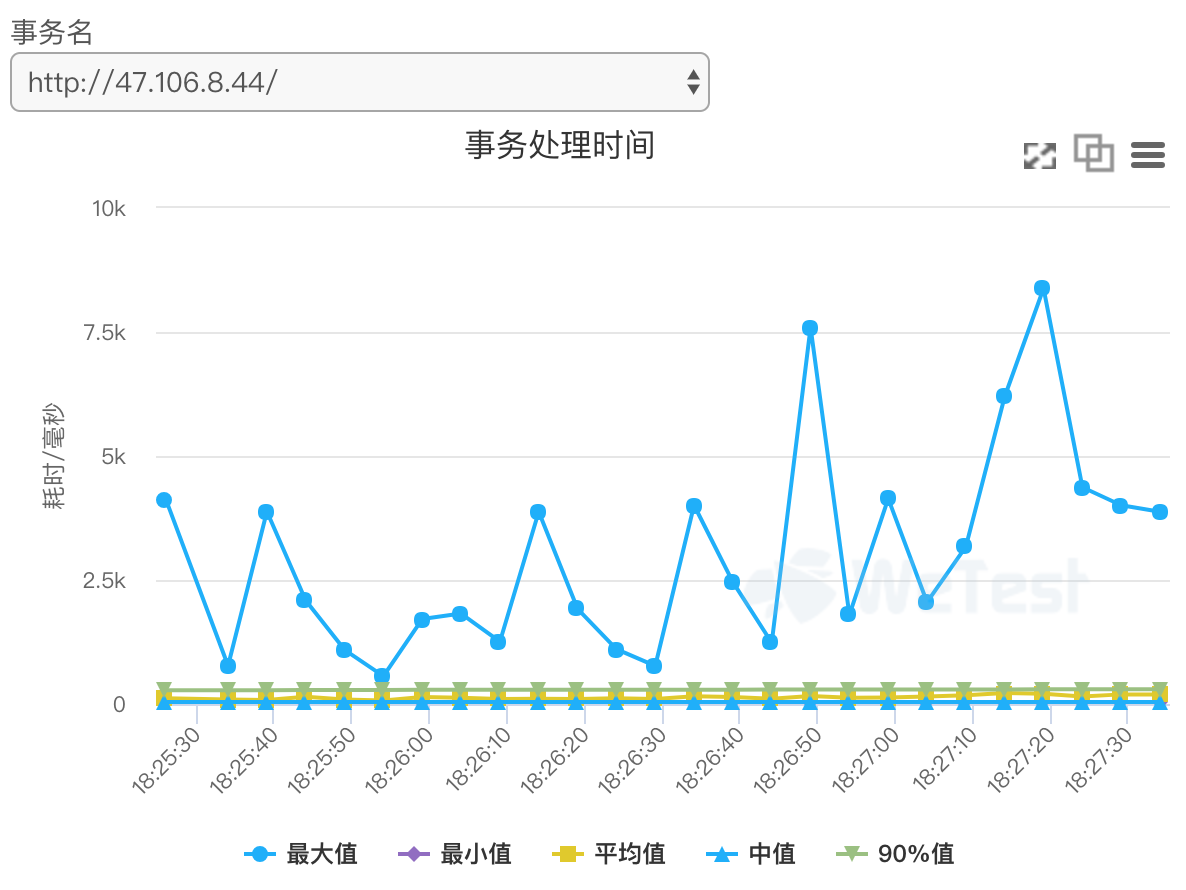
| **测试** | **总事务** | **成功率** | **平均响应时间** | **平均TPS** | **平均收包率** | **平均发包率** | **内存占用均值** | **CPU占用均值** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 测试一 | 16307 | 99.97% | 901.92msms | 98.80/s | 98.2/s | 98.84/s | 0.43% | 6.70% |
| 测试二 | 33771 | 99.97% | 1709.76ms | 100.48/s | 100.18/s | 100.8/s | 0.74% | 6.97% |

