

JMeter 的简单使用

1. JMeter 简介

Apache JMeter 是 Apache 组织开发的基于 Java 的压力测试工具。用于对软件做压力测试，它最初被设计用于 Web 应用测试，但后来扩展到其他测试领域。它可以用于测试静态和动态资源，例如静态文件、Java 小服务程序、CGI 脚本、Java 对象、数据库、FTP 服务器，等等。JMeter 可以用于对服务器、网络或对象模拟巨大的负载，来自不同压力类别下测试它们的强度和分析整体性能。另外，JMeter 能够对应用程序做功能 / 回归测试，通过创建带有断言的脚本来验证你的程序返回了你期望的结果。为了最大限度的灵活性，JMeter 允许使用正则表达式创建断言。

2. 安装与配置

2.1. 工具 / 原料

JDK 1.7

Apache JMeter 3.0

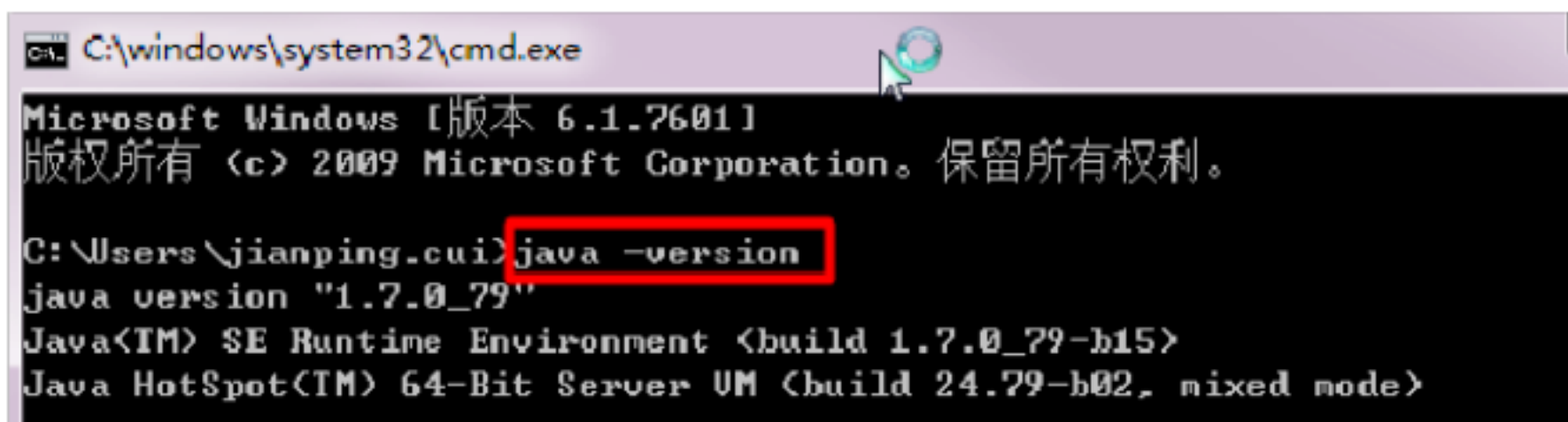
2.2. 环境

JMeter 可以在 Windows/Linux/Mac 平台上运行，这里介绍在 Windows

上的安装配置。

2.3. 安装步骤

安装 JDK，网上下载一个 JDK，配置好环境变量即可使用。在 dos 界面输入 `java -version`，能显示版本号，则证明 JDK 安装 OK，可以使用。

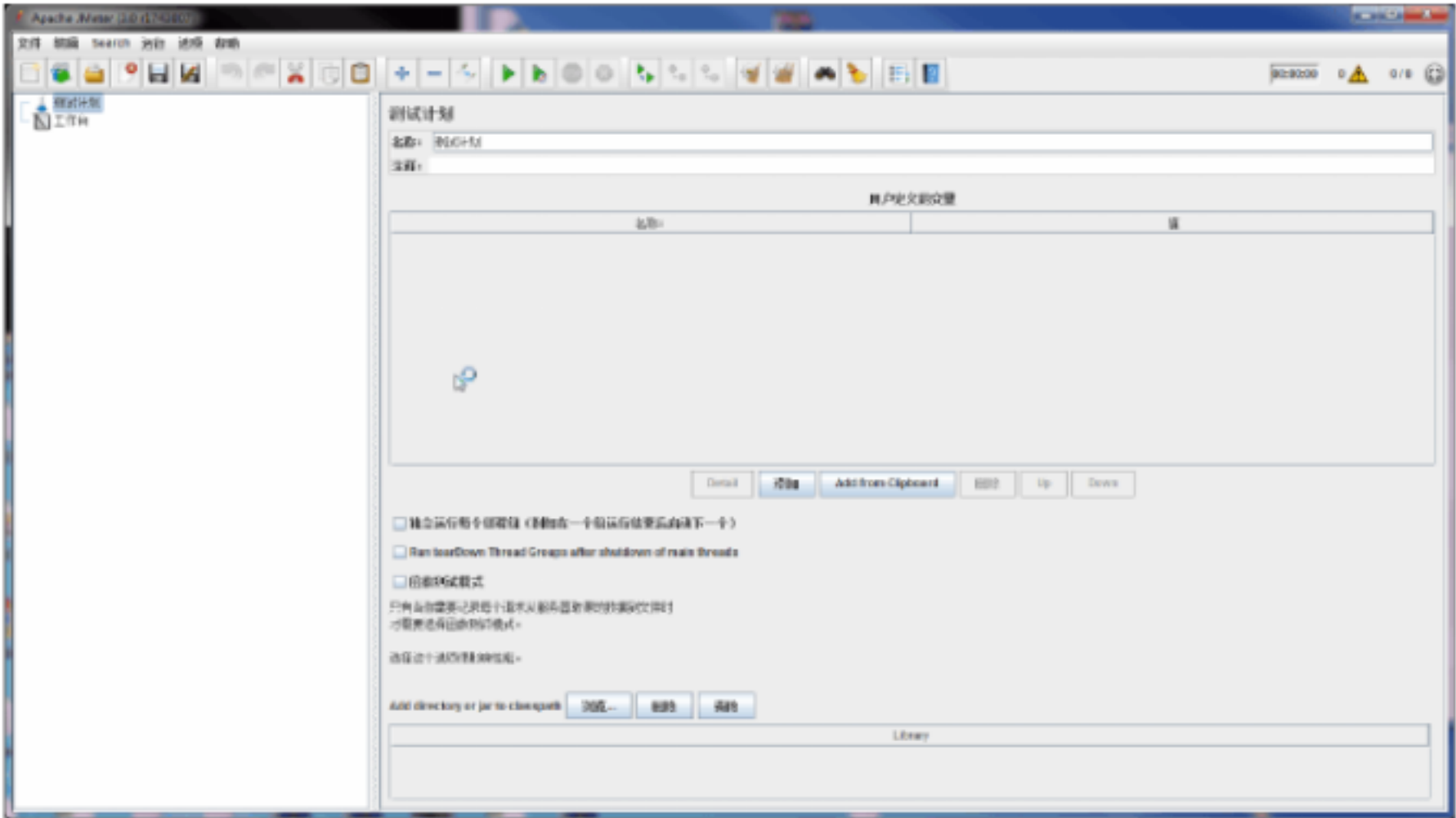


```
C:\windows\system32\cmd.exe
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\jianping.cui>java -version
java version "1.7.0_79"
Java(TM) SE Runtime Environment (build 1.7.0_79-b15)
Java HotSpot(TM) 64-Bit Server VM (build 24.79-b02, mixed mode)
```

安装 Jmeter：Jmeter 不需要安装，解压之后即可使用；

双击 Jmeter 的 bin 目录下的 `jmeter.bat`，出现下面界面，则 Jmeter 启动成功，Jmeter 就绪。



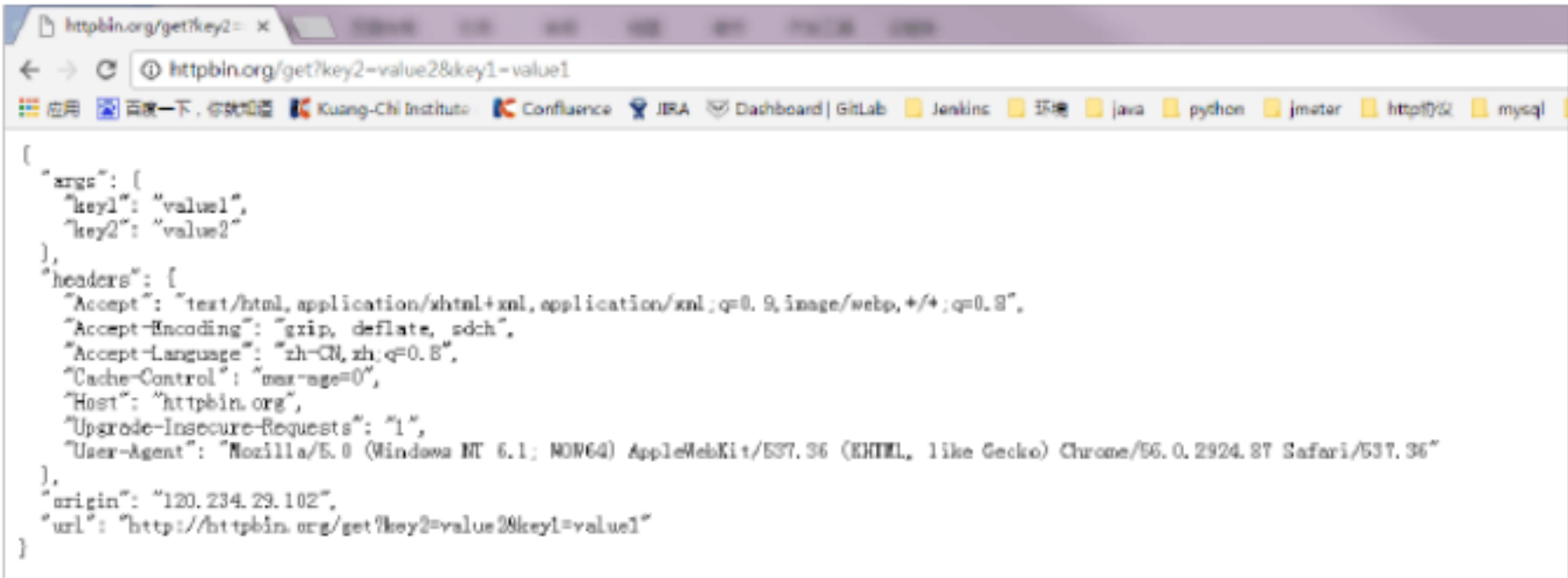
3. HTTP接口测试

3.1. 测试实例

以一个简单的接口为例：

<http://httpbin.org/get?key2=value2&key1=value1>

在浏览器中打开的效果如下：

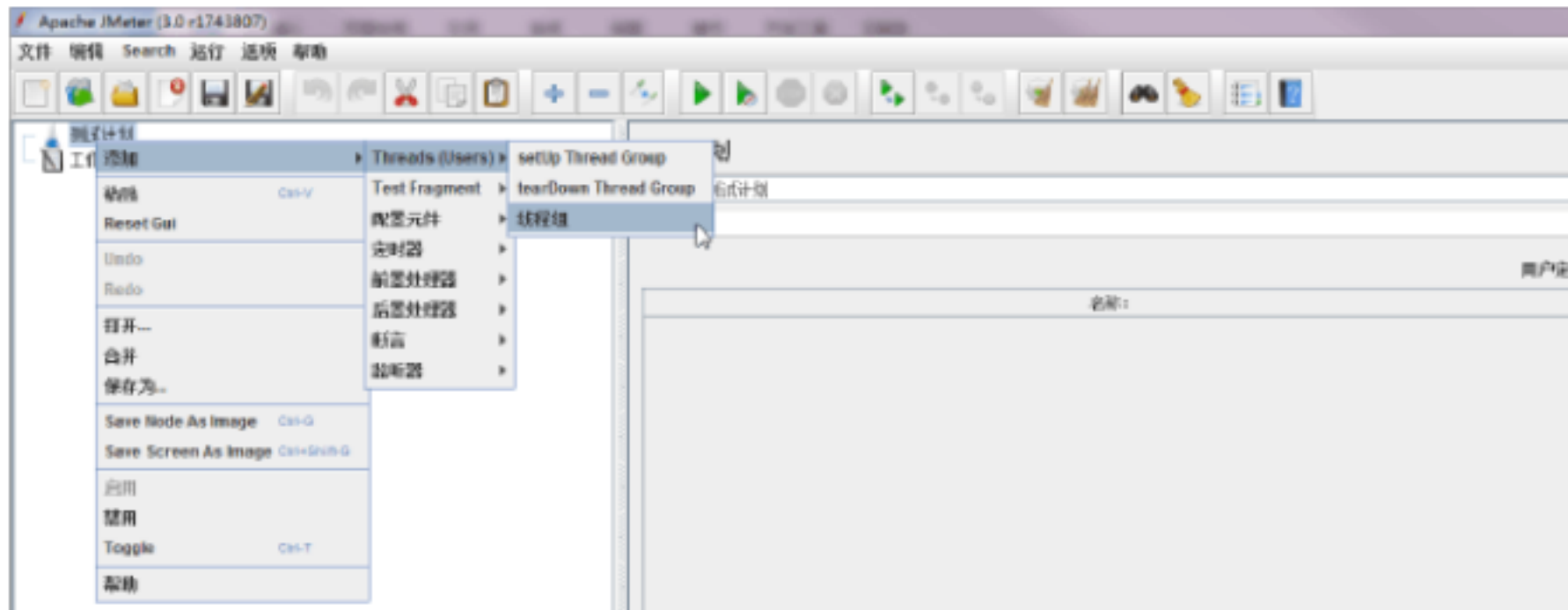


在这里，成功的用浏览器访问了这个接口，并得到了成功的返回报文。下面我们使用 Jmeter 来访问这个接口。

3.2. 使用 Jmeter 进行 HTTP 测试

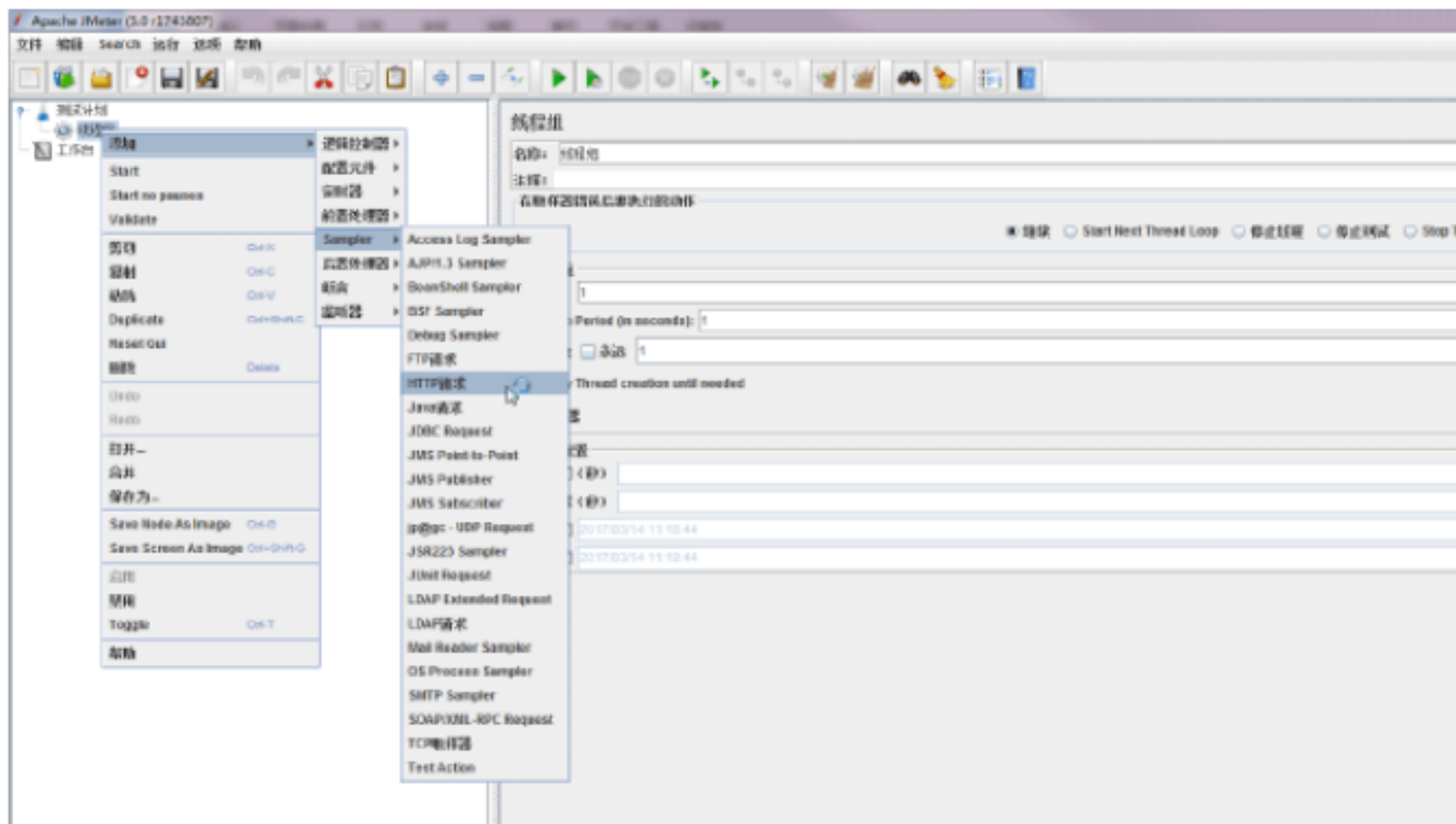
3.2.1. 添加线程组

右键“测试计划” -> “添加” -> “Threads(Users)” -> “线程组”

