**实验1 D-Link DIR-645 溢出漏洞分析**

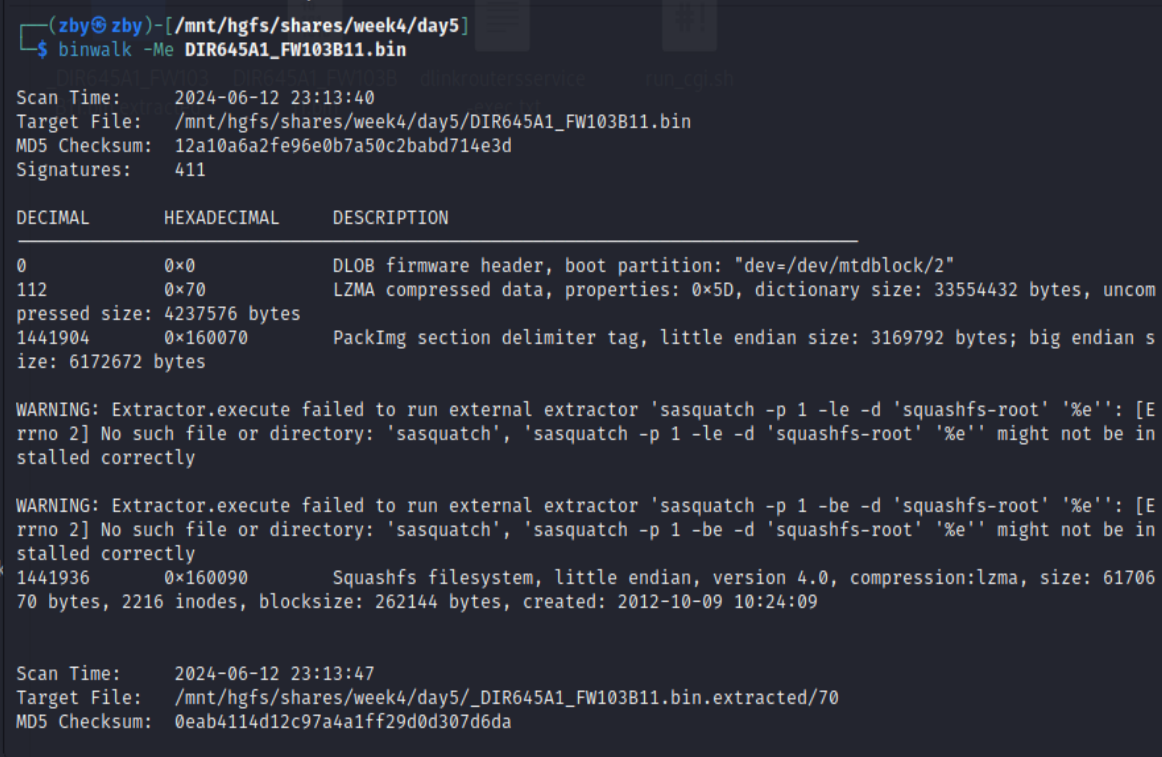
**实验目的：**

* + 学会栈溢出
  + 学会MIPS汇编

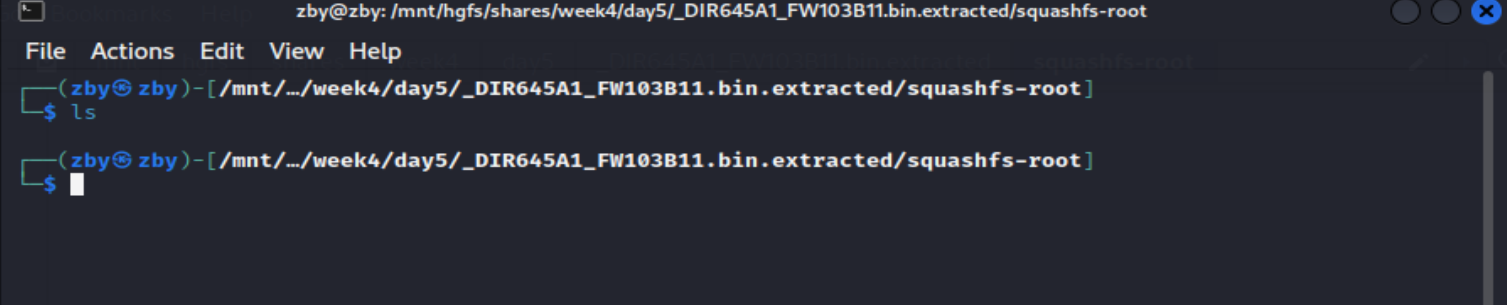
**实验题目：**

1. 完成DIR-645溢出漏洞，通过栈溢出漏洞利用，获取shell - 截图运行结果