### 4.2.3 静态资源服务

#### (1) 压力测试一

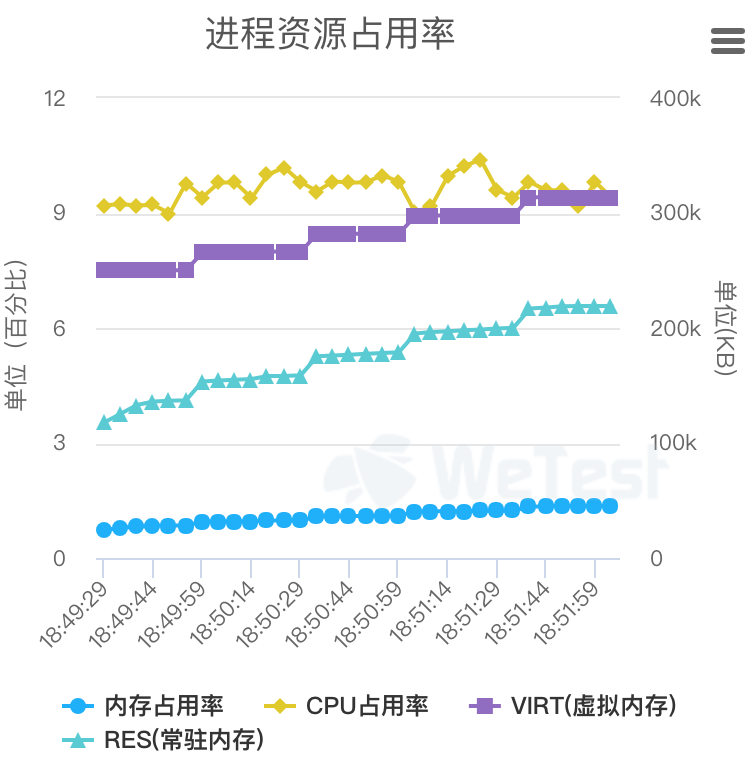
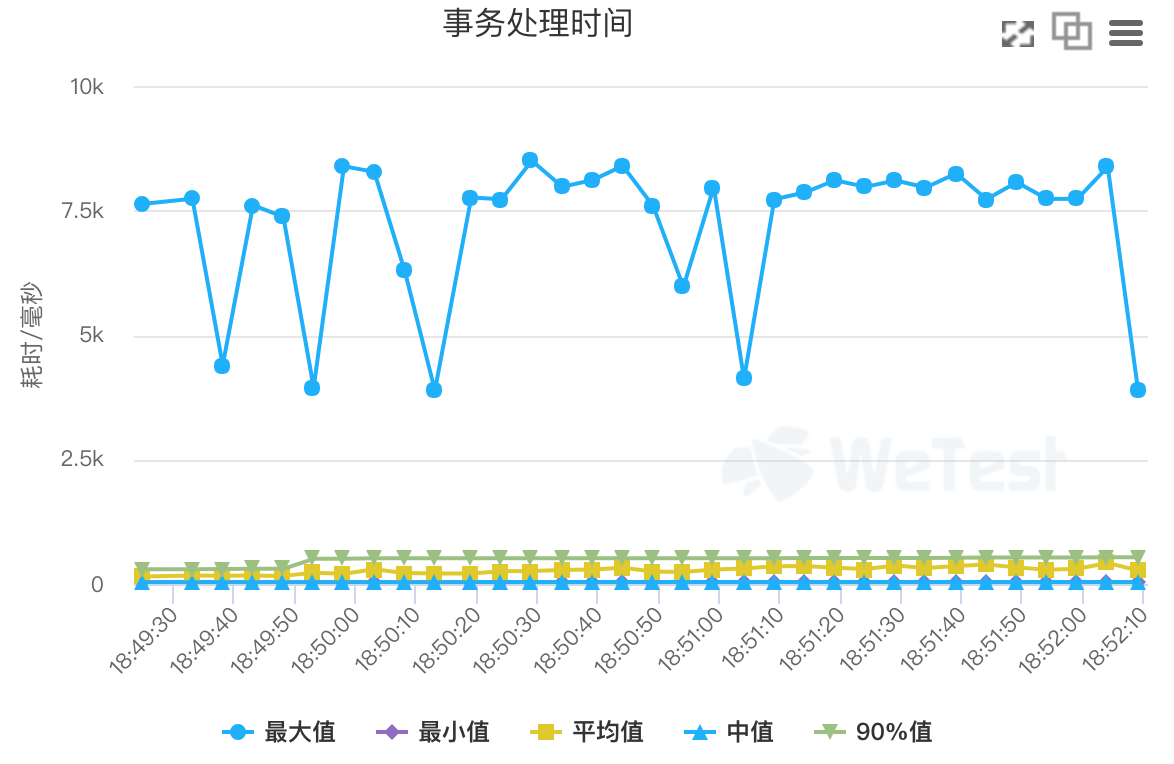