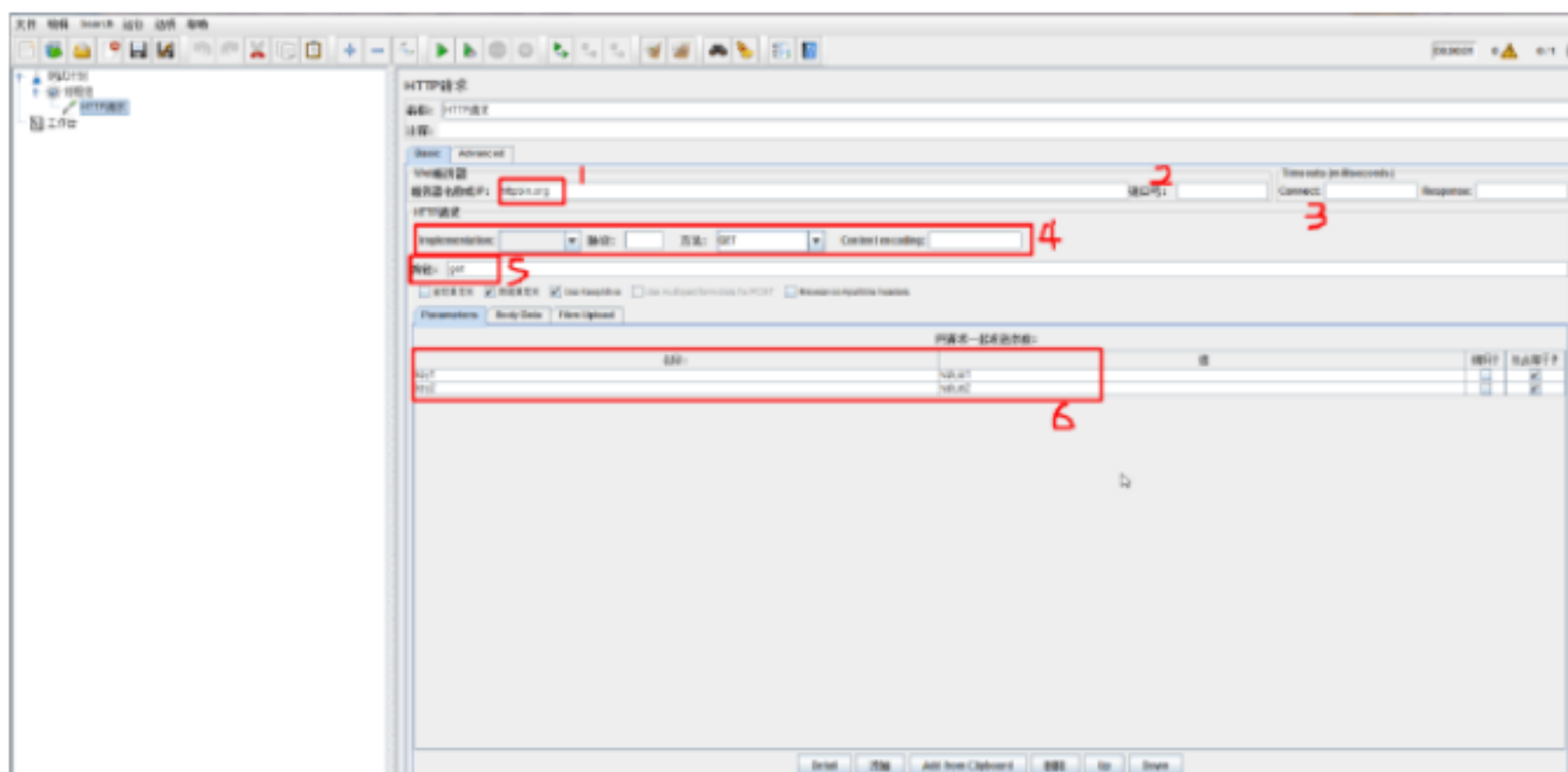


3.2.2. 添加 “ HTTP 请求 ”

右键“线程组” -> “添加” -> “Sampler” -> “HTTP 请求”



打开之后，填入访问该接口所需要的参数，效果如下图



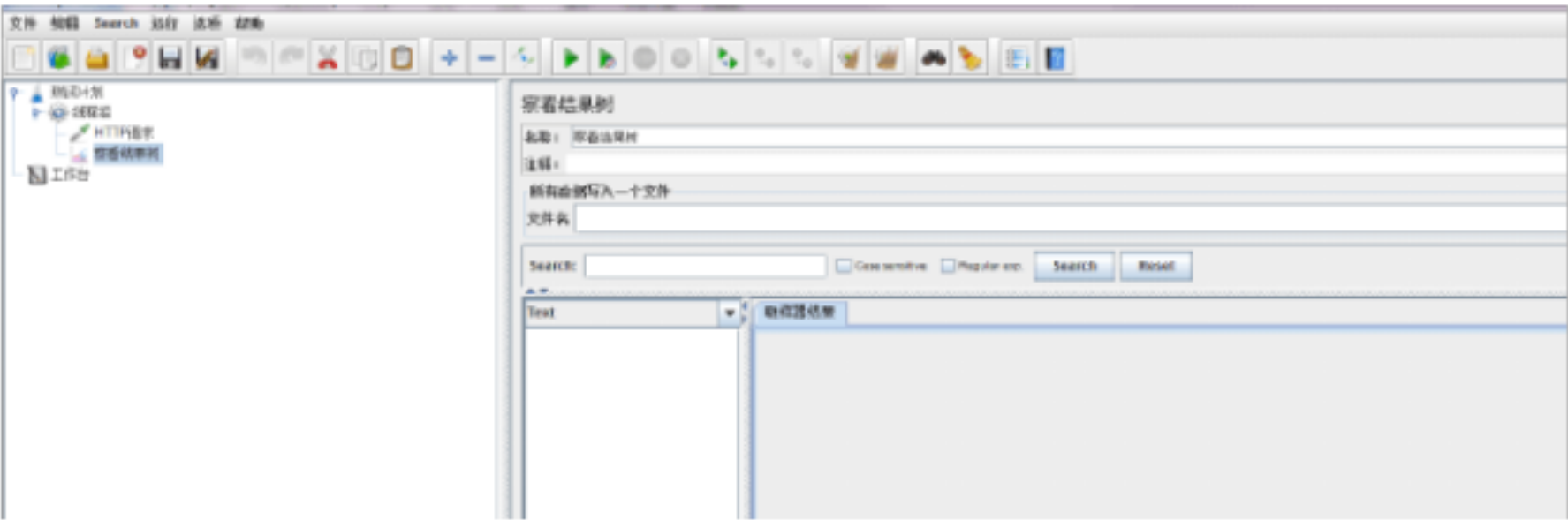
Tips :

- 1. 服务器名或 IP：这里填写服务 IP 地址（不带端口），有域名的直接填写域名
- 2. 端口号：这里填写端口号，有域名的这里为空就好。
- 3. Timeouts(milliseconds)：超时时间，单位是毫秒； Connect：连接超时； Response：响应超时。二者都可以不填
- 4. Implementa：采用什么实例来运行这次 HTTP 请求（HttpClient4/HttpClient3.1/Java），这里我们不用选择，直接默认就好。
协议：该接口采用什么协议（http/https），这里不填默认是 http，根据情况填写
方法：请求该接口的方法（GET/POST等），根据接口请求类型选择。
Content encoding：这里是返回报文的编码格式（如 utf-8），采用默认值就好
- 5. 同请求一起发送参数：接口的参数，可以通过下方“添加”按钮，添加参数，左边为“key”，右边填“value”，如上图。（这里只适用于传参数的接口，传 Json 和文件等接口使用这种方式不支持，暂不做介绍）

3.2.3. 添加监听器


完成上述 3.2.1 和 3.2.2 步，一个接口的简单脚本已经编写完成了；但是此时我们还不知道在哪里查看脚本运行的结果，这里我们需要添加一个监听器。

右键“线程组” -> “添加” -> “监听器”，选择“察看结果树”



Tips：Jmeter 提供了需要监听器，有兴趣的可以去一一添加，察看效果。

3.2.4. 运行脚本

点击  按钮，运行脚本，察看结果



在“响应数据”下，我们看到了与使用浏览器一样的返回结果（报文中 headers 有差异，这

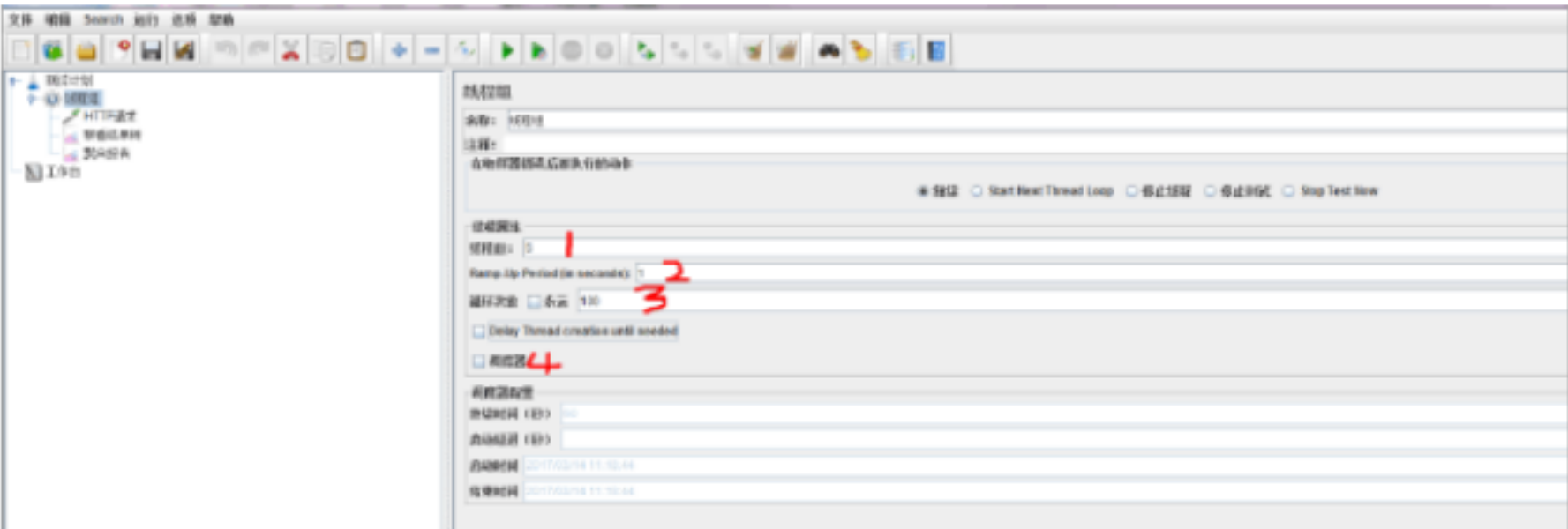
部分可以忽略，想知道为什么不一样的可以去学学 http 协议，在这里不影响结果）。

到这里，一个 http 接口测试的 Jmeter 脚本已经完成。“ Ctrl+S ”保存一下吧，下次就可以直接拿来使用了（ Jmeter 脚本默认保存在 bin 目录下）

3.3. 压力测试

在上面脚本配置上线程数，即可对该接口进行压力测试啦。

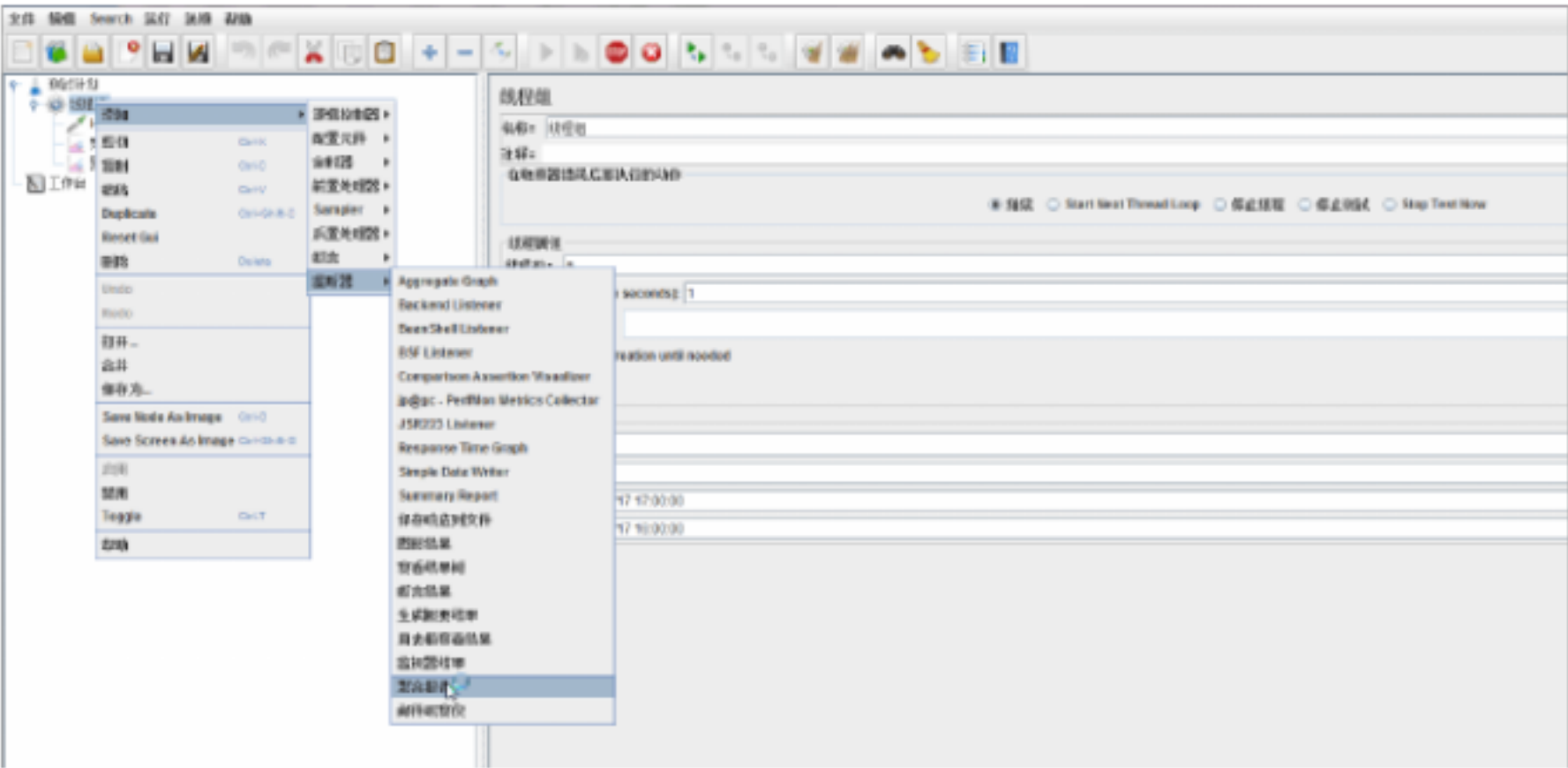
点击“线程组”



