

软件测试上机报告



第六次上机作业 - Lab6 Jmeter

学 院 智能与计算学部

专 业 软件工程

姓 名 郎文翀

学 号 **3019244247**

年 级 2019 级

班 级 软件工程 5 班

1. Experimental requirements

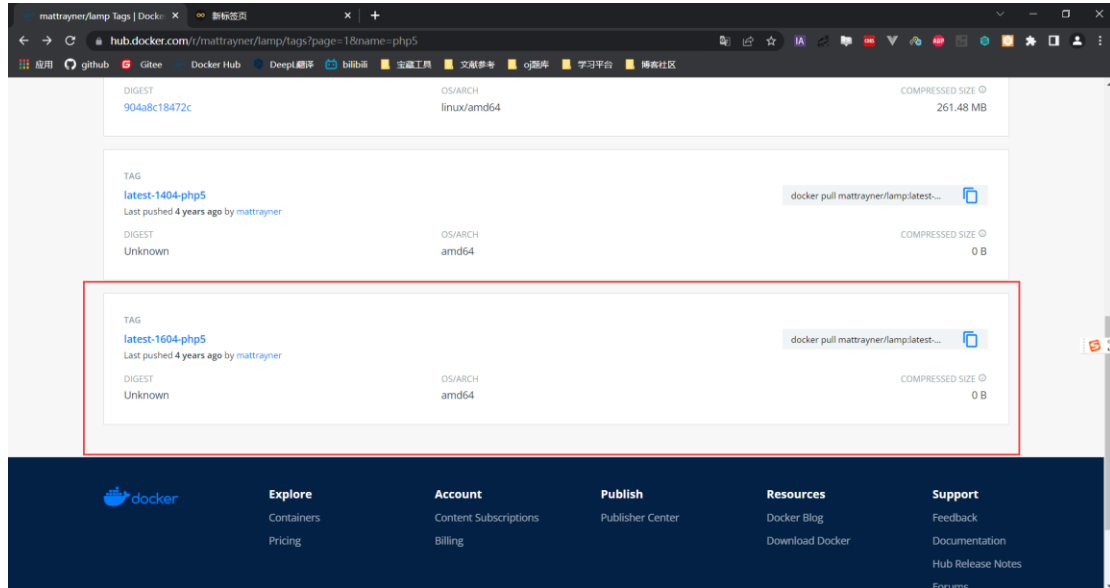
安装虚拟机，在虚拟机上搭建 LAMP（Linux+Apache+Mysql+PHP）工作环境，并基于此安装待测系统，推荐 ECShop（<http://www.ecshop.com>）或 BugFree（<https://www.bugfree.cn/>）（也可以尝试最新的禅道项目管理软件），基于此进行 Jmeter 压力测试，并在测试后得出 Jmeter 测试报告，并根据 top 得出 Linux 服务器的 CPU、Memory 等信息。

1. 安装 LAMP: Linux 服务器（Centos/Ubuntu）、Apache、MySQL 及 PHP;
2. 安装 ECShop 或 BugFree 待测系统（或类似 B/S 结构的系统）
3. 使用 Jmeter 录制 ECShop/BugFree 创建新订单/Bug 描述的行为，并尝试更改部分行为（例如不同商品的订单、Bug 标题等）；
4. 使用 Jmeter 进行 5*10、50*20 的压力测试并得出 Jmeter Aggregate Report，同时运用 sysstat 对服务器信息进行统计。
5. 添加 Beanshell 代码对压测结果进行分析，测试功能自定义。
6. 基本要求：实践压力测试工具 Jmeter
7. 相关日期：
 - i. 提交报告日期：2022.4.20 按照格式提交到智慧树
8. 实验报告应包括：
 - i. Linux 下 top 命令结果截图
 - ii. 访问的 B/S 系统截图
 - iii. Jmeter 的 Testplan 展开截图
 - iv. Beanshell 代码（功能可自定义）
 - v. 运行 Jmeter 测试之后的 Aggregate Report Result
 - vi. 运行 Jmeter 测试之后的服务器性能截图
 - vii. 根据 sysstat 结果对压力测试对服务器性能产生的影响作简要的分析
 - viii. 如发生错误，则简单描述错误并分析错误产生的原因

2. Configuration

2.1 配置 LAMP 环境

Lamp 环境是实际上就是一个 centos 系统上同时集成了 Apache/Mysql/PHP 环境，我们需要安装这些环境来为后面的 ecsshop 提供运行环境。首先我们可以选择逐一配置，但是太过麻烦了，因此我选择使用 docker 来进行环境的安装，首先我们前往 Dockerhub 寻找 lamp 镜像，随意看到如下的镜像符合我们的需求：



考虑到我找到的版本是较老版本，同时 Mysql 普遍使用 5.7 更加稳定，因此我选择这个版本，接下来我们输入如下命令创建这个版本的容器，由于我们需要将项目运行到容器中，而我们需要通过公网 Ip 访问到运行的 ecsshop，因此我们在创建 LAMP 容器时要为其暴露一个端口，同时为了方便后面我们管理，我们进行卷挂载将需要放置项目的文件夹与服务器的 /home/docker/lamp/app 路径挂载，因此代码如下即可创建好容器：

```
docker run --name=lamp01 -p 8088:80 -p 3401:3306 -v /home/docker/lamp/app:/app matttrayner/lamp:latest-1604-php5
```

云服务器 - root@VM-0-7-centos:/home/docker/lamp - Xshell 7 (Free for Home/School)

文件(F) 编辑(E) 查看(V) 工具(T) 选项卡(B) 窗口(W) 帮助(H)

ssh://root:*****@1.117.169.85:22

要添加当前会话，点击左侧的箭头按钮。

1 云服务器

```
[root@VM-0-7-centos ~]# pwd
/root
[root@VM-0-7-centos ~]# cd /home
[root@VM-0-7-centos home]# ls
christmas  deliverySystem  docker  hexo  react-chart-editor  sexamStudent  welcome
compile    demo             food    major  sexamAdmin          vclass        wenchong
[root@VM-0-7-centos home]# cd docker
[root@VM-0-7-centos docker]# ls
[root@VM-0-7-centos docker]# mkdir lamp/app
mkdir: cannot create directory 'lamp/app': No such file or directory
[root@VM-0-7-centos docker]# mkdir lamp
[root@VM-0-7-centos docker]# ls
lamp
[root@VM-0-7-centos docker]# cd lamp
[root@VM-0-7-centos lamp]# mkdir app
[root@VM-0-7-centos lamp]# docker run --name=lamp01 -p 8081:80 -p 3401:3306 -v /home/docker/lamp/app:/a
pp mattarrayner/lamp:latest
Unable to find image 'mattarrayner/lamp:latest' locally
latest: Pulling from mattarrayner/lamp
35807b77a593: Pull complete
ccfecfa17ed6: Pull complete
499764c8dc6b: Pull complete
c6b0ddc4cdc0: Pull complete
8ae8ee891eaf: Pull complete
510e2baf4b24: Pull complete
4103a3ecbaab: Pull complete
3066e34dc7b7: Pull complete
b626f562f102: Pull complete
fad4a5a21cfc: Pull complete
9c2a40162cc1: Pull complete
a8b002eab6c2: Pull complete
cda2d831f3ac: Pull complete
36cd807eb11e: Pull complete
7418cc89daa3: Pull complete
c5690feb59a2: Pull complete
b836b3984976: Pull complete
078cd7ee493e: Pull complete
1c368befafd5: Pull complete
7ed553304ebc: Pull complete
eca287b55180: Pull complete
cd928c8d6aad: Pull complete
```

ssh://root@1.117.169.85:22

SSH2 xterm 103x41 41,1 1 会话 CAP NUM

```
云服务器 - root@VM-0-7-centos:/home/docker/lamp - Xshell 7 (Free for Home/School)
文件(F) 编辑(E) 查看(V) 工具(T) 选项卡(B) 窗口(W) 帮助(H)
ssh://root:*****@1.117.169.85:22
要添加当前会话，点击左侧的箭头按钮。
1 云服务器 x +

^H^H^H

^Z^H^H^H^C2022-04-13 03:18:17,960 WARN received SIGINT indicating exit request
2022-04-13 03:18:17,961 INFO waiting for apache2, mysqld to die
2022-04-13 03:18:20,963 INFO waiting for apache2, mysqld to die
2022-04-13 03:18:23,966 INFO waiting for apache2, mysqld to die
2022-04-13 03:18:26,969 INFO waiting for apache2, mysqld to die
2022-04-13 03:18:27,970 WARN killing 'mysqld' (272) with SIGKILL
2022-04-13 03:18:28,973 INFO stopped: mysqld (terminated by SIGKILL)
2022-04-13 03:18:29,076 INFO stopped: apache2 (exit status 0)
/usr/local/lib/python3.8/dist-packages/supervisor-4.2.2-py3.8.egg/supervisor/options.py:474: UserWarning: Supervisor is running as root and it is searching for its configuration file in default locations (including its current working directory); you probably want to specify a "-c" argument specifying an absolute path to a configuration file for improved security.
  self.warnings.warn(
[root@VM-0-7-centos lamp]# docker ps
CONTAINER ID   IMAGE          COMMAND         CREATED        STATUS        PORTS          NAMES
[root@VM-0-7-centos lamp]# docker ps -a
CONTAINER ID   IMAGE          COMMAND         CREATED        STATUS        PORTS          NAMES
f2313fa7a697   matrayner/lamp:latest   "/run.sh"       7 minutes ago   Exited (0) 8 seconds ago   lamp01
[root@VM-0-7-centos lamp]# docker docker start lamp01
docker: 'docker' is not a docker command.
See 'docker --help'
[root@VM-0-7-centos lamp]# docker start lamp01
lamp01
[root@VM-0-7-centos lamp]# docker ps
CONTAINER ID   IMAGE          COMMAND         CREATED        STATUS        PORTS          NAMES
f2313fa7a697   matrayner/lamp:latest   "/run.sh"       7 minutes ago   Up 4 seconds   0.0.0.0:8088->80/tcp, :::8088->80/tcp, 0.0.0.0:3401->3306/tcp, :::3401->3306/tcp   lamp01
[root@VM-0-7-centos lamp]#
```

