用户中心V1.0性能测试报告

|  |  |  |  |
| --- | --- | --- | --- |
| 编写报告 ： | 李华 | 日 期 ： | 2017-1-3 |
| 脚本执行 ： | 罗利君 | 日 期 ： | 2016-12-28至2016-12-30 |
| 批 准 ： |  | 日 期 ： |  |

# 概述

## 编写目的

本次测试报告为公有云管理控制台（用户中心V1.0版本）的性能测试报告总结。根据产品提供的的性能测试指标，目的在于总结文档中三个场景的性能测试工作，并分析测试结果。

预期参考人员包括产品人员、测试人员、开发人员、项目管理者、用户和需要阅读本报告的高层经理。

## 项目背景

易云公有云平台是面向最终客户提供的管理控制台，最新版本‘用户中心V1.0’即将上线，在交付给客户使用之前，需对ECSC业务系统的重要环节进行性能测试与优化，才能保证上线后的质量。

## 测试目标

简要列出进行本次压力测试的主要目标：

1.多用户并发登录测试

2.多用户压力广义并发测试

3.创建云主机、创建子网的请求响应时间

## 名词解释

Maximum Running Vusers:最大运行虚拟用户数

Total Throughput(byte):总吞吐量

Average Throughput(byte/second):平均吞吐量

Total Hits:总点击数

Average Hits per Second：平均每秒点击次数

Total Error:错误的个数

Transaction Name：事务名称

Minimum：事务中最小的时间（单位：秒）

Average：事务的平均时间（单位：秒）

Maximum：事务中最大的时间（单位：秒）

Std.Deviation：标准偏差，方差越小系统越稳定，方差越大系统越不稳定

90 Percent：90%的事务在X秒内通过

Pass：通过的事务数

Fail：失败的事务数

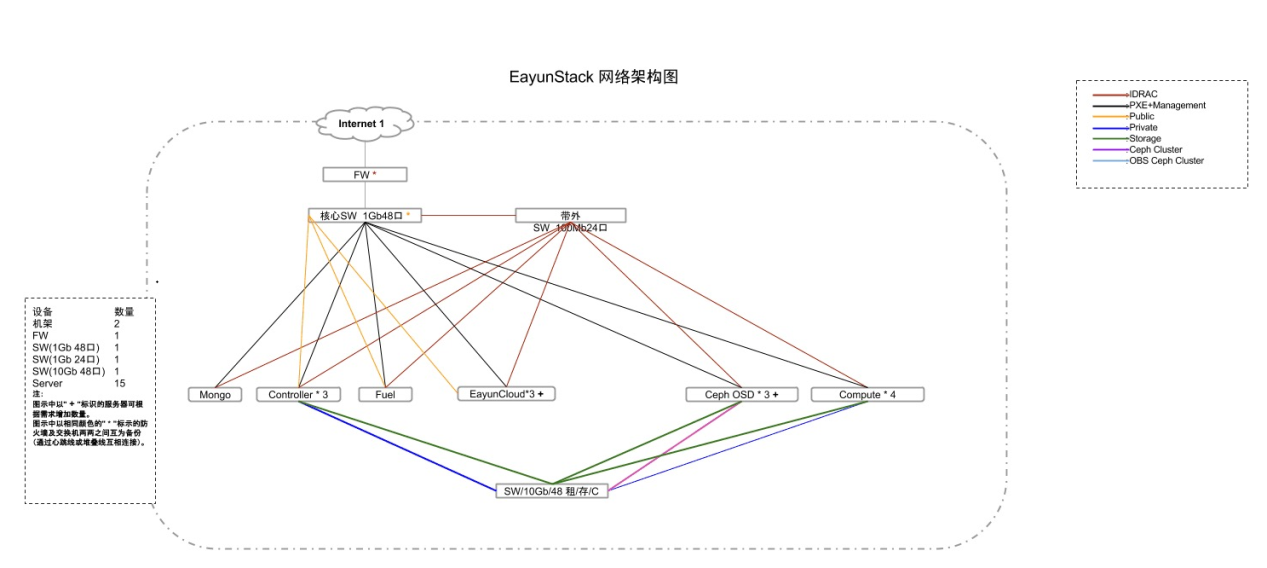
Stop：停止的事务数

# 测试环境说明

## 环境配置

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 集群服务器配置 | | | | | |
| 机器名（IP） | | CPU（核） | 内存（G） | 硬盘（G） | 软件环境（操作系统、应用软件） |
| 25.0.3.17 | | 4 | 8 | 200 | CentOs6.8 Tomcat ECSC-1 |
| 25.0.3.39 | | 4 | 8 | 200 | CentOs6.8 Tomcat ECSC-2 |
| 25.0.3.27 | | 2 | 4 | 100 | CentOs6.8 Tomcat ECMC-1 |
| 25.0.3.40 | | 2 | 4 | 100 | CentOs6.8 Tomcat ECMC-2 |
| 25.0.3.16 | | 2 | 4 | 100 | CentOs6.8 Tomcat mail-sms-1 |
| 25.0.3.37 | | 2 | 4 | 100 | CentOs6.8 Tomcat mail-sms-2 |
| 25.0.3.26 | | 4 | 8 | 100 | CentOs6.8 Tomcat Schedule-1 |
| 25.0.3.38 | | 4 | 8 | 100 | CentOs6.8 Tomcat Schedule-2 |
| 25.0.3.51 | | 4 | 8 | 100 | CentOs6.8 Tomcat Schedule-res-1 |
| 25.0.3.61 | | 4 | 8 | 100 | CentOs6.8 Tomcat Schedule-res-2 |
| 25.0.3.15 | | 4 | 8 | 500 | CentOs6.8 Tomcat mysql-1 |
| 25.0.3.35 | | 4 | 8 | 500 | CentOs6.8 Tomcat mysql-2 |
| 25.0.3.19 | | 4 | 8 | 200 | CentOs6.8 Tomcat api-1 |
| 25.0.3.29 | | 4 | 8 | 200 | CentOs6.8 Tomcat api-2 |
| EayunStack环境 | | | | | |
| 底层（IP） | | | | | |
| 25.0.0.3 | | | | | |
| 客户端配置 | | | | | |