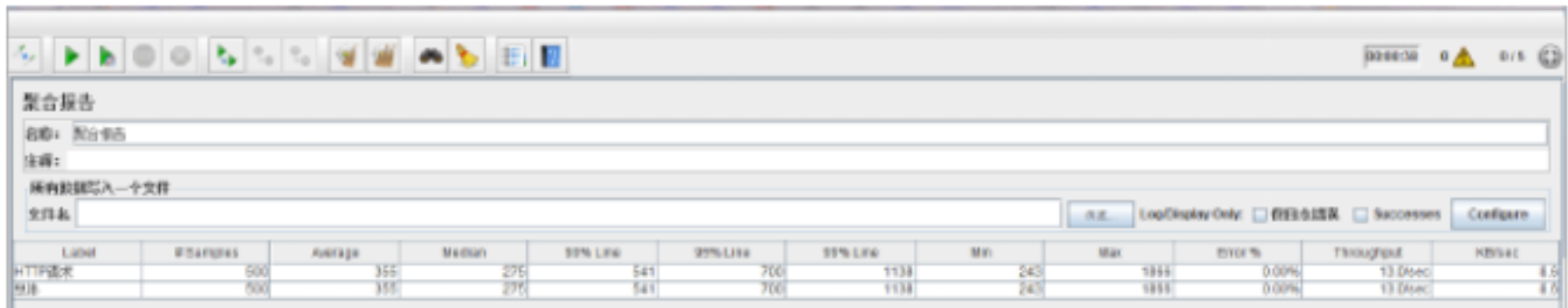
1. 线程数：需要采用多少个并发。
2. Ramp-Up Period(in seconds): 设置到达最大并发所需时间 （如上图，从 0 增加到 5 个并发，花费 1 秒钟）
3. 循环次数：每个并发执行几遍脚本（如上图，每个线程执行 100 遍）。勾选永远，则脚本会被无限循环执行下去。
4. 调度器：勾选则启用调度器
持续时间（秒）：设置脚本跑多久（如需要让脚本跑一分钟，则填入 60）
启动延迟（秒）：延迟多少秒才开始执行脚本

添加“聚合报告”


聚合报告是一个监听器，作用是，为本次压力测试整理出一份“报表”

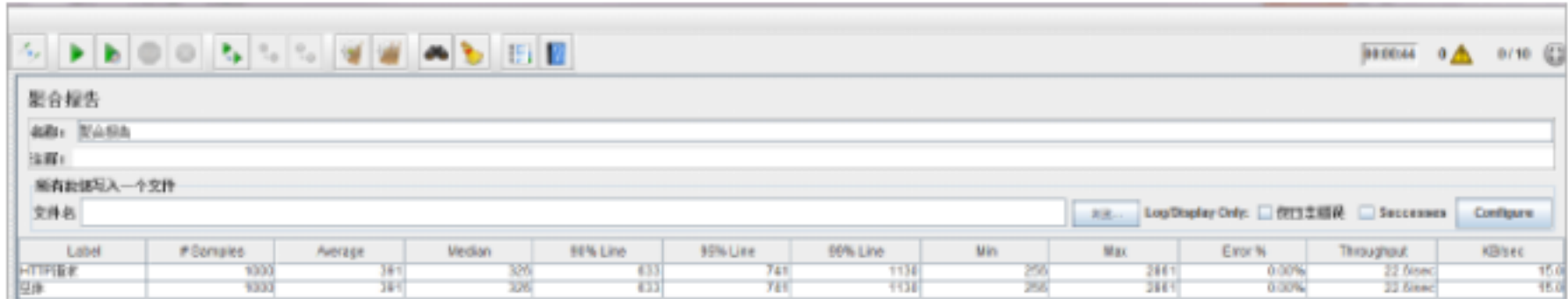


点击 按钮让我们来看看 5 并发循环 100 次的结果



Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Max	Error %	Throughput	KB/sec
HTTP请求	500	355	275	541	700	1138	243	1885	0.00%	13.0/sec	8.5
连接	500	355	275	541	700	1138	243	1885	0.00%	13.0/sec	8.5


 按钮，清除上次测试结果，然后来看看 10 并发循环 100 次结果



Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Max	Error %	Throughput	KB/sec
HTTP请求	1000	181	105	633	741	1138	250	2881	0.00%	22.6/sec	15.0
连接	1000	181	105	633	741	1138	250	2881	0.00%	22.6/sec	15.0

可以看到 5 并发时，Throughput 为 13.0/sec；10 并发时：Throughput 为 22.6/sec。
 理论上，并发数与吞吐量成正比，并发越多，吞吐量越大。
 实际上，服务器的处理能力和网络带宽是有限的，当并发数达到一定数量之后，吞吐量不随并发数增加而增长，出现响应时间变长，甚至服务器宕机等状况，称之为服务器的性能“瓶颈”。
 简单来说，性能测试就是根据服务器的“瓶颈”，来评估服务的性能是否能够达到期望。

附：表头说明

- Label - 如图，在不勾选 "Include group name in label?" 复选框的情况下，为请求取样器的名称，否则为 “请求取样器所在线程组 :请求取样器名称”
- # Samples - 用同一个请求取样器，发送请求的数量 (注意：该值是不断累计的)。比如，10 个线程数设置为 10，迭代 10 次，那么每运行一次测试，该值就增加 10*10=100
- Average - 默认情况下是单个 Request 的平均响应时间，当使用了 Transaction Controller 时，也可以以 Transaction 为单位显示平均响应时间
- Median - 中位数。表示响应时间本不大于该时间值的请求样本数占总数的 50%
- 90% Line - 表示响应时间不大于该时间值的请求样本数占总数的 90%
- Min - 针对同一请求取样器，请求样本的最小响应时间
- Max - 针对同一请求取样器，请求样本的最大响应时间
- Error % - 出现错误的请求样本的百分比
- Throughput - 吞吐量以 “requests/second requests /minute、requests /hour 来衡量。时间单位已经被选取为 second，所以，显示速率至少是 1.0，即每秒 1 个请求。当吞吐量被保存到 CVS 文件时，采用的是 requests/second 所以 30.0 requests/second 在 CVS中被保存为 0.5Kb/sec - 以 Kilobytes/seond 来衡量的吞吐量

3.4. 附上演示脚本

上面演示步骤最后保存的脚本，可以直接用 Jmeter 打开，运行。



演示脚本 .jmx

4. 录制 HTTP脚本

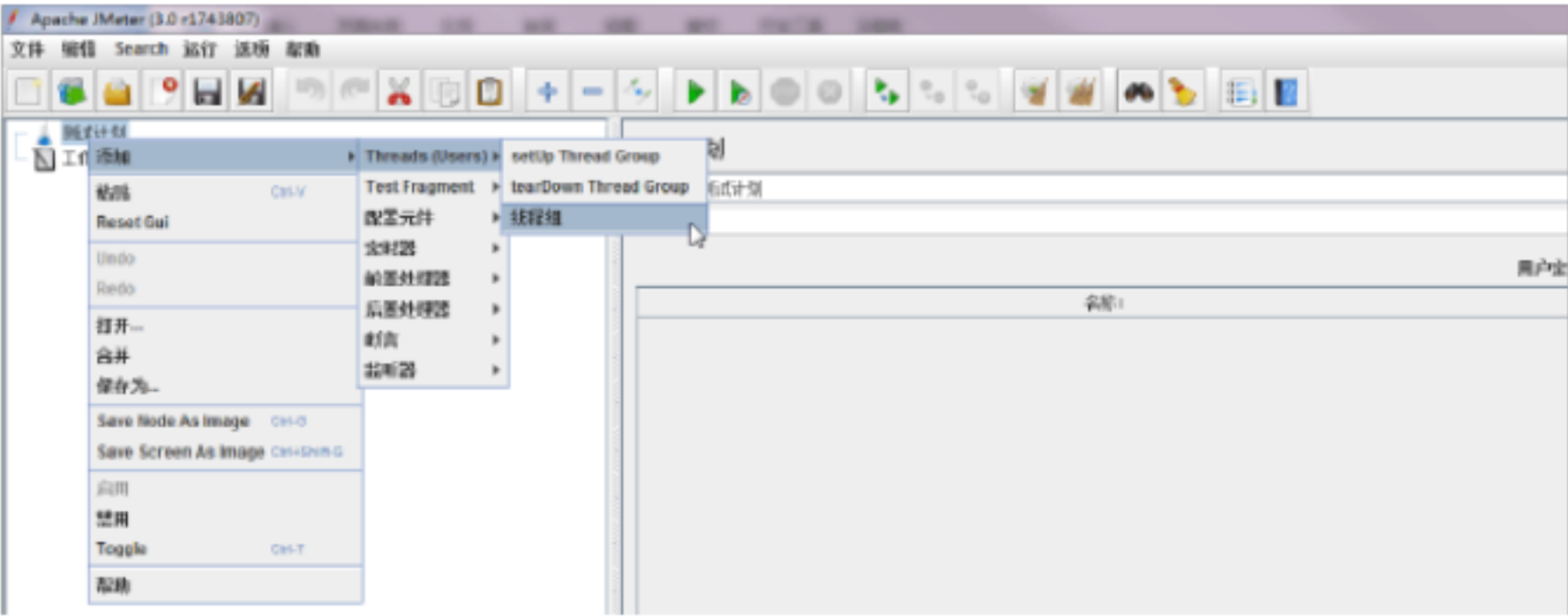
当需要为多个接口做脚本时，手动编制就显得太麻烦，这时候我们可以使用 Jmeter 自带的脚本录制功能。

录制原理是 Jmeter 启用一个代理服务器，然后配置浏览器代理， Jmeter 就可以对浏览器进行抓包，然后生成脚本。

4.1. Jmeter 配置代理服务器并录制脚本

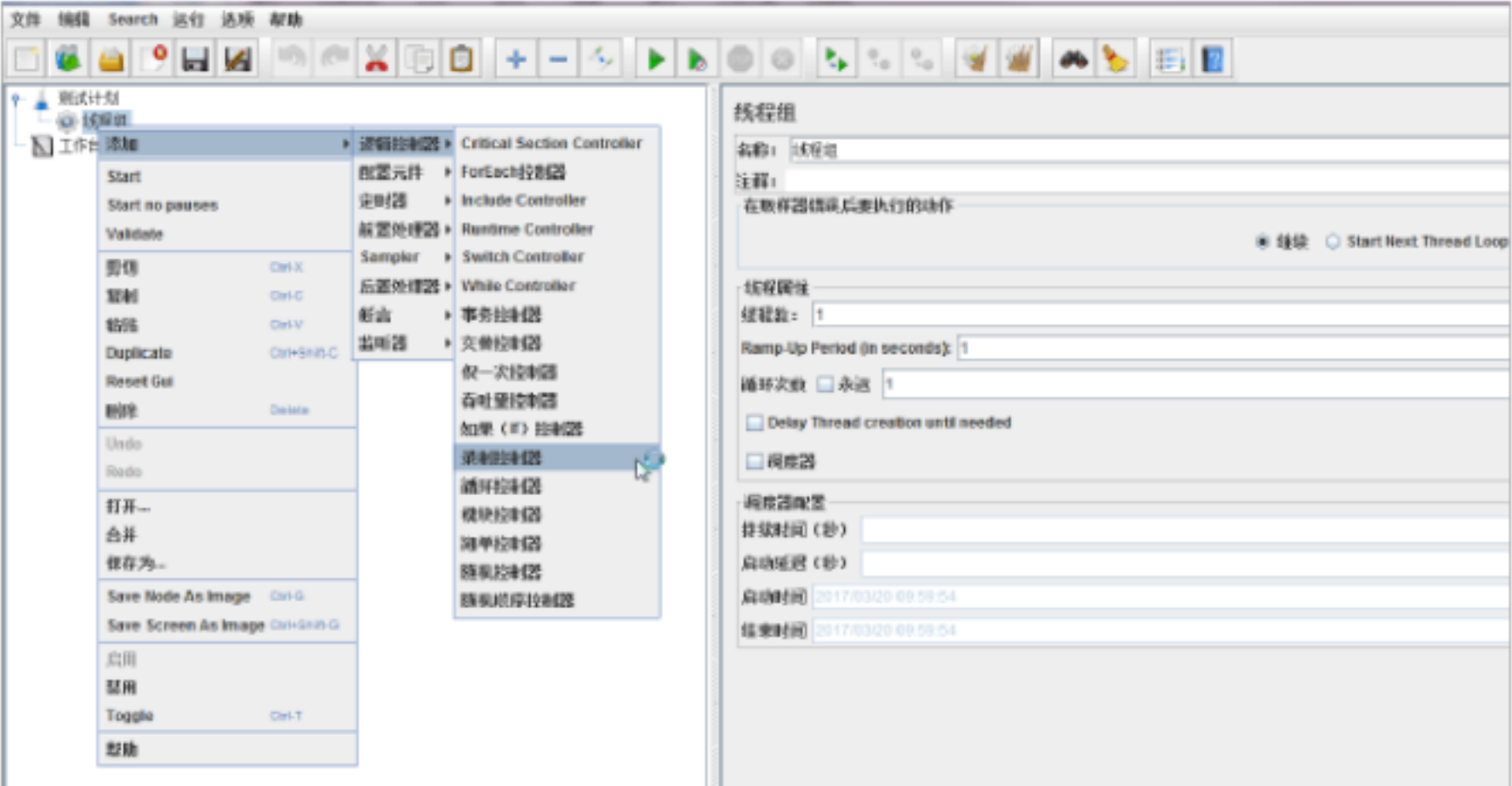
4.1.1. 添加线程组

右键“测试计划” -> “添加” -> “ Threads(Users)” -> “ 线程组 ”



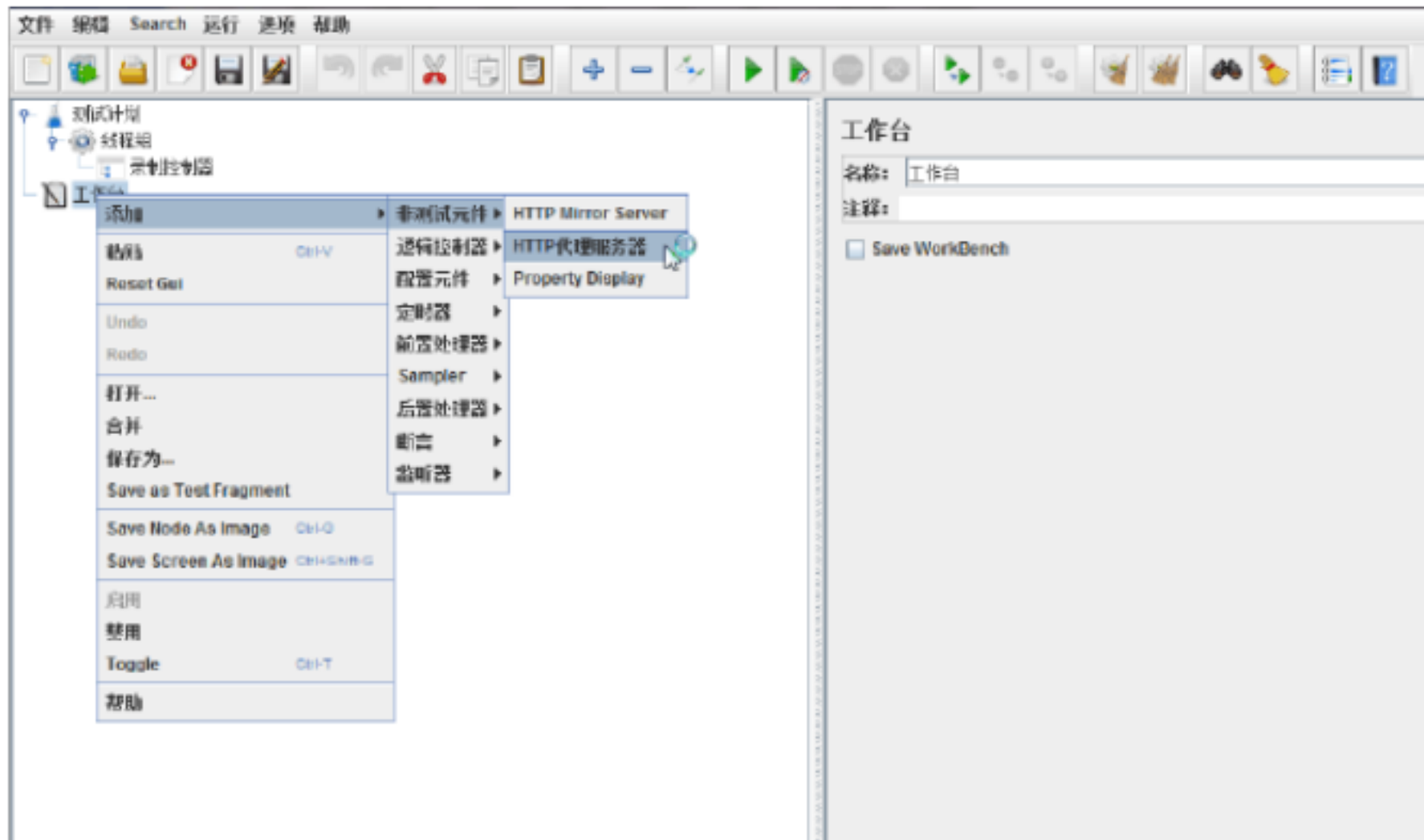
4.1.2. 添加录制控制器

右键“线程组” -> “添加” -> “ 逻辑控制器 ” -> “ 录制控制器 ”

