伴随云计算日益普及以及人工智能技术日益成熟，推动信息科技向物联网时代转变，特别在IoT＋AI融合下使得万物具有感知能力，使用场景已在军事、农业、医疗等领域投入应用。面对海量的终端设备，平台接入层的压测是一个必要的环节。

下面结合一个简单的私有协议使用JMeter自带的TCP Sample完成并发的TCP性能测试。

## 设备通信协议

此协议只是为了说明压测流程，并非真实的私有协议。

* 第一步：身份认证

左边是终端request，右侧是server端的response

* Message\_id：表示请求的唯一id
* Device\_id：表示终端设备的id
* Message\_type：表示报文类型
* Req\_time：表示请求发送的时间
* Sign：表示签名，防止被中间人篡改
* Data：表示具体的业务数据



* 第二步：上传地理位置信息（认证通过后）



* 第三步：上传设备基本信息

同上

* 第四步：定时上传心跳和地理位置

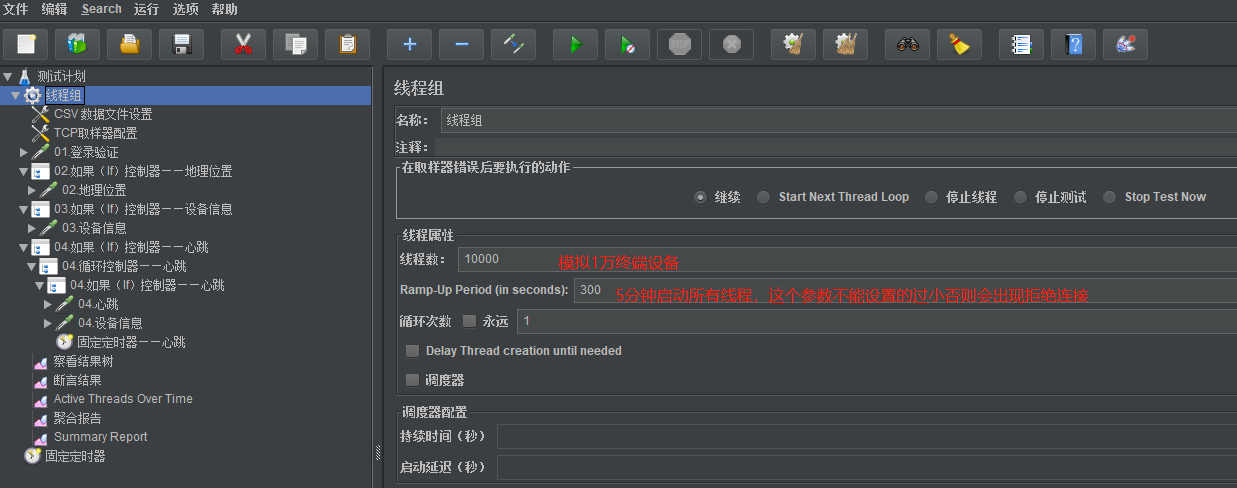
心跳时间间隔是15秒，为了加大对服务端的压力，在心跳的时候同时上传一次设备信息报文