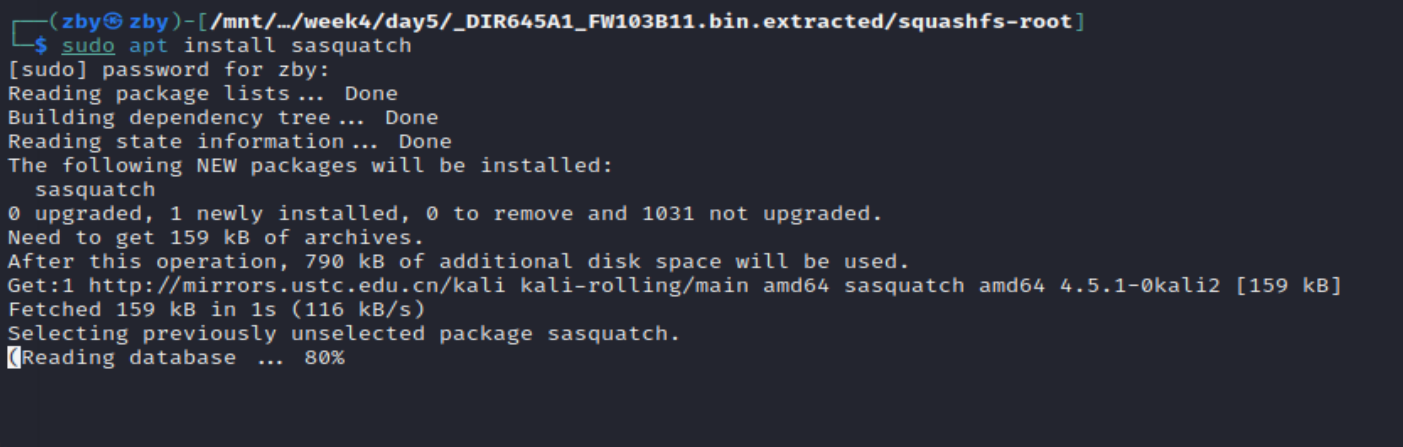
首先在kali虚拟机中使用binwalk提取得到固件根目录



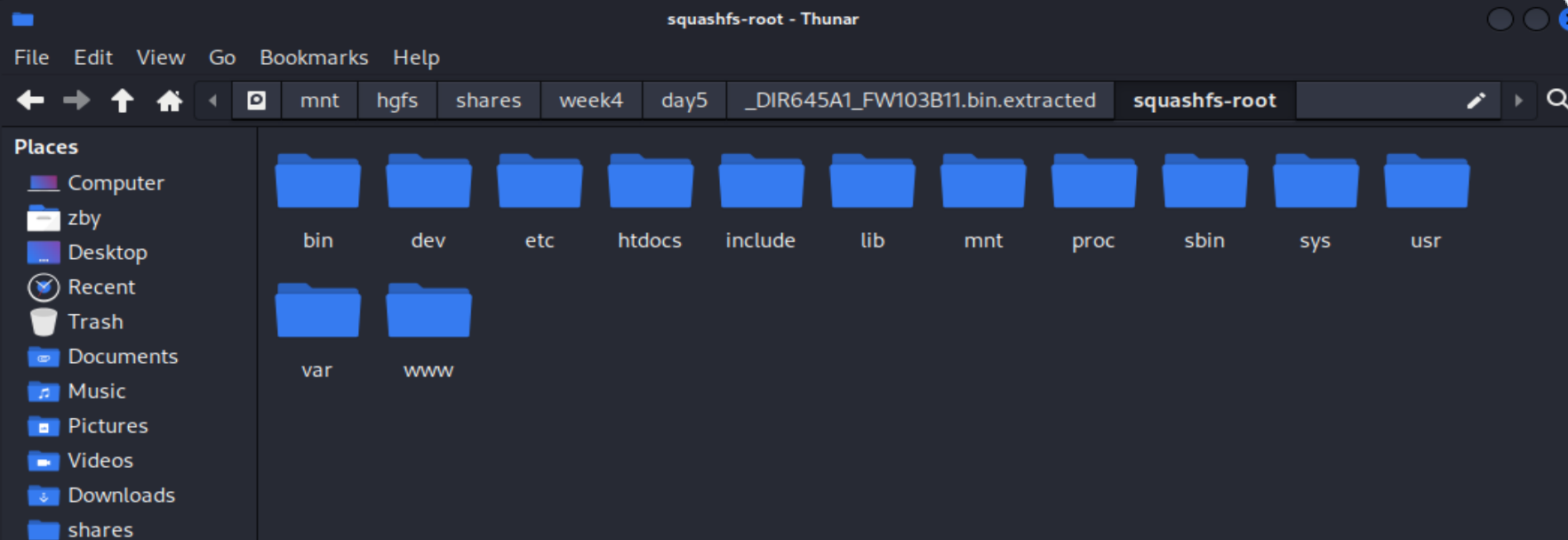
发现并没有提取到目标文件系统



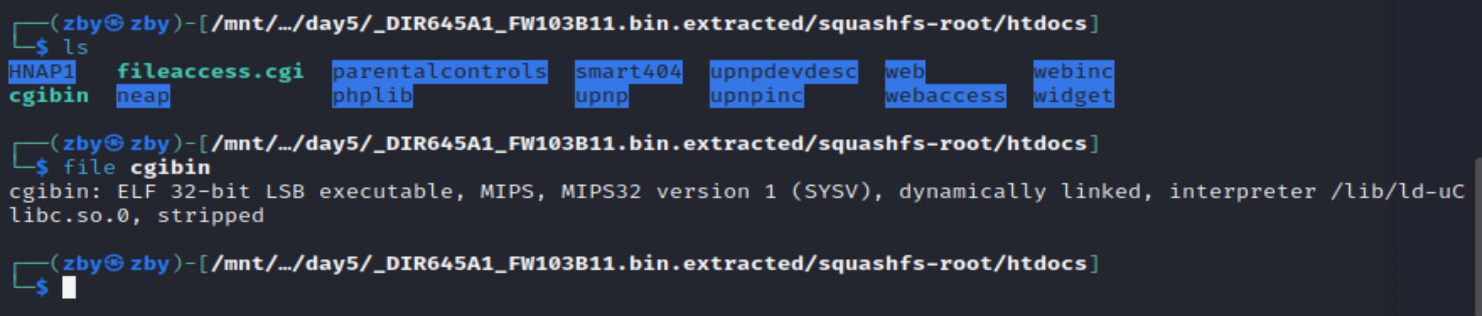
观察binwalk运行的信息可以发现是因为缺少sasquatch库导致，所以先下载sasquatch库