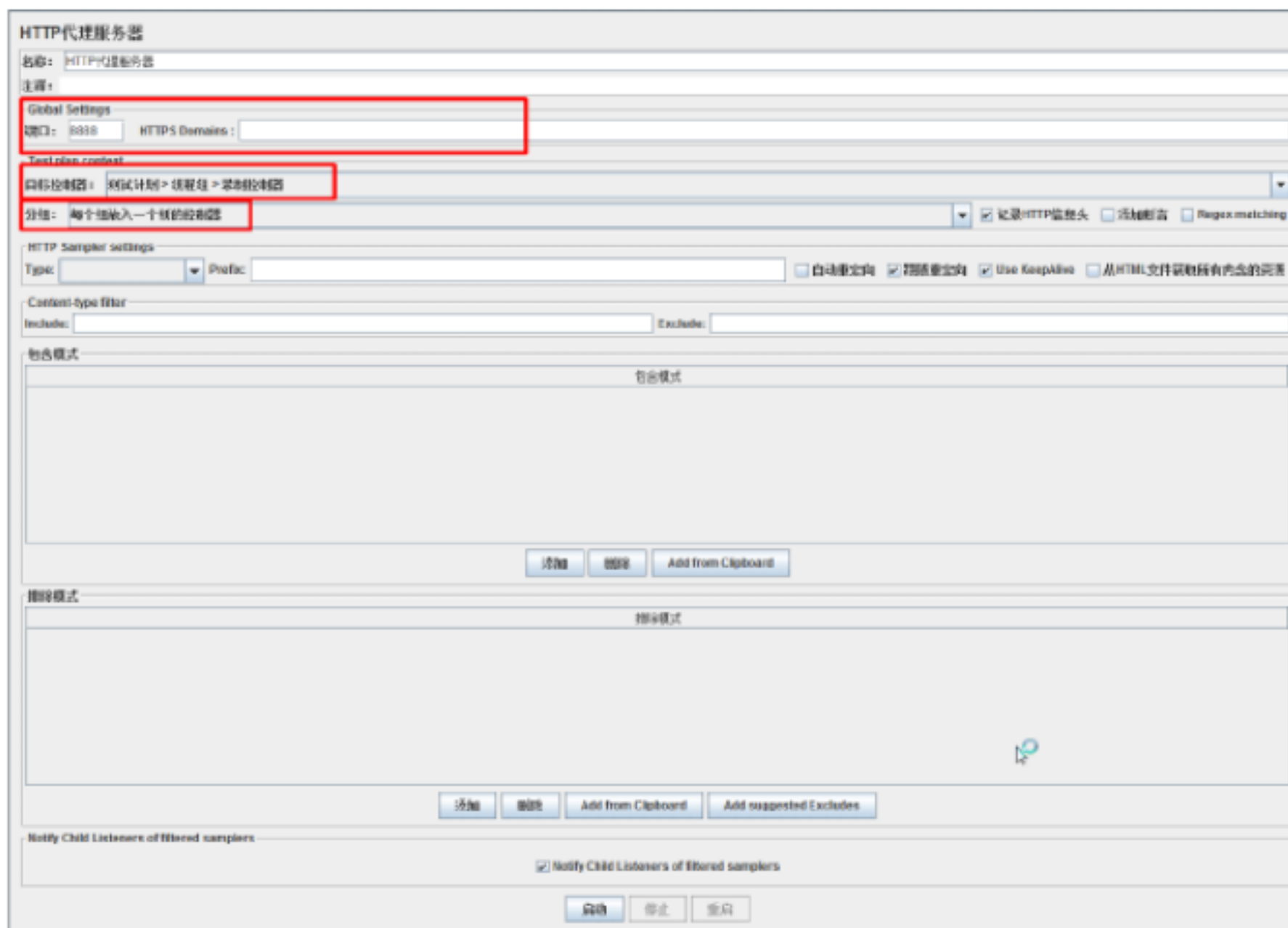


4.1.3. 添加代理服务器

右键“工作台” -> “添加” -> “ 非测试元件 ” -> “ Http 代理服务器 ”



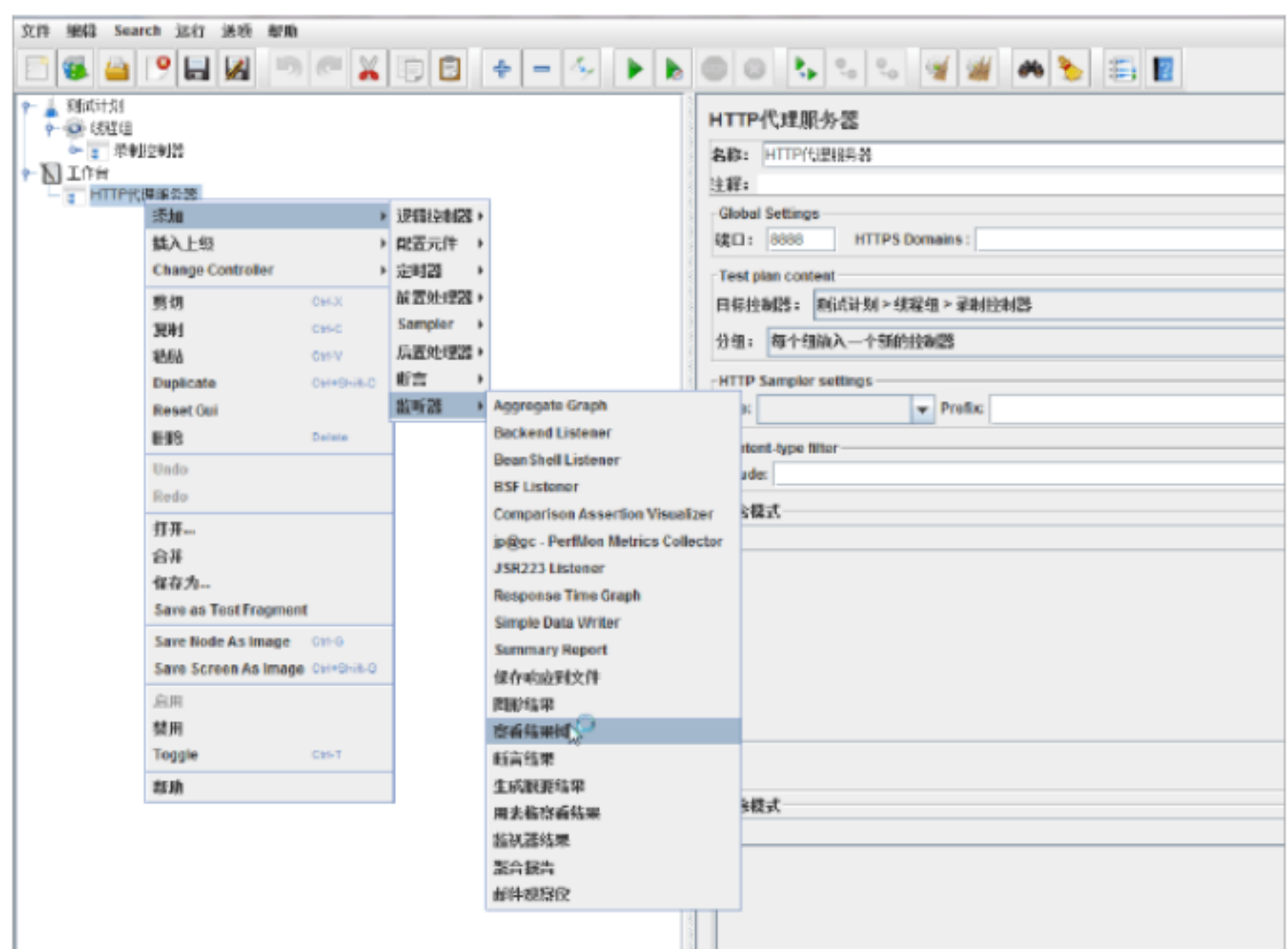
配置代理服务器



1. 端口：用默认的 8888 就好，如果冲突，可以换一个不冲突的端口号。
 2. HTTPS Domains: 默认本机 localhost
 3. 目标控制器：选择 测试计划 > 线程组 > 录制控制器
 4. 分组：选择 每个组放入一个新的控制器
- 然后点击“启动”，弹出的提示框点击“确定”，代理就启动成功了。

添加“察看结果树”

右键“HTTP代理服务器” -> “添加” -> “监听器” -> “察看结果树”