| 机器名（IP） | | CPU（核） | 内存（G） | | 软件环境（操作系统、应用软件） |
| 10.10.10.182 | | 4 | 12G | |  |
| 说明 | 访问地址： https://ecsc.eayun.cn | | | | |

**Eayunstack底层网络拓扑图：**



## 测试概要

### 2.2.1 基本信息

测试地址: <https://ecsc.eayun.cn>

测试时间：2016-12-28 至 2016-12-30

脚本执行者：罗利君

性能测试工具：loadrunner 11

测试场景：3个

脚本数量：9个

### 2.2.2 模拟数据信息

以下准生产环境中造的数据全部存在数据库、mongo中，与Eayunstack底层无关。

客户数量：100

用户数量：2599—平均一个客户下25个用户

订单总览：51980—平均一个用户下20个订单记录

交易记录：44335—平均一个客户下443条交易记录

报警信息：50028

网络记录：5000

路由记录：5000

子网记录：25000

云主机记录：25000

云硬盘记录：5000

云硬盘快照记录：25000

资源统计数据涉及MongoDB四个集合：

cpu\_util.detail ： 4578 documents

memory.usage.detail：4509 documents

bandwidth.network.incoming.detail：24140 documents

bandwidth.network.outgoing.detail：24140 documents

# 测试策略&结果

## 3.1测试方案

产品的三个指标分别为：多用户并发登录测试、多用户压力广义并发测试、请求响应时间。根据以上指标，分析测试可执行过程并与实际情况相结合，分别列出了以下三个场景方案；

**多用户并发登录测试**：根据2-5-8原则，页面请求响应时间与用户数量结合得出最优、较优、良好的情况。最优：即2s内能够响应的最大狭义并发用户数量；较优：2s-5s内能够响应的最大并发用户数量；良好：5s-8s内能够响应的最大狭义并发用户数量。并发用户数量以100为基础再逐渐增加，以‘事务成功率’100% 或90percent响应时间8s以内为场景的结束值，得出并发登录的最大用户数量。

**多用户压力广义并发登录测试**：通常用户登录ecsc平台后，不同的用户会有不同的操作，所以结合实际场景，估算用户常用模块功能，模拟此场景为：40%的用户进行查询云主机、云硬盘、云硬盘快照、私有网络的操作；30%的用户进行资源监控、报警信息查询；20%的用户进行交易记录、费用报表（后付费）、订单列表、资源统计（云主机、云硬盘、私有网络）查询；10%的用户进行创建工单、创建标签、创建标签类别。在CPU使用率允许范围内、事务数失败率10%以内得出最大在线人数。

**创建云主机、创建子网的请求响应时间**：根据产品的需求需要得出创建30台window2008标准（按需付费、自管子网、受管子网、1核1GB、1台）、30台liunx（按需付费、自管子网、受管子网、1核1GB、1台）、30个受管子网、30个自管子网的请求响应时间。

**备注：以上所有的响应时间均以‘90percent’的数据为准。**

## 3.2测试用例以及结果分析

### 3.2.1多用户并发登录测试

|  |  |  |  |
| --- | --- | --- | --- |
| **用例名称** | 多用户并发登录测试 | **用例编号** | 001 |
| **测试步骤** | 1. 使用loadrunner录制ecsc登录脚本。 2. 登录前设置‘集合点’，参数化登录账户，添加‘登录’、‘退出’两个事务。 3. Controller并发运行脚本。 | | |
| **场景设计** | 1、 设计用户数量为100至550人。  2、 测试方案：并发测试，用户在集合点同时登录，退出时全部立即退出。  3、 运行场景。 | | |
| **执行时间** | (1~6)分钟 \* 10（10次并发） | | |
| **实际结果** | 1、 对比10次并发（100、200、360、370、380、400、500、550），其中500人与550人分别做了两次并发。而550人并发时事务数已有失败率、500人时无失败率，故500人为登录并发的最大数。  2、 500人登录并发为机器获得的绝对并发数据，换算成真实用户数一般都会\*10，所以，真实用户并发登录数为5000=500\*10。 最大并发用户数为5000人。 | | |

