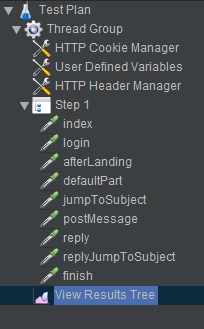
JMter实训报告

**1、 录制** （由于系统原因，BugFree、然之协同有问题，在此使用了Discuz,录制操作如下）

登录

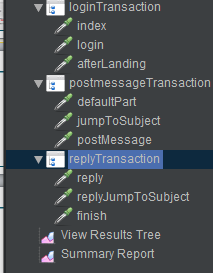
发帖

回复帖子



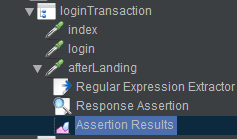
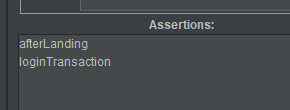
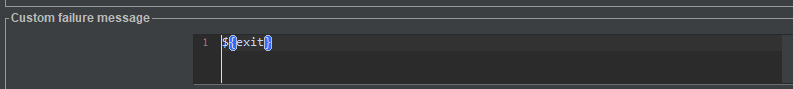
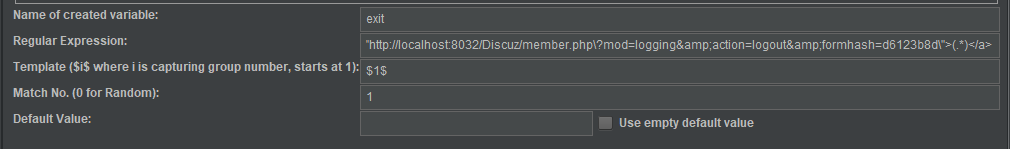
**2、 加事务**

对录制好的脚本添加事务控制器

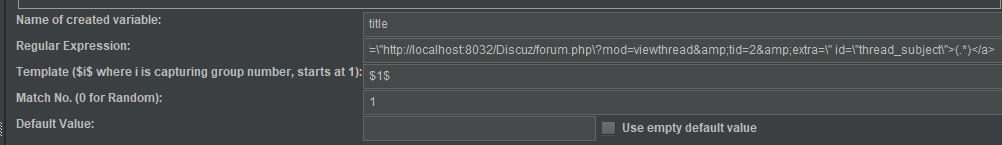


**3、 加检查点**

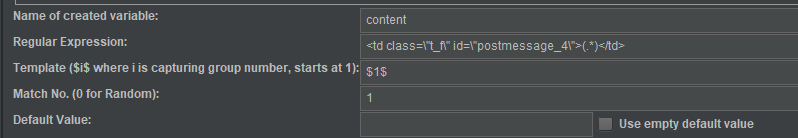
登录检查：登录成功后网页出现退出按钮，以此作为检查点



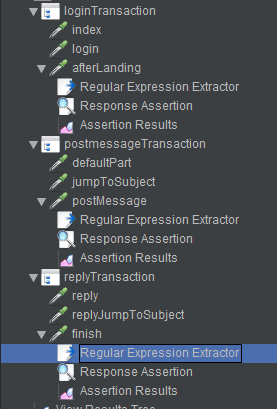
发帖：检测发帖成功之后的标题



回复：检测回复帖子的内容



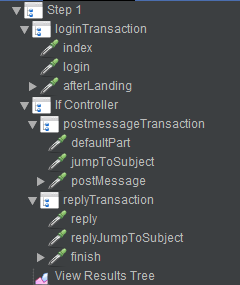
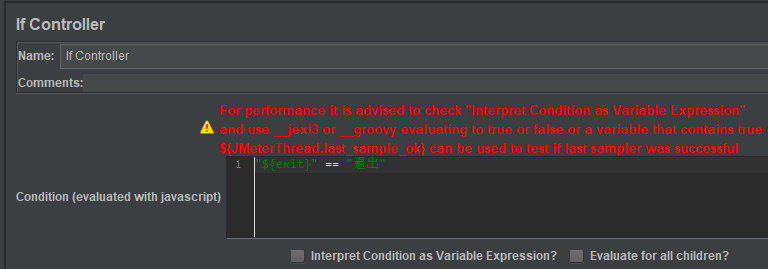
分布如下：



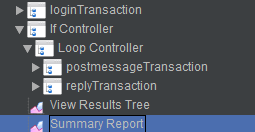
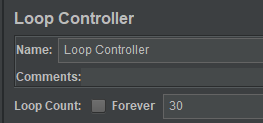
运行无错