## 模拟场景

本次压测，模拟1万台终端设备在5分钟内完成建立全部连接，平均每秒建立33个连接。连接建立后每隔15秒发送一次心跳和业务数据，长连接维持1万个终端设备同时在线。

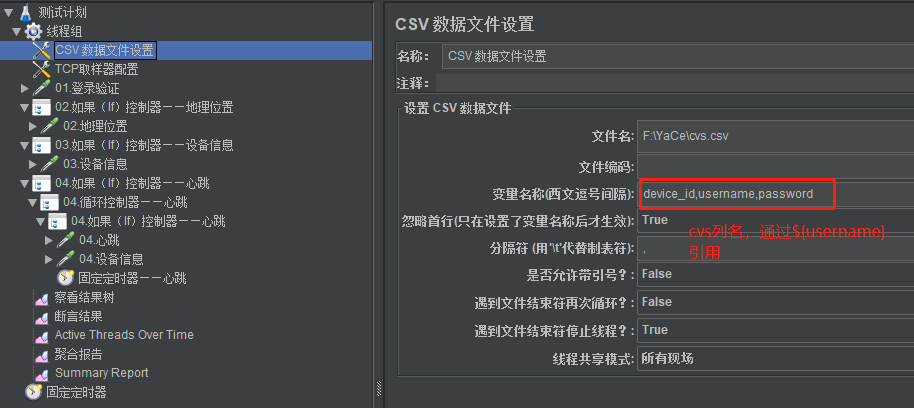
## JMeter 参数配置

### 设置线程数：10000， ramp-Up Period:300

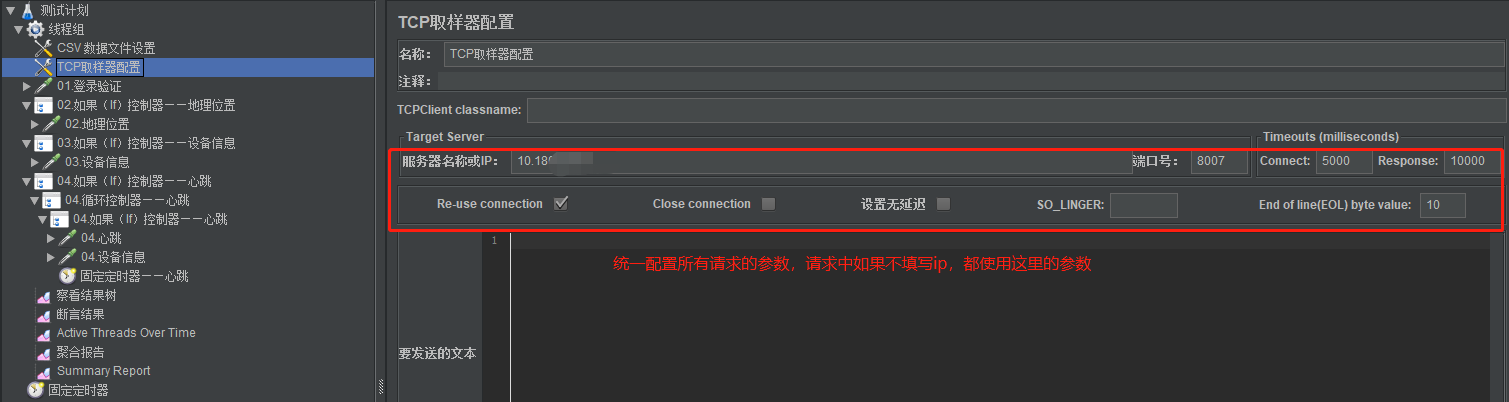


### 设置.cvs文件的路径

.cvs 文件中包含3列分别是终端id、登录名、密码

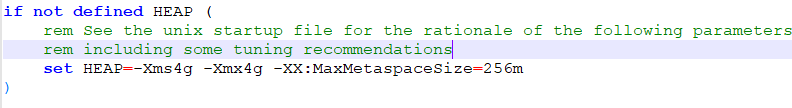


### 设置统一的ip地址等参数



## JMeter 参数优化

修改JMeter的/bin 目录下的jmeter.properties文件。默认最大内存为1G，如果不修改可能会内存溢出。