**并发测试结果：**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **并发人数** | **时间段** | **通过事务数** | | **90percent（s）** | | **Std.Deviation** | |
| **login** | **logout** | **login** | **logout** | **login** | **logout** |
| 100 | 2016/12/28 11:22- 2016/12/28 11:23 | 100 | 100 | 0.253 | 0.625 | 0.028 | 0.244 |
| 200 | 2016/12/28 15:20- 2016/12/28 15:21 | 200 | 200 | 0.741 | 2.982 | 0.067 | 1.177 |
| 360 | 2016/12/28 16:31- 2016/12/28 16:34 | 360 | 360 | 1.525 | 4.656 | 0.895 | 1.422 |
| 370 | 2016/12/28 16:39- 2016/12/28 16:42 | 370 | 370 | 1.583 | 5.269 | 1.6 | 1.904 |
| 380 | 2016/12/28 16:35- 2016/12/28 16:38 | 380 | 380 | 2.218 | 4.168 | 1.149 | 1.569 |
| 400 | 2016/12/28 15:29- 2016/12/28 15:31 | 400 | 400 | 2.425 | 5.026 | 0.844 | 1.75 |
| 500 | 2016/12/28 16:19- 2016/12/28 16:23 | 500 | 500 | 3.162 | 5.569 | 1.342 | 1.857 |
| 500 | 2016/12/28 17:13- 2016/12/28 17:17 | 500 | 500 | 4.729 | 6.691 | 1.533 | 2.269 |
| 550 | 2016/12/28 17:28- 2016/12/28 17:32 | 550 | **548** | 4.577 | 7.107 | 1.889 | 2.432 |
| 550 | 2016/12/28 17:36- 2016/12/28 17:42 | 550 | **541** | 3.693 | 5.795 | 1.809 | 2.004 |

表1

结果分析：

1. 表1 为从100人并发逐步升到550人并发，从‘通过事务数’中可以看出550人时，退出的事务已有失败的现象（第一次有2个失败，第二次有9个失败），所以可以得出最大并发数为500人，从而得到最大并发用户数为5000人。（由于在6s内550人时已有失败的事务，所以未做8s内的最大并发人数。）

2. 90%的用户在2s内并发登录的最大值为370，最大并发用户数为3700=370\*10

3. Std.Deviation 为‘标准偏差’，此值越小代表系统越稳定，std<1时并发登录在360（即3600人=360\*10）时系统比较稳定。

**500登录并发Analysis Summary：**

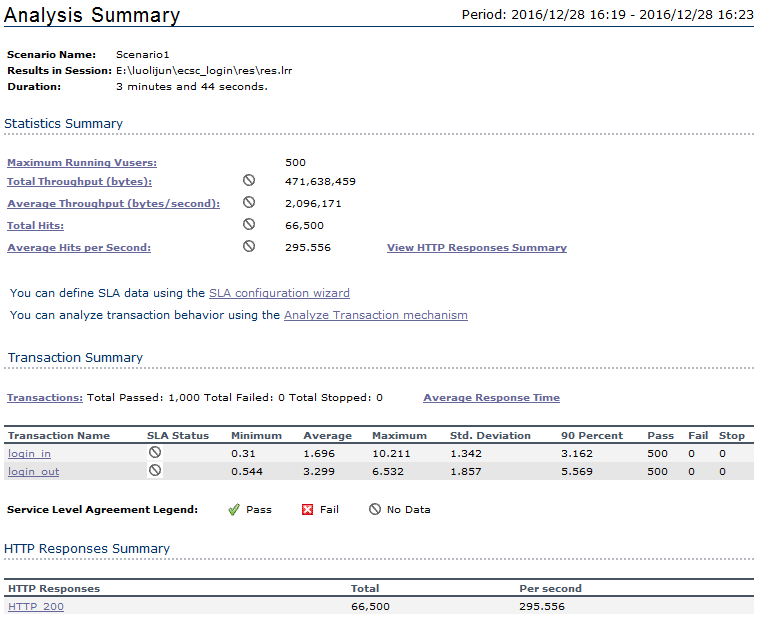


图3-1

结果分析：

1. 数据总结：并发用户数为500，总吞吐量为471,638,459，平均吞吐量2,096,171每秒，总点击数66,500。

2. 事务总结：500个登录事务，最小登录时长0.31秒，最大登录时长10.211秒，平均登录时长1.696秒，90%的事务在3.162秒内登录成功；500个退出事务，最小登录时长0.544秒，最大登录时长6.532秒，平均登录时长3.299秒，90%的事务在5.569秒内登录成功；可以得出并发500（即：最大并发用户数5000人）登录响应时长为3.299秒，退出响应时长为5.569秒。