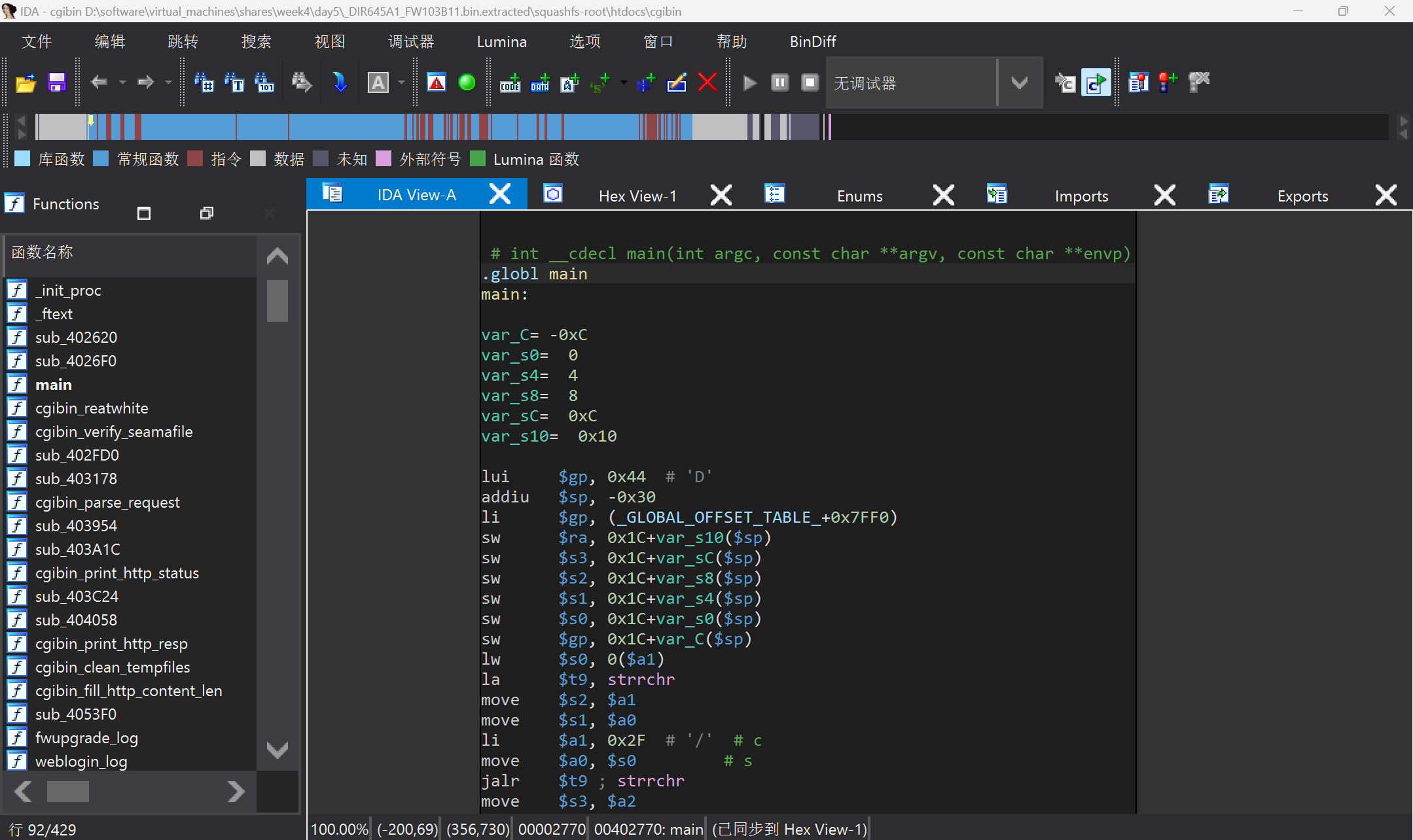
重新进行提取可以发现成功获取到目标文件系统



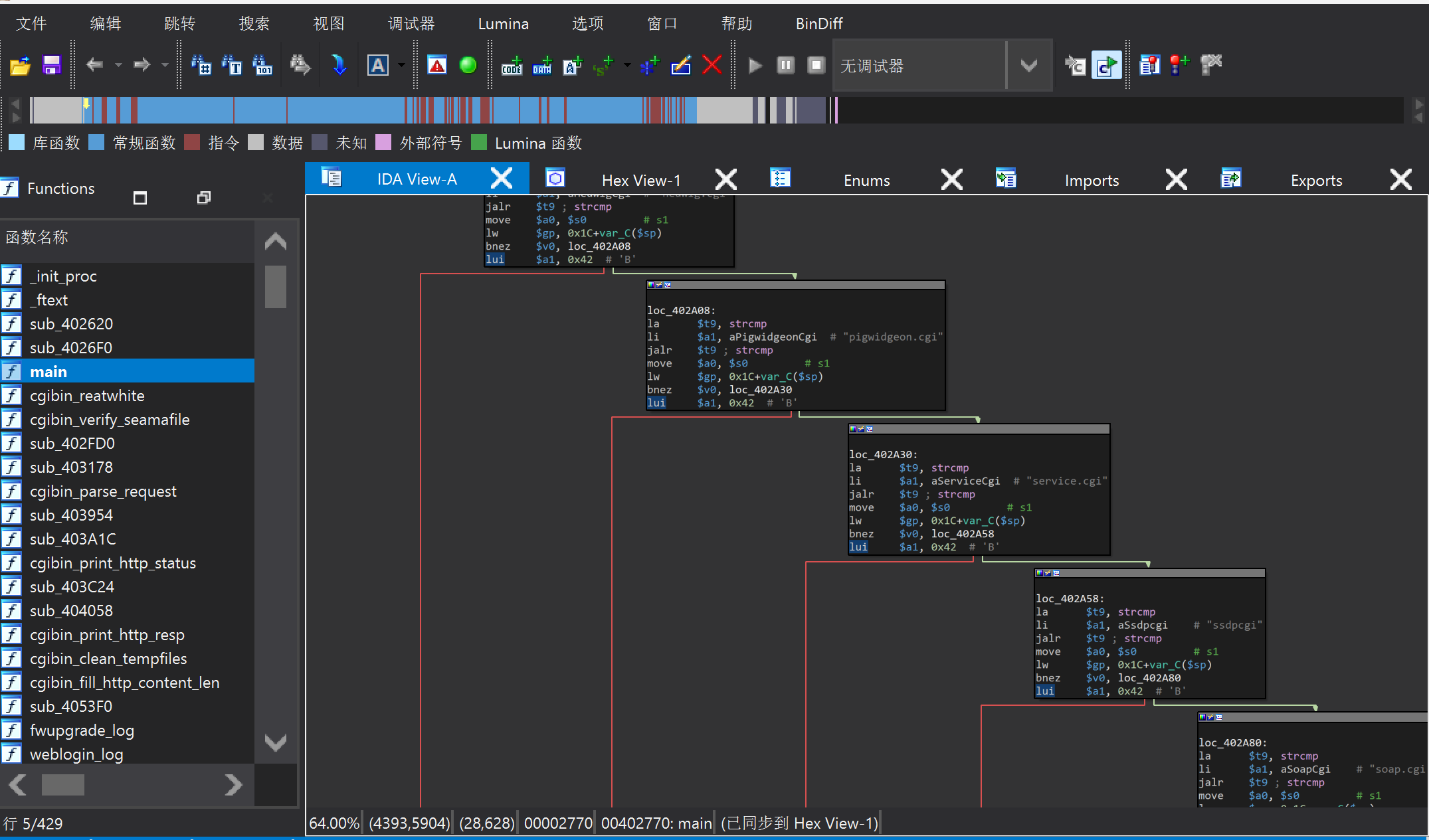
查看位于htdocs目录下的cgibin文件的基本信息，可以发现该文件是32位小端序可执行文件，指令架构为MIPS