4.1.4. 设置浏览器代理

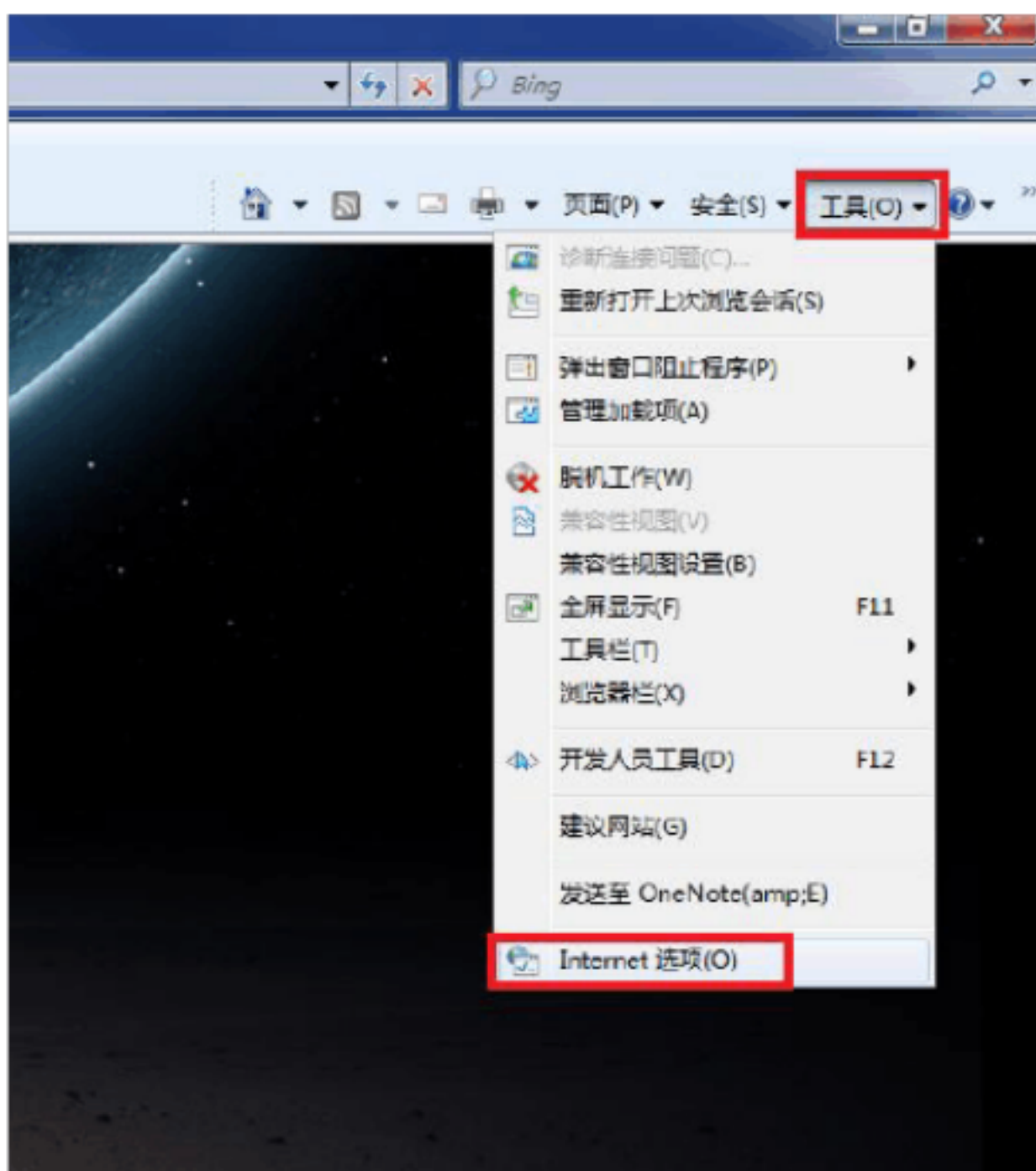
Jmeter 代理服务器启动之后，再设置浏览器代理，就可以抓包了。

不同浏览器的代理设置方式不一样。一般来说，浏览器都是在设置系统代理，一个浏览器设置了代理，其它浏览器也会用这个代理。

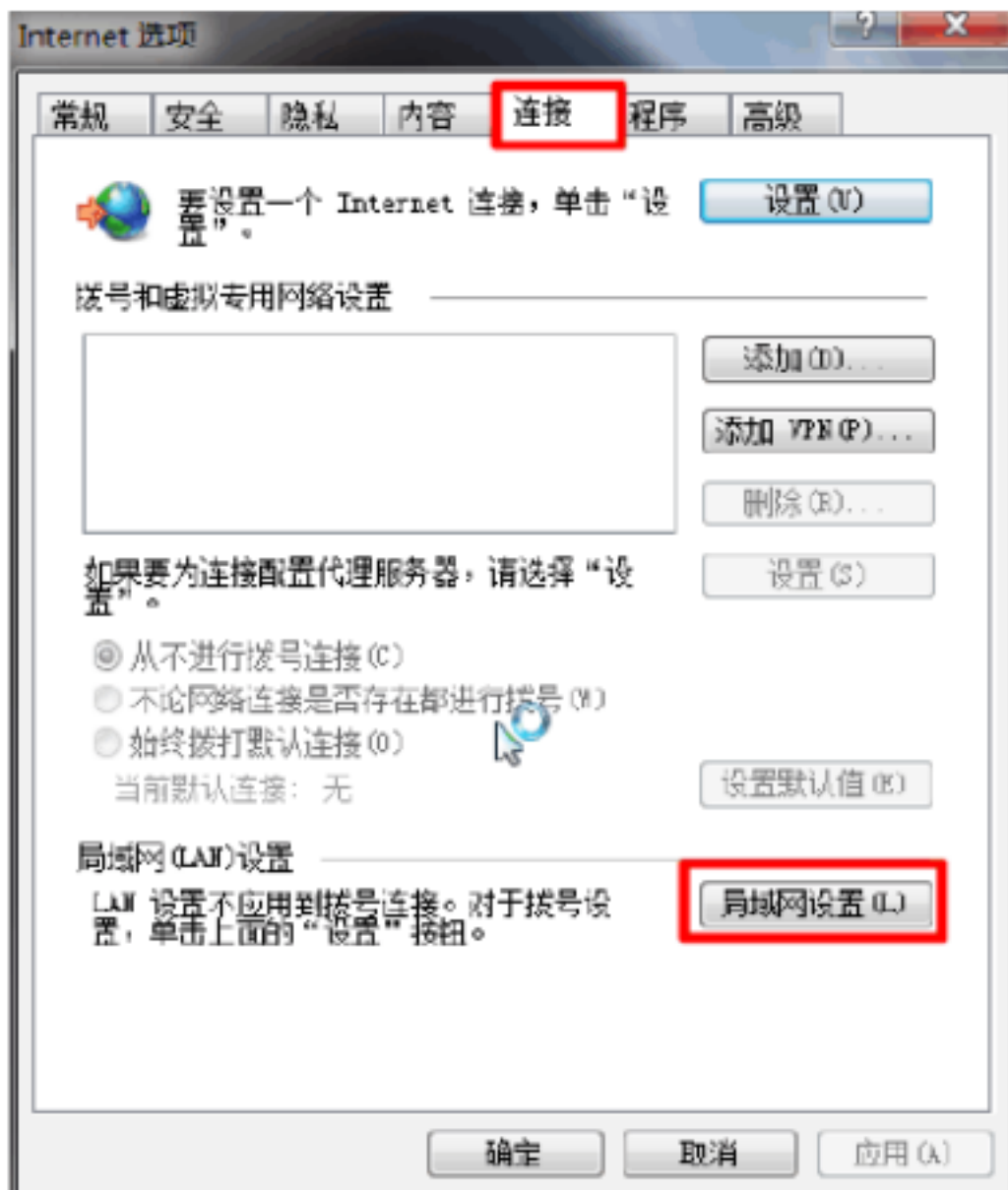
下面介绍三种常用浏览器代理的设置方法。

4.1.4.1. IE 浏览器

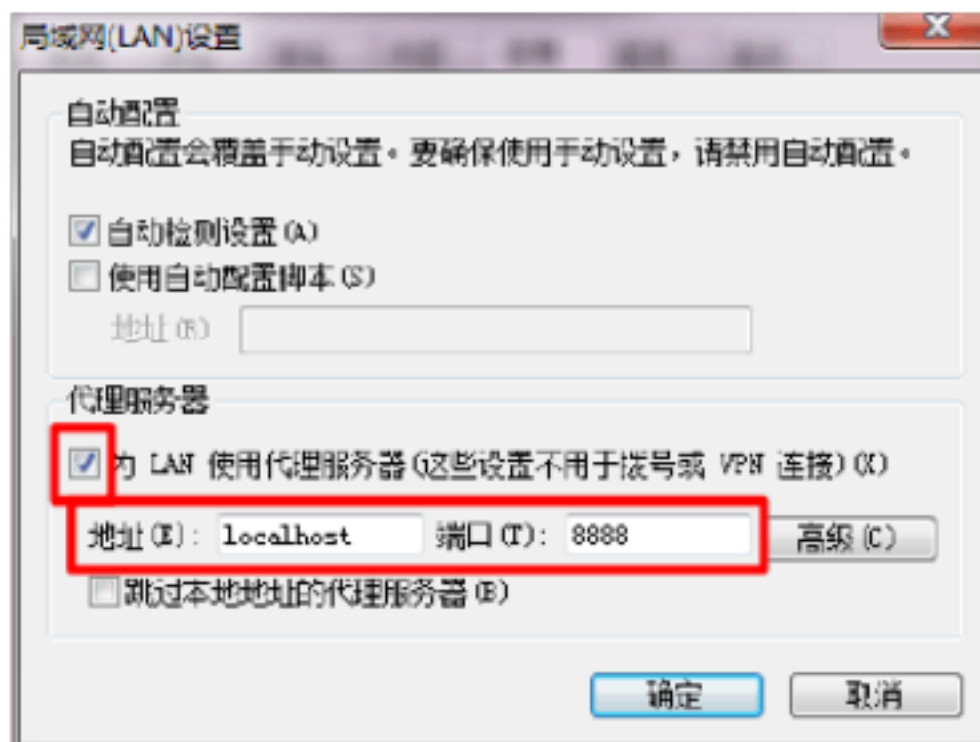
1. 点击“工具” -> “Internet 选项”



2. 在新打开的窗口中选择“连接”，点击“局域网设置（ L ）”



3. 在弹出窗口，选中“为 LAN 使用代理服务器（这些设置不用于拨号或 VPN 连接）（ X ）”，然后在“地址（ E ）”填入 localhost，“端口（ T ）”填入刚设置的代理端口号 8888。然后点击“确定”完成配置。

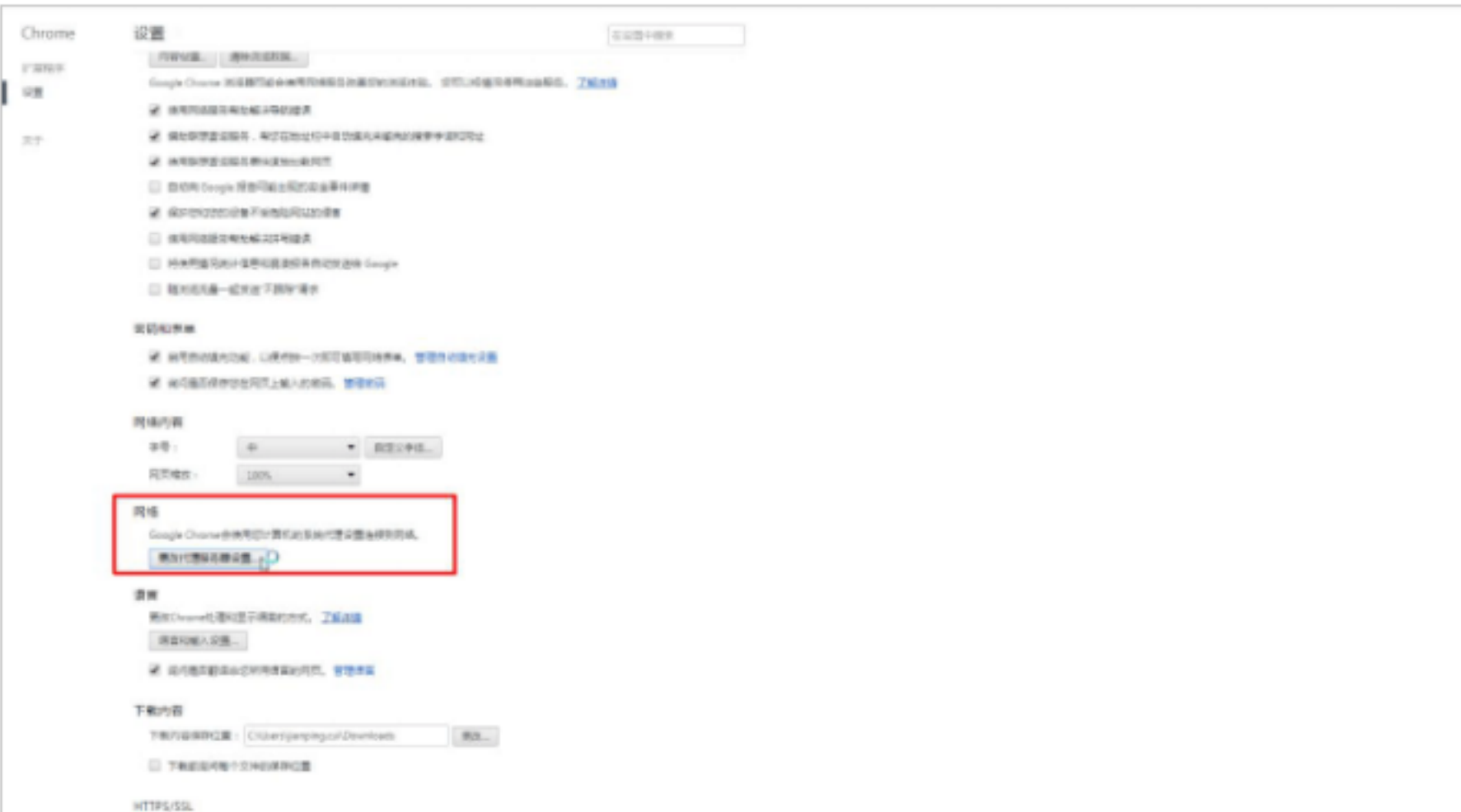


4.1.4.2. Google Chrome （谷歌浏览器）

1. 点击右上角，展开菜单，点击“设置”



2. 点击最下方“显示高级设置 ...”，找到“更改代理服务器设置 ...”，点击



3. 弹出新窗口后，剩下的操作照着 IE 浏览器设置代理的 2、3 就可以了

4.1.4.3. Firefox（火狐浏览器）

1. 点击右上角，展开菜单，点击“选项”

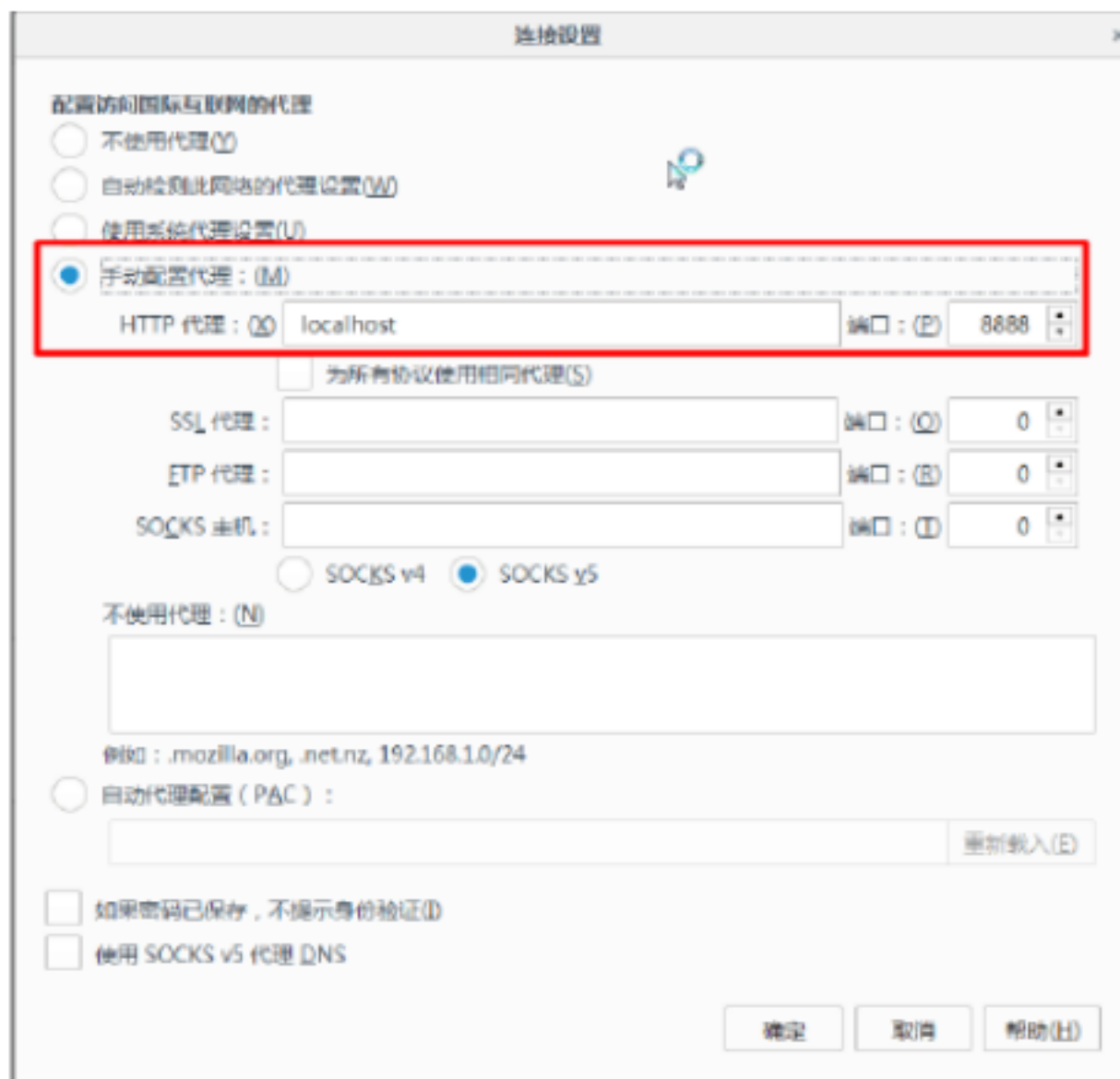


2. 选择“高级”->“网络”，点击“设置（E）...”



3. 在新窗口配置代理： 选择“手动配置代理 : (M)”，“ HTTP代理 : (M)”填入“ localhost ”，
“ 端口 : (P)”填入“ 8888 ”，点击“ 确定 ”，设置完成。