创建完成以后我们可以输入 `docker images` 和 `docker ps` 查看到对应的 LAMP 容器，可以看到他的状态为 `up` 正在运行：

```
云服务器 - root@VM-0-7-centos:/home/docker/lamp - Xshell 7 (Free for Home/School)
文件(F) 编辑(E) 查看(V) 工具(T) 选项卡(B) 窗口(W) 帮助(H)
ssh://root:*****@1.117.169.85:22
要添加当前会话，点击左侧的箭头按钮。
1 云服务器 x +
=> Creating MySQL admin user with random password
ERROR 1133 (42000) at line 1: Can't find any matching row in the user table
=> Done!
=====
You can now connect to this MySQL Server with qG6bfiae9r5S

mysql -uadmin -pqG6bfiae9r5S -h<host> -P<port>

Please remember to change the above password as soon as possible!
MySQL user 'root' has no password but only allows local connections

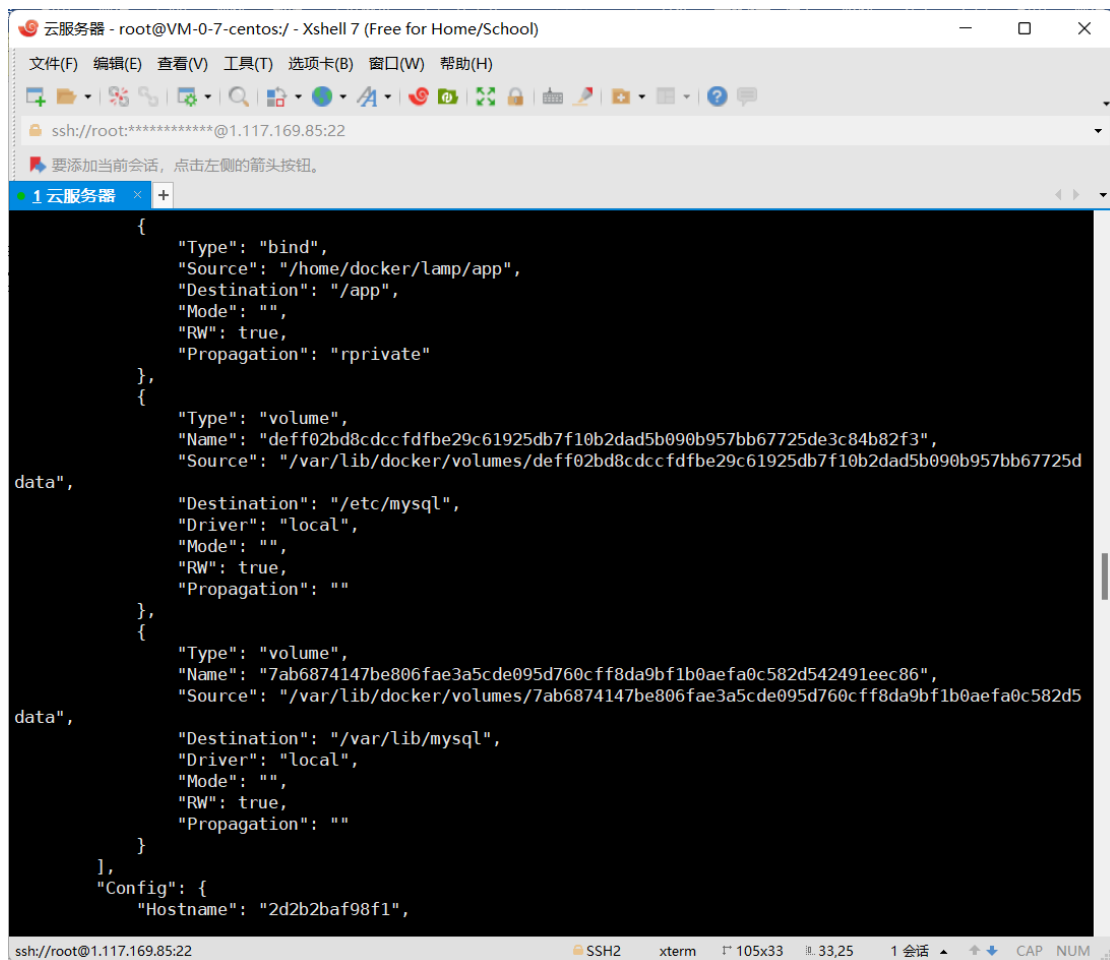
enjoy!
=====
/usr/lib/python2.7/dist-packages/supervisor/options.py:297: UserWarning: Supervisor is running as root and it is searching for its configuration file in default locations (including its current working directory); you probably want to specify a "-c" argument specifying an absolute path to a configuration file for improved security.
  'Supervisor is running as root and it is searching '
2022-04-13 05:51:21,947 CRIT Supervisor running as root (no user in config file)
2022-04-13 05:51:21,947 WARN Included extra file "/etc/supervisor/conf.d/supervisord-apache2.conf" during parsing
2022-04-13 05:51:21,947 WARN Included extra file "/etc/supervisor/conf.d/supervisord-mysqld.conf" during parsing
2022-04-13 05:51:21,963 INFO RPC interface 'supervisor' initialized
2022-04-13 05:51:21,963 CRIT Server 'unix_http_server' running without any HTTP authentication checking
2022-04-13 05:51:21,963 INFO supervisord started with pid 1
2022-04-13 05:51:22,965 INFO spawned: 'mysqld' with pid 489
2022-04-13 05:51:22,967 INFO spawned: 'apache2' with pid 490
2022-04-13 05:51:24,442 INFO success: mysqld entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
2022-04-13 05:51:24,443 INFO success: apache2 entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
2022-04-13 05:52:49,157 WARN received SIGINT indicating exit request
2022-04-13 05:52:49,158 INFO waiting for mysqld, apache2 to die
2022-04-13 05:52:49,260 INFO stopped: apache2 (exit status 0)
2022-04-13 05:52:52,264 INFO waiting for mysqld to die
2022-04-13 05:52:55,267 INFO waiting for mysqld to die
2022-04-13 05:52:58,270 INFO waiting for mysqld to die
2022-04-13 05:52:59,272 WARN killing 'mysqld' (489) with SIGKILL
2022-04-13 05:52:59,272 INFO stopped: mysqld (terminated by SIGKILL)
ssh://root@1.117.169.85:22 SSH2 xterm 117x37 37,28 1 会话 CAP NUM

Last login: Sun Apr 17 23:18:25 2022 from 117.136.1.15
[root@VM-0-7-centos ~]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
justb4/jmeter        latest              4a7b2894ddaf       5 weeks ago        242MB
matttrayner/lamp     latest             c15726116aef       7 months ago       1.06GB
matttrayner/lamp     latest-1604-php5   ac1d42d13c2a       3 years ago        841MB
[root@VM-0-7-centos ~]# docker ps
CONTAINER ID        IMAGE                                     COMMAND             CREATED             STATUS              PORTS
2d2b2baf98f1       matttrayner/lamp:latest-1604-php5      "/run.sh"          4 days ago         Up 4 days          0.0.0.0:8088->80/tcp, :::8088->80/tcp, 0.0.0.0:3401->3306/tcp, :::3401->3306/tcp
[root@VM-0-7-centos ~]#
```

然后我们输入指令

```
Docker inspect lamp01
```

可以查看卷挂载是否正确，如下是 lamp01 容器的详细信息，我们可以看到卷挂载正确：

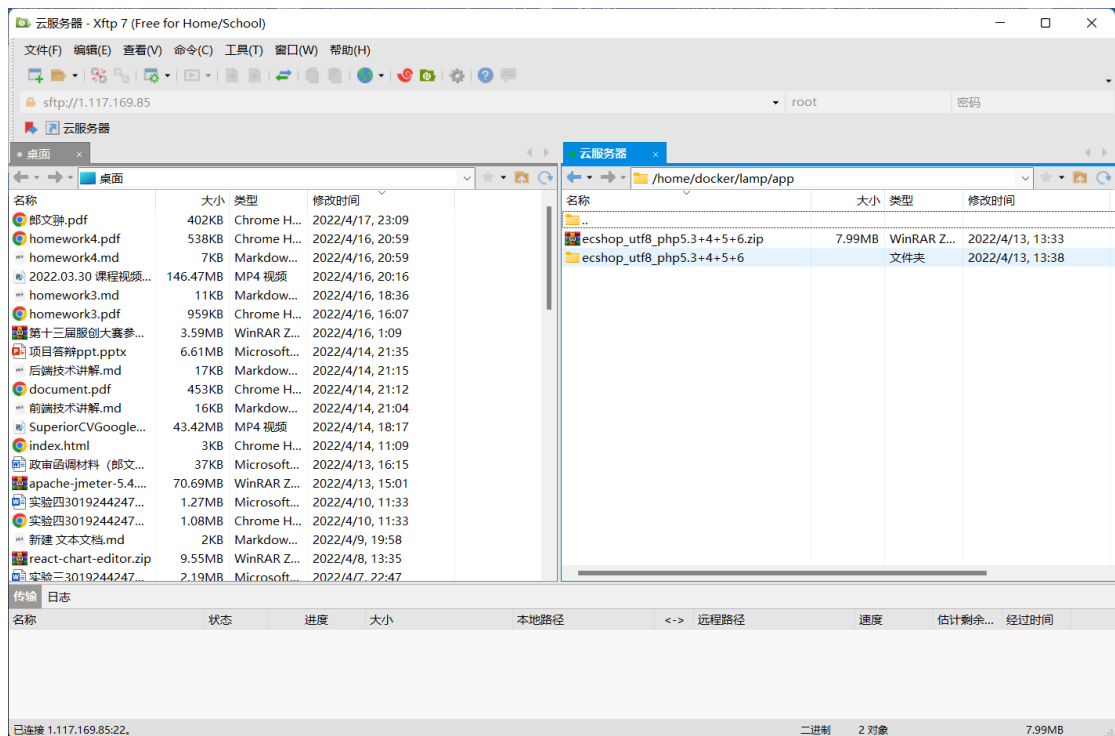


```
{
  "Type": "bind",
  "Source": "/home/docker/lamp/app",
  "Destination": "/app",
  "Mode": "",
  "RW": true,
  "Propagation": "rprivate"
},
{
  "Type": "volume",
  "Name": "deff02bd8cdccdfbe29c61925db7f10b2dad5b090b957bb67725de3c84b82f3",
  "Source": "/var/lib/docker/volumes/deff02bd8cdccdfbe29c61925db7f10b2dad5b090b957bb67725d",
  "Destination": "/etc/mysql",
  "Driver": "local",
  "Mode": "",
  "RW": true,
  "Propagation": ""
},
{
  "Type": "volume",
  "Name": "7ab6874147be806fae3a5cde095d760cfff8da9bf1b0aefa0c582d542491eec86",
  "Source": "/var/lib/docker/volumes/7ab6874147be806fae3a5cde095d760cfff8da9bf1b0aefa0c582d5",
  "Destination": "/var/lib/mysql",
  "Driver": "local",
  "Mode": "",
  "RW": true,
  "Propagation": ""
}
],
"Config": {
  "Hostname": "2d2b2baf98f1",
```

自此我们通过 docker 就轻松配置完成了 Lamp 环境。只不过 Mysql 用的是默认密码这也没有什么问题，通过输入 `docker log` 我们就可以看到 mysql 账户的密码。

2.2 安装 ecsshop 项目

接下来我们需要将 ecsshop 放置到 lamp01 容器中让其运行，然后我们就可以通过公网 ip+ 端口查看到这个项目了。之前我们卷挂载了 app 路径，实际上就是为了方便这里进行 ecsshop 的安装，我们将提前准备好的 ecsshop 压缩包通过 xftp 上传到服务器上的 `/home/docker/lamp/app` 下如下图所示：

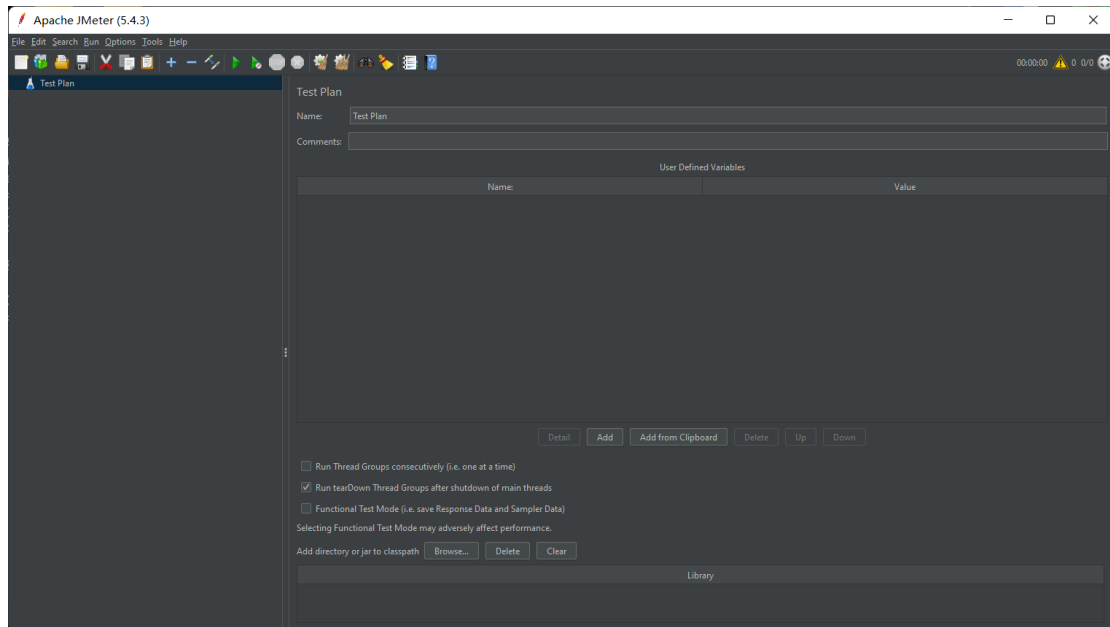


然后我们解压缩这个项目,由于我们已经将本地服务器上的 app 文件夹与 lamp01 容器的 app 文件夹进行了卷挂载,因此实际上此时相当于 lamp01 下也已经安装上了 ceshop,并且由于 docker 容器正在运行,因此我们可以通过之前暴露的端口号 8088 访问到这个项目了:
我的服务器的公网 ip 是 1.117.169.85, 因此访问项目的 url 就是 http://1.117.169.85:8088/ecshop_utf8_php5.3+4+5+6/
我们进入这个项目之前首先需要注册一下管理员账号,并且配置一下 mysql,接下来我们提交配置即可以进入进入到这个项目中如下图所示:



2.3 安装 jmeter 并启动运行

现在我们已经配置好了项目并且部署到了容器中，可以通过公网 url 访问到这个项目了，接下来我们就来安装一下 jmeter 来对我们部署的项目进行压力测试。我们下载下来 jmeter 安装以后不需要配置环境变量，直接解压缩然后进入到文件夹中可以看到在 `/bin/` 下有一个 `jmter.sh` 脚本文件可以直接启动 jmeter,因此我们在 jmeter 根路径下输入 `sh jmeter.sh` 启动 jmeter 如果可以正常启动，那么就可以看到如下图所示的 UI 界面了：

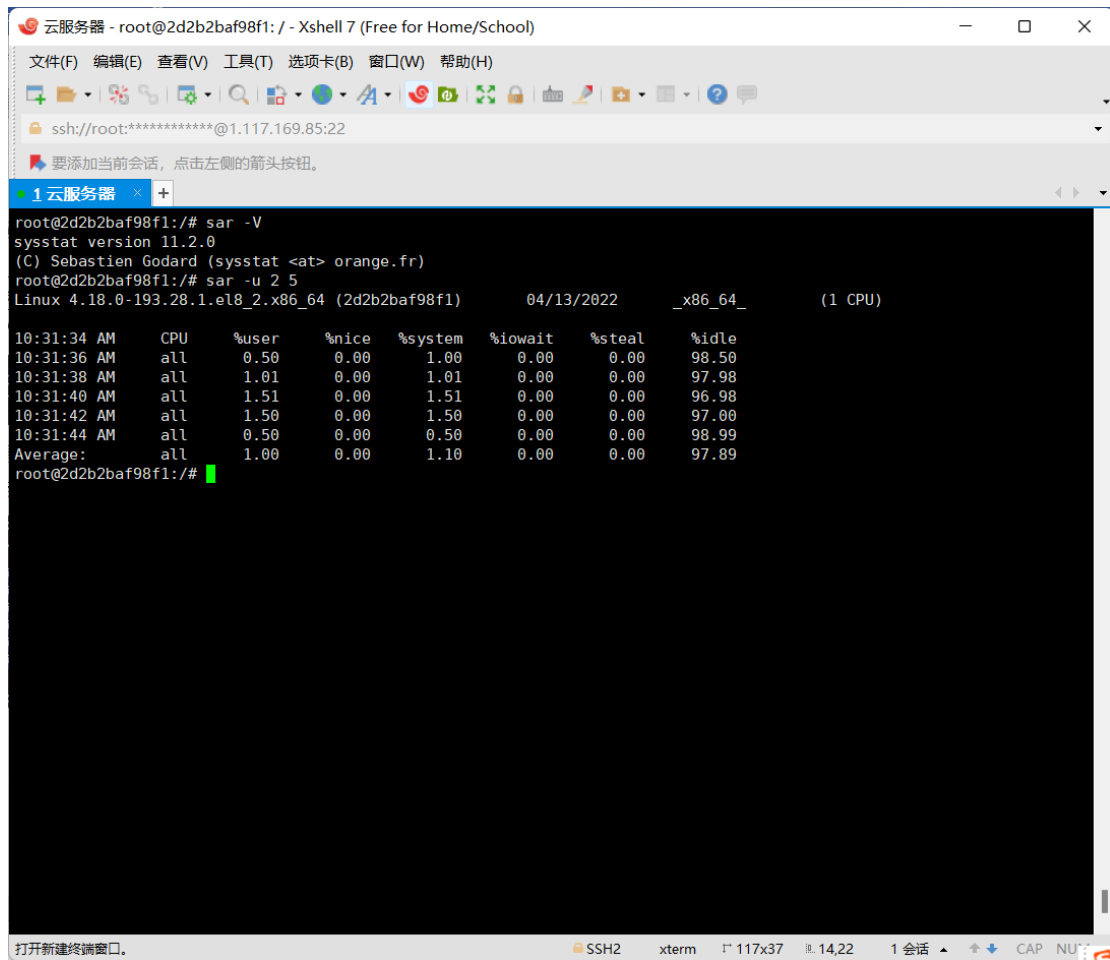


2.4 安装 systat

由于我们是在容器内部署的项目，因此我们在查看负载信息时应该是查看容器内部，所以首先我们输入如下执行进入容器：

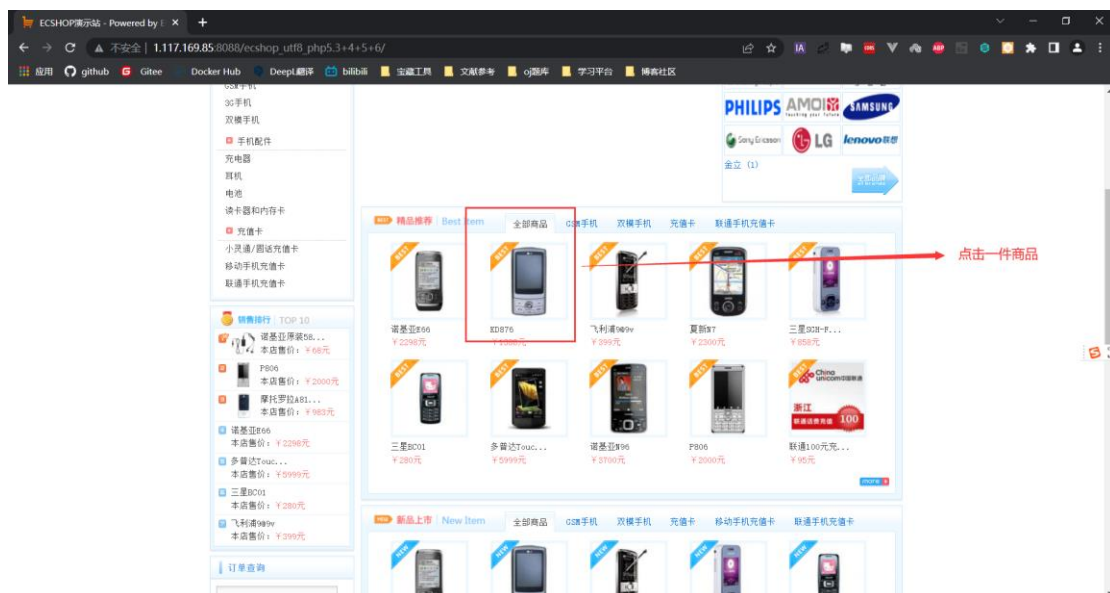
```
Docker exec -it lamp01 /bin/bash
```

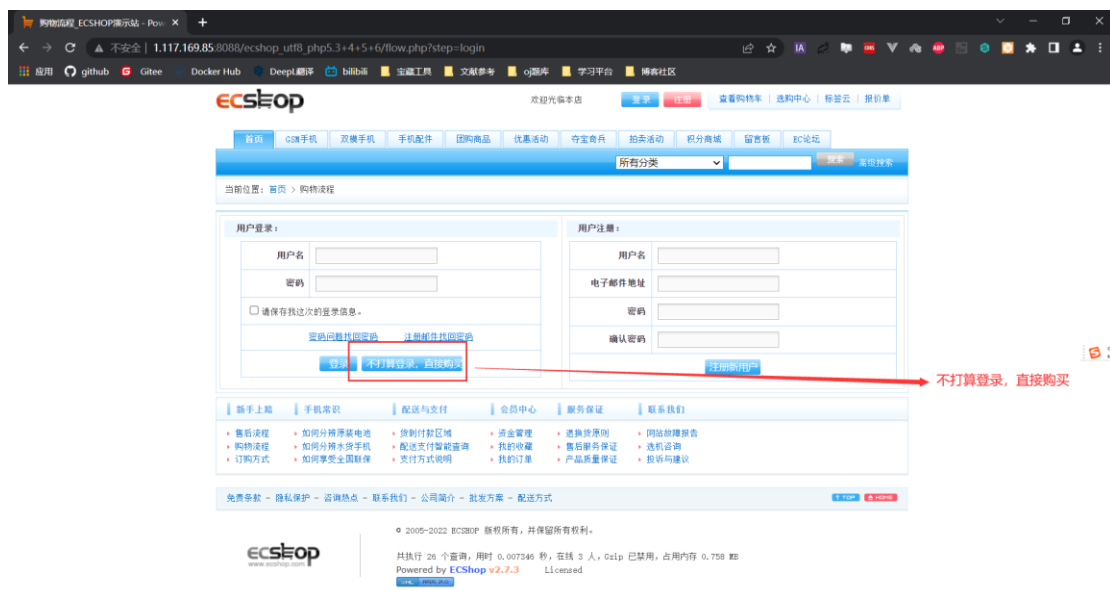
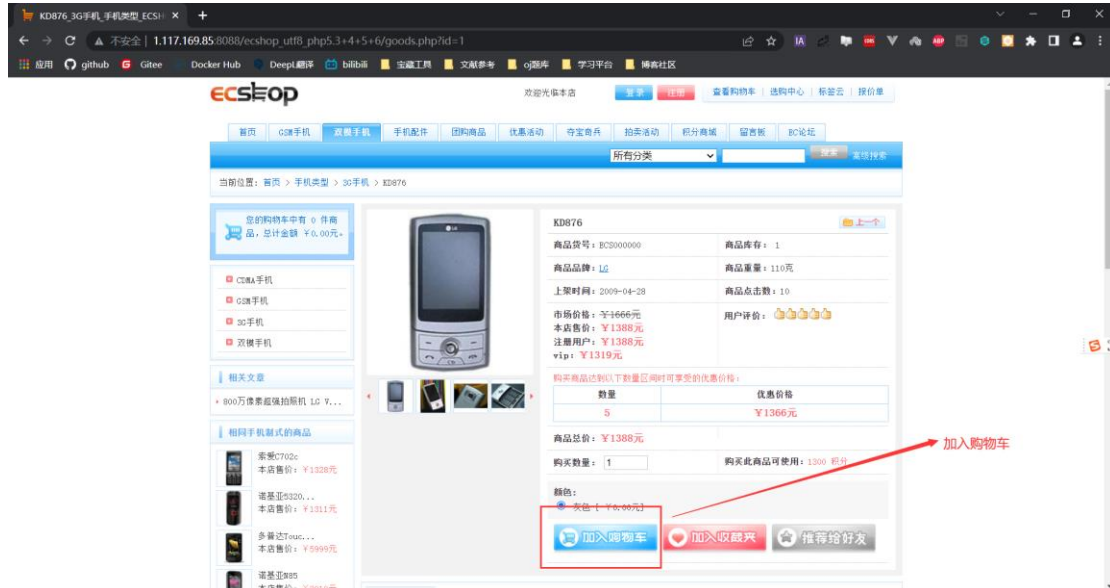
然后在这里输入 `top` 首先查看压力测试前的负载情况，具体图片以及分析请见第三部分，然后我们还需要在使用 `systat` 查看负载信息，因此首先需要在内部安装 `systat`,使用 `yum install systat` 安装即可，如果成功安装，我们输入 `sar -V` 应该可以查看到版本号如下图所示：