**4、 加逻辑控制**

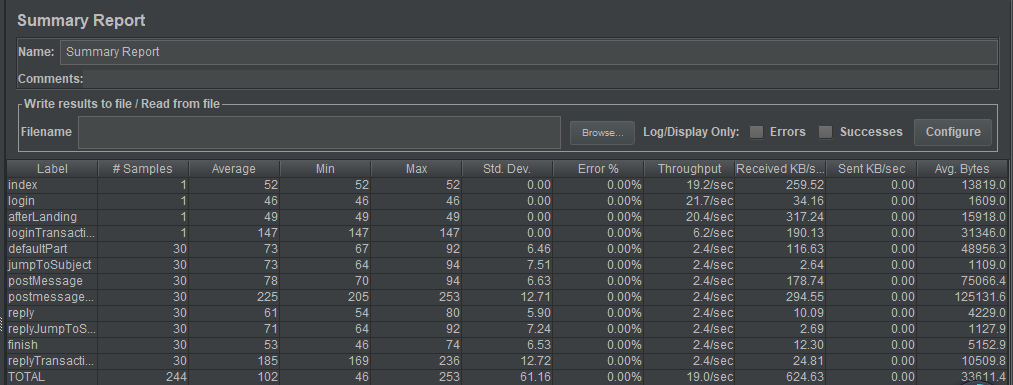
如果登录成功，则可以进行发帖和回复



当成功登录后，连续做30次发帖和回复（在if controller里加一个循环控制器Loop Controller）

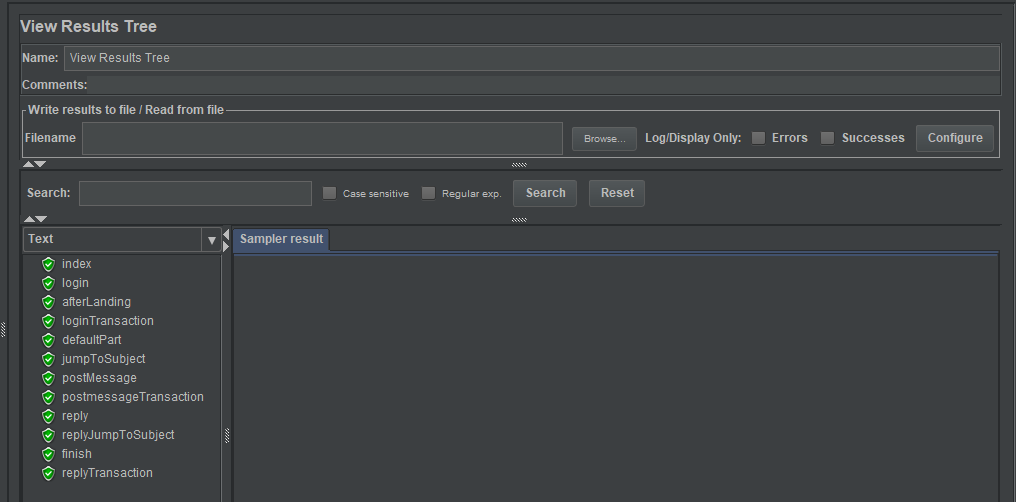
 

发帖和回帖运行30次结果如下：

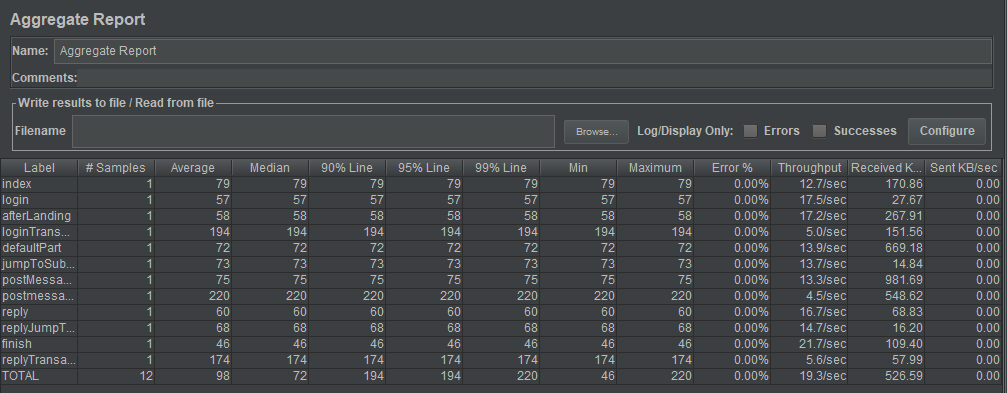


**5、 加监听器**

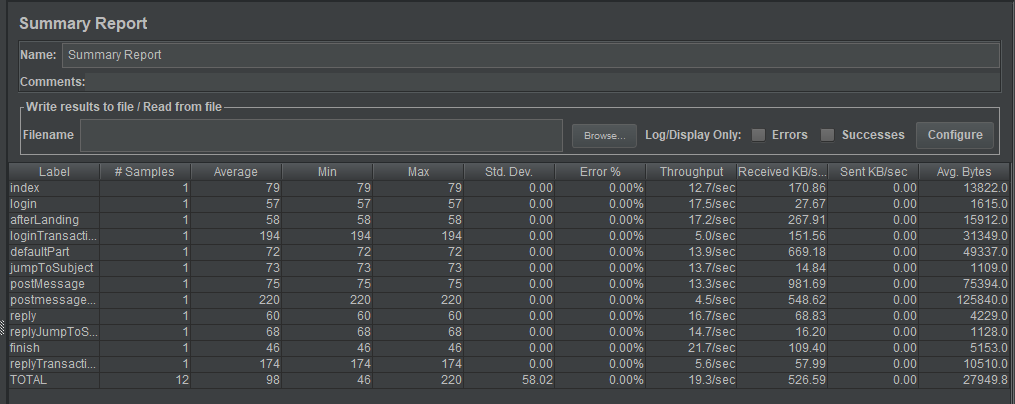
树形监听器（View Results Tree）：如下图，运行成功



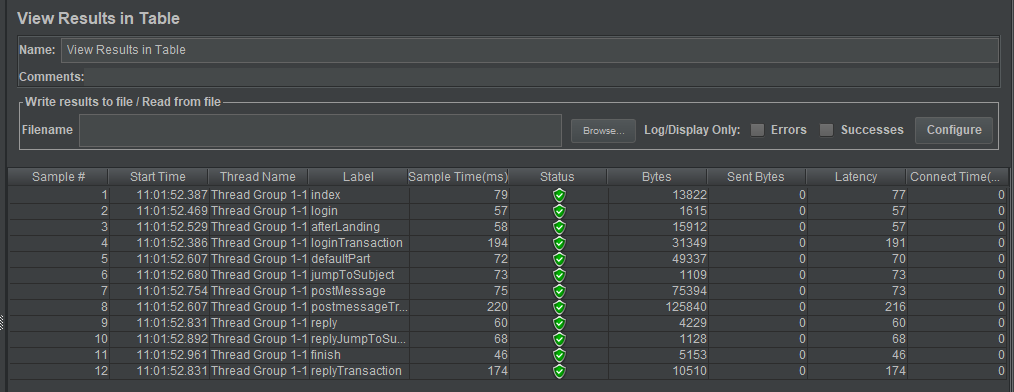
聚合报告（Aggregate Report）：如下图，统计每个请求的响应信息并提供请求数，平均值，最大，最小值，中位数、90%、95%用户响应时间、错误率，吞吐量(以请求数/秒为单位)和以kb/秒为单位的吞吐量