### 3.2.2多用户压力广义并发登录测试

|  |  |  |  |
| --- | --- | --- | --- |
| **用例名称** | 多用户压力广义并发登录测试 | **用例编号** | 002 |
| **测试步骤** | 1、使用loadrunner录制ecsc脚本。  2、录制4个脚本：  脚本1：查询云主机、云硬盘、云硬盘快照、私有网络；  脚本2：查询资源监控、报警信息；  脚本3：查询交易记录、费用报表（后付费）、订单列表、资源统计（云主机、云硬盘、私有网络）；  脚本4：创建工单、创建标签、创建标签类别。 | | |
| **场景设计** | 1、 设计用户数量500，700。  2、 加压方案：脚本1（40%）、脚本2（30%）、脚本3（20%）、脚本4（10%）,四个脚本按着比例逐渐增加，以10s增加3个用户的数量逐渐加压500/700人。  3、 设计运行时间为全部用户登录成功后在线运行30min，再全部退出所有用户。  4、 运行场景。 | | |
| **执行时间** | 140min\*2（2次压力） | | |
| **实际结果** | 1、 500人登录成功后，前后不停的循环运行1个多小时，每秒的点击率在50左右，共有7个事务错误；700人登录时，已有260个事务错误，而且数据库的CPU一直在90%以上，虽然事务数一直在产品需求10%以内，但是ecsc平台已经非常缓慢，数据库的CPU使用率很高，故以500为机器获取的最大虚拟人数。   1. 此场景的并发并非绝对并发，可以根据公式‘C=nL/T’与‘C‘ = C + 3\*根号C’获取最大在线人数。 其中C‘为并发用户的峰值，C为平均的并发用户数，n为最大在线人数，L为线上用户从登录到退出的时长，T为用户使用ecsc时间段。   可以看出C‘=500，根据公式C‘ = C + 3\*根号C，可以得出C=440，再根据公式C=nL/T，最大在线人数n=440T/L，当T越大L越小时，在线人数的值越大，反之越小。（由于没有提供具体的T与L值，所以无法估算最大在线人数。举例说明：用户平均从登录到退出为1个小时，而用户只在上班时间9:00-18:00，9个点一直使用ecsc，那么最大在线人数为n=(440\*9)/1=3960人） | | |

**500广义并发Analysis Summary：**

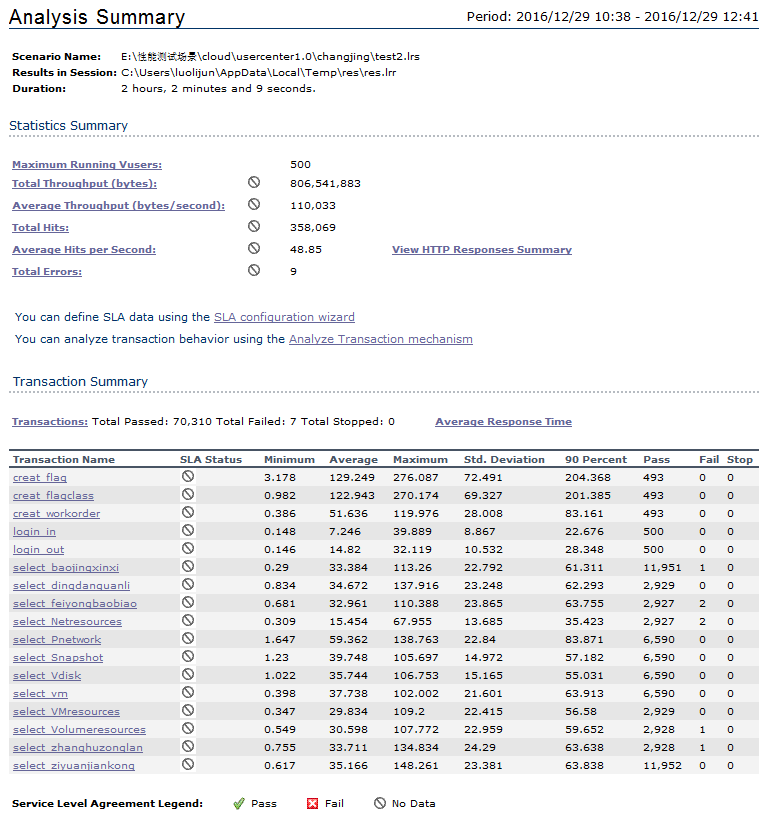


图3-2



图3-3

结果分析：

1. 图1中可以看出500人在123分钟内执行17个事务共70310次，失败的事务为7次，90%的事务都以分钟级别的基础上通过，标准方差已经非常的大，说明此时系统已经非常缓慢，（经排查数据盘CPU使用率90%以上），无需再进行进一步的加压测试。
2. 图2中是运行用户数与每秒点击率结合的图片，左纵坐标为每秒点击率，右纵坐标为运行用户数。在用户逐渐增加的基础上，每秒点击率再逐渐降低，与理想状态偏离，平均点击率为48.241。

### 3.2.3 创建云主机、创建子网的请求响应时间

|  |  |  |  |
| --- | --- | --- | --- |
| **用例名称** | 创建云主机、创建子网的请求响应时间 | **用例编号** | 003 |
| **测试步骤** | 1、使用loadrunner录制ecsc脚本  2、录制四个脚本：创建Windows云主机脚本、创建liunx云主机脚本、创建自管子网脚本、创建受管子网脚本。 | | |
| **场景设计** | 1、 将四个脚本放到一个场景中，登录与退出只运行一次，创建云主机、创建子网的事务运行N次  2、 运行场景，求取请求响应时间 | | |
| **执行时间** | 6min（1次） | | |
| **实际结果** | 1.创建39次Linux云主机平均请求响应时间：23.787秒；创建39次Windows云主机平均请求响应时间：20.739秒；创建42次受管子网平均请求响应时间：4.768秒；创建42次自管子网平均请求响应时间：3.352秒。 | | |