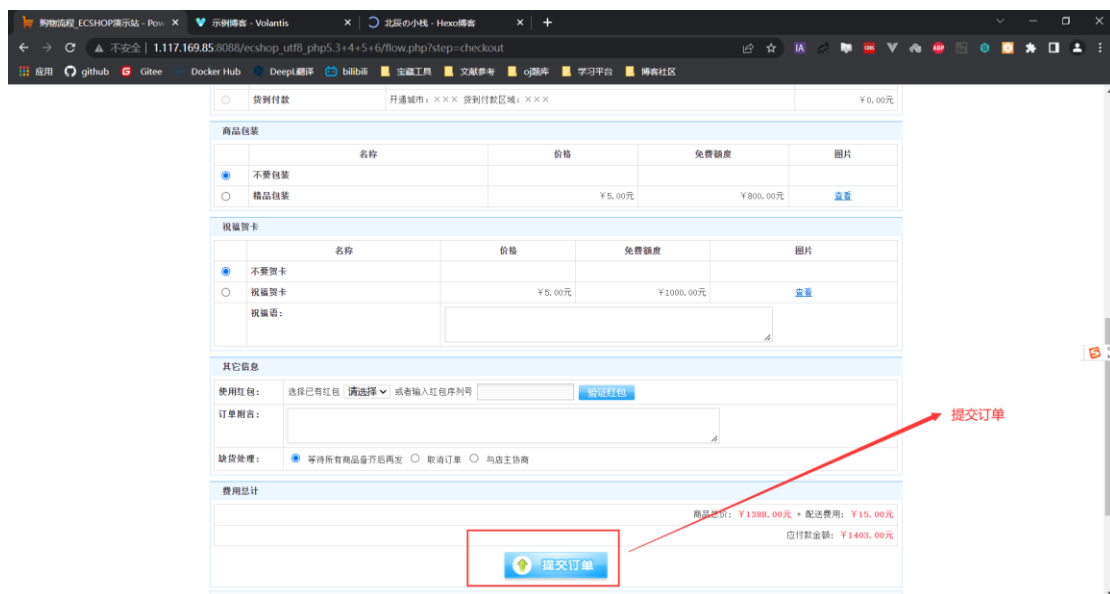
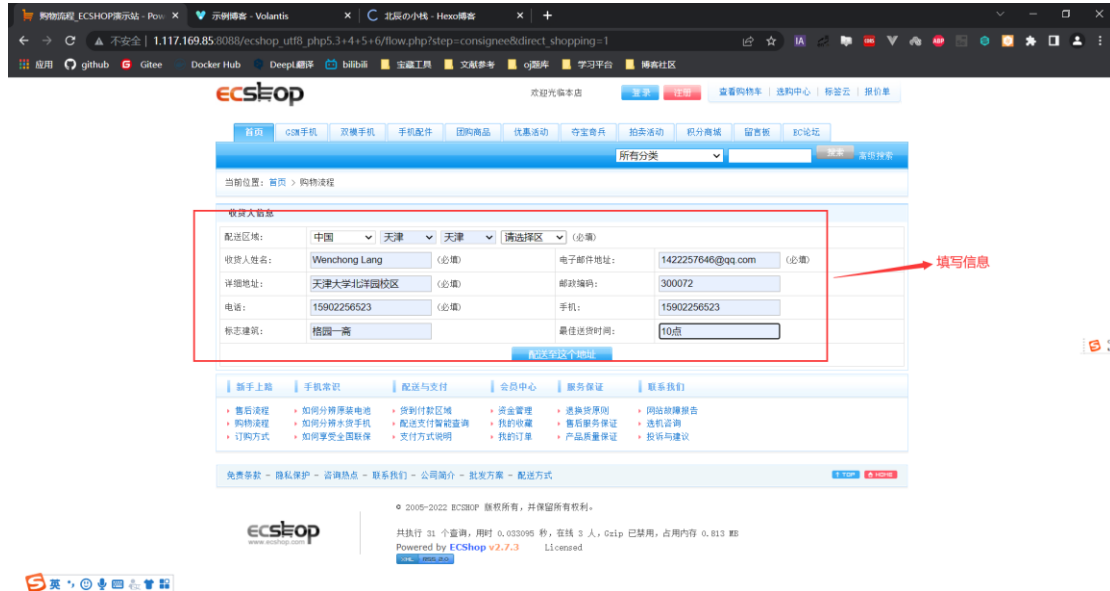


2.5 录制脚本

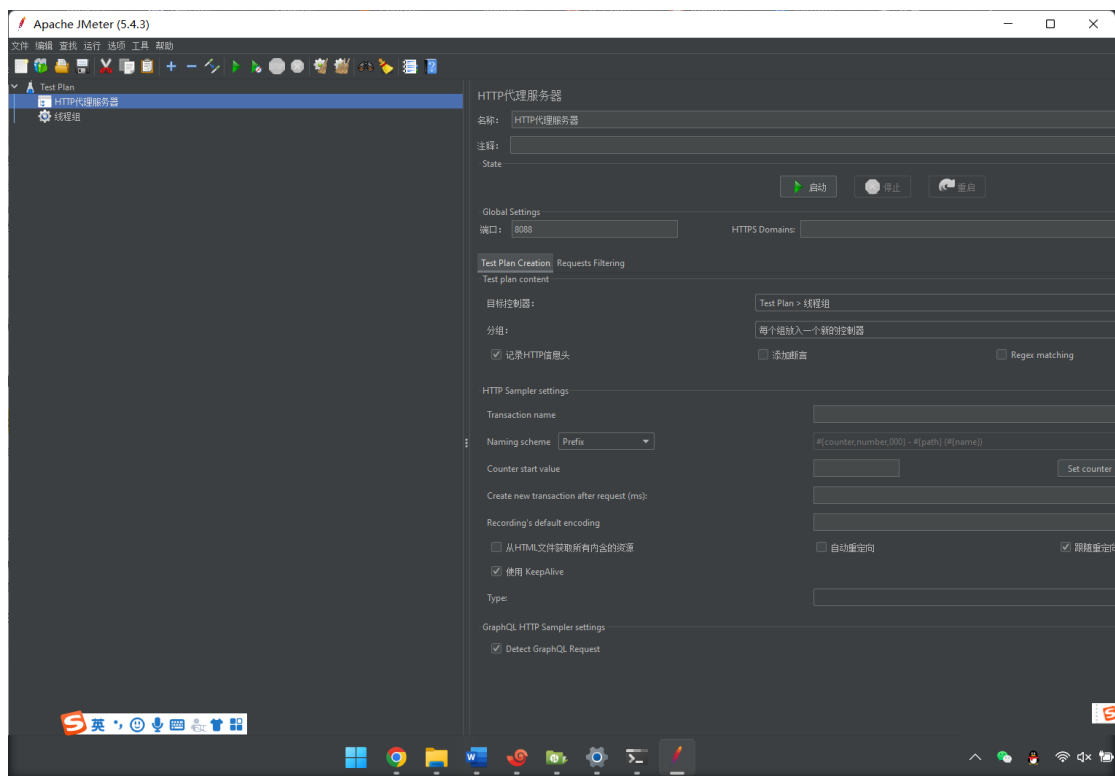
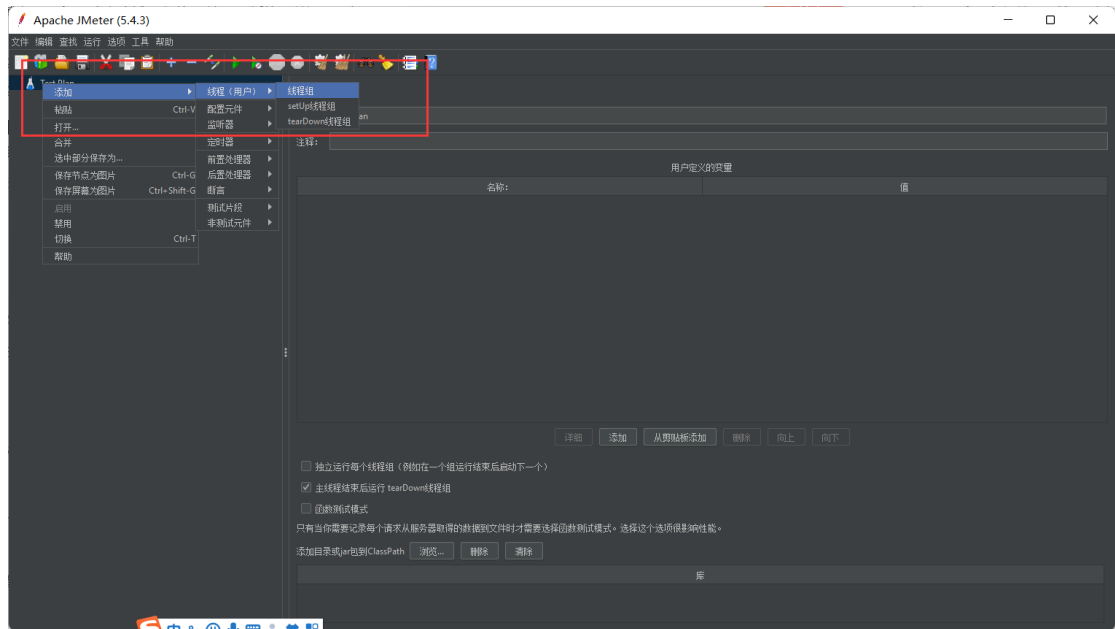
然后我们启动 jmeter 尝试对一个操作进行录制，这里我们发现可以点击商品加入购物车，然后结算，这个操作具体流程如下：







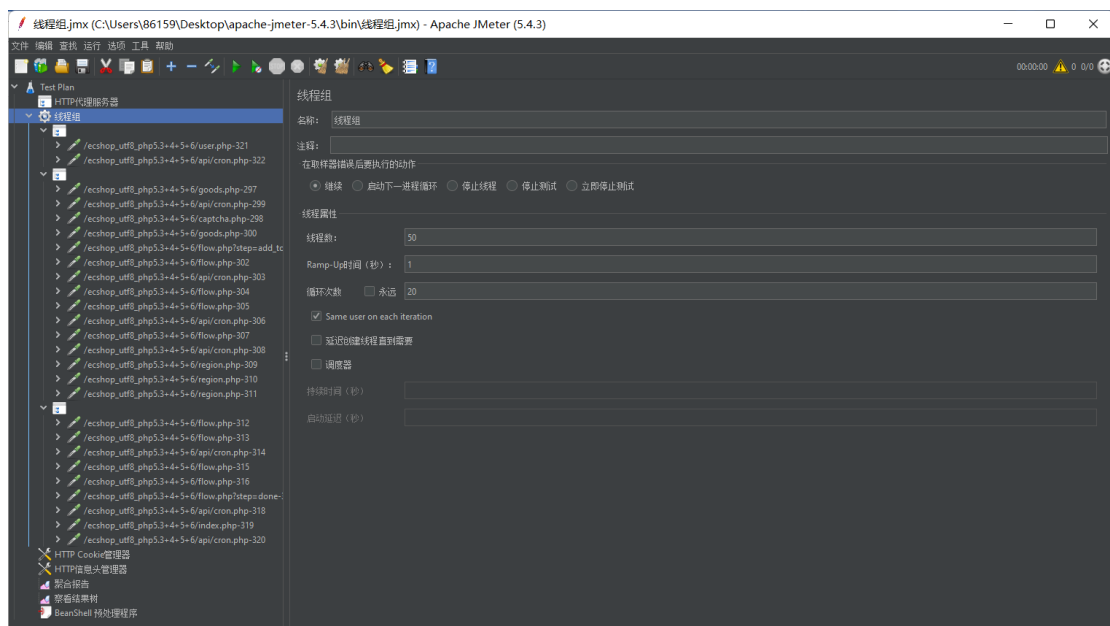
以上就是 ecshop 一个购买商品到计算订单的过程，接下来我们就尝试录制这个过程，首先我们需要在 jmeter 中，然后首先我们创建一个线程组，命名随意，他是用来存储记录我们的操作的，一会进行压力测试时也是不断地调用执行这个线程组，接下来我们还需要创建一个 HTTP 代理服务器用来录制我们的操作，然后以后在进行压力测试时他将会自动执行线程组中的操作来帮助我们完成多次重复的接口测试从而实现压力测试。