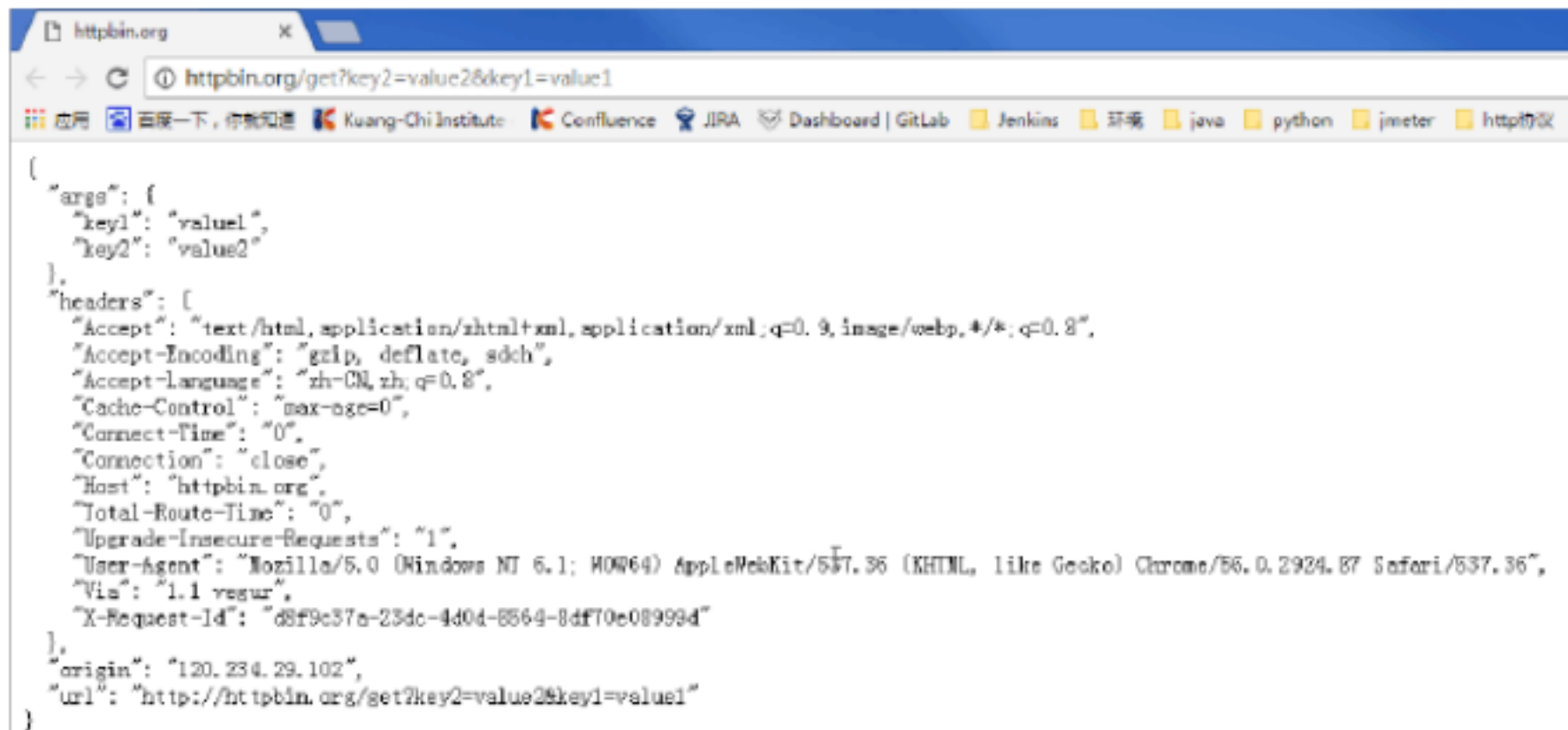
Tip：这里设置的代理是设置的自己的，不会影响系统代理设置。



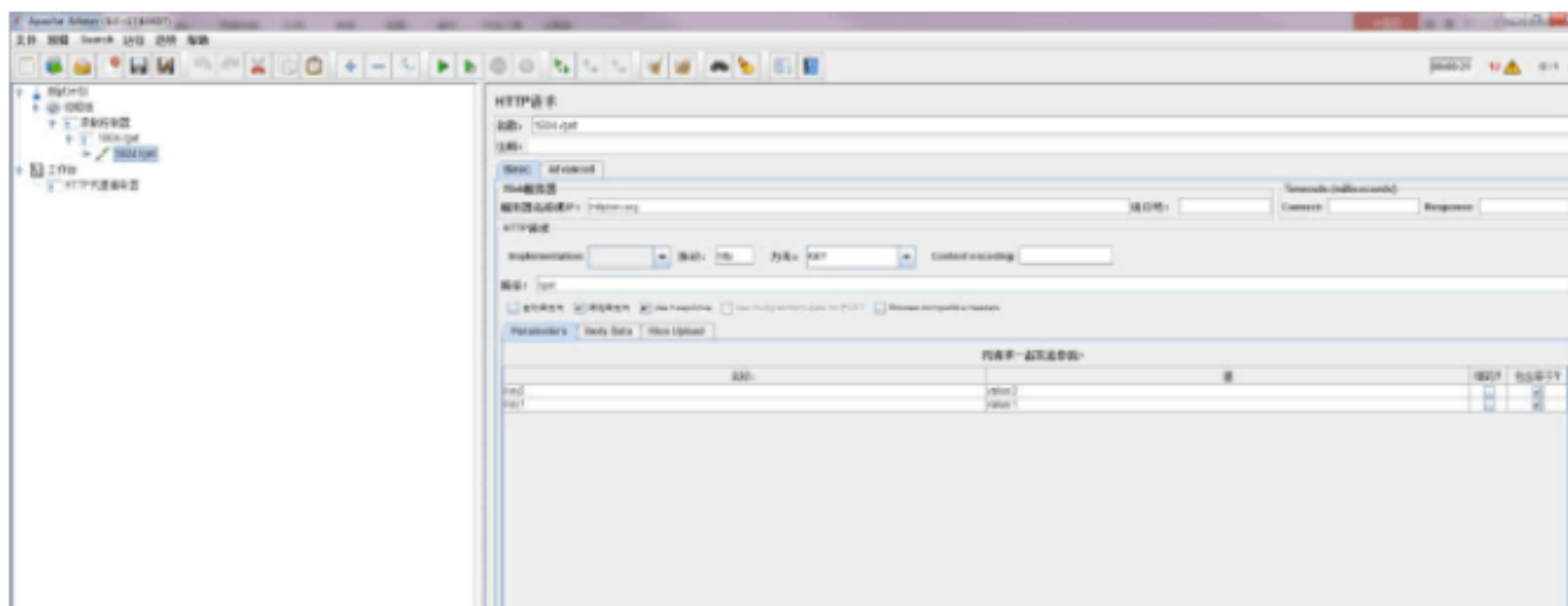
4.1.5. 录制脚本

所有准备工作都做完， 可以开始录制脚本了， 还用上面举例的接口， 这里尝试录制该接口的脚本。

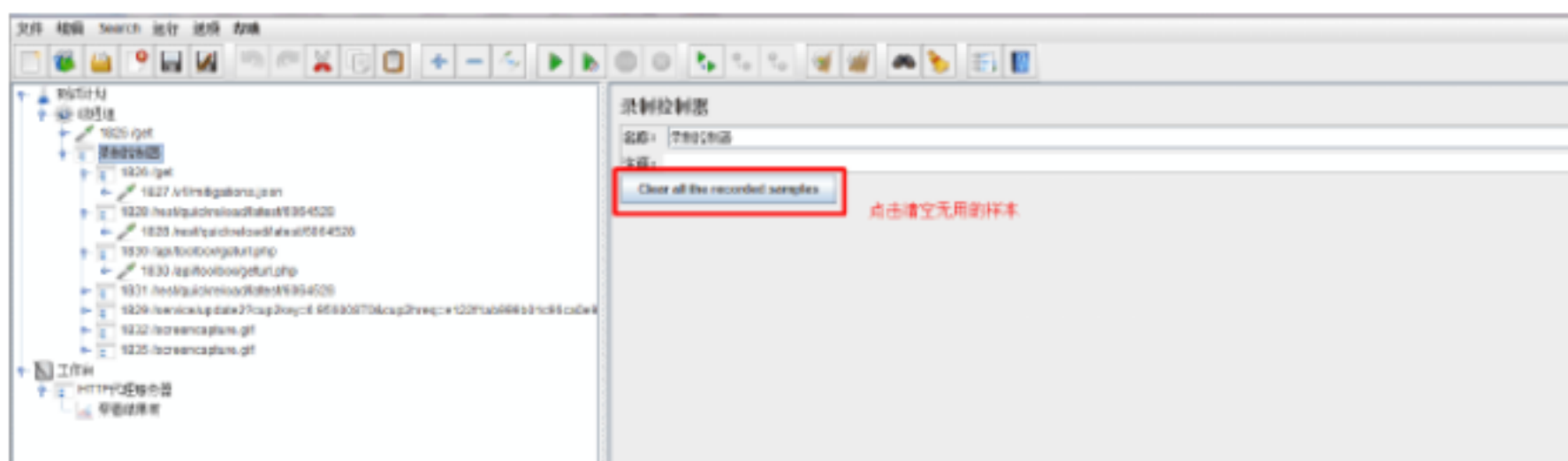
1. 在一个设置好代理的浏览器中打开该接口，用 Google Chrome 演示一下



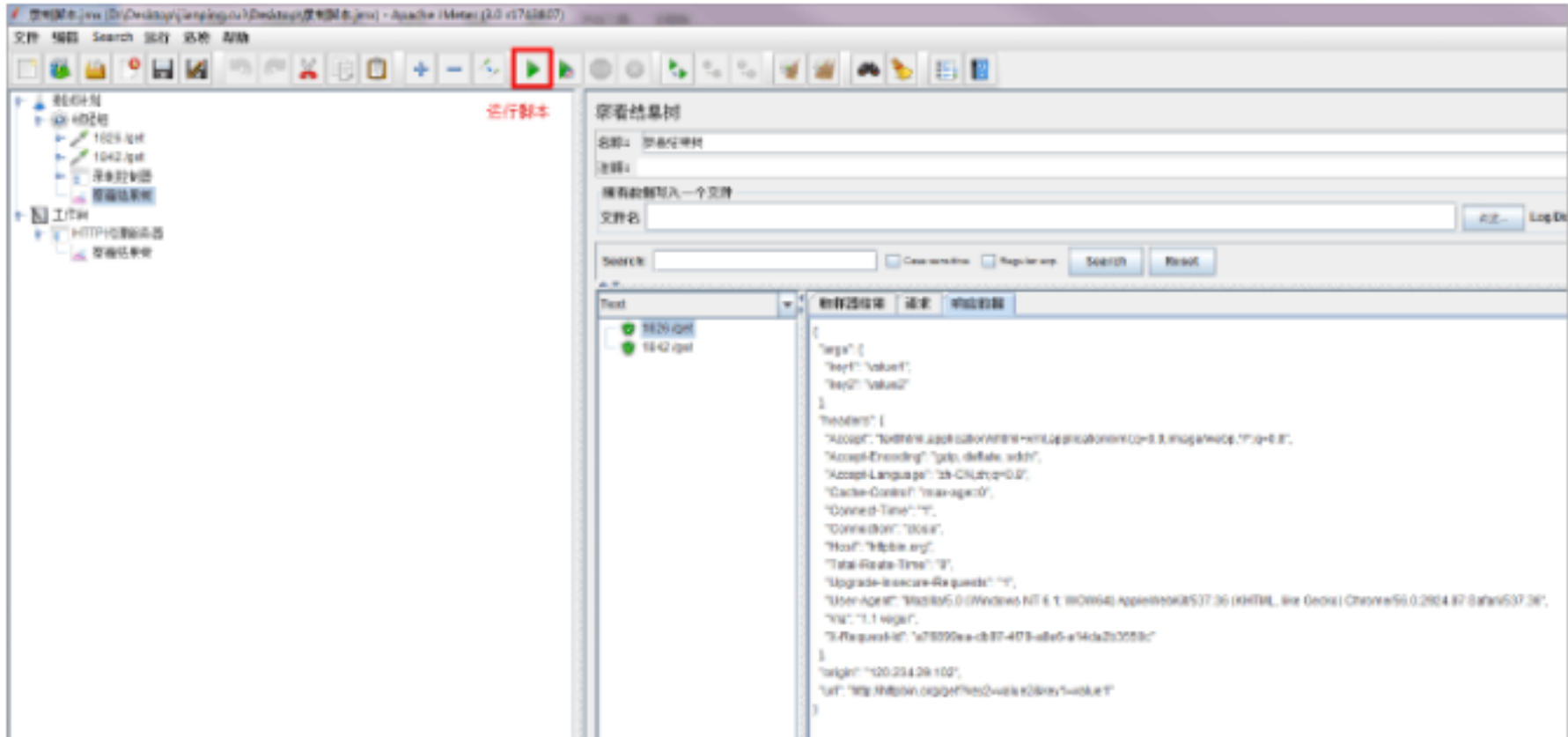
2. 然后我们看看 Jmeter，这里捕获了这次请求的信息，并在 “录制控制器” 下生成了一条 Http 请求，如下



3. 将需要的 Http 请求拉倒 “线程组” 下，然后清空 “录制控制器”，然后就可以开始下一个接口的录制



4. 录制完成之后，在线程组下添加 “察看结果树”，“Ctrl+S” 保存脚本，运行一下，看到两条请求都成功了，一个脚本就录制成功了！



4.1.6. 一个未解决的问题

在录制脚本时，会发现很多无用的请求干扰，在这里 Jmeter 代理提供了一个过滤器，如接口是 *.html 的，使用 “ ./*.*.html ” 过滤。

这里配置了过滤，总不好用，要么什么也没过滤掉，要么都过滤没了，还有把需要的一部分也过滤没了。

鉴于此，录制脚本时，尽量少开其它网页，减少干扰。
如有哪位知道怎么设置这个过滤器，请告知，感激不尽。

5. TCP测试

就像简介上说的， Jmeter 不单可以做 Web 测试，也可以 TCP测试。TCP脚本跟 HTTP脚本做成十分类似，但是有些地方会不同，还是啰嗦介绍一下。

5.1. 创建一个 TCP 测试桩

先创建一个简单的 TCP服务器，用来学习 Jmeter 的 TCP测试，这个服务器只做一个事情，就是收到一个报文，然后给一个响应。



threadServer.py

这是从网上找的一份简单的 TCP 服务器代码， python 语言写的，直接运行就好，端口是 8888，IP 则为运行该脚本的机器的 IP 地址。

问题：怎么运行这个服务器？

答：有很多方法可以启动，介绍一种简单的，找一个 Linux 服务器，上传这个文件，然后执行命令 python threadServer.py 就 OK了。

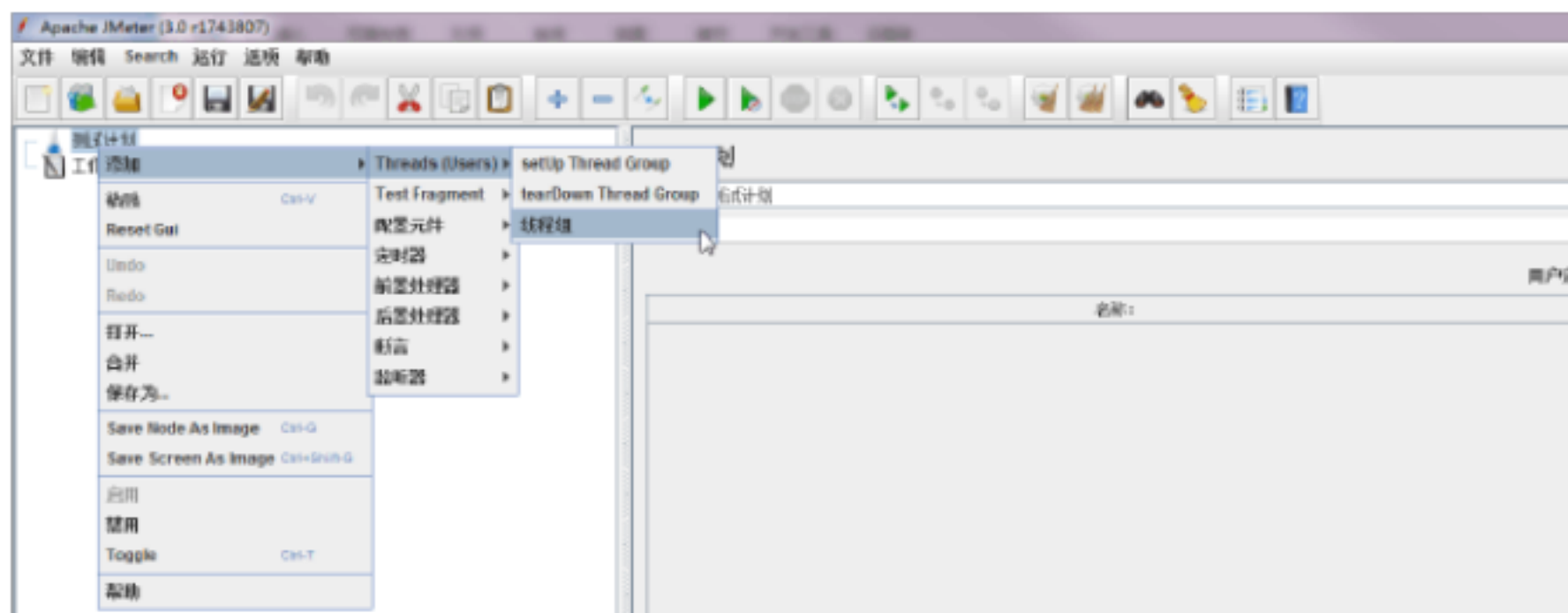
```
[root@model opt]# python threadServer.py
Socket created
Socket bind complete
Socket now listening
```

5.2. 创建 TCP 测试脚本

和 HTTP测试脚本类似

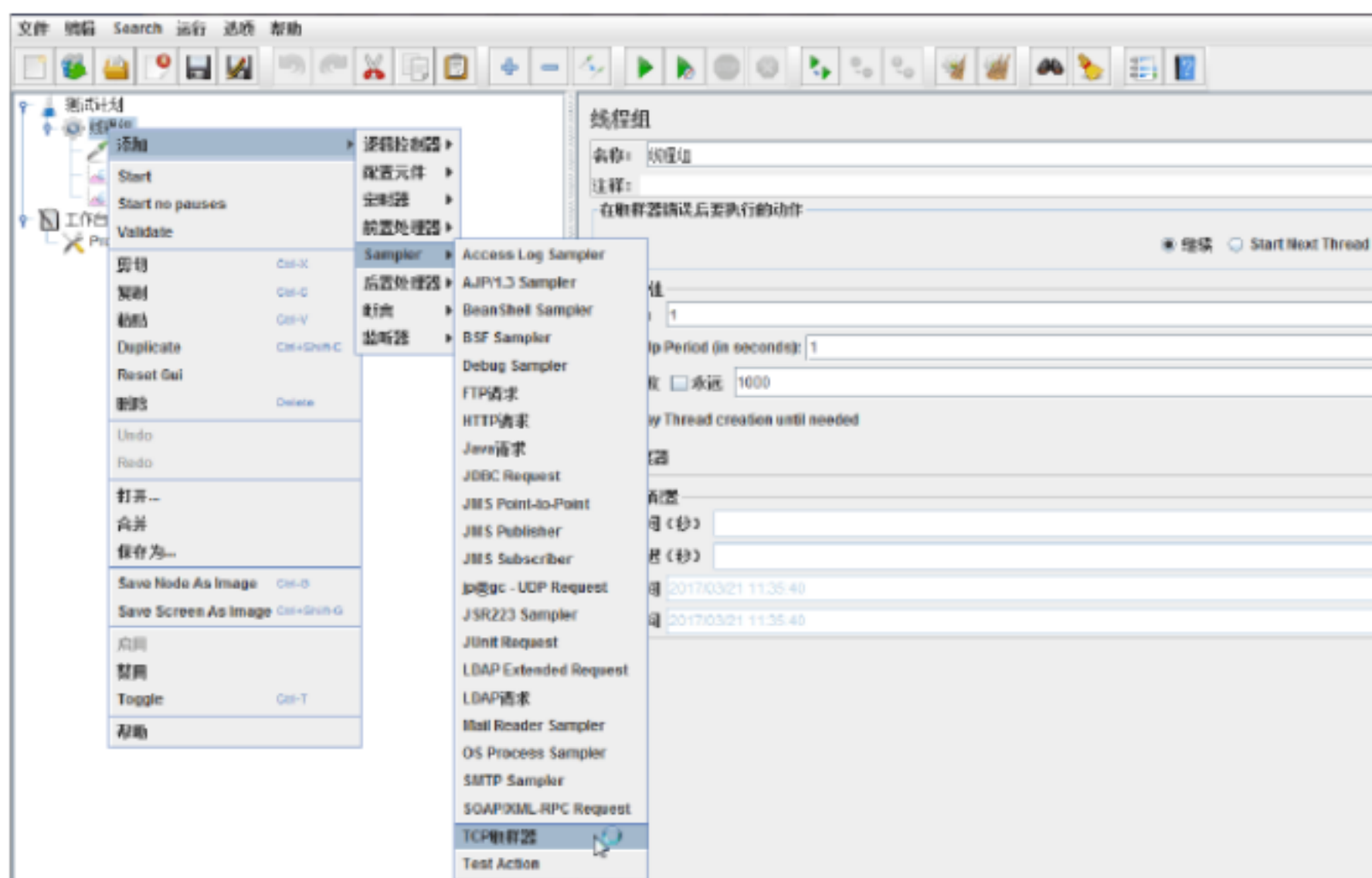
5.2.1. 添加线程组

右键“测试计划” -> “添加” -> “Threads(Users)” -> “线程组”