**请求响应时间Analysis Summary：**

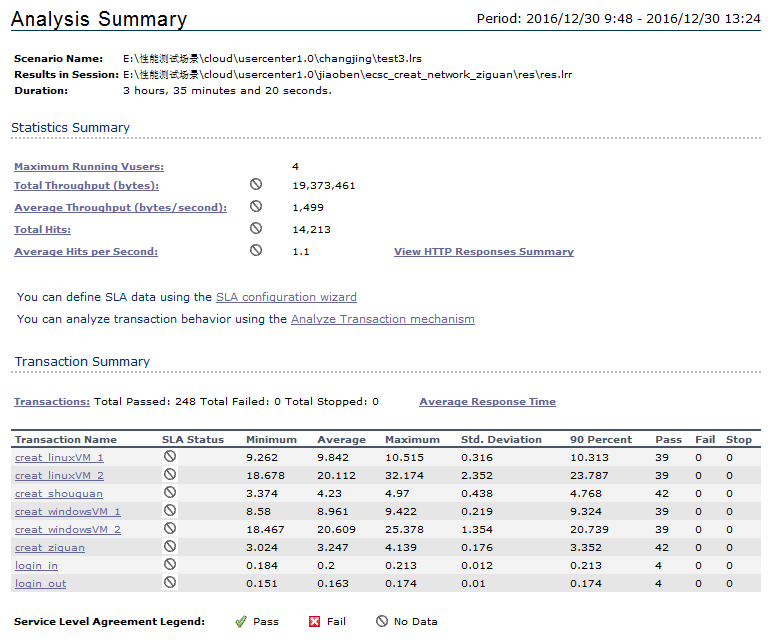


图3-4

事务名称解释：

creat\_linuxVM\_1： 点击创建虚拟机按钮到订单提交之前的操作  
creat\_linuxVM\_2：点击提交订单到订单状态变为完成  
creat\_shouguan： 点击创建子网到受管子网创建完成  
creat\_windowsVM\_1： 点击创建虚拟机按钮到订单提交之前的操作  
creat\_windowsVM\_2：点击提交订单到订单状态变为完成  
creat\_ziguan：点击创建子网到自管子网创建完成

结果分析：

1. 创建liunx与windows云主机各运行39次，全部pass；创建自管与受管子网个运行42次，全部pass。

2. 需求中创建云主机的请求响应时间为‘云主机购买界面点击提交订单按钮到订单变为‘已完成’状态’，故得出的数据为creat\_linuxVM\_2 90%事务通过的时间23.787秒，creat\_windowsVM\_2 90%事务通过的时间20.739秒。

3. 创建子网的平均请求响应时间为‘点击创建子网按钮到子网创建成功’，得出自管子网90%事务通过时间3.352秒，受管子网90%事务通过时间4.768秒。

# 调优前后性能分析对比

## 4.1性能优化点

第一轮的性能测试结果虽然已经满足线上用户并发量的需求，但针对场景一与场景二，在可优化的基础上，研发在JDBC配置、Tomcat配置、MySQL配置、SQL优化四个方面做了的参数调整，使ecsc平台的稳定性、用户并发量、90%用户的响应时间、每秒点击量等等有了非常明显的提升。

1）JDBC配置调整

修改JDBC配置的目的是建立的数据库连接池连接数，增加处理能力，效果十分明显，查看MySQL连接数，在压力时，能最多开启了712个连接，相比之前最大使用了277个连接，是有了长足的进步。

2）Tomcat配置调整

本次参数调整，主要涉及的是增加线程池，增加线程池中的最大线程数（单核CPU配置为200左右是正常的，对于我们的四核CPU的MySQL服务器，可以开到800没问题的），开启NIO，调整使用NIO模型时接收线程的数目，增加等待连接队列长度（acceptCount），禁用DNS查询。

1. MySQL配调整

主要是增加query\_cache\_size及打开慢查询。慢查询是将哪些进入数据库的请求超过X秒后，将其记录在指定日志中，方便研发后期查询问题。query\_cache\_size，当数据库打开了Query Cache（简称QC）功能后，数据库在执行SELECT语句时，会将其结果放到QC中，当下一次处理同样的SELECT请求时，数据库就会从QC取得结果，而不需要去数据表中查询，优化了查询的速度。

1. SQL优化

基于第一轮、第二轮压测结果，响应时间最慢的事务，沟通事务点，找到对应的查询，对SQL进行分析、查看解释计划、查看相关表索引设计等，最终在尽可能不修改业务的情况下，合理添加索引，提高查询的速度。