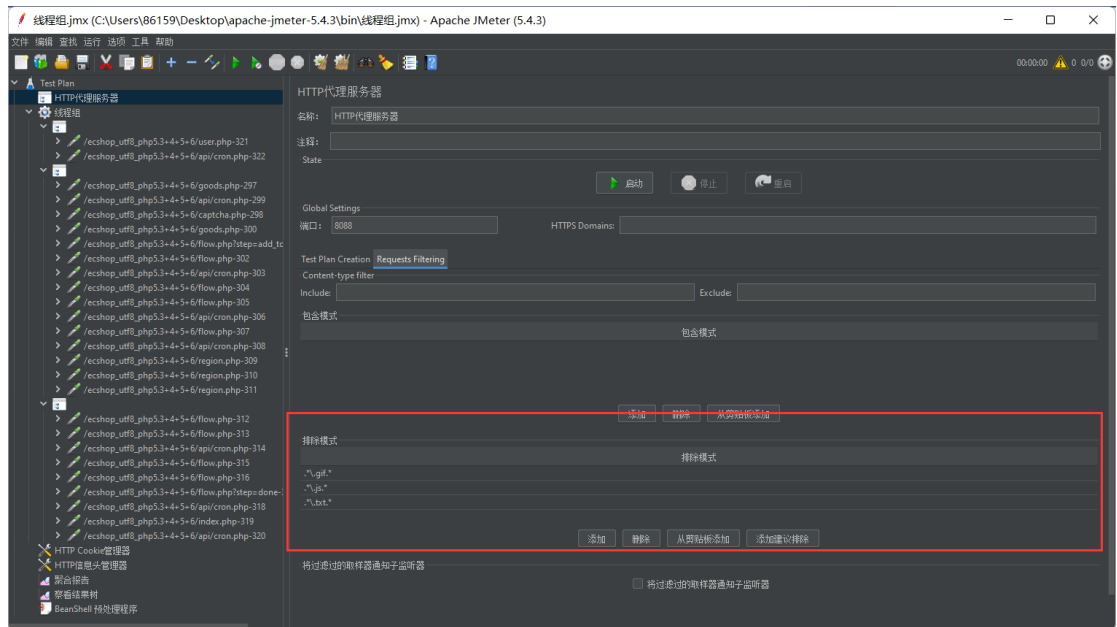
配置完上面的两个组件以后，我们只需要点击启动就可以开启录制操作，但是我们在之前还需要配置一下浏览器代理，我们打开 **chrome** 浏览器然后点击设置搜索代理即可打开如下界面：



我们需要将代理服务器开启，这样 http 代理服务器接下来才能执行，ip 地址就是回环地址 127.0.0.1，端口号就是 http 代理服务器上的端口即可。然后我们点击启动以后开始录制执行一遍刚刚上图所演示的购买商品流程，录制完成以后结束我们就可以看到线程组中出现了大量的记录：



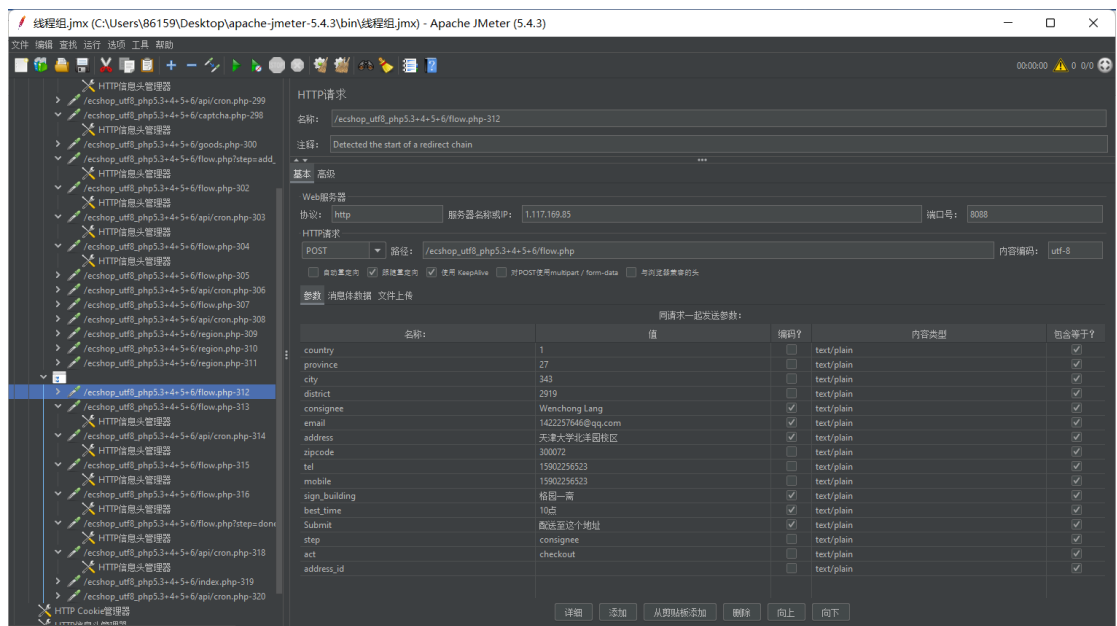
但是里面包含了太多无用的信息，因此我们可以借助 Http 代理服务器中的筛选设置筛掉无用的信息：



这样线程组中剩下的就是我们可能会用到的有用的操作，自此我们就完成了脚本记录。

2.6 修改记录的部分行为

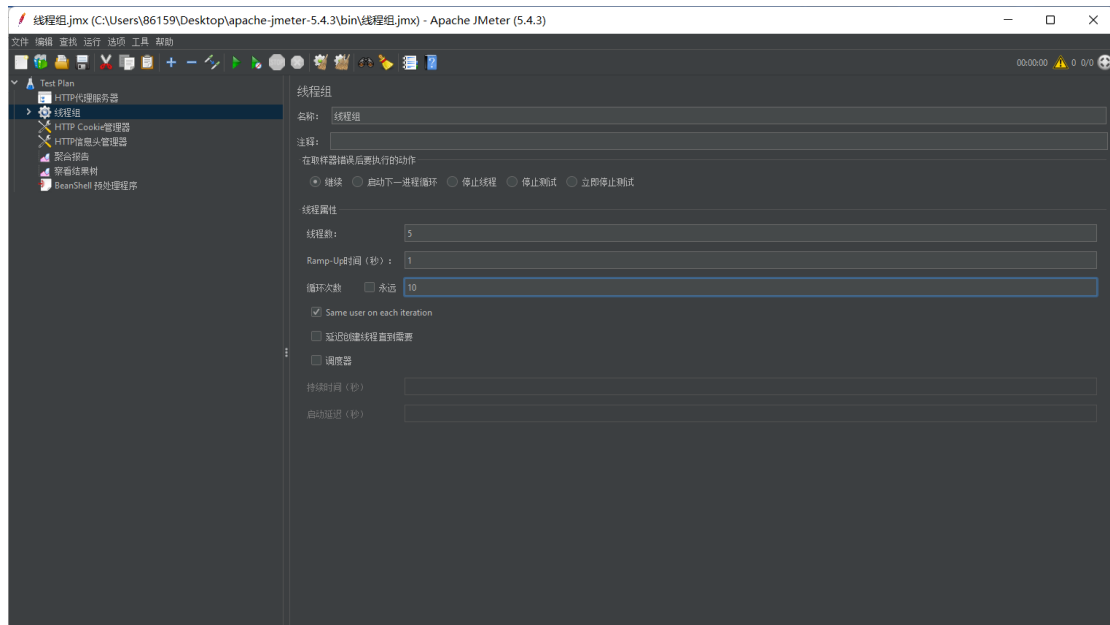
我们发现这里线程组中存储的每一条都是一个操作，有点类似于 **selenium**，如下所示就是刚刚记录的我们填写订单个人信息的操作：



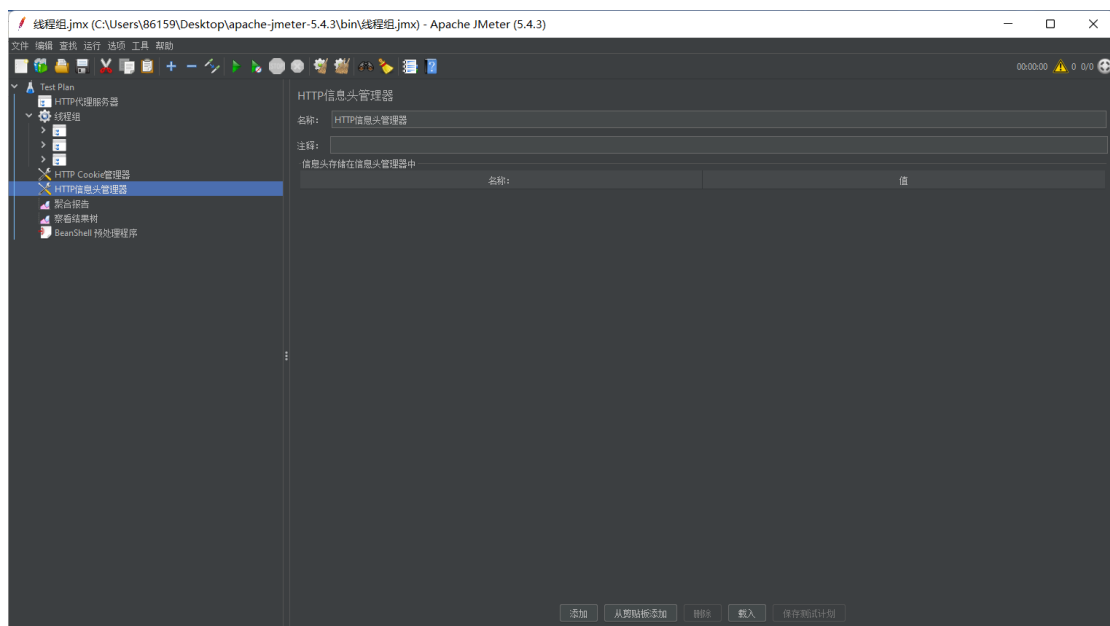
因此我们只要修改这里面操作的信息即可完成行为的修改

2.7 压力测试

接下来我们尝试进行压力测试，其实就是选择 **http** 代理服务器要执行的次数，然后他将代替我们自动执行刚刚的记录操作来完成压力测试。首先我们尝试 **5*10** 的压力测试，配置就是：



即每秒都执行 5 个线程，并且执行 10 次，然后点击上方的绿色三角既可以完成测试，为了方便我们一会查看系统的负载信息，我们可以现在运行前输入 **top** 以及 **sar -u 2 5** 查看一下压力测试前的负载情况。同时实验要求还要展示聚合报告，因此我们再创建一个聚合报告然后接下来我们尝试进行 **5*10** 的压力测试，首先我们还需要在创建 **http** 信息头管理器和 **cookie** 管理器，因此现在 **testplan** 应该如下图所示：



自此我们就完成了压力测试前的基本配置，开始进行压力测试，等待执行完成以后我们可以

查看聚合报告如下所示:

飞眼数据 jmx [C:\Users\86159\Desktop\lapanche-jmeter-5.4.3\bin\test\jmx] - Apache JMeter (5.4.3)

Test Plan
 根目录
 根目录
 JMeter Console管理
 配置系统参数
 JMeter性能参数管理
 配置数据

报告报告

名称: 测试报告

注释:

将数据输入一个文件

文件名

图表: 柱状图内容 | 饼图柱状图 | 饼图柱状图 | 饼图

| Label | 线程数 | 平均值 | 中位数 | 90% 百分位 | 95% 百分位 | 99% 百分位 | 最小值 | 最大值 | 误差 % | 吞吐量 | 接收 K/s | 发送 K/s |
|------------------------|------|-----|-----|---------|---------|---------|-----|-------|-------|----------|--------|--------|
| /ecshop_utf8.php5.3... | 40 | 295 | 195 | 629 | 682 | 1038 | 70 | 1038 | 0.00% | 1.4/sec | 14.19 | 0.9 |
| /ecshop_utf8.php5.3... | 40 | 43 | 37 | 52 | 59 | 277 | 30 | 277 | 0.00% | 1.4/sec | 0.38 | 0.8 |
| /ecshop_utf8.php5.3... | 40 | 107 | 67 | 310 | 384 | 429 | 39 | 429 | 0.00% | 1.4/sec | 5.71 | 0.8 |
| /ecshop_utf8.php5.3... | 40 | 42 | 39 | 75 | 91 | 84 | 31 | 84 | 0.00% | 1.4/sec | 0.38 | 0.7 |
| /ecshop_utf8.php5.3... | 40 | 70 | 66 | 102 | 112 | 138 | 44 | 138 | 0.00% | 1.4/sec | 0.68 | 1.0 |
| /ecshop_utf8.php5.3... | 40 | 103 | 62 | 206 | 381 | 563 | 43 | 563 | 0.00% | 1.4/sec | 6.80 | 0.9 |
| /ecshop_utf8.php5.3... | 40 | 39 | 35 | 49 | 59 | 69 | 30 | 69 | 0.00% | 1.4/sec | 0.28 | 0.6 |
| /ecshop_utf8.php5.3... | 40 | 186 | 124 | 395 | 518 | 623 | 79 | 623 | 0.00% | 1.4/sec | 7.50 | 1.9 |
| /ecshop_utf8.php5.3... | 40 | 124 | 62 | 353 | 419 | 421 | 39 | 421 | 0.00% | 1.4/sec | 7.15 | 0.9 |
| /ecshop_utf8.php5.3... | 40 | 37 | 33 | 46 | 48 | 83 | 30 | 83 | 0.00% | 1.4/sec | 0.28 | 0.6 |
| /ecshop_utf8.php5.3... | 40 | 149 | 90 | 367 | 418 | 425 | 50 | 425 | 0.00% | 1.4/sec | 7.36 | 0.8 |
| /ecshop_utf8.php5.3... | 40 | 49 | 43 | 137 | 48 | 286 | 30 | 286 | 0.00% | 1.4/sec | 0.38 | 0.8 |
| /ecshop_utf8.php5.3... | 40 | 42 | 42 | 53 | 57 | 81 | 31 | 81 | 0.00% | 1.4/sec | 0.48 | 0.8 |
| /ecshop_utf8.php5.3... | 40 | 37 | 36 | 46 | 52 | 57 | 31 | 57 | 0.00% | 1.4/sec | 0.35 | 0.8 |
| /ecshop_utf8.php5.3... | 40 | 39 | 36 | 50 | 54 | 80 | 30 | 80 | 0.00% | 1.4/sec | 0.74 | 0.8 |
| /ecshop_utf8.php5.3... | 40 | 182 | 118 | 367 | 388 | 502 | 87 | 502 | 0.00% | 1.4/sec | 9.52 | 2.8 |
| /ecshop_utf8.php5.3... | 40 | 141 | 83 | 416 | 470 | 763 | 44 | 763 | 0.00% | 1.5/sec | 9.26 | 1.0 |
| /ecshop_utf8.php5.3... | 40 | 38 | 38 | 48 | 50 | 55 | 30 | 55 | 0.00% | 1.5/sec | 0.28 | 0.8 |
| /ecshop_utf8.php5.3... | 40 | 63 | 42 | 141 | 169 | 563 | 44 | 563 | 0.00% | 1.5/sec | 0.84 | 0.8 |
| /ecshop_utf8.php5.3... | 40 | 87 | 66 | 125 | 136 | 322 | 44 | 322 | 0.00% | 1.5/sec | 0.95 | 0.8 |
| /ecshop_utf8.php5.3... | 40 | 298 | 236 | 390 | 583 | 1752 | 125 | 1752 | 0.00% | 1.5/sec | 6.33 | 0.8 |
| /ecshop_utf8.php5.3... | 40 | 103 | 36 | 64 | 103 | 1083 | 30 | 1083 | 0.00% | 1.5/sec | 0.29 | 0.8 |
| /ecshop_utf8.php5.3... | 40 | 512 | 117 | 336 | 724 | 16362 | 47 | 16362 | 0.00% | 1.5/sec | 9.49 | 0.8 |
| /ecshop_utf8.php5.3... | 40 | 36 | 35 | 47 | 52 | 55 | 1 | 55 | 2.08% | 1.5/sec | 0.35 | 0.8 |
| 总计 | 1188 | 119 | 53 | 284 | 368 | 673 | 1 | 16362 | 0.17% | 33.0/sec | 86.06 | 23.4 |

在群集中添加电话铃声 | 保存测试数据 | 保存测试数据

然后我们在调整线程组配置改为 50*20 再次进行压力测试聚合报告如下图所示:

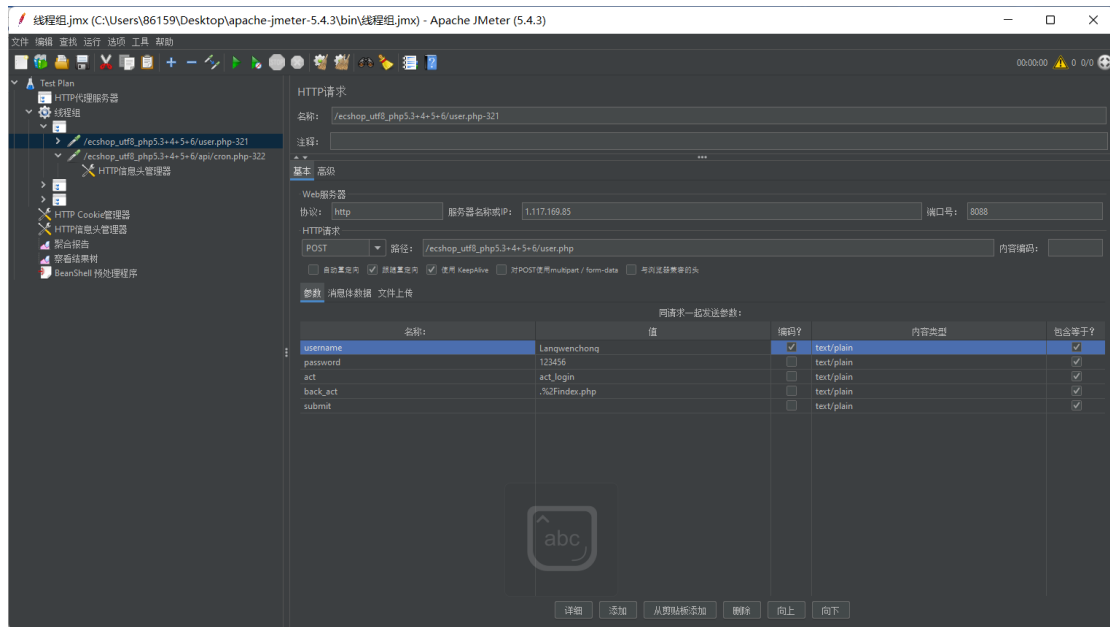
Summary Table Data:

| Label | # 样本 | 平均值 | 中位数 | 90% 百分位 | 95% 百分位 | 99% 百分位 | 最小值 | 最大值 | 异常 % | 吞吐量 | 错误 K/s | 失败 K/s | 发送 K/s |
|-----------------------|------|------|-----|---------|---------|---------|-----|-------|-------|----------|--------|--------|--------|
| recshop_utf8.php5.3-4 | 378 | 1843 | 634 | 2734 | 7431 | 23481 | 70 | 46290 | 0.00% | 1.2/sec | 12.14 | 0.75 | 0.75 |
| recshop_utf8.php5.3-4 | 378 | 52 | 38 | 79 | 126 | 200 | 0 | 360 | 0.00% | 1.2/sec | 0.21 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 372 | 346 | 152 | 407 | 887 | 2402 | 37 | 16139 | 0.00% | 1.2/sec | 4.90 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 372 | 53 | 41 | 72 | 88 | 300 | 0 | 575 | 0.27% | 1.2/sec | 0.33 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 372 | 94 | 80 | 144 | 171 | 354 | 43 | 646 | 0.00% | 1.2/sec | 0.57 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 369 | 433 | 207 | 869 | 1153 | 2786 | 41 | 9603 | 0.00% | 1.2/sec | 5.48 | 0.75 | 0.75 |
| recshop_utf8.php5.3-4 | 369 | 142 | 39 | 60 | 82 | 296 | 0 | 34138 | 0.94% | 1.2/sec | 0.25 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 367 | 660 | 395 | 984 | 1324 | 2307 | 0 | 63156 | 0.27% | 1.2/sec | 6.21 | 1.53 | 1.53 |
| recshop_utf8.php5.3-4 | 362 | 432 | 225 | 813 | 1074 | 1970 | 39 | 47709 | 0.00% | 1.2/sec | 5.82 | 0.75 | 0.75 |
| recshop_utf8.php5.3-4 | 366 | 91 | 58 | 74 | 104 | 302 | 0 | 15962 | 0.27% | 1.2/sec | 0.34 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 364 | 419 | 236 | 877 | 1000 | 1787 | 0 | 4081 | 0.27% | 1.2/sec | 6.05 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 364 | 48 | 38 | 72 | 100 | 233 | 30 | 296 | 0.00% | 1.2/sec | 0.23 | 0.75 | 0.75 |
| recshop_utf8.php5.3-4 | 364 | 51 | 39 | 59 | 84 | 237 | 30 | 1545 | 0.00% | 1.2/sec | 0.40 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 364 | 45 | 36 | 58 | 75 | 271 | 29 | 423 | 0.00% | 1.2/sec | 0.29 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 363 | 69 | 38 | 84 | 280 | 354 | 30 | 3578 | 0.00% | 1.2/sec | 0.61 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 362 | 402 | 194 | 1276 | 1574 | 3627 | 87 | 17732 | 0.00% | 1.2/sec | 7.60 | 0.75 | 0.75 |
| recshop_utf8.php5.3-4 | 361 | 620 | 447 | 1038 | 1337 | 2641 | 0 | 15811 | 1.11% | 1.2/sec | 7.52 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 361 | 76 | 39 | 73 | 97 | 575 | 0 | 8010 | 0.00% | 1.2/sec | 0.24 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 361 | 126 | 81 | 208 | 291 | 670 | 0 | 1582 | 0.28% | 1.2/sec | 0.75 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 361 | 171 | 88 | 172 | 237 | 400 | 43 | 2362 | 0.00% | 1.2/sec | 0.75 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 352 | 839 | 499 | 1235 | 1477 | 4794 | 62 | 61888 | 0.00% | 1.1/sec | 4.62 | 1.00 | 1.00 |
| recshop_utf8.php5.3-4 | 351 | 205 | 39 | 76 | 119 | 1023 | 0 | 33238 | 0.57% | 1.1/sec | 0.23 | 0.00 | 0.00 |
| recshop_utf8.php5.3-4 | 349 | 801 | 461 | 1046 | 1196 | 3962 | 0 | 62627 | 0.36% | 1.1/sec | 7.12 | 0.75 | 0.75 |
| recshop_utf8.php5.3-4 | 348 | 57 | 46 | 83 | 132 | 527 | 0 | 654 | 0.00% | 1.1/sec | 0.23 | 0.00 | 0.00 |
| 总结 | 8723 | 335 | 74 | 712 | 1023 | 2348 | 0 | 66290 | 0.38% | 27.6/sec | 72.96 | 16.75 | 16.75 |