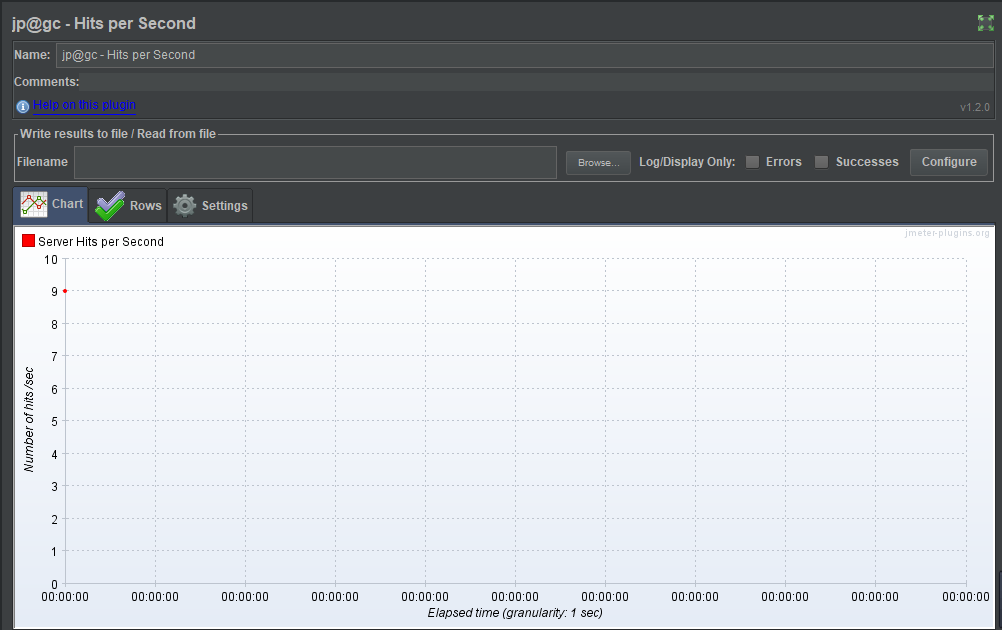
概要报告(Summary Report)