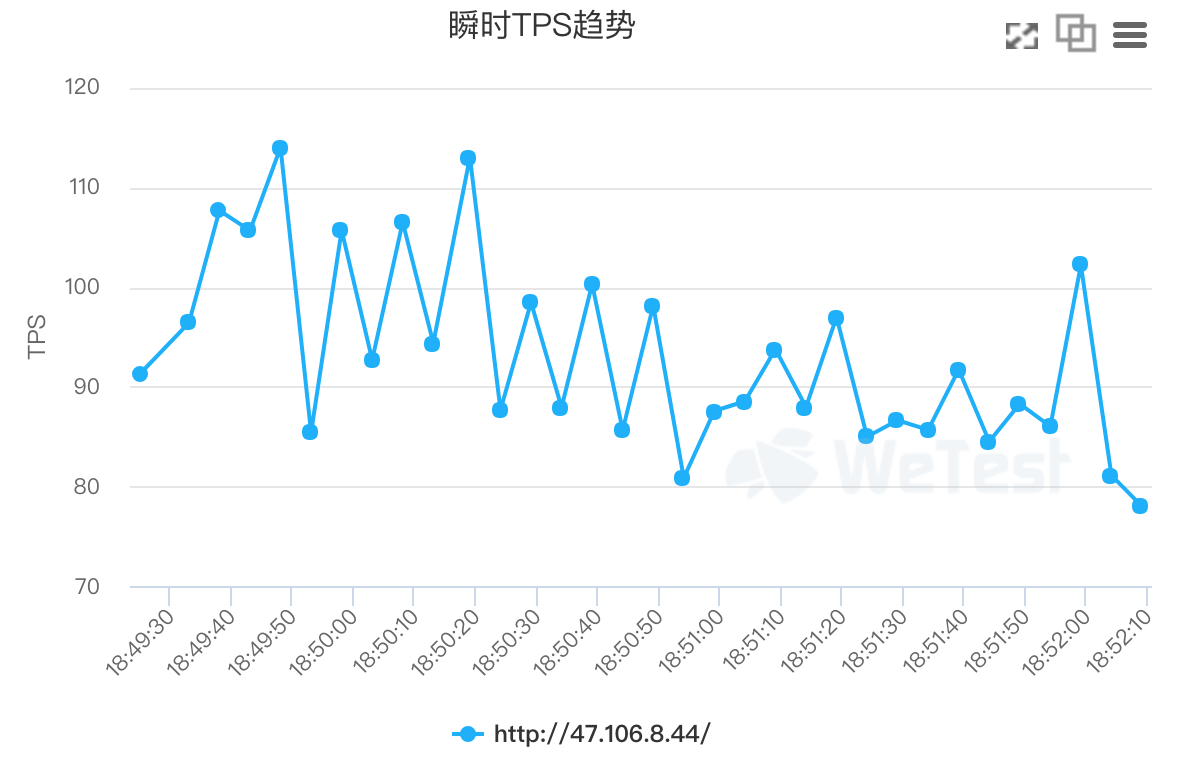
起始人数 5，每阶段增加人数 5，每阶段持续时间 30s，最大人数 20