5.2.2. 添加 “ TCP取样器 ”

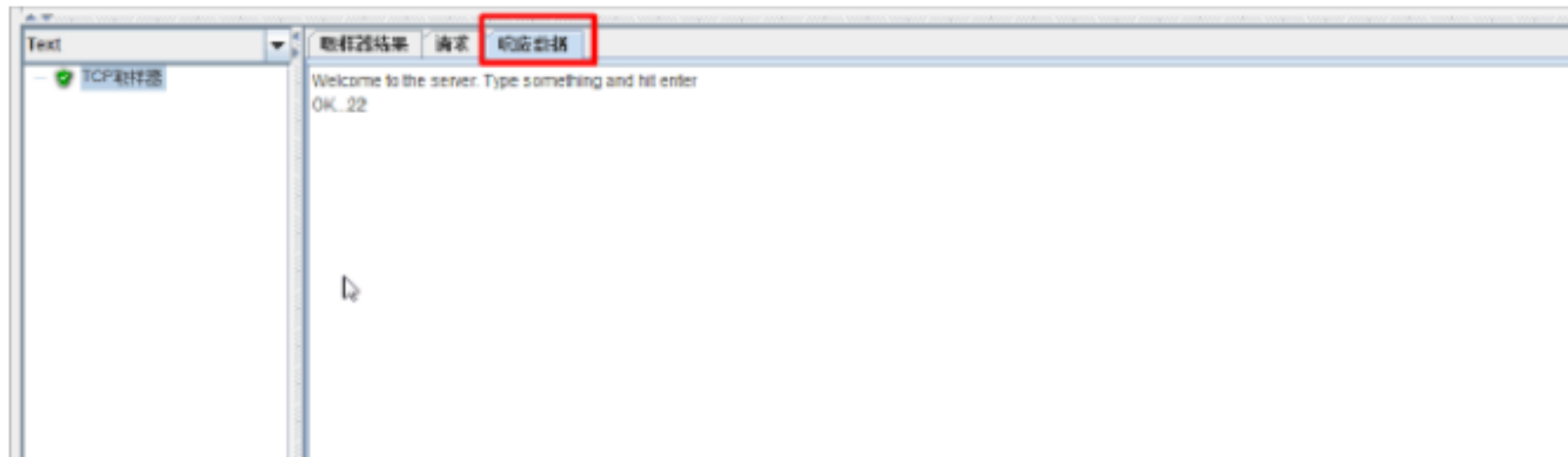
右键“线程组” -> “添加” -> “Sampler” -> “TCP取样器”



看下 “ TCP取样器 ” 的参数

5.2.4. 运行脚本

点击  按钮，运行脚本，察看结果



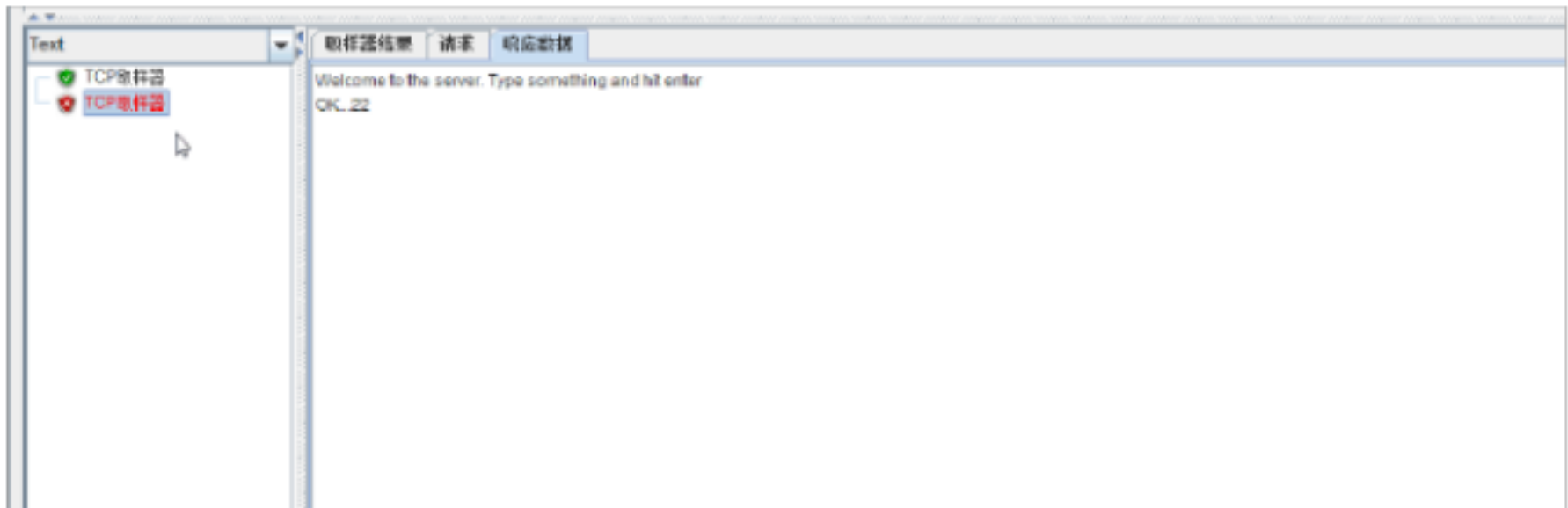
右边框里的就是 TCP服务器响应报文，脚本这就跑通了
服务器后台响应

```
[root@model opt]# python threadServer.py
Socket created
Socket bind complete
Socket now listening
Connected with 10.4.91.40:64838
Connected with 10.4.91.40:64898
Connected with 10.4.91.40:64907
Connected with 10.4.91.40:64910
Connected with 10.4.91.40:65090
Connected with 10.4.91.40:65093
Connected with 10.4.91.40:65177
Connected with 10.4.91.40:65186
Connected with 10.4.91.40:65383
Connected with 10.4.91.40:65405
Connected with 10.4.91.40:49836
Connected with 10.4.91.40:50462
```

注：这里对 EOL 特殊说明一下，第一次运行脚本，不知道这个 EOL 填写什么，可以先运行脚本，然后手动停止运行脚本，如下



然后见到如下结果



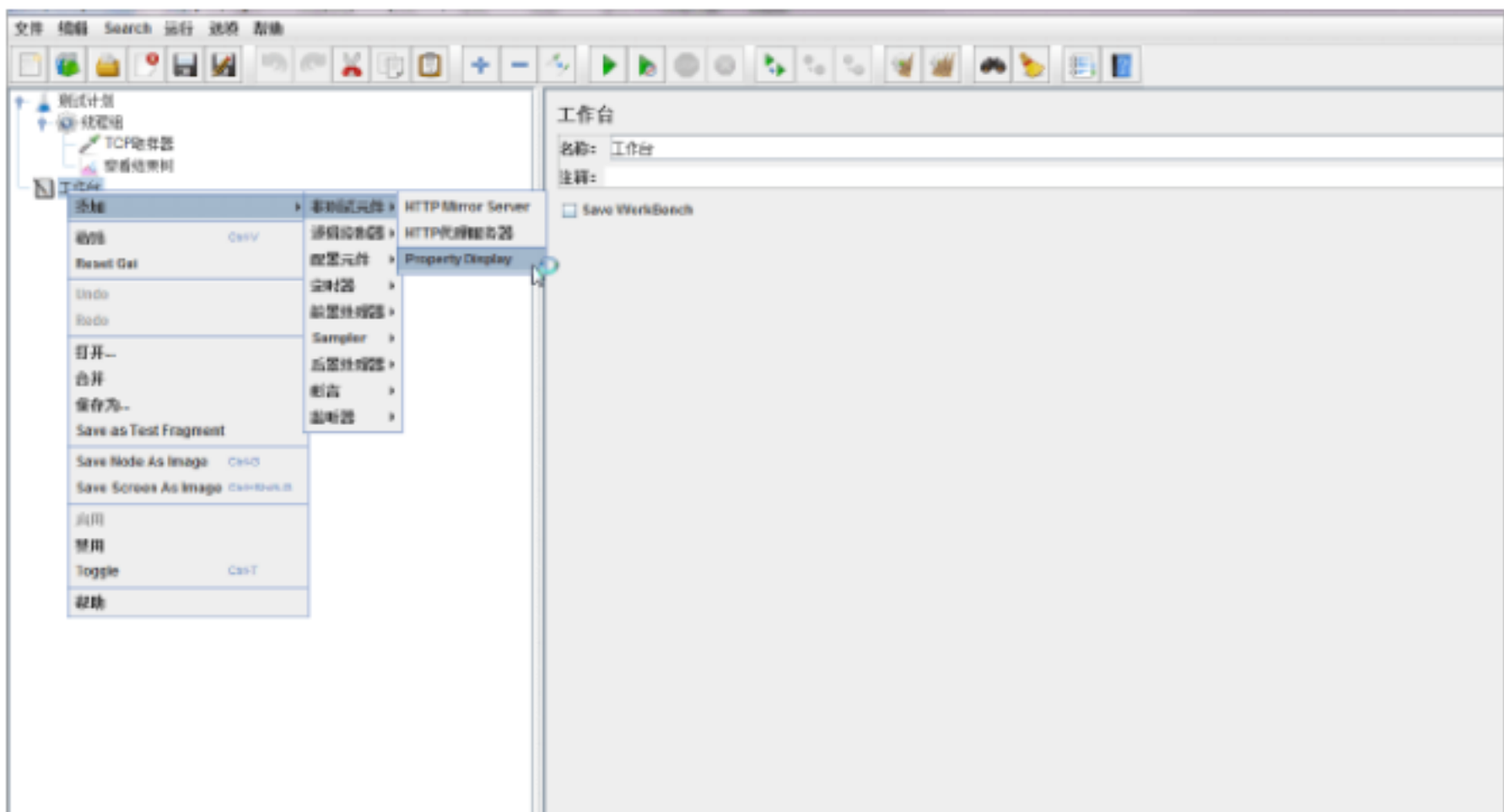
脚本执行失败，但是我们仍然获得了响应报文，将最后一个字符 ‘ 2 ’ 转换为 byte 值就是 50，所以之前的 EOL填入 50。

5.2.5. 发送 16 进制字节流报文

上面介绍的都是工具默认的文本报文， Jmeter 同样支持发送 16 进制报文，我们只需要修改一个参数值就好了。

修改 Jmeter 参数

右键 “ 工作台 ” -> “ 非测试元件 ” -> “ Property Display ”



找到参数 tcp.handler ，如下



udpServer.py

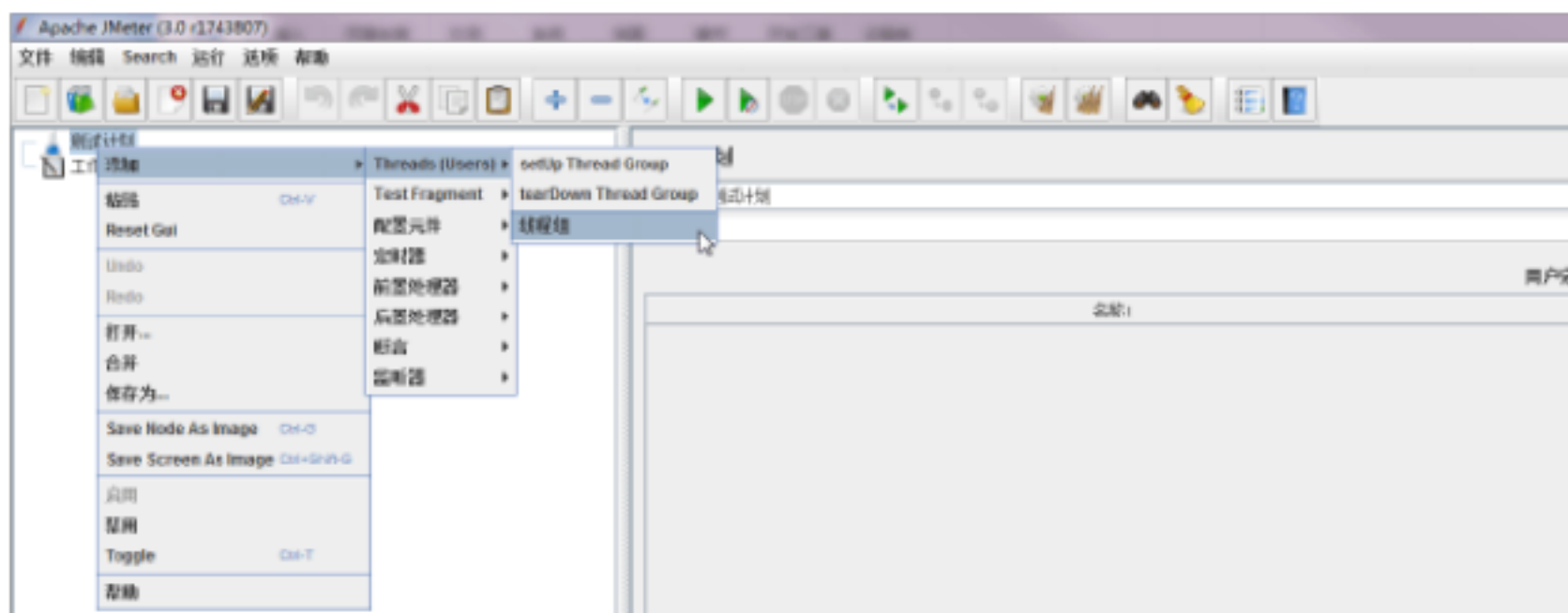
运行该服务器的方法同 5.1 一样，找一个 Linux 服务器，上传该 python 文件，执行命令“python udpServer.py”，该服务器端口为 54321，IP 为 Linux 服务器的 IP 地址。

```
root@ubuntu:/opt# python udpServer.py
start to recieve data
```

6.2. 创建 UDP 测试脚本

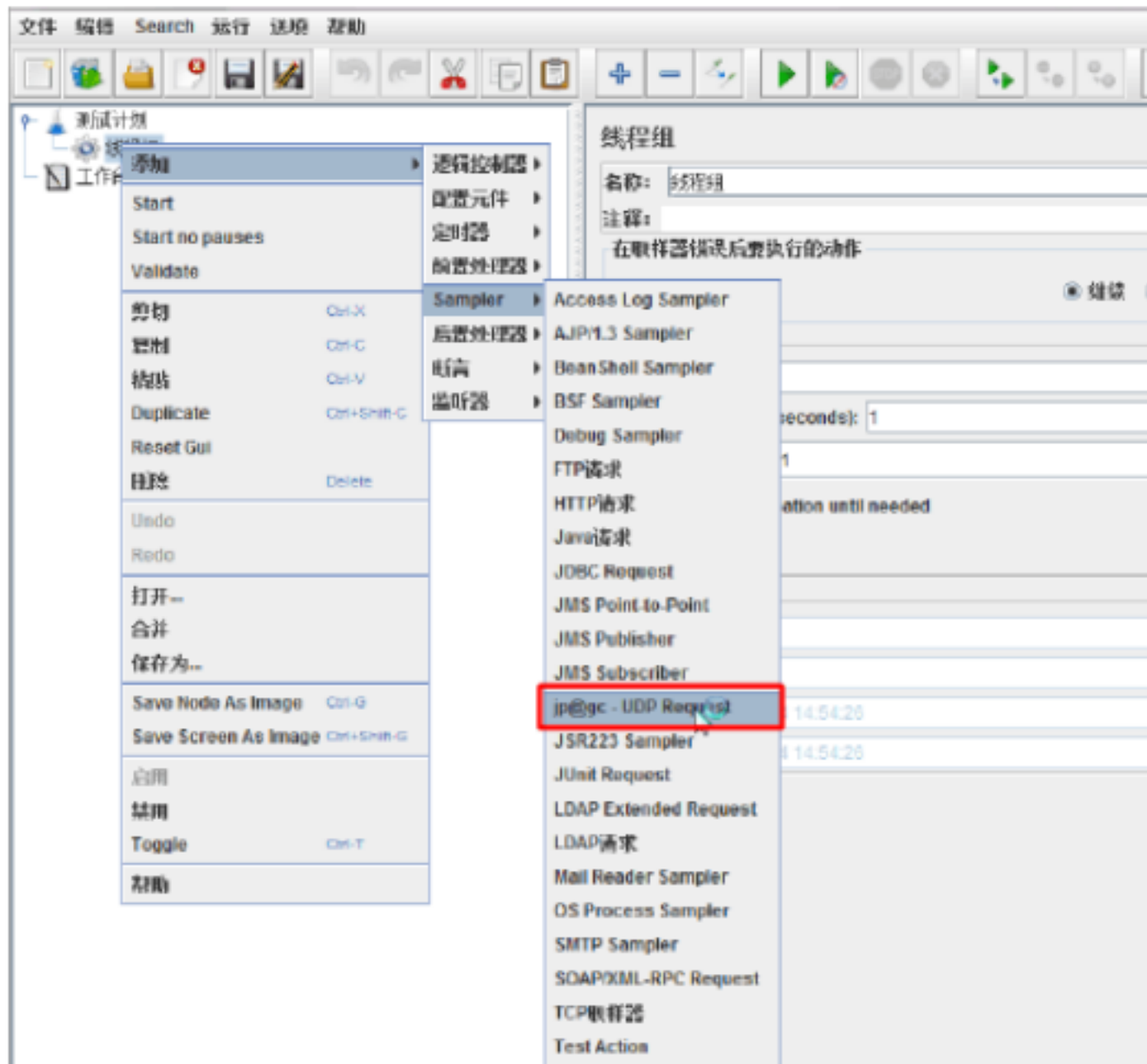
6.2.1. 添加线程组

右键“测试计划” -> “添加” -> “Threads(Users)” -> “线程组”