## 4.2 性能结果分析对比

### 4.2.1 多用户并发登录优化前后性能对比（场景一）

优化前500用户绝对并发：

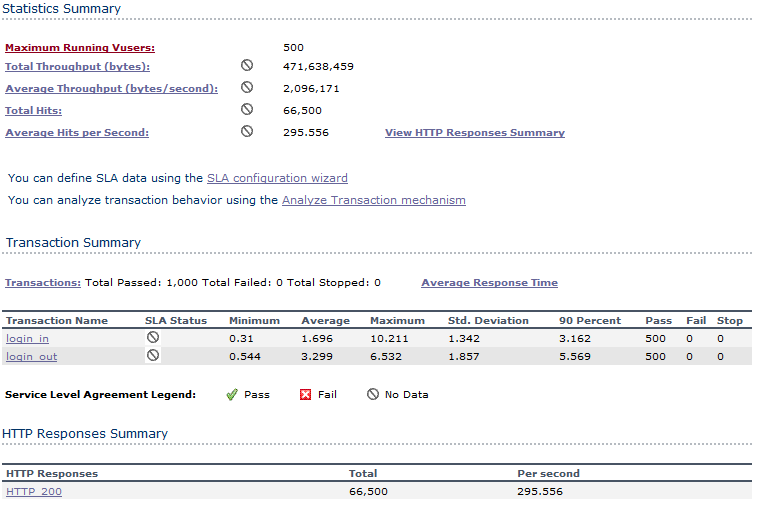


图4-1

优化后500用户绝对并发：

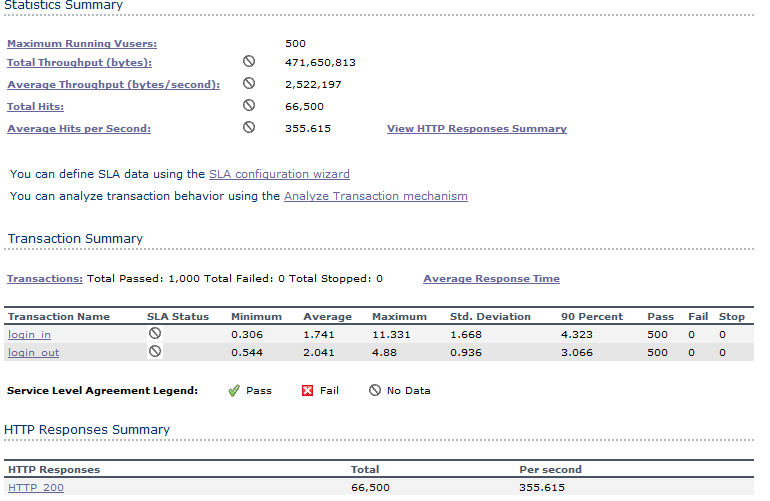


图4-2

优化后600用户绝对并发：

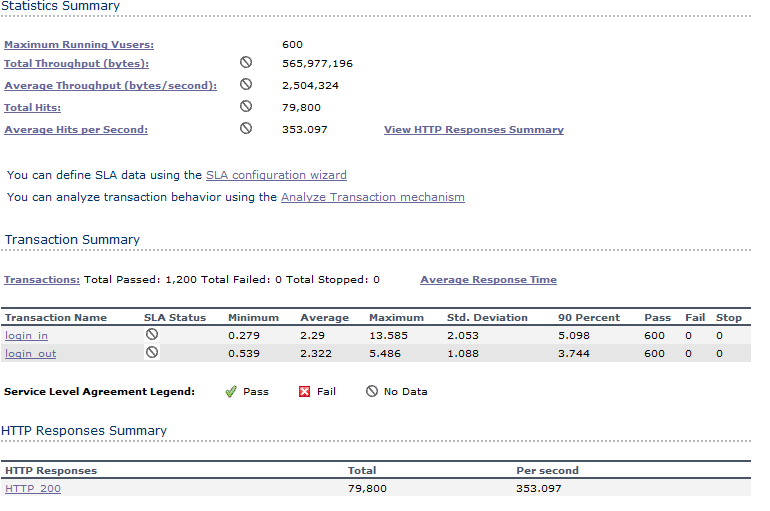


图4-3

对比分析表：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **场景1绝对并发分析表** | | | | | | |
| **并发用户数** | **平均点击数** | | **标准偏差** | | **90%用户通过响应时长** | |
| **login** | **logout** | **login** | **logout** | **login** | **logout** |
| 500优化前 | 295.556 | | 1.342 | 1.857 | 3.162 | 5.569 |
| 500优化后 | 355.615 | | 1.668 | 0.936 | 4.323 | 3.066 |
| 600优化后 | 353.097 | | 2.053 | 1.088 | 5.098 | 3.744 |

表2

优化前后对比分析：

前提：优测试脚本一直未变化，数据库中的数据量未变化

图4-1是优化前第一次500并发测试的Analysis Summary， 图4-2与图4-3是优化后500并发登录、600并发登录的Analysis Summary。首先，从并发用户数来看，优化后的虚拟用户并发数由500提高到了600，三个并发的事务成功率都在100%，说明实际用户并发人数提高到了6000人，提高了20%的用户并发数。平均点击率由295提高到了355左右，提高了16.7%速率。‘标准偏差’与‘90%用户通过的响应时长’都是随着并发数量的增大而增大，优化后的稳定性和响应时间相对变化不大。