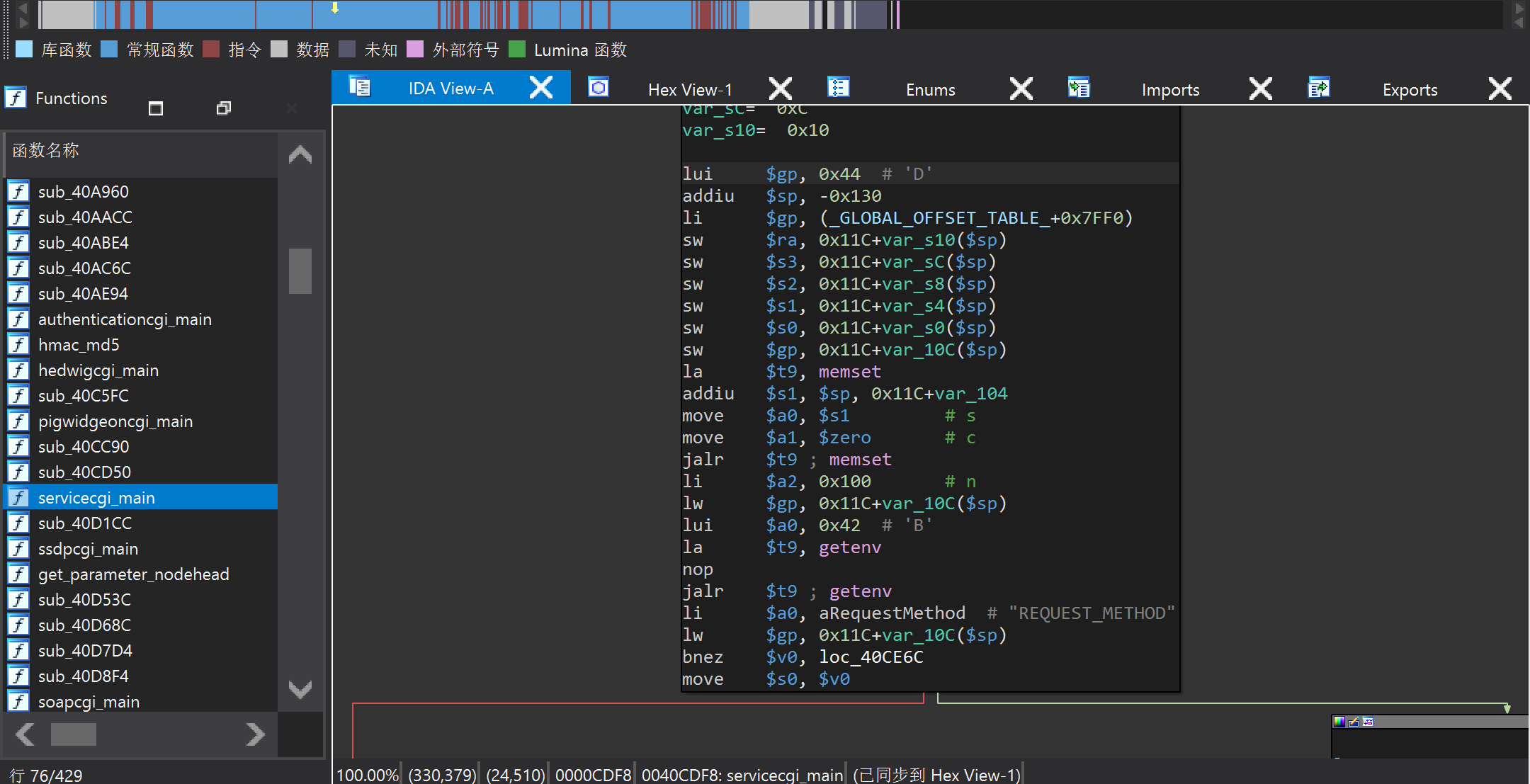
将该可执行文件放入到IDA中进行分析



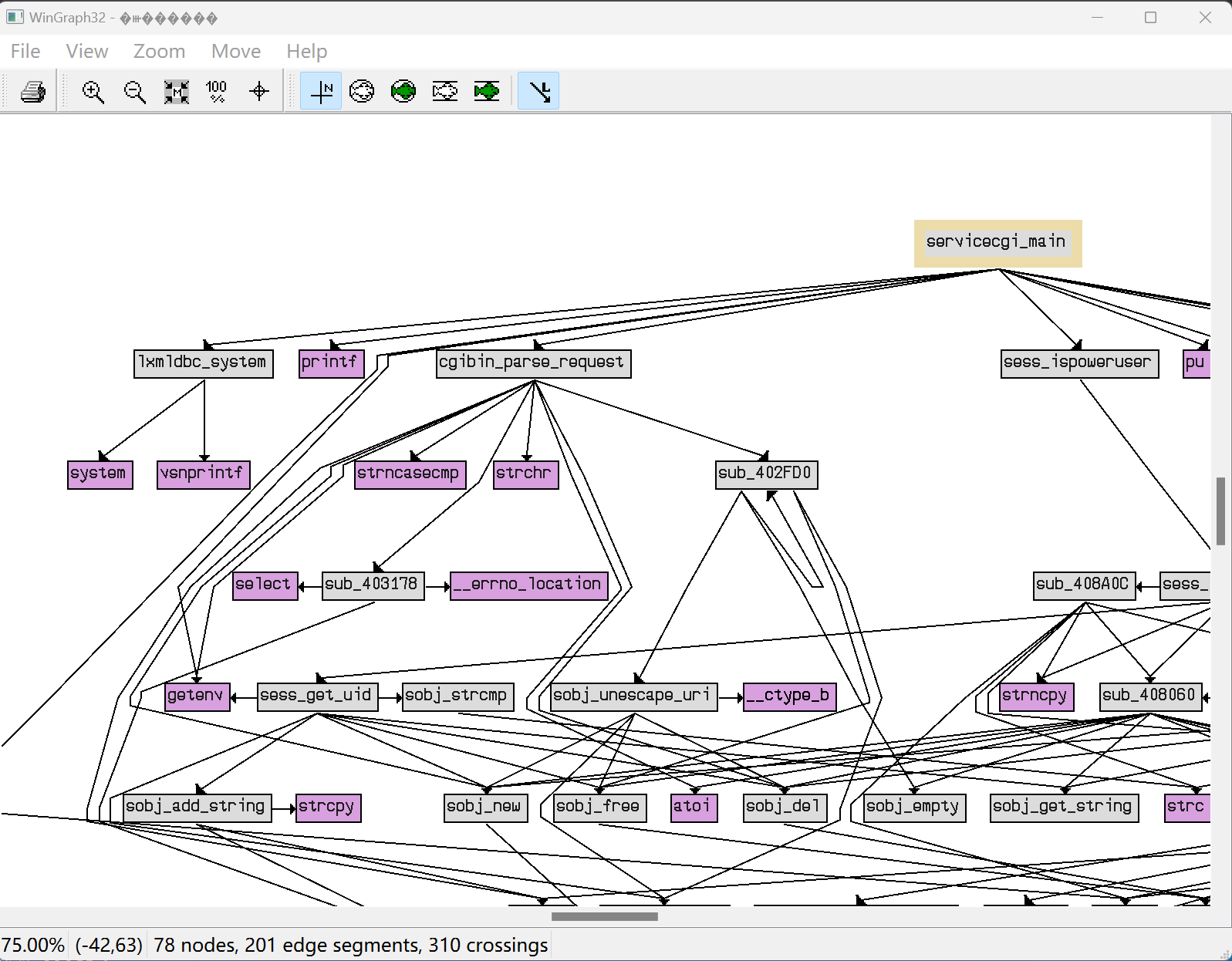
在汇编代码中查找可以发现service.cgi字段



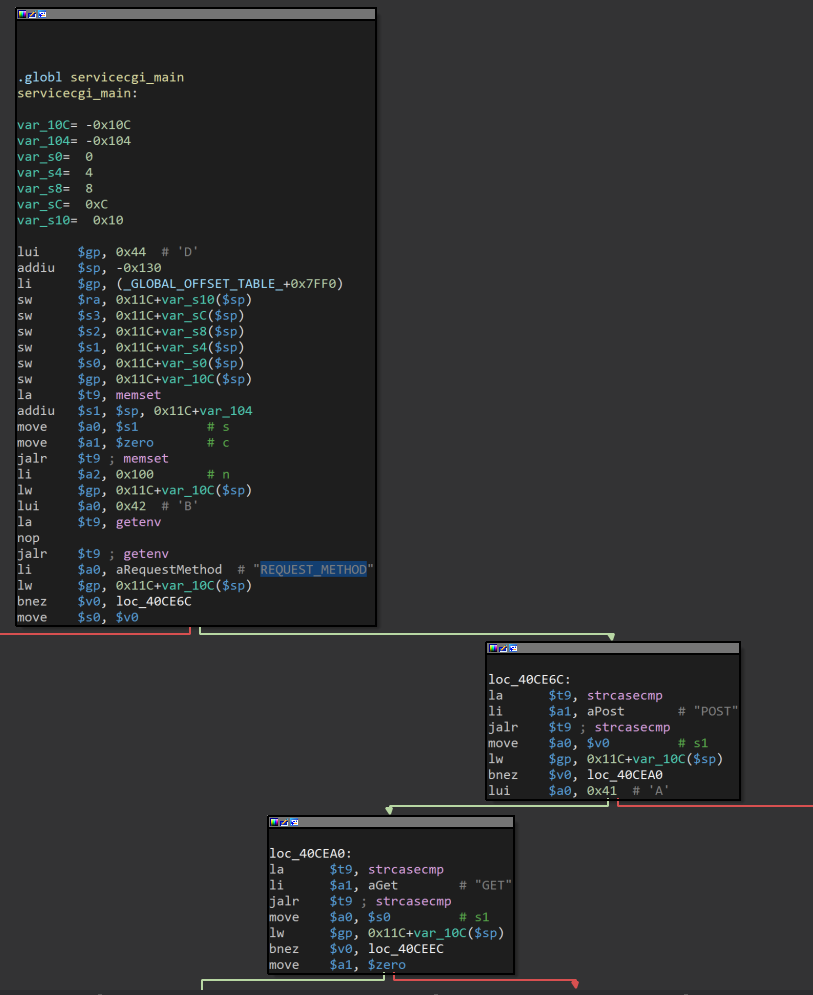
在函数名称列表中可以找到servicecgi\_main函数如下