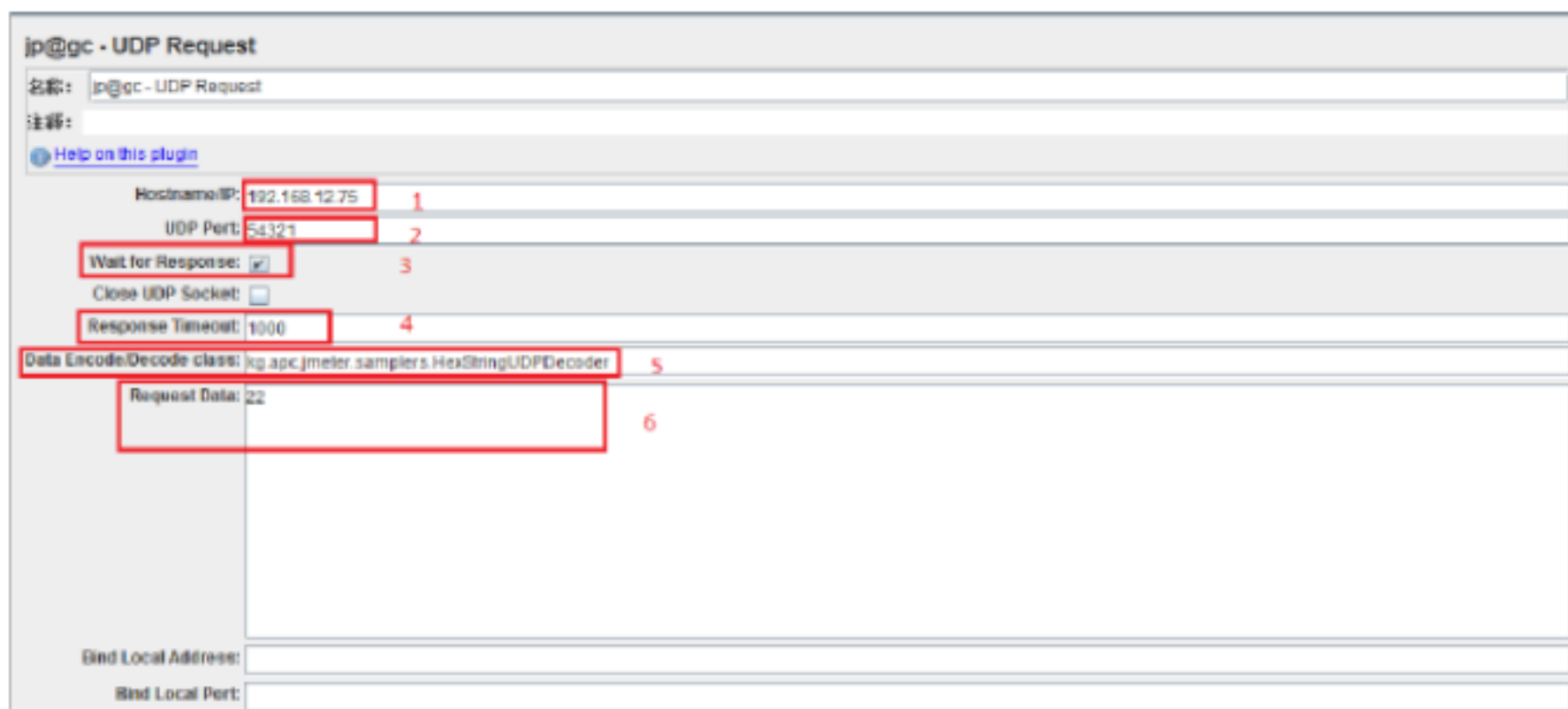


6.2.2. 添加“jp@gc - UDP Request”

右键“线程组” -> “添加” -> “Sampler” -> “jp@gc - UDP Request”



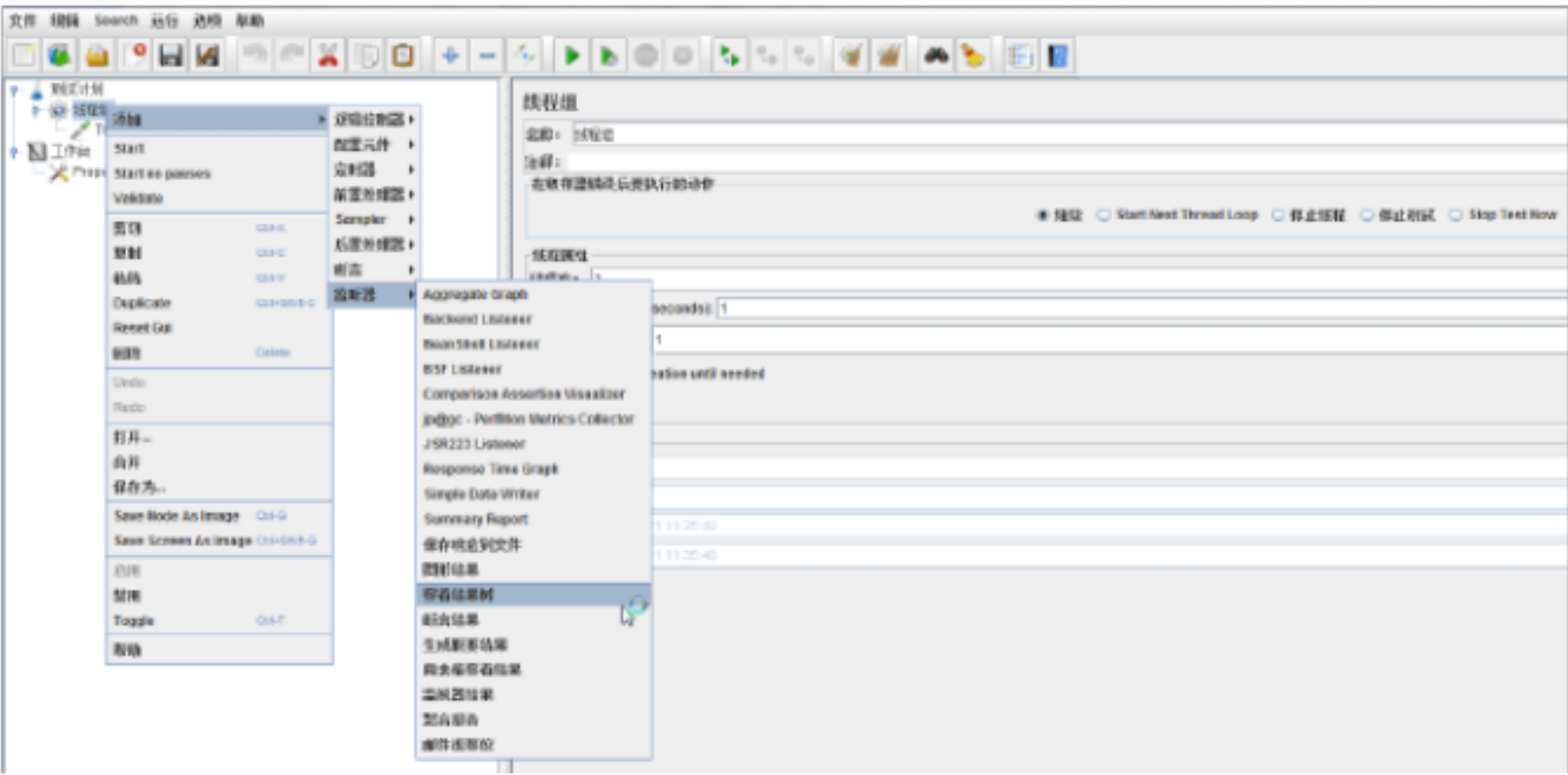
配置 “ jp@gc - UDP Request ”




- 1.服务器 IP 地址
- 2.UDP 服务器端口号
- 3.勾选等待返回报文，不勾选则只管发送报文，不接收
- 4.等待返回报文的超时时间
- 5.发送报文的编码格式，这里可以手动编辑
kg.apc.jmeter.samplers.HexStringUDPEncoder 直接发送 16 进制数据， HEX-encoded
kg.apc.jmeter.samplers.UDPSampler 填写字符串
kg.apc.jmeter.samplers.DNSJavaDecoder dns 解析填写格式如 : www.6san.com. A IN
kg.apc.jmeter.samplers.UDPTrafficDecoder 接口可以自定义编码 /解码
- 6.需要发送的报文

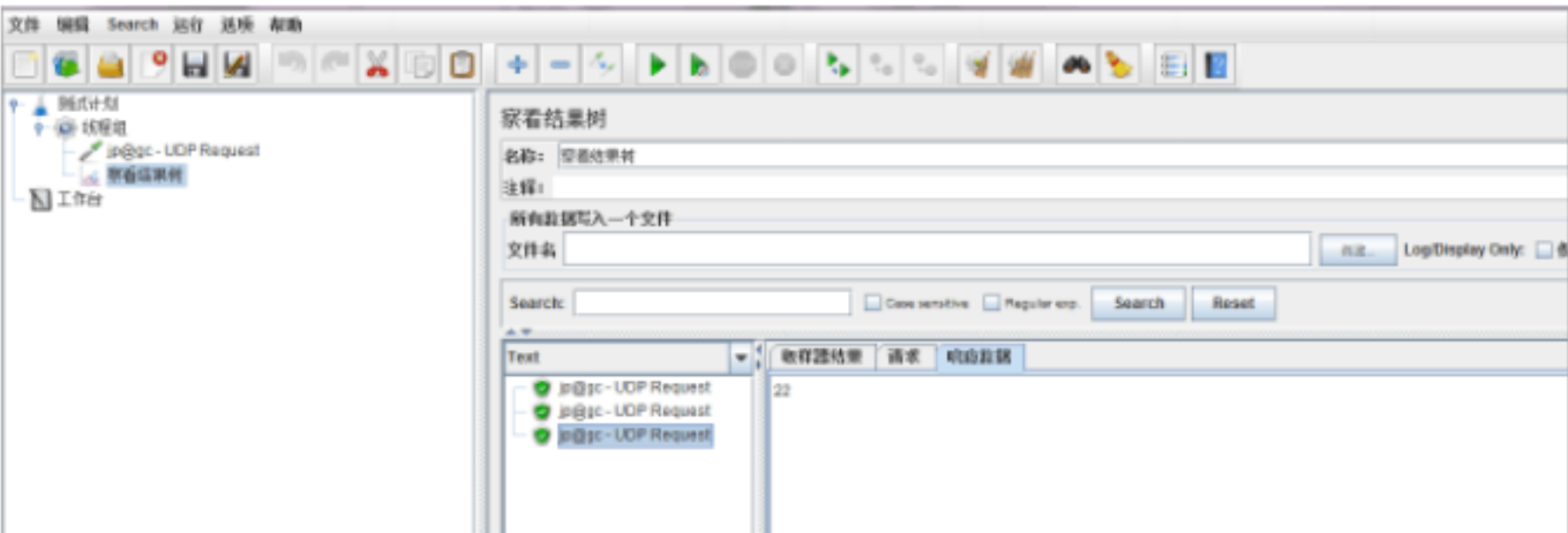
6.2.3. 添加监听器

右键“线程组” -> “添加” -> “监听器”，选择“察看结果树”



6.2.4. 运行脚本

点击  按钮，运行脚本，察看结果



这里看到， Jmeter 已经接收到了 UDP服务器的返回值
服务器后台也有简单日志打出来

```
root@ubuntu:/opt# python udpServer.py
start to recieve data
Got data from ('10.4.91.64', 53877) : "
Got data from ('10.4.91.64', 53879) : "
Got data from ('10.4.91.64', 53880) : "
Got data from ('10.4.91.64', 63615) : "
```

6.2.5. 设置 UDP 报文格式

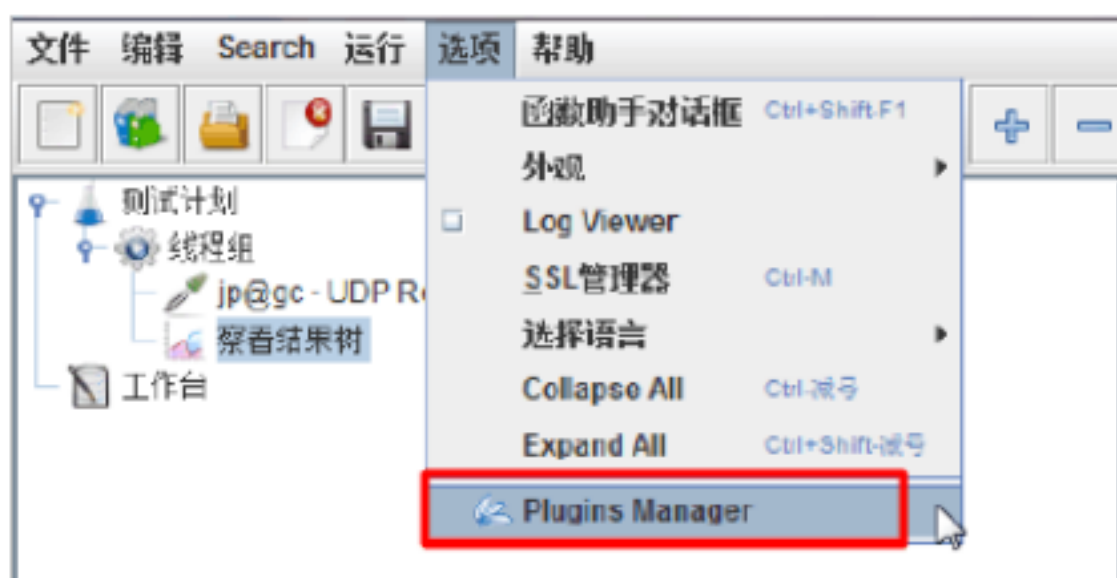
修改字段 “ kg.apc.jmeter.samplers.UDPSampler ” 的值，默认发送 16 进制字符流。

如果想发送字符串，则将该字段值改成 kg.apc.jmeter.samplers.UDPSampler

7. Jmeter 分布式部署

8. 配置 Jmeter 插件管理

1. 上官网 <https://jmeter-plugins.org/downloads/all/>
2. 下载插件管理软件 plugins-manager.jar
3. 然后将 plugins-manager.jar 放到 Jmeter 中的目录 lib/ext 下，重启 Jmeter
4. 可以看到“选项”下多了个“plugins manager”，如下图



5. 该插件能管理 Jmeter 插件，下载升级，如果要下载新插件，只需在“Available Plugins”中勾选，然后点击“Apply Changes and Restart Jmeter”，等待插件下载完，自动重启 Jmeter 即可。