用表格查看（View Results in Table）：

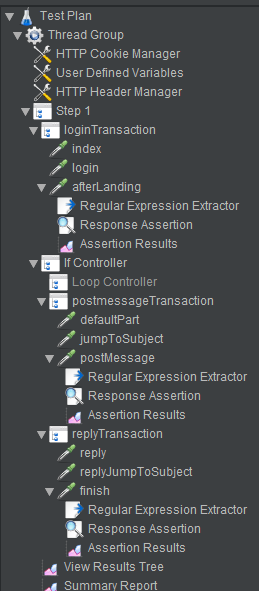


还可以进行图形化（以点击率为例）：

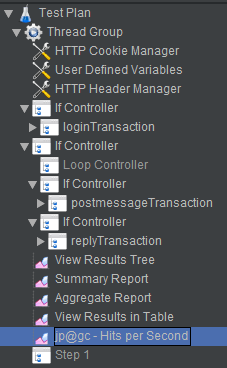


**6、 创建场景**

在创建场景前，层级关系及如下（step1为录制脚本后有的循环控制器Loop Controller，为了后面的创建脚本，将循环控制器去掉）：

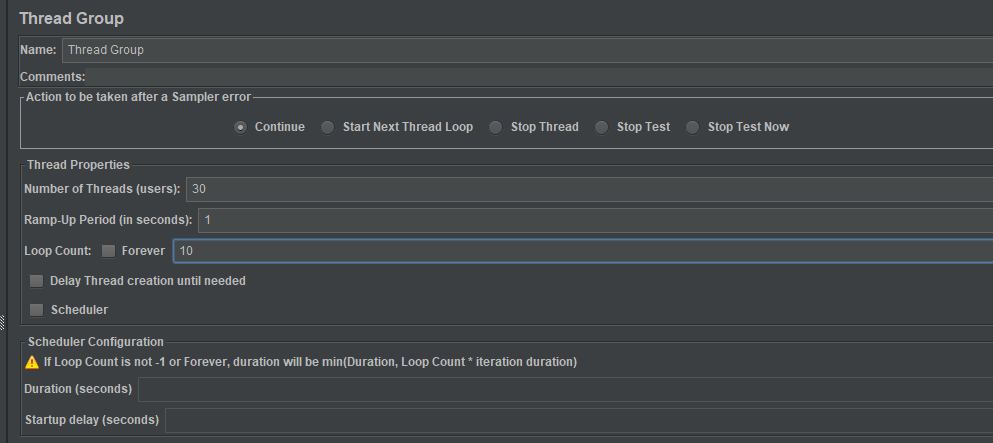


通过If Controller和线程迭代计数函数（ ${\_\_counter(true,)} ）来实现用户组式场景

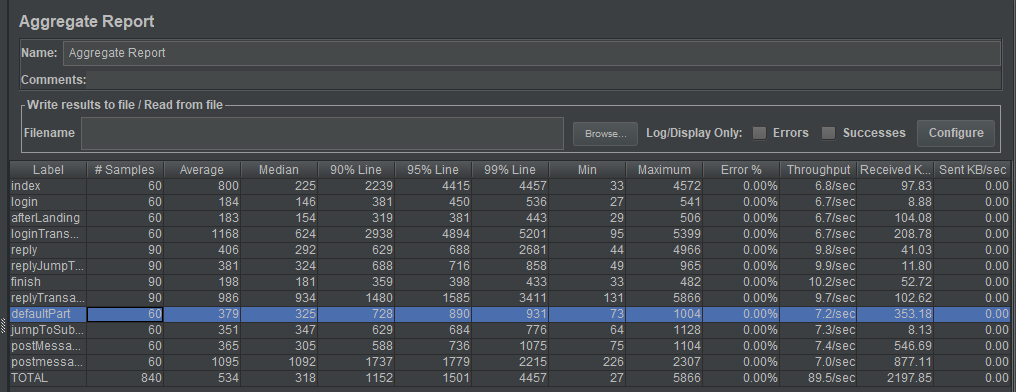


30并发量：20%用户做登录、30%做发帖、50%做回帖

修改用户数为30，运行



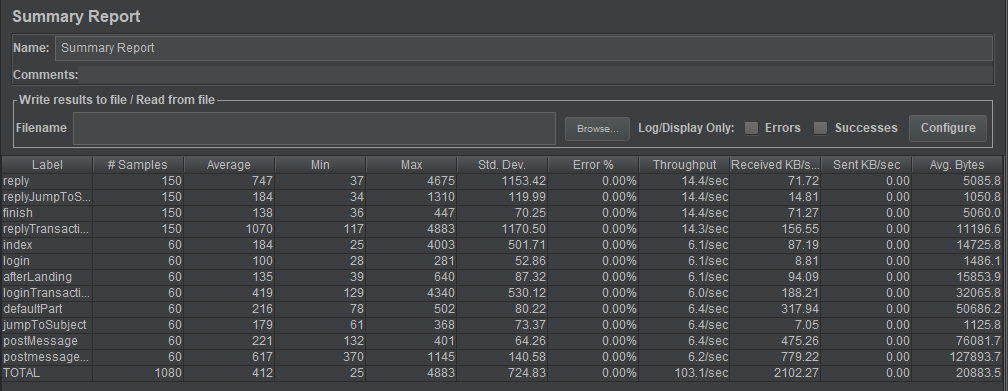
此时运行结果



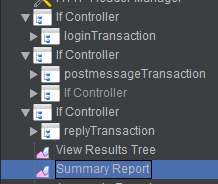
不符合20% 30% 50%,猜测是检查点的ifControler的问题，

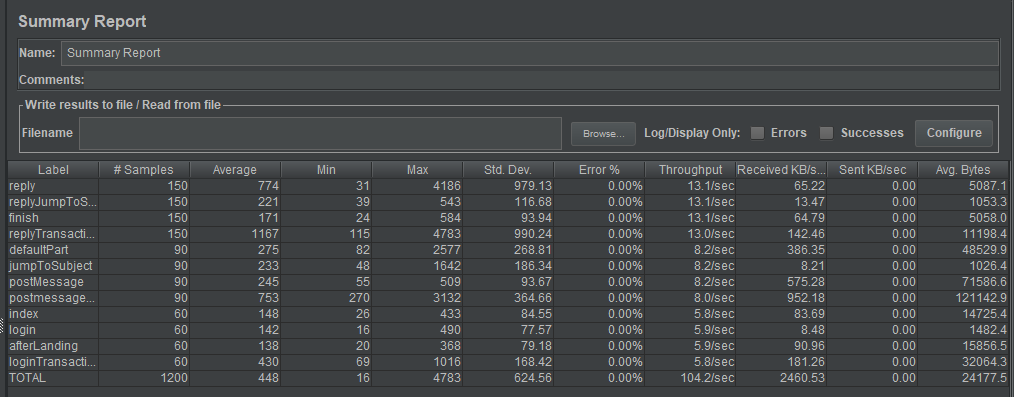
将检查点改为如果登录成功，则可以进行发帖；如果登录成功，则可以进行回复，分别作为带有\_\_counter函数的控制器的子节点，结果同上图，不符合20% 30% 50%