### 4.2.2 多用户广义并发登录优化前后性能对比（场景二）

优化前700用户广义并发：

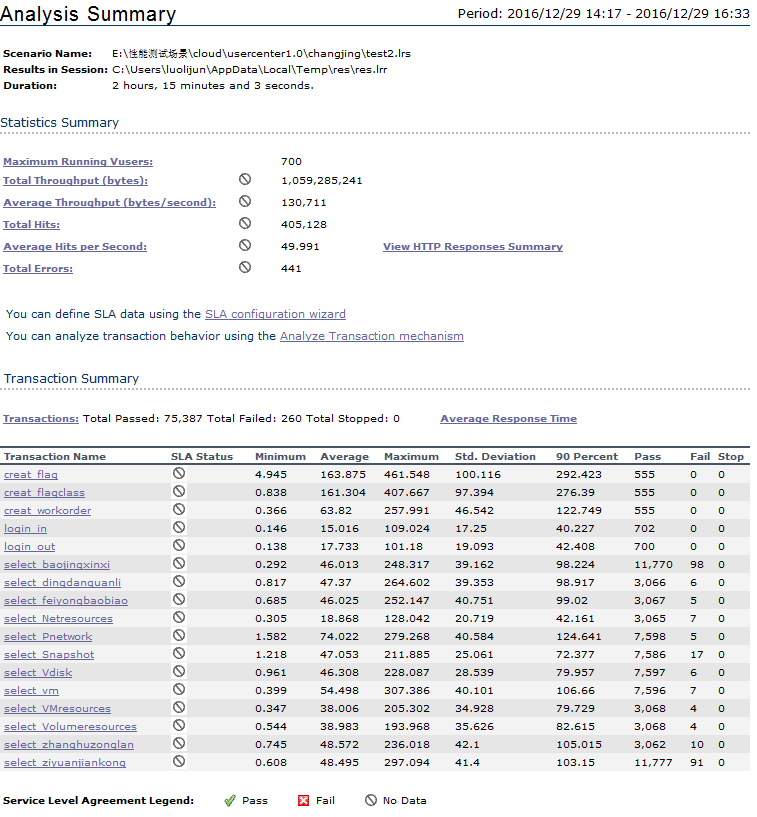


图4-4

优化后700用户广义并发：

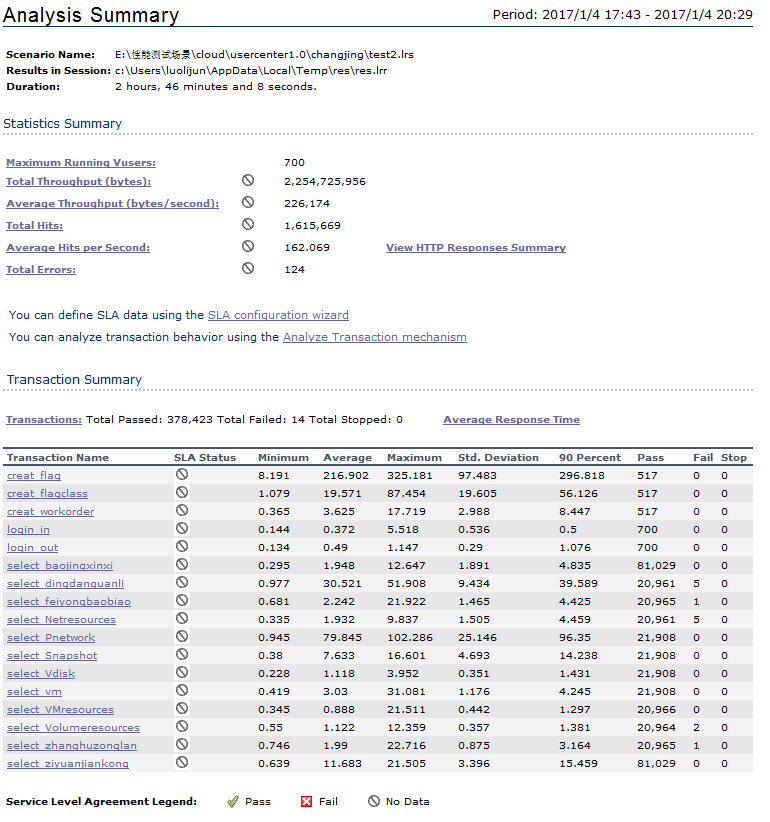


图4-5

二次优化后700用户广义并发：

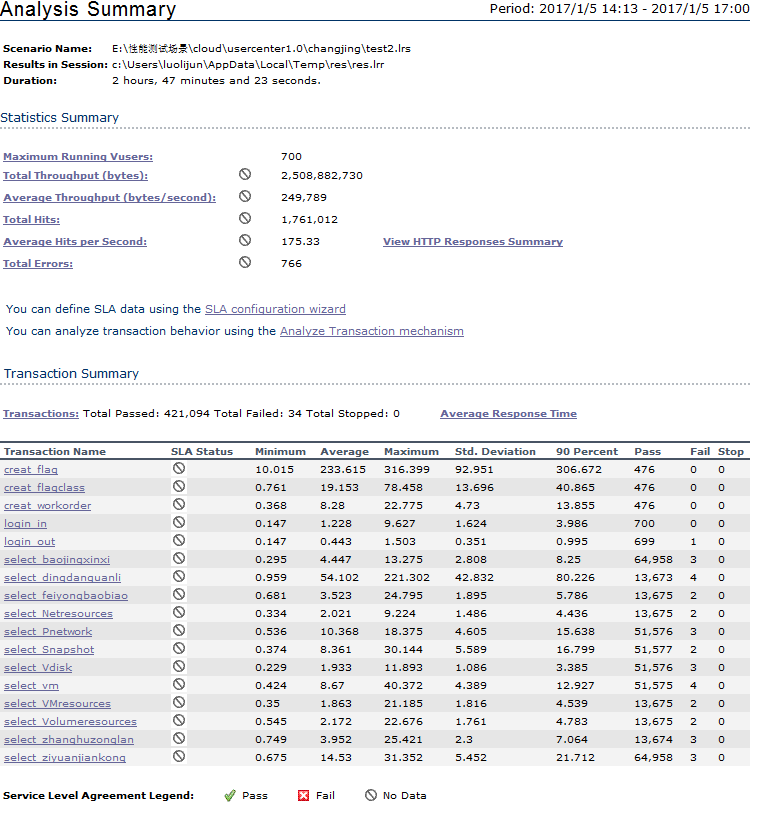


图4-6

|  |  |  |  |
| --- | --- | --- | --- |
| **Transaction Name** | **二次优化后** | **优化后** | **优化前** |
| [creat\_flag](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(creat_flag)0000) | 306.672 | 296.818 | 292.423 |
| [creat\_flagclass](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(creat_flagclass)0000) | 40.865 | 56.126 | 276.39 |
| [creat\_workorder](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(creat_workorder)0000) | 13.855 | 8.447 | 122.749 |
| [login\_in](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(login_in)0000) | 3.986 | 0.5 | 40.227 |
| [login\_out](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(login_out)0000) | 0.995 | 1.076 | 42.408 |
| [select\_baojingxinxi](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(select_baojingxinxi)0000) | 8.25 | 4.835 | 98.224 |
| [select\_dingdanguanli](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(select_dingdanguanli)0000) | 80.226 | 39.589 | 98.917 |
| [select\_feiyongbaobiao](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(select_feiyongbaobiao)0000) | 5.786 | 4.425 | 99.02 |
| [select\_Netresources](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(select_Netresources)0000) | 4.436 | 4.459 | 42.161 |
| [select\_Pnetwork](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(select_Pnetwork)0000) | 15.638 | 96.35 | 124.641 |