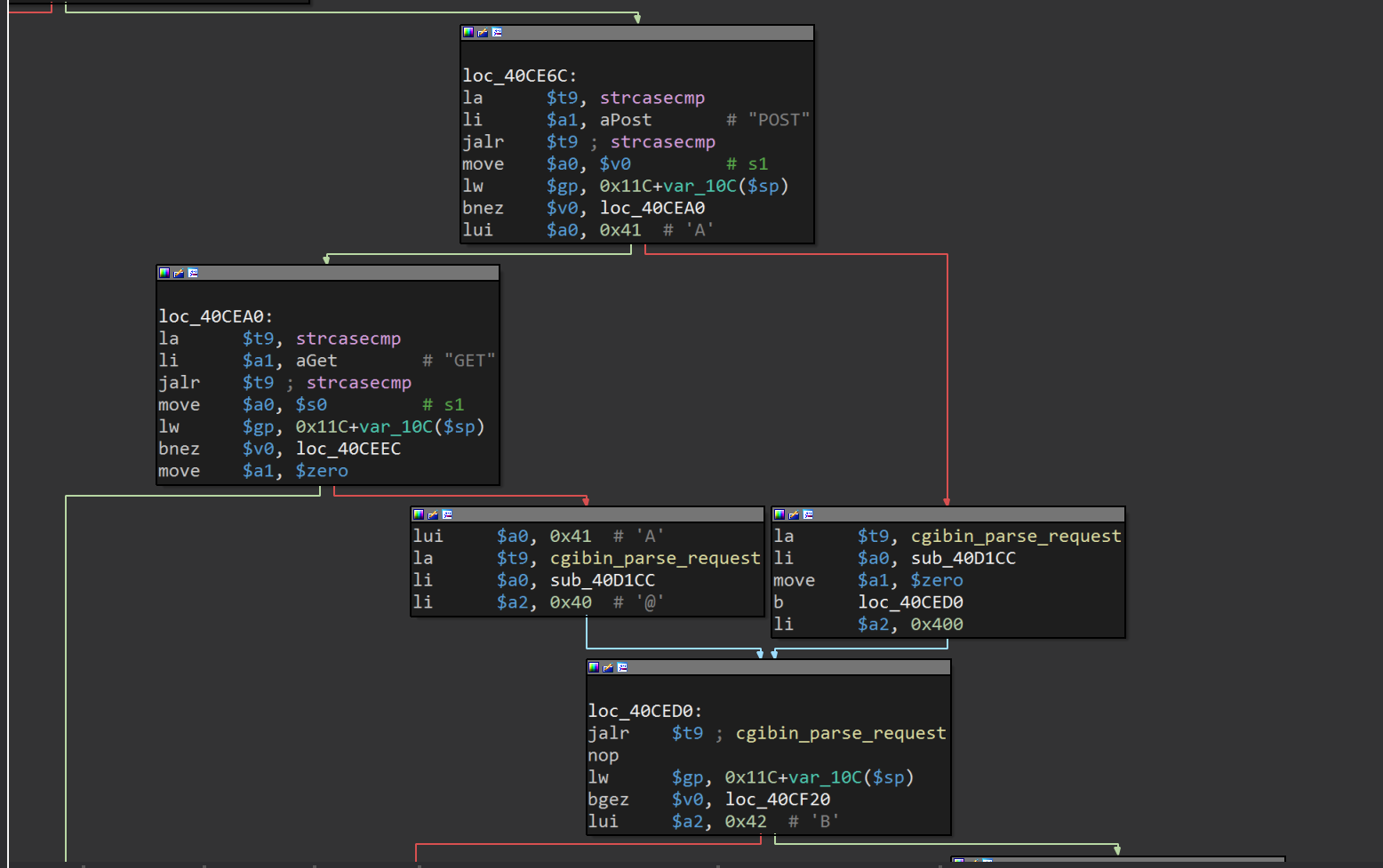
查看该函数的交叉引用情况如下可以发现servicecgi\_main函数会通过lxmldbc\_system调用到system函数



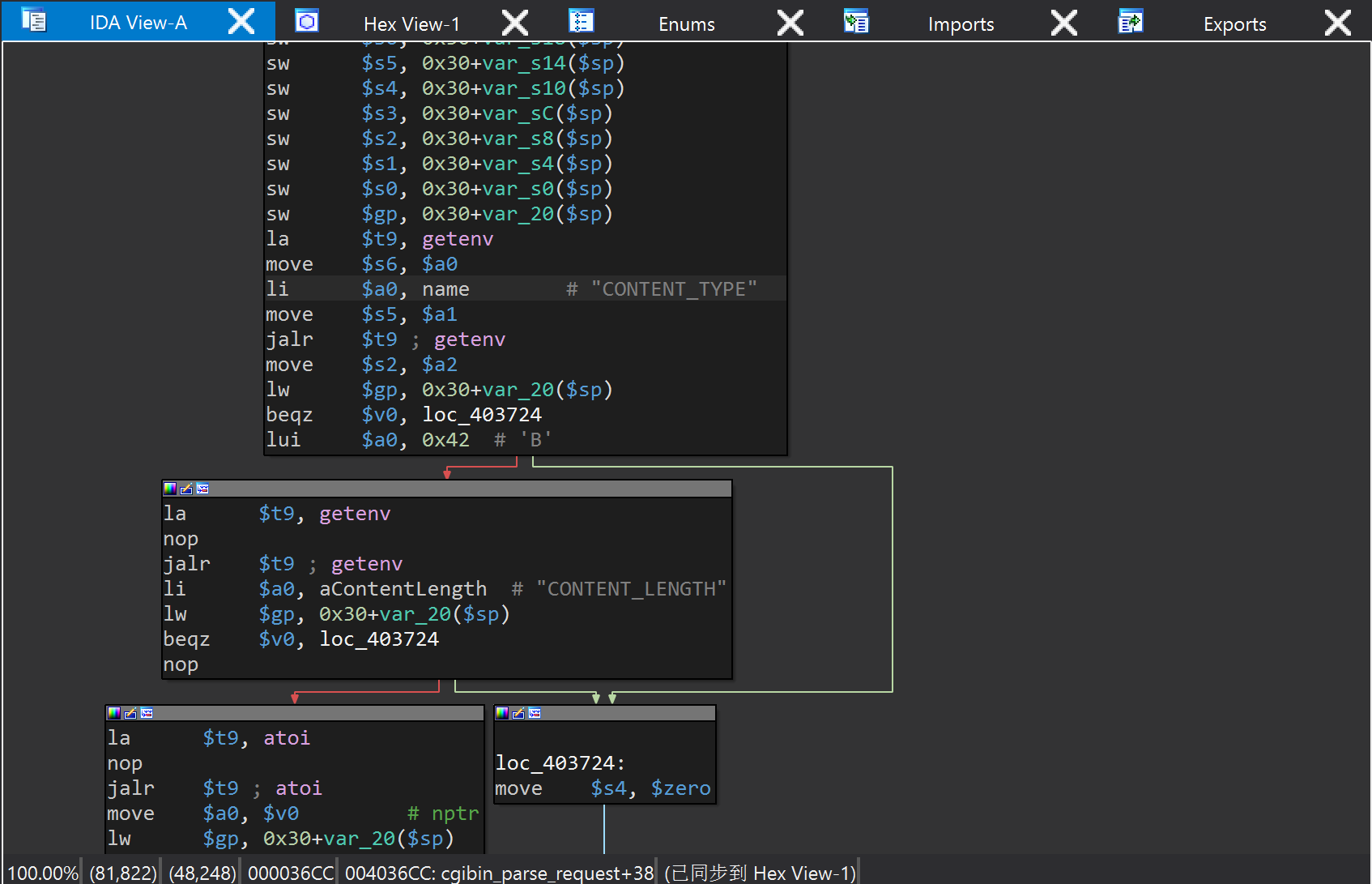
查看servicecgi\_main函数的汇编代码如下，可以看到程序会调用REQUEST METHOD请求方法来处理get和post请求，但是会同时进入到loc\_40CE6C函数中。