将检查点改为如果登录成功，则可以进行发帖作为带有\_\_counter函数的控制器的子节点，结果同下图，reply改变，但是仍然不符合20% 30% 50%



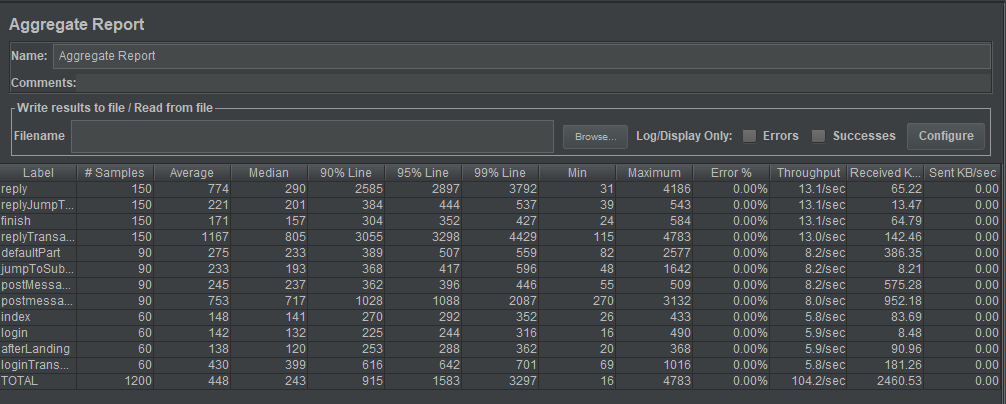
此时考虑到可能是系统的问题，则去掉检查点

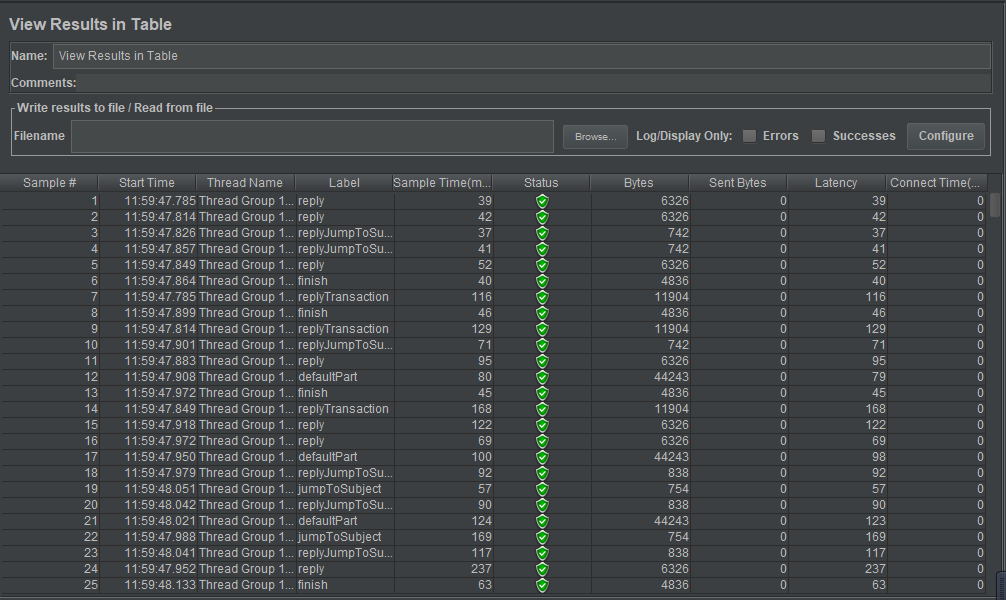
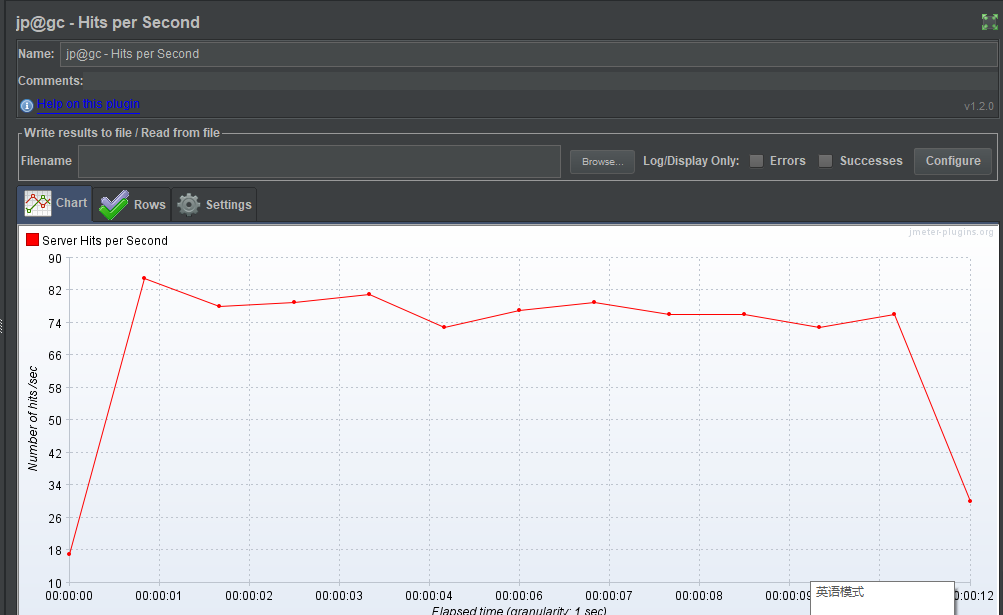
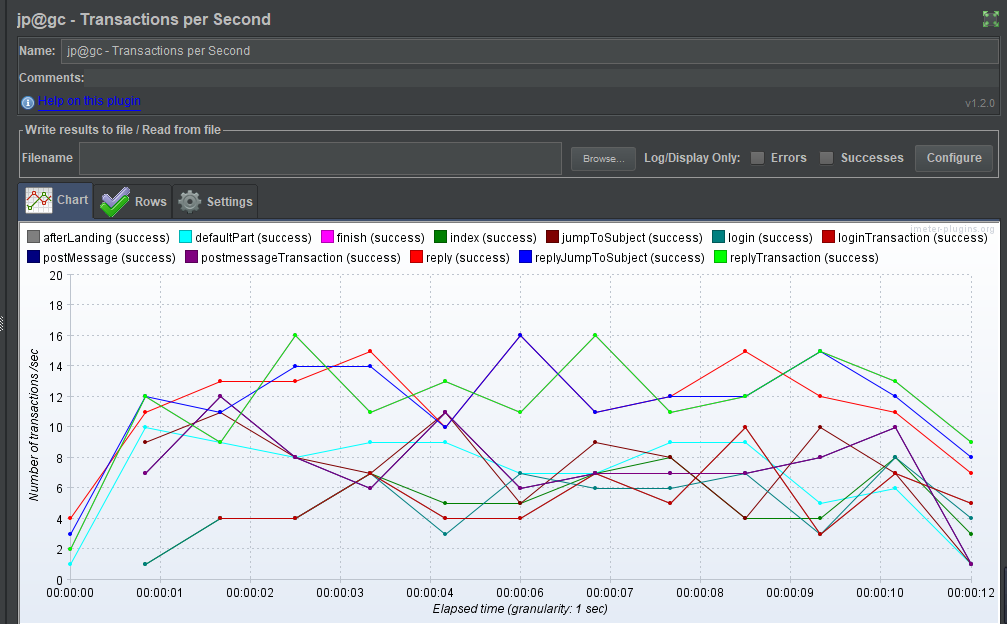