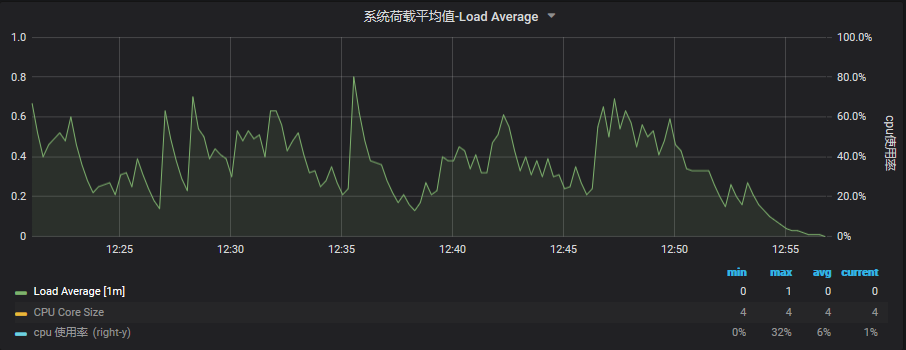
## 服务端监控和压测报告

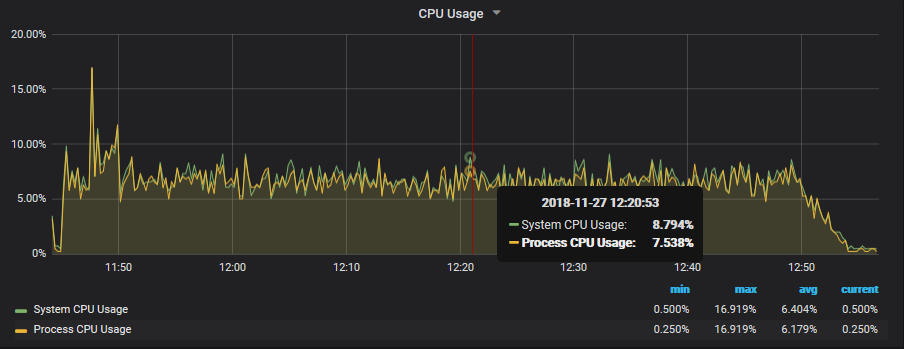
### 文件句柄数监控

### 文件句柄数

为啥模拟了1万终端数，句柄数会占用到2万多呢？这是因为我的程序中每个终端会占用一个日志文件这样就导致句柄数翻倍。同时，系统默认的句柄数是1024，可以通过ulimit -n 30000 命令暂时修改最大句柄数为30000，同时在此终端窗口中启动应用，此时应用的最大句柄数将是30000。

### CPU和系统负载监控





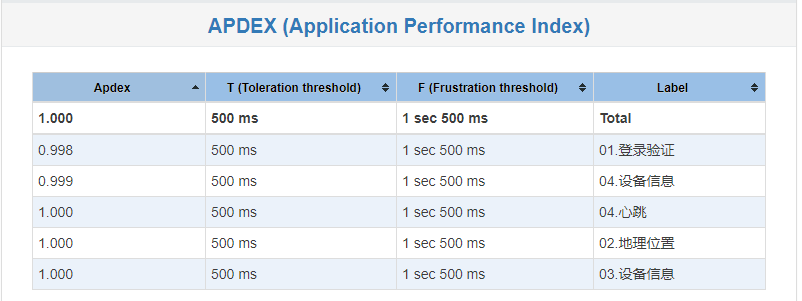
系统负载基本在0.6以下，CPU使用率也不超过10%。 测试用的服务器是8核16G内存。