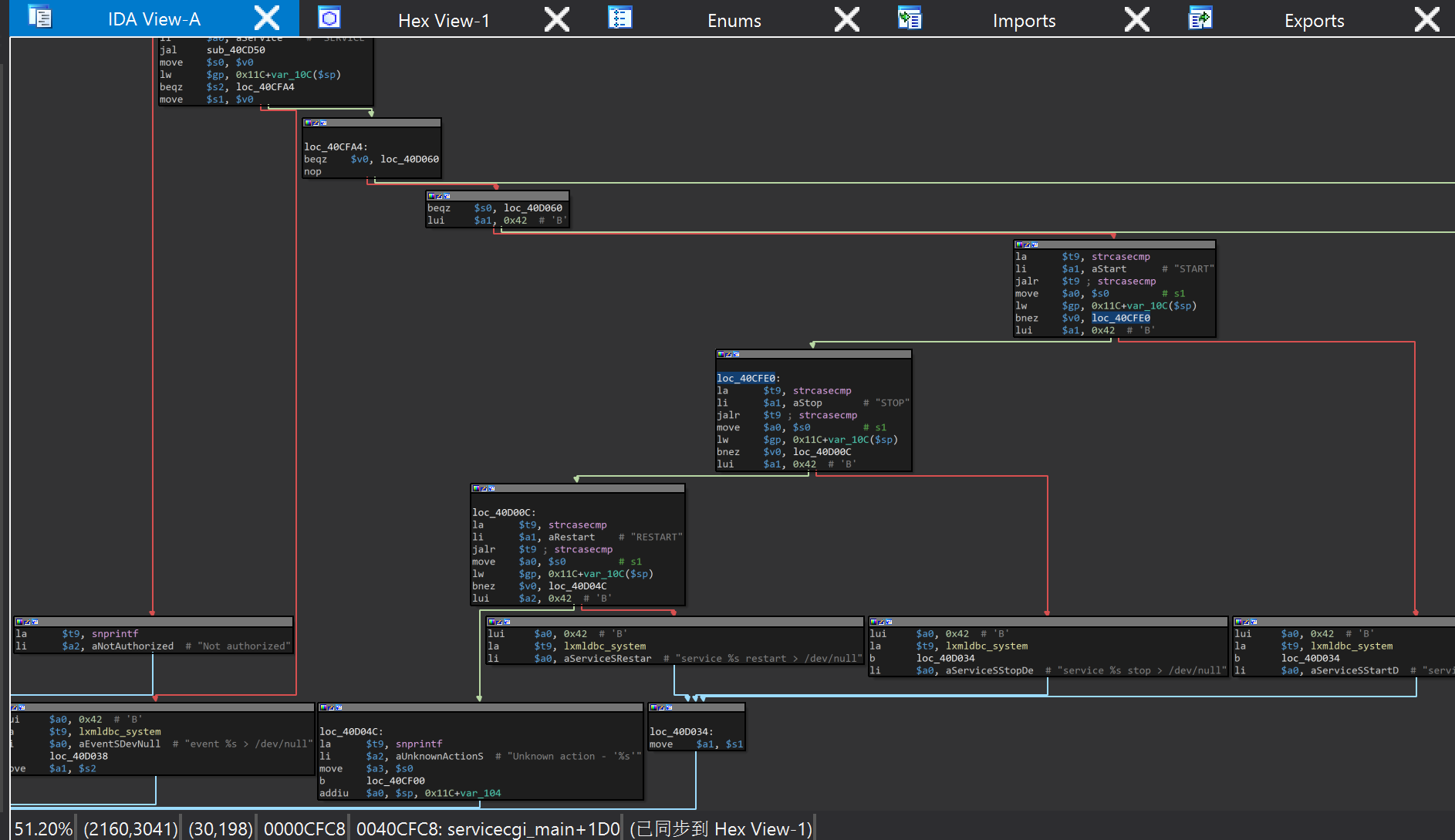
查看loc\_40CE6C函数的内容如下，可以发现两个请求运行路径基本相同



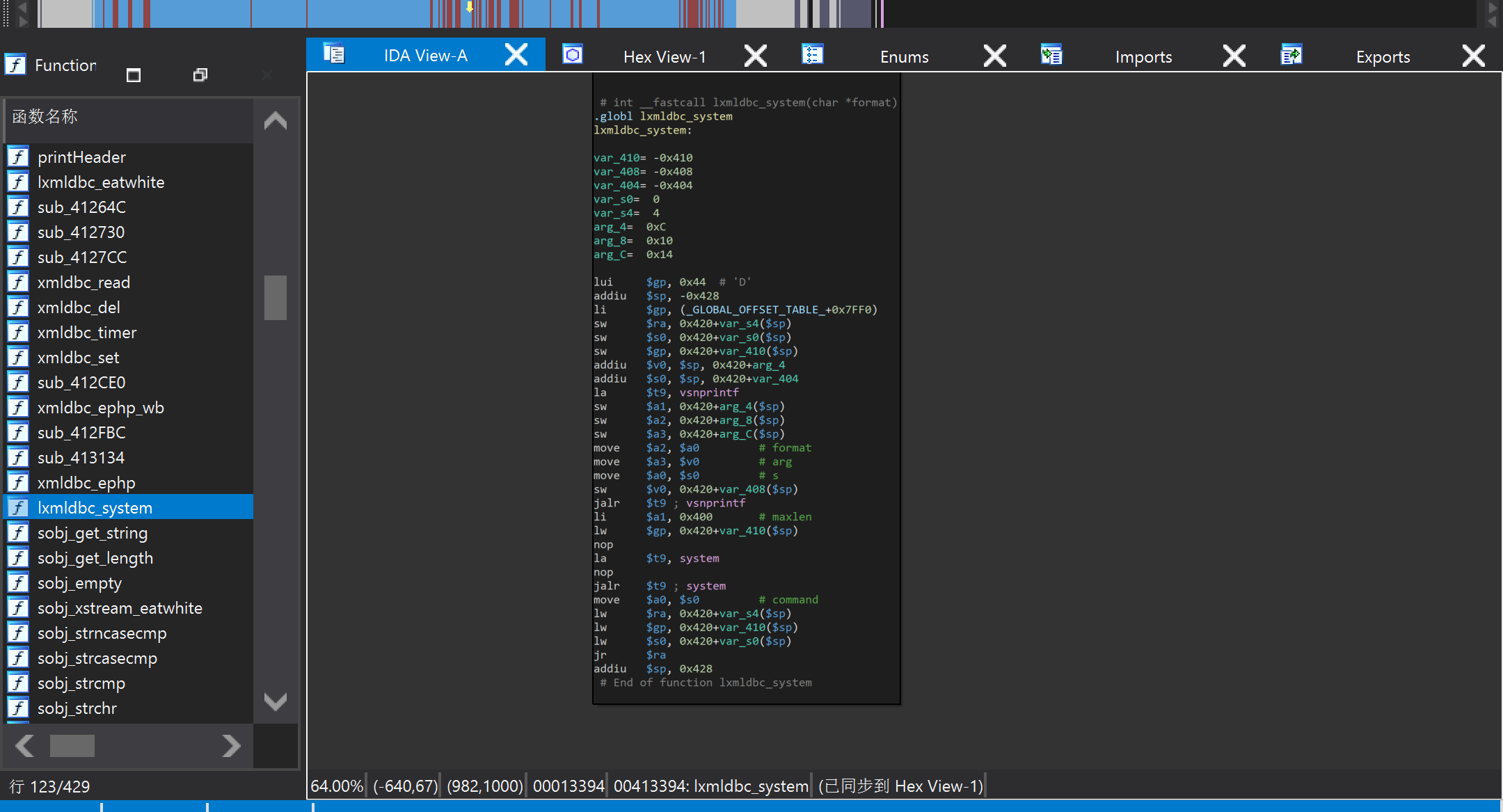
因为两个处理函数均会调用cgibin\_parse\_request函数，在IDA中查看该函数的内容，可以发现该函数会对CONTENT\_TYPE和CONTENT\_LENGTH进行解析



在该函数中对参数进行解析完毕之后就会回到servicecgi\_main函数中，可以发现EVENT和SERVICE都会调用lxmldbc\_system函数



查看lxmldbc\_system函数可以发现该函数会调用system函数，同时该函数在调用system执行command时并没有进行命令过滤，我们可以在这里引发命令注入漏洞。

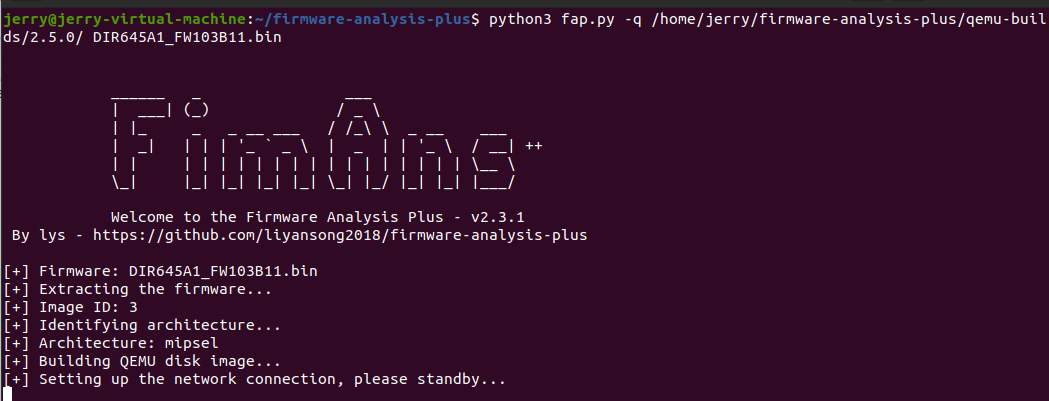


综上所述，我们可以在访问的时候加上EVENT或者利用已经加入ACTION字段的SERVICE，在阶段之后加上恶意指令即可，注入命令之后因为&在http请求中为请求域分隔符，所以需要将&转码为urlencode编码的%26。