此时满足并发条件

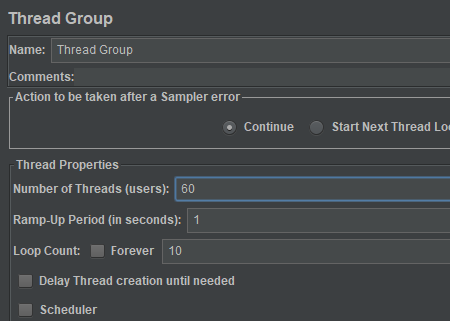
其他监听器结果如下：

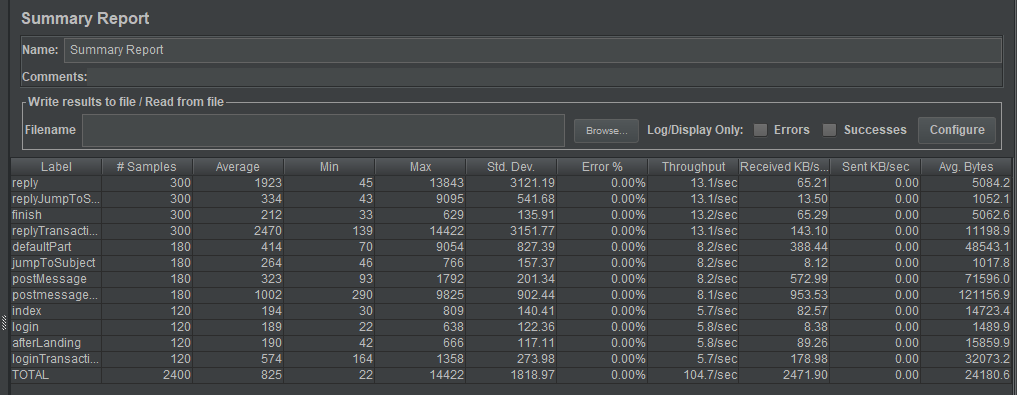
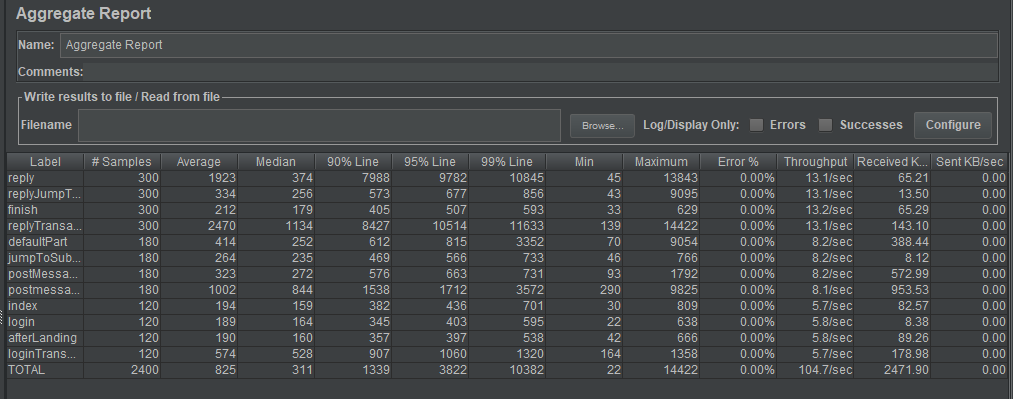