| [select\_Snapshot](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(select_Snapshot)0000) | 16.799 | 14.238 | 72.377 |
| [select\_Vdisk](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(select_Vdisk)0000) | 3.385 | 1.431 | 79.957 |
| [select\_vm](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(select_vm)0000) | 12.927 | 4.245 | 106.66 |
| [select\_VMresources](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(select_VMresources)0000) | 4.539 | 1.297 | 79.729 |
| [select\_Volumeresources](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(select_Volumeresources)0000) | 4.783 | 1.381 | 82.615 |
| [select\_zhanghuzonglan](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(select_zhanghuzonglan)0000) | 7.064 | 3.164 | 105.015 |
| [select\_ziyuanjiankong](C:\\Users\\luolijun\\AppData\\Local\\Temp\\ResponseTime0000(select_ziyuanjiankong)0000) | 21.712 | 15.459 | 103.15 |

表3

前提：1）测试脚本一直未变化，同样是4个脚本，而且所以每个脚本用户量的比例同优化前后一致。

2）数据库中的数据量未变化。

3）测试时长都在2个半小时左右，相差不大

优化先后对比分析：

图4-4是优化前第一次性能测试时场景二的700用户广义并发，图4-5是首次优化后场景二的700用户广义并发。从两个图标中可以看出：优化后执行的事务总数为378423，优化前是75387，在相同时间内执行的事务数‘优化后’是‘优化前’的5倍，说明执行的速率提高了5倍。 每秒的点击率提高了三倍多（由49.991 提高到162.069），错误失败率由3.45%降低到了0.0037%，失败事务数（260降低14个）。90%用户的响应时间相对于第一次来讲只有创建标签和查询私有网络变动不大（需要优化），其他事务都有非常显著的性能提高。 优化后的标准偏差小于1的有6个事务，方差在1-5之间的有7个， 方差大于5的有4个事务。 系统稳定性整体来讲比第一轮有非常显著的提高。

图4-6是针对‘私有网络’进行的优化，私有网络的查询时间由96.35秒降低到了15.638，但是失败率由原来的0.0037%提高到了0.0081%，因为调优后的第一次性能测试是在下班后，二次调优后的测试在上班时间，初步分析受网络影响很大。

综上所述，性能调优后，用户数量、系统稳定性、90%用户通过响应时间、每秒的点击率等等都有比较明显的提高。