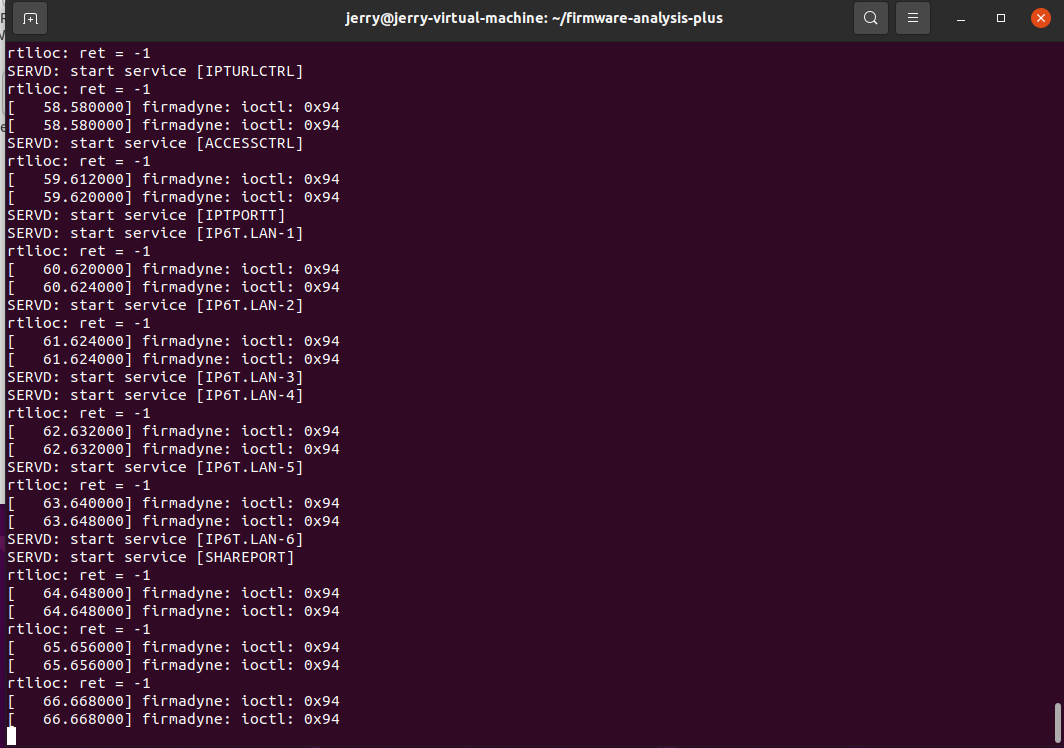
因为在本机中下载了firmware-analysis-toolkit的所有版本以及firmware-analysis-plus均不能运行该硬件，所以接下来采用firmware-analysis-plus的作者在github上发出的ubuntu20.04的镜像进行接下来的步骤

在下载好的镜像虚拟机中执行如下指令启动软件

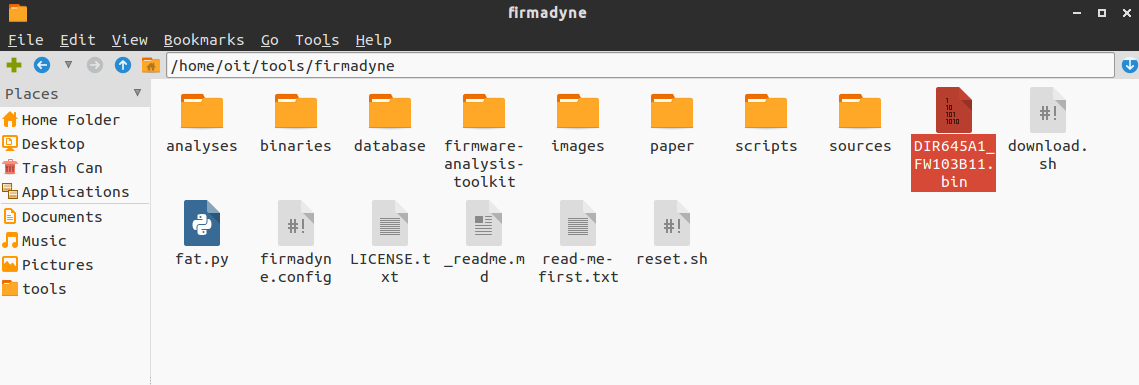
python3 fap.py -q /home/jerry/firmware-analysis-plus/qemu-builds/2.5.0/ DIR645A1\_FW103B11.bin