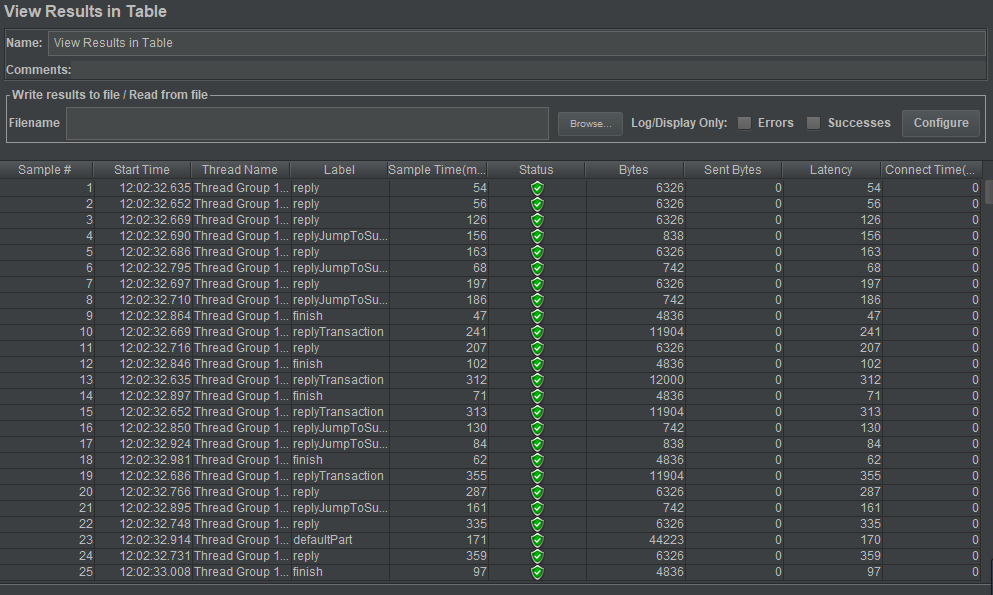
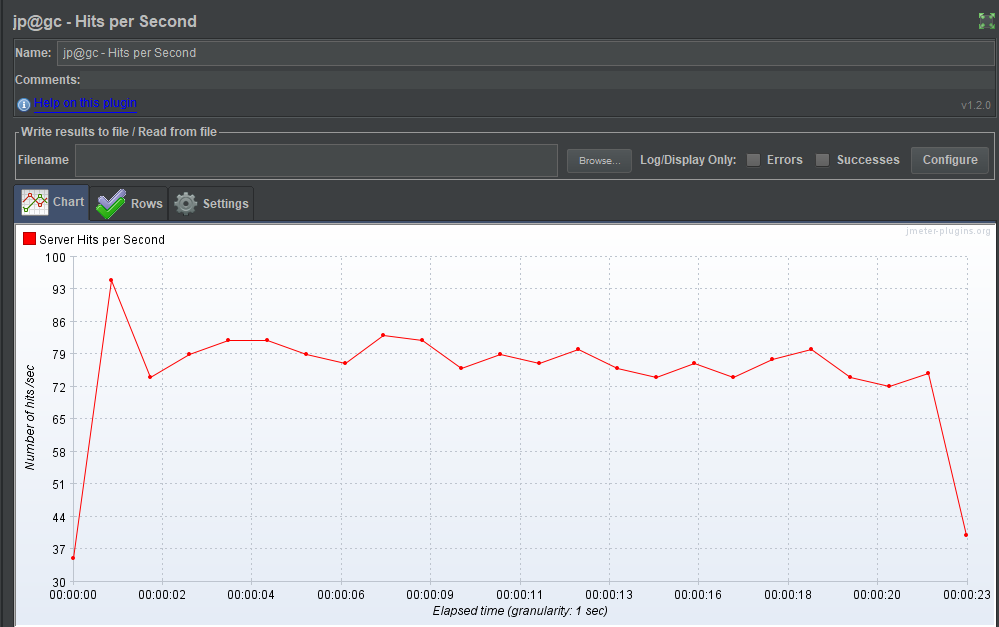
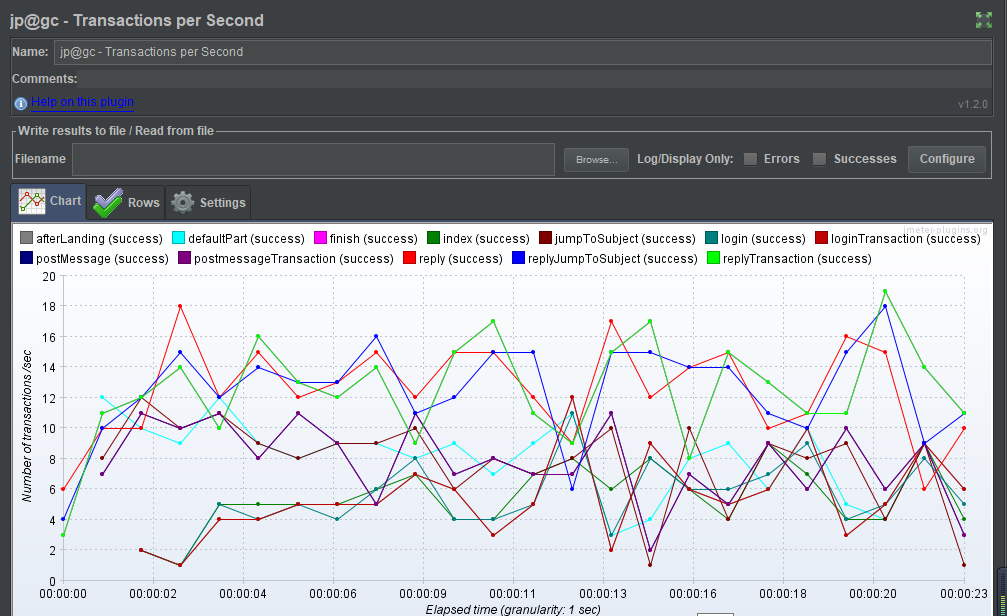