### 日志监控

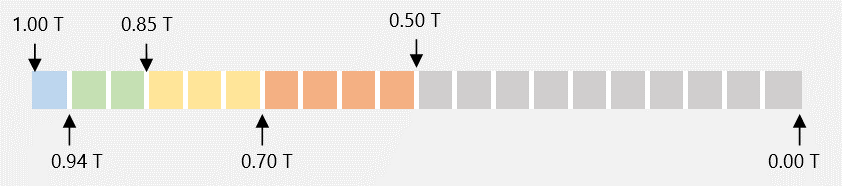


压测过程中没有出现warn和error级别的日志，debug级别日志超过2000tps，info级别的日志也超过600tps。

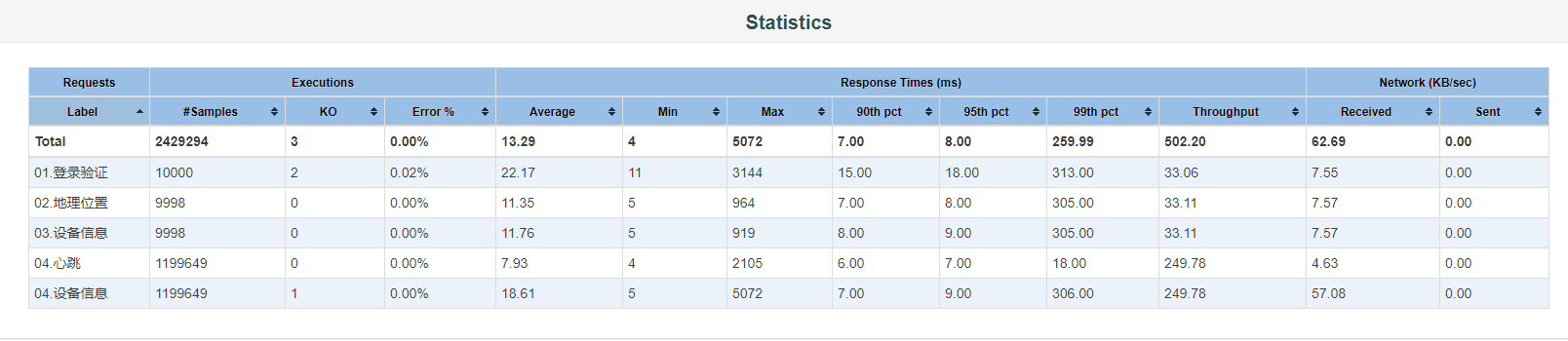
### JMeter压测报告局部展示



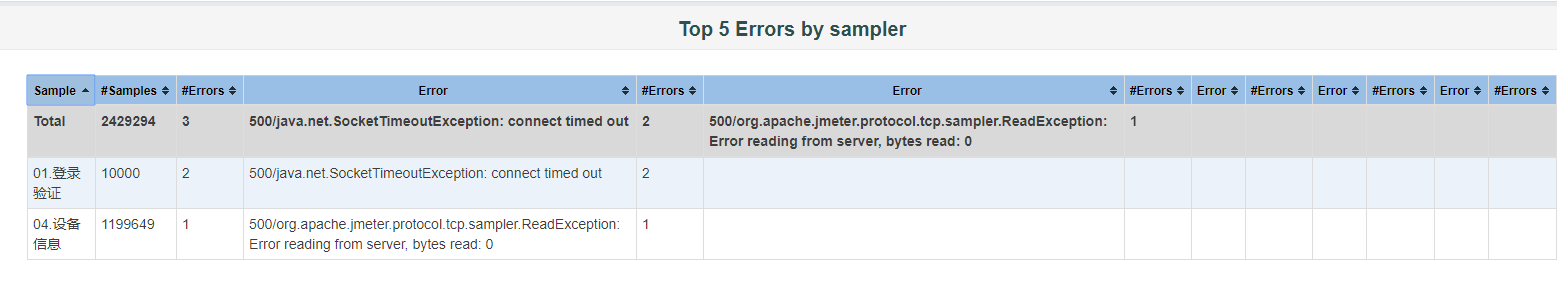
Apdex(Application Performance Index)是一个国际通用标准，Apdex 是用户对应用性能满意度的量化值。它提供了一个统一的测量和报告用户体验的方法，把最终用户的体验和应用性能作为一个完整的指标进行统一度量。 0 到 1之间的数值即 “Apdex 指数”，0 代表没有满意用户，1则代表所有用户都满意



statistics 中包含了很多的信息。



整个压测过程中出现3个连接断开，2个因为连接超时，1个因为读超时。总体来说还是可以接受。



### 总结

作为一个开发人员做压力测试确实挺辛苦的，着实感叹能力不足，压测确实是一门深不可测的学科（因为还需要通过压测结果进行优化）。虽然我们开始说是1万个终端的压测，实则只是一个数字，并非压测工具或者服务端性能的极限。

美中不足之处：

1. 此次压测没有涉及到二进制的通信协议，只是一个简单的json协议。
2. 不能对服务器推送的消息进行响应，无法测试服务端的推送能力。