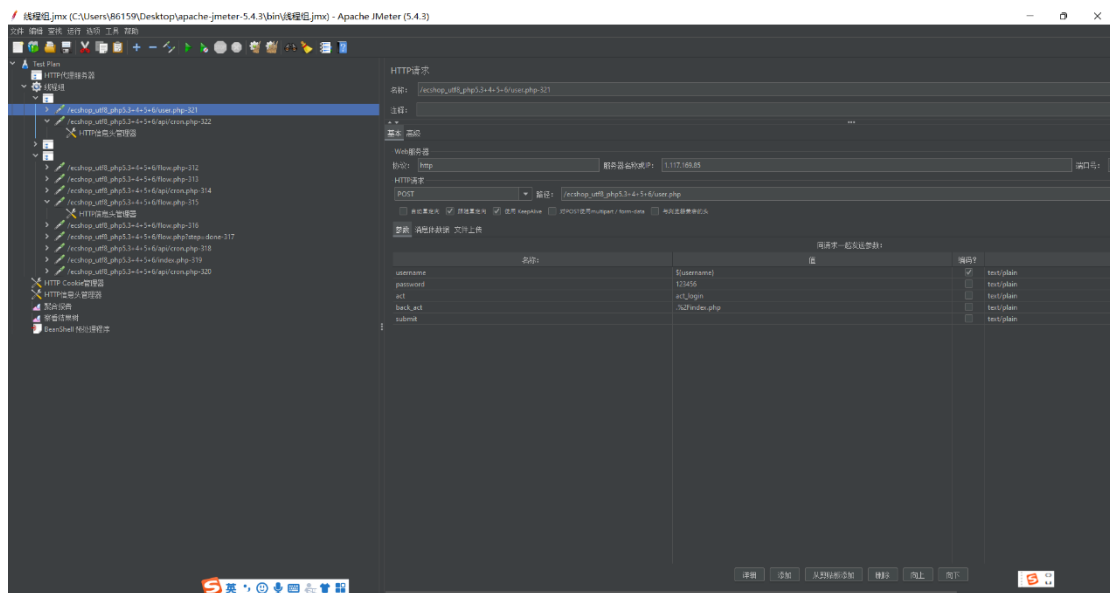
自此我们就完成了压力测试，可以看到当 50^29 时吞吐量变大，同时异常率提高这可能是因为服务器不能及时进行多个请求的相应导致的，也有可能是后台出现了问题。然后我们在执行一下 top 以及 sar -u 2 5 记录一下此时的负载情况，具体的分析见第三部分。

2.8 Beanshell 编写代码自定义测试

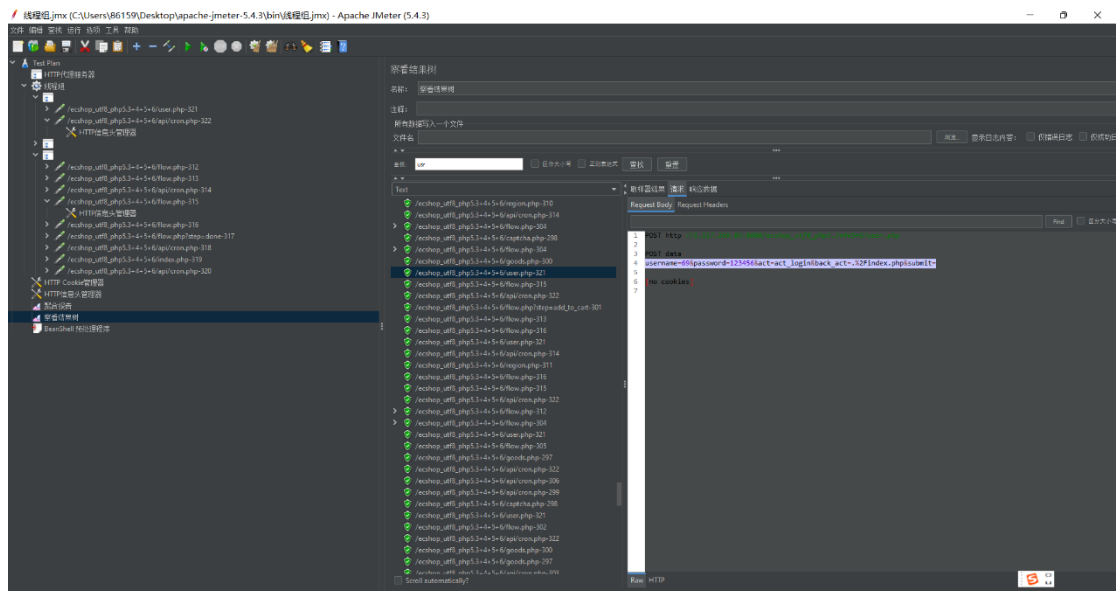
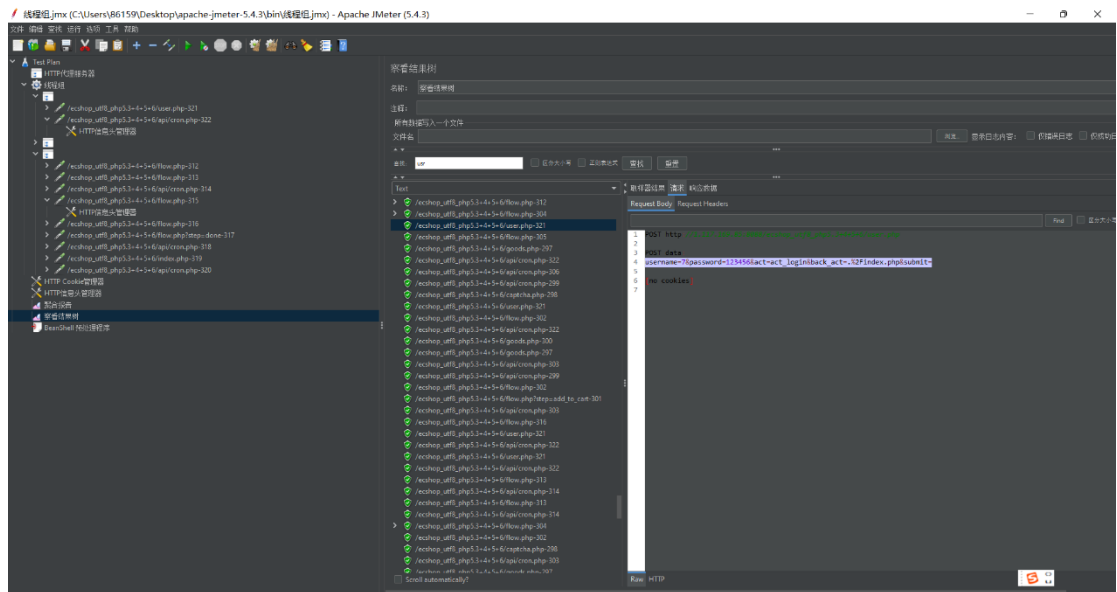
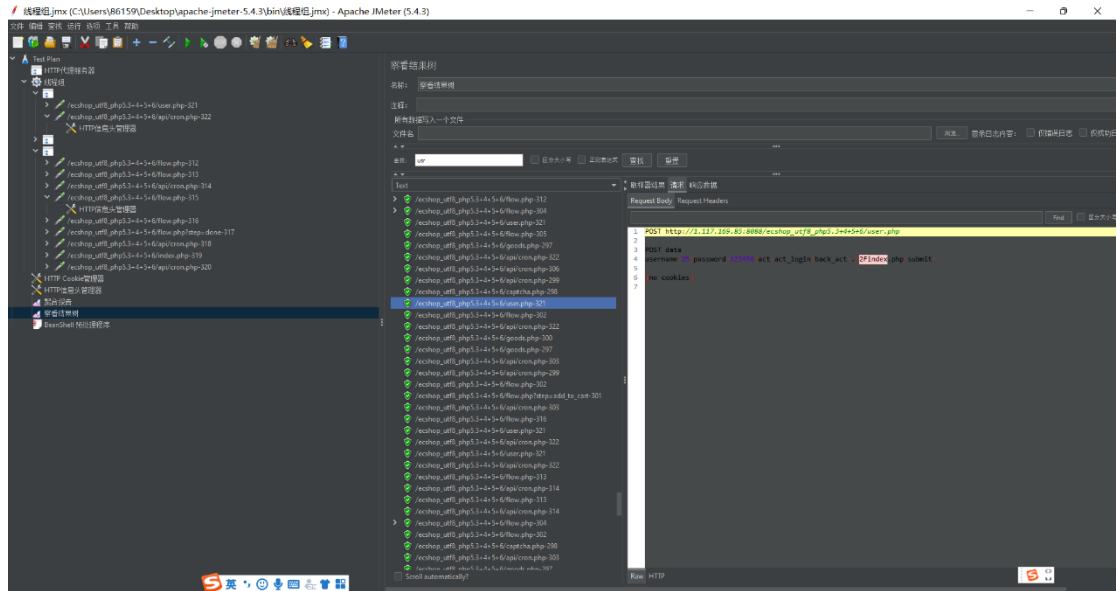
所谓 beanshell 待处理程序，其实是可以让我们自定义对某个操作进行测试前进行修改，比如登录接口的测试，如果我们按照上面的方法进行会发现一直都是填入的同一个数据来进行测试，这明显有时候不能满足我们的测试需求，比如现在我们可以让 jmeter 在进行测试前首先自动生成一下随机数据填充到将要测试的内容之中然后再执行压力测试，此时我们就会用到 beanshell 来对测试进行预处理。首先我们需要创建一个 beanshell 预处理程序，其实里面就可以直接编写我们要进行的 java 代码即可，甚至无需配置包路径等内容，非常简单上手，由于我这里是 对登录进行测试，因此我先又录制了一个进行登录的操作可以查看到这里我填写的账号一直是 Langwenchong



现在我们在 beanshell 中编写一个 java 程序调用 random 来随机生成一个随机数作为账号名填充到操作中，具体的代码见第四部分。我们将要填充的随机数账号存储到了变量名 username 中，然后修改这个接口的变量名为\${username}



然后为了方便我们一会查看压力测试时填充的账号是否为我们生成的随机数，我们在创建一个察看结果树。创建完成以后我们再次启动压力测试，执行完成以后我们可以从查看结果中查看到每一个测试操作中填充的账号都变成了我们生成的随机数，这就是 beanshell 预处理程序的功能：



3. Result analysis

完成了压力测试以后我们再来分析一下压力测试前后的负载情况，首先给出压力测试前后的top 指令分析结果：

压力测试前：

云服务器 - root@f2313fa7a697: / - Xshell 7 (Free for Home/School)

文件(F) 编辑(E) 查看(V) 工具(T) 选项卡(B) 窗口(W) 帮助(H)

ssh://root:*****@1.117.169.85:22

要添加当前会话，点击左侧的箭头按钮。

1 云服务器

top - 03:23:42 up 313 days, 21:50, 0 users, load average: 0.00, 0.04, 0.06
Tasks: 13 total, 1 running, 12 sleeping, 0 stopped, 0 zombie
%Cpu(s): 1.0 us, 0.7 sy, 0.0 ni, 98.0 id, 0.0 wa, 0.3 hi, 0.0 si, 0.0 st
MiB Mem : 1827.0 total, 70.8 free, 1104.6 used, 651.6 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 549.0 avail Mem

| PID | USER | PR | NI | VIRT | RES | SHR | S | %CPU | %MEM | TIME+ | COMMAND |
|-----|----------|----|----|---------|--------|-------|---|------|------|---------|-------------|
| 1 | root | 20 | 0 | 34084 | 23984 | 9316 | S | 0.0 | 1.3 | 0:00.39 | supervisord |
| 29 | root | 20 | 0 | 257220 | 34712 | 26240 | S | 0.0 | 1.9 | 0:00.05 | apache2 |
| 30 | root | 20 | 0 | 25572 | 17100 | 6908 | S | 0.0 | 0.9 | 0:00.09 | pidproxy |
| 31 | www-data | 20 | 0 | 257268 | 10596 | 2116 | S | 0.0 | 0.6 | 0:00.00 | apache2 |
| 32 | www-data | 20 | 0 | 257416 | 12144 | 3596 | S | 0.0 | 0.6 | 0:00.00 | apache2 |
| 33 | www-data | 20 | 0 | 257252 | 10596 | 2116 | S | 0.0 | 0.6 | 0:00.00 | apache2 |
| 34 | www-data | 20 | 0 | 257252 | 10596 | 2116 | S | 0.0 | 0.6 | 0:00.00 | apache2 |
| 35 | www-data | 20 | 0 | 257252 | 10596 | 2116 | S | 0.0 | 0.6 | 0:00.00 | apache2 |
| 36 | root | 20 | 0 | 2612 | 1708 | 1560 | S | 0.0 | 0.1 | 0:00.00 | mysqld_safe |
| 182 | www-data | 20 | 0 | 1299080 | 404292 | 35260 | S | 0.0 | 21.6 | 0:01.31 | mysqld |
| 226 | www-data | 20 | 0 | 257252 | 10596 | 2116 | S | 0.0 | 0.6 | 0:00.00 | apache2 |
| 227 | root | 20 | 0 | 10148 | 3936 | 3416 | S | 0.0 | 0.2 | 0:00.21 | bash |
| 239 | root | 20 | 0 | 12044 | 3840 | 3316 | R | 0.0 | 0.2 | 0:00.00 | top |