60并发量：20%用户做登录、30%做发帖、50%做回帖

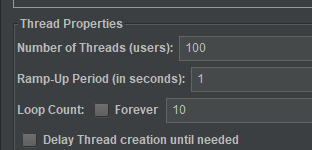
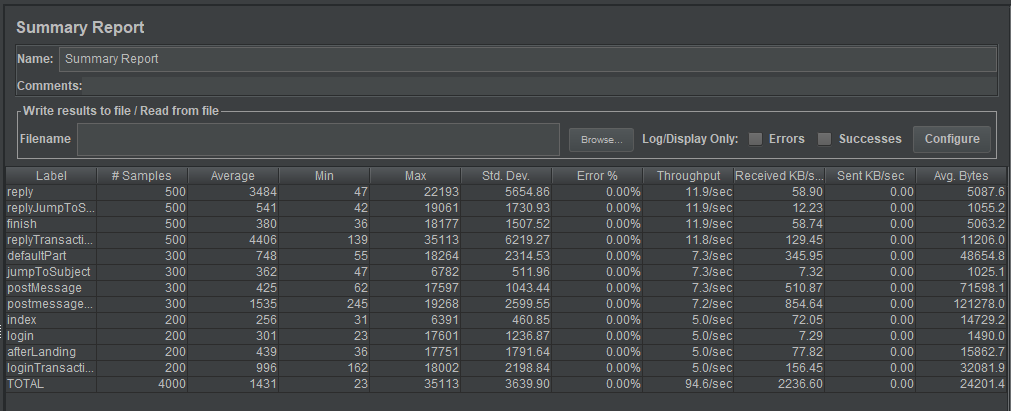
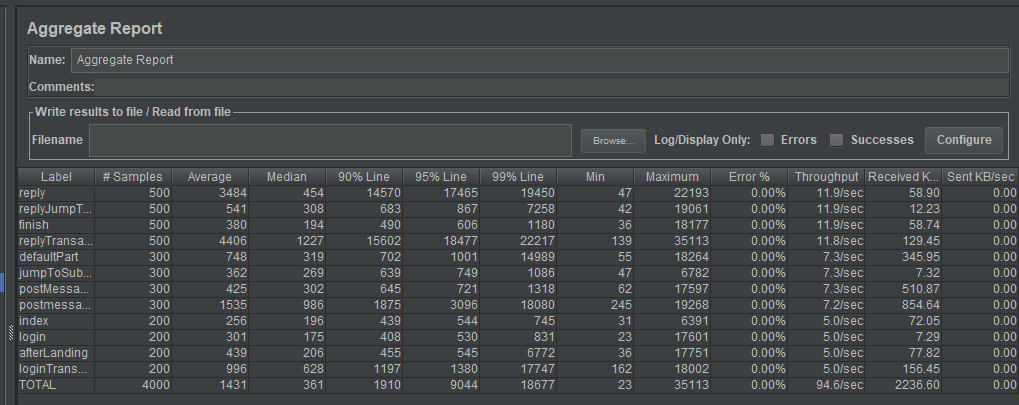
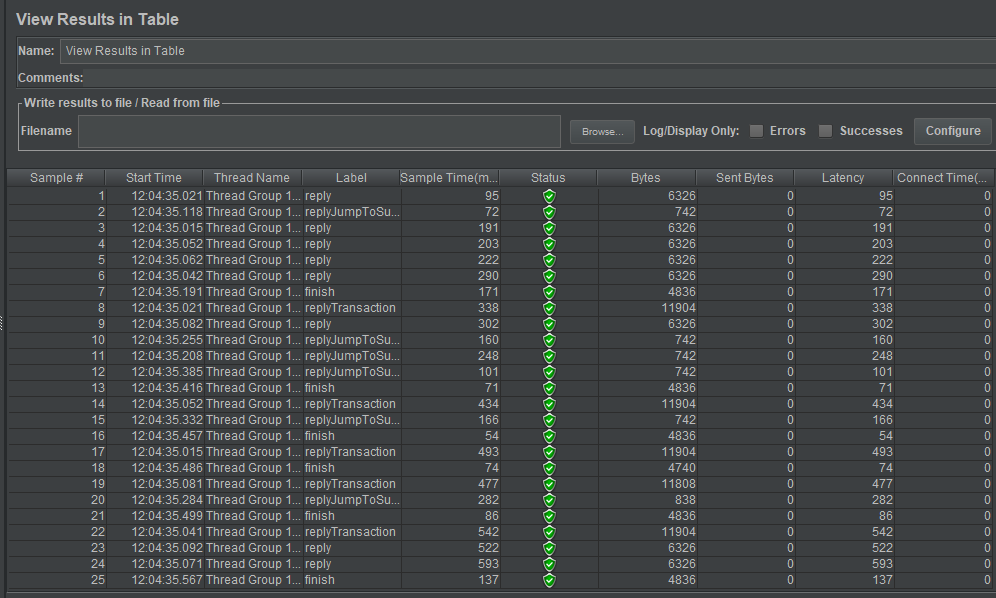
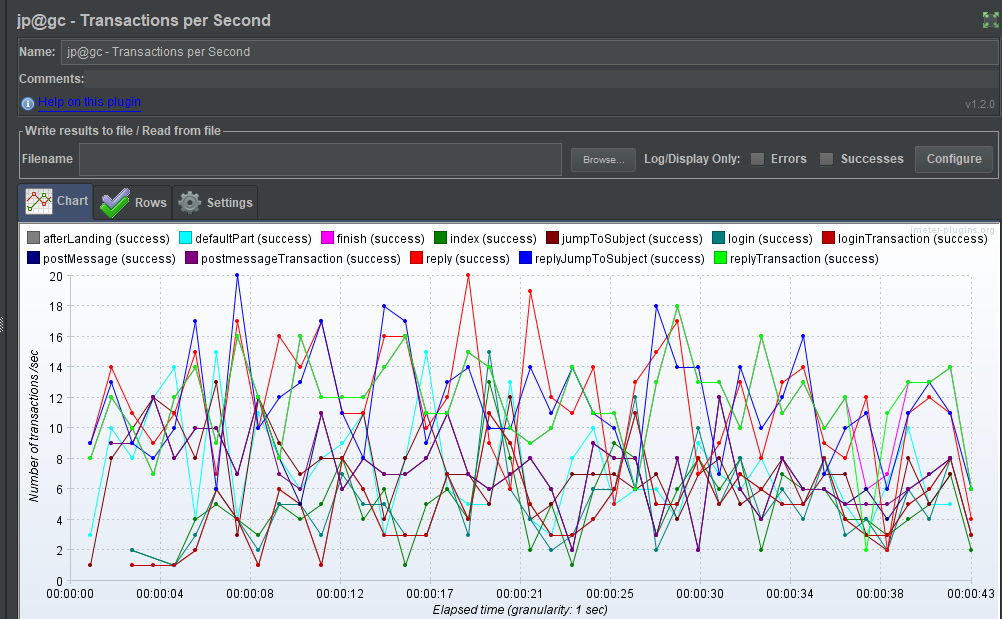
修改用户数为60，监听器报告如下：



100并发量：20%用户做登录、30%做发帖、50%做回帖

修改用户数为60，监听器报告如下：

**7、 总结报告**

问题：在一个事务中定义的用正则表达式表示出的参数，在与本事务同级的事务中按理来说不可以调用，但是在这里可以