#### (2)压力测试二

起始人数 30，每阶段增加人数 5，每阶段持续时间 30s，最大人数 50



| 测试 | 总事务 | 成功率 | 平均响应时间 | 平均TPS | 平均收包率 | 平均发包率 | 内存占用均值 | CPU占用均值 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 测试一 | 8250 | 81.52% | 165.85ms | 50.95/s | 50.57/s | 61.71/s | 0.39% | 7.31% |
| 测试二 | 32666 | 80.58% | 360.93ms | 160.51/s | 159.54/s | 198.75/s | 1.09% | 9.62% |

## 4.3集成测试

### 4.3.1执行情况与记录

测试人员：王见思

测试时间：2018年7月24日

测试内容：“查看历史视频”，“搜索目标”两项用例

测试版本：1.1

### 4.3.2覆盖分析

|  |  |  |
| --- | --- | --- |
| 需求/功能 | 测试内容 | 是否通过 |
| 查看历史视频 | 选择并播放历史视频 | 是 |
| 搜索目标 | 上传目标图片返回搜索结果 | 是 |

需求覆盖率：100%

### 4.4 兼容性测试

测试人员：王见思

测试时间：2018年7月25日

测试内容：部署到服务器后不同浏览器的兼容性情况

测试版本：1.1

测试环境：PC端（MacOS与Windows）FireFox，Safari，Chrome

|  |  |  |  |
| --- | --- | --- | --- |
| 浏览器 | 视频播放 | 图片加载 | 其他功能 |
| Chrome | √ | 需要自动代理 | √ |
| Safari· | 不支持webm | 需要全局代理 | √ |
| FireFox | √ | 需要全局代理 | √ |

Safari不支持webm格式视频：历史视频·存储时存储为mp4或其他格式即可解决

图片加载：由于某些图片是用谷歌搜索得到，图片存储于国外CDN，不挂全局代理就无法访问到。实际进入生产环境后采用国内CDN即可

## 4.4 性能测试

### 4.1.1 执行情况与记录

测试人员：李翌珺

测试时间：2018年7月27日

测试内容：针对实时视频和历史视频的“搜索目标”

测试版本：1.1

测试环境：MacBook Air

### 4.1.2 性能分析

|  |  |
| --- | --- |
| 需求/功能 | 平均耗费时间 |
| 单张图片（1280\*720）的物体检测 | 4.0s |
| 对一个摄像头（1280\*720）进行实时搜索 | 59.8s |
| 对两个摄像头（1280\*720）进行实时搜索 | 125.7s |
| 对历史视频进行重识别（65张图片资源） | 64.6s |

制约性能的因素：

1. 物体检测与 重识别之间大量的文件读写
2. 重识别过程中对历史图片重复计算embedding
3. 重识别本身计算的复杂性

## 4.5缺陷统计与分析

缺陷总数：0

缺陷分布：

|  |  |  |
| --- | --- | --- |
| 小错误 | 大错误 | 严重错误 |
| 0 | 0 | 0 |

## 4.6测试结论与建议

本次迭代功能基本实现，并且将相关服务实际部署于服务器上，进行了单元测试、集成测试、压力测试、兼容性测试、性能测试等。

单元测试保证了前端代码的正确性以及修改代码能够及时发现错误。但是由于使用了视频、画布、截图框等，有很多代码难以通过自动化代码测试，而需要通过肉眼观察进行测试。同时大量使用antd组件，很多状态难以监听，也给单元测试带来难度。

将服务部署后，每个服务启动了两个docker 镜像以及一个HAProxy负载均衡后进行压力测试。但是由于服务器内存与带宽所限，并发量并不能很高。后端动态资源只传输很短的json，因此200并发量时，响应时间稍慢，但正确率仍有99.97%；前端页面体积较大，video.js 的CDN加载也较慢，50并发量时，正确率仅有80.58%

集成测试将基本功能与需求以及可能得分支操作考虑，全部符合预期。