ssh://root@1.117.169.85:22 SSH2 xterm 103x41 21,1 1 会话 CAP NUM

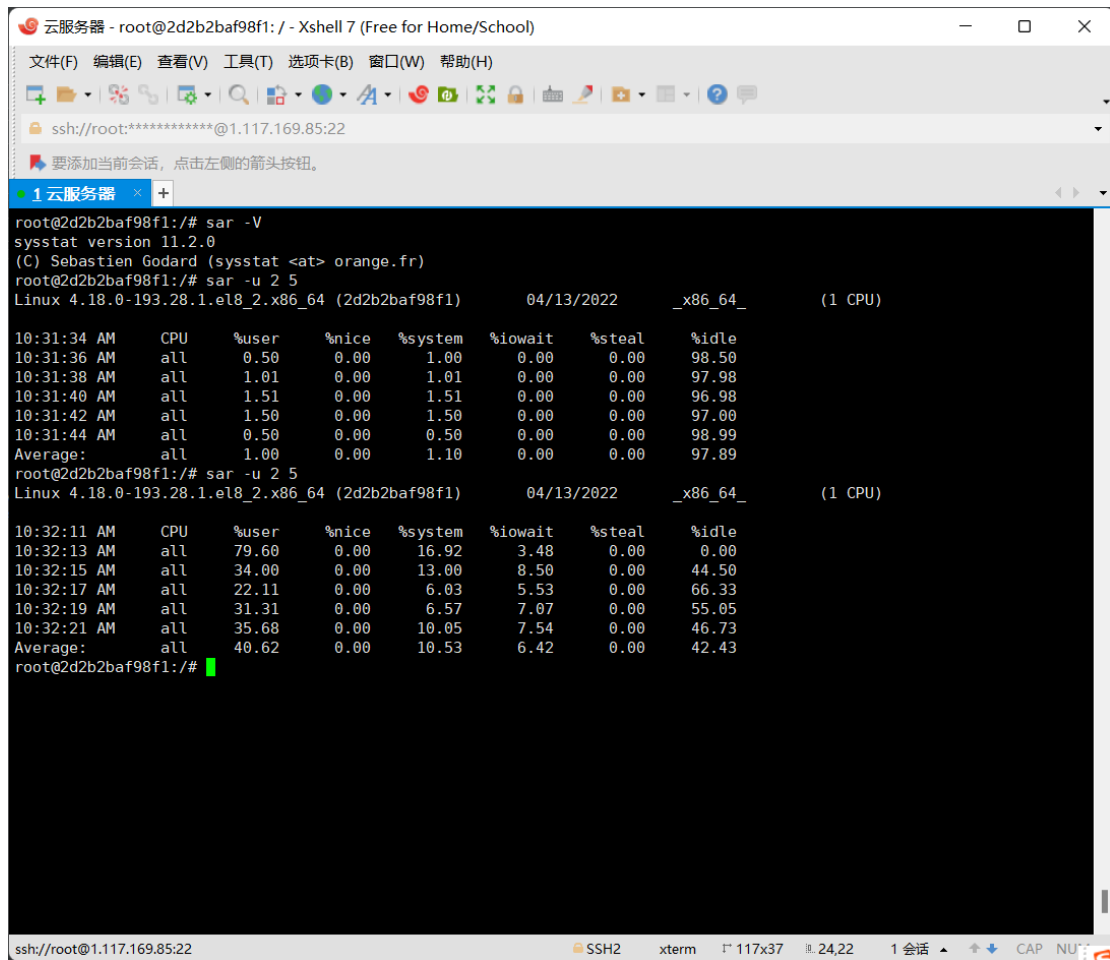
压力测试后：

```
云服务器 - root@2d2b2baf98f1: / - Xshell 7 (Free for Home/School)
文件(F) 编辑(E) 查看(V) 工具(T) 选项卡(B) 窗口(W) 帮助(H)
ssh://root:*****@1.117.169.85:22
要添加当前会话，点击左侧的箭头按钮。
1 云服务器
top - 09:50:02 up 314 days, 4:16, 0 users, load average: 1.44, 0.64, 0.47
Tasks: 76 total, 1 running, 74 sleeping, 0 stopped, 1 zombie
%Cpu(s): 26.7 us, 7.8 sy, 0.0 ni, 56.4 id, 7.4 wa, 0.7 hi, 1.0 si, 0.0 st
KiB Mem : 1870828 total, 76576 free, 1140768 used, 653484 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 466916 avail Mem

  PID USER      PR  NI   VIRT    RES    SHR  S  %CPU  %MEM     TIME+ COMMAND
 428 mysql      20   0 1192472 250368 19524  S   9.3  13.4   0:35.74 mysqld
 734 www-data   20   0 455648  25280 15764  S   1.3   1.4   0:00.06 apache2
 742 www-data   20   0 454600  26308 17032  S   1.3   1.4   0:00.07 apache2
 500 www-data   20   0 455888  31320 21312  S   1.0   1.7   0:03.31 apache2
 573 www-data   20   0 456140  32304 21828  S   1.0   1.7   0:03.74 apache2
 720 www-data   20   0 455644  26256 16628  S   1.0   1.4   0:00.13 apache2
 721 www-data   20   0 454876  26600 17140  S   1.0   1.4   0:00.14 apache2
 732 www-data   20   0 455396  26260 17008  S   1.0   1.4   0:00.08 apache2
 746 www-data   20   0 454216  24576 15764  S   1.0   1.3   0:00.05 apache2
 786 www-data   20   0 454496  24692 15700  S   1.0   1.3   0:00.04 apache2
 730 www-data   20   0 454496  24980 15828  S   0.7   1.3   0:00.04 apache2
 743 www-data   20   0 454248  25536 16468  S   0.7   1.4   0:00.06 apache2
 745 www-data   20   0 454216  24608 15764  S   0.7   1.3   0:00.04 apache2
 747 www-data   20   0 453316  24632 16468  S   0.7   1.3   0:00.03 apache2
 750 www-data   20   0 454112  24584 15956  S   0.7   1.3   0:00.03 apache2
 762 www-data   20   0 453868  22296 13972  S   0.7   1.2   0:00.03 apache2
 781 www-data   20   0 453316  24312 16148  S   0.7   1.3   0:00.03 apache2
 782 www-data   20   0 454112  24584 15956  S   0.7   1.3   0:00.03 apache2
 784 www-data   20   0 453316  24312 16148  S   0.7   1.3   0:00.03 apache2
 787 www-data   20   0 453316  24308 16148  S   0.7   1.3   0:00.03 apache2
 726 www-data   20   0 454620  25348 16084  S   0.3   1.4   0:00.06 apache2
 744 www-data   20   0 454252  22444 13520  S   0.3   1.2   0:00.06 apache2
 748 www-data   20   0 453868  22296 13972  S   0.3   1.2   0:00.03 apache2
 780 www-data   20   0 454220  22560 13904  S   0.3   1.2   0:00.03 apache2
 785 www-data   20   0 453584  21380 13072  S   0.3   1.1   0:00.02 apache2
   1 root        20   0  51172  18608  7024  S   0.0   1.0   0:02.30 supervisord
  25 root        20   0  4504    1744   1592  S   0.0   0.1   0:00.01 mysqld_safe
  26 root        20   0 452532  33236 26004  S   0.0   1.8   0:00.55 apache2
 460 www-data   20   0 453264  33356 25124  S   0.0   1.8   0:07.91 apache2
 483 www-data   20   0 456172  32624 22320  S   0.0   1.7   0:04.38 apache2

ssh://root@1.117.169.85:22  SSH2  xterm  117x37  37,117  1 会话  CAP  NUM
```

我们可以明显看到服务器的运行程序增多，同时 cpu 利用率增大，说明压力测试以后服务器负载情况增大。同时我们也可以通过 sysstat 的前后对比图进一步看出：



```
root@2d2b2baf98f1:/# sar -V
sysstat version 11.2.0
(C) Sebastien Godard (sysstat <at> orange.fr)
root@2d2b2baf98f1:/# sar -u 2 5
Linux 4.18.0-193.28.1.el8_2.x86_64 (2d2b2baf98f1)      04/13/2022      _x86_64_      (1 CPU)

10:31:34 AM    CPU     %user   %nice   %system %iowait  %steal   %idle
10:31:36 AM    all       0.50      0.00      1.00      0.00      0.00    98.50
10:31:38 AM    all       1.01      0.00      1.01      0.00      0.00    97.98
10:31:40 AM    all       1.51      0.00      1.51      0.00      0.00    96.98
10:31:42 AM    all       1.50      0.00      1.50      0.00      0.00    97.00
10:31:44 AM    all       0.50      0.00      0.50      0.00      0.00    98.99
Average:      all       1.00      0.00      1.10      0.00      0.00    97.89
root@2d2b2baf98f1:/# sar -u 2 5
Linux 4.18.0-193.28.1.el8_2.x86_64 (2d2b2baf98f1)      04/13/2022      _x86_64_      (1 CPU)

10:32:11 AM    CPU     %user   %nice   %system %iowait  %steal   %idle
10:32:13 AM    all      79.60      0.00     16.92      3.48      0.00     0.00
10:32:15 AM    all     34.00      0.00     13.00      8.50      0.00    44.50
10:32:17 AM    all     22.11      0.00      6.03      5.53      0.00    66.33
10:32:19 AM    all     31.31      0.00      6.57      7.07      0.00    55.05
10:32:21 AM    all     35.68      0.00     10.05      7.54      0.00    46.73
Average:      all     40.62      0.00     10.53      6.42      0.00    42.43
root@2d2b2baf98f1:/#
```

可以看到 iowai, system 等指令都明显增大, 因此 jmeter 可以有效的对部署项目的服务器进行压力测试。

4. Source code

```
import java.util.Random;

//随机设置用户名
random = new Random(System.currentTimeMillis());
int r1 = random.nextInt(100);
String username = Integer.toString(r1);

//将 username 保存到 jmeter 中
vars.put("username", username);

//查看用户名
String getUsername = vars.get("username");
System.out.println("User's name is " + getUsername);
```

5. 实验心得

本次实验花费的时间很长，主要是环境配置，项目部署以及熟悉 `jmeter` 的工作方法都需要很多时间与精力的投入。对于 `LAMP` 环境的配置我一开始选择的是自己配置，但是由于版本每个组件的版本兼容问题，最终我放弃了选择更轻松的 `docker` 容器的方法进行环境搭建，然后 `ecsshop` 也是第一次部署的 `php` 程序中途出现了一些配置数据库的问题，最终还是完美解决。而对于 `jmeter` 则是主要学习记录操作，线程组压力测试，同时中途发现需要使用 `http cookie` 管理和 `http` 信息头管理以及本地服务器代理的配置以后才能正常进行压力测试。同时由于使用了容器，而这个 `lamp` 容器 `centos` 是阉割版，没有 `yum` 因此我又进一步安装了 `yum` 才完成了最后的 `sysstat` 的安装以及性能分析。经过本次实验我对 `jmeter` 的使用以及压力测试的具体原理和 `beanshell` 预处理自定义测试有了深入的理解，相信对未来自己的软件测试有着巨大的帮助。