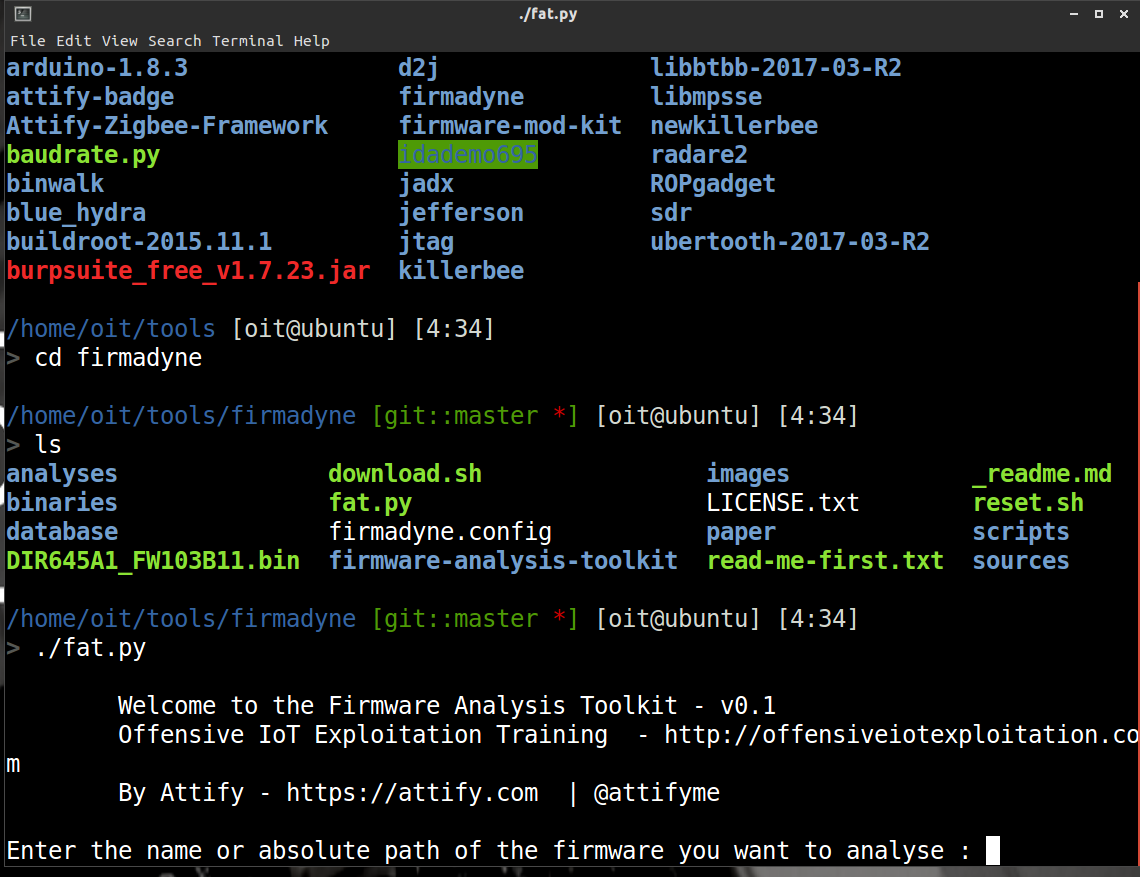
运行一段时间后发现工具仍然会在一个位置死循环



于是尝试使用affityOS,将固件放置到/home/oit/tools/firmadyne目录下



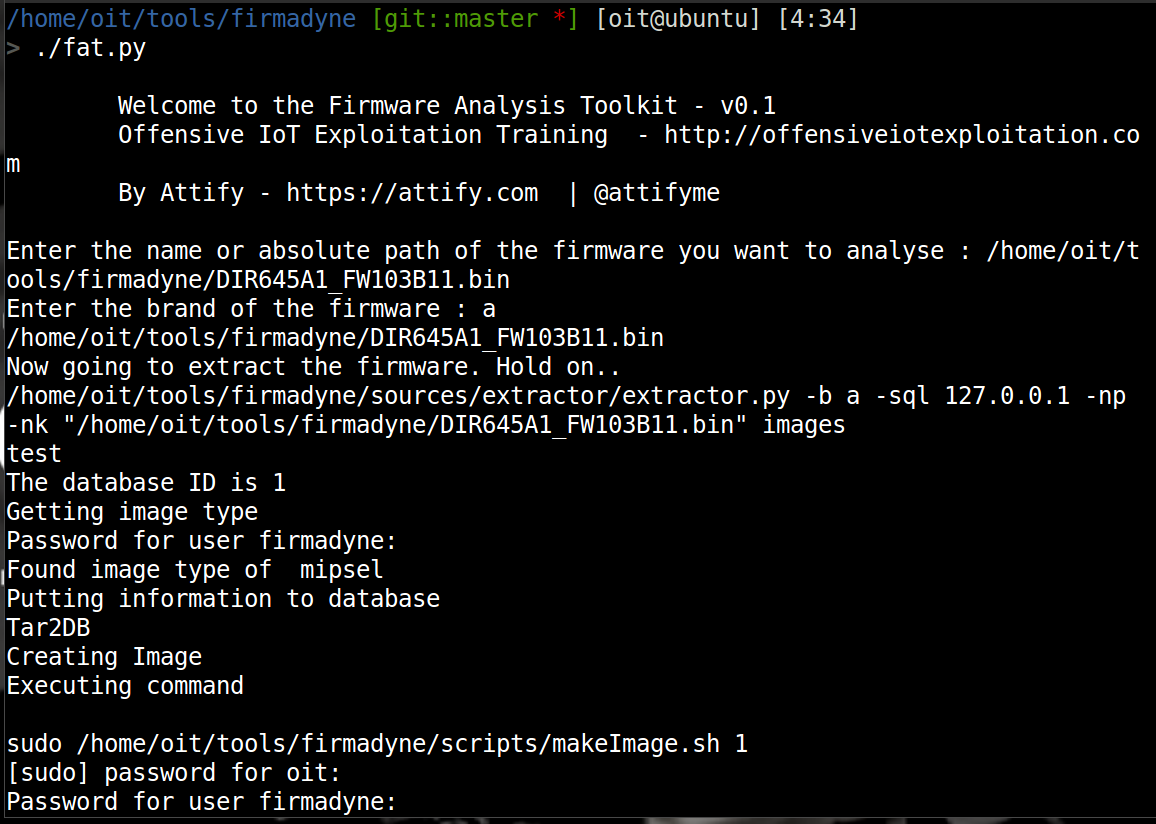
运行fat.py程序



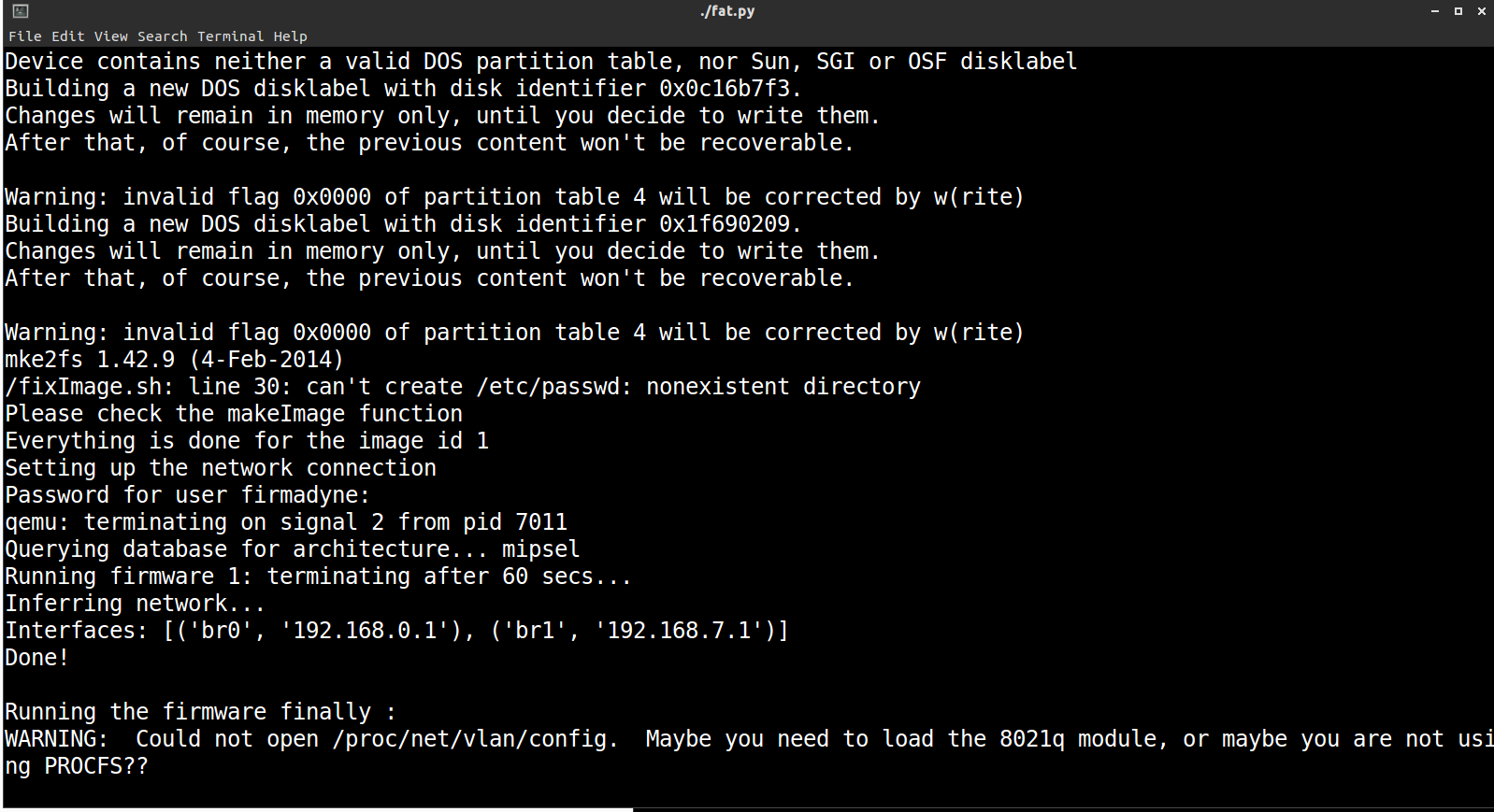
在文件名称选项中输入我们的固件路径/home/oit/tools/firmadyne/DIR645A1\_FW103B11.bin

固件品牌名随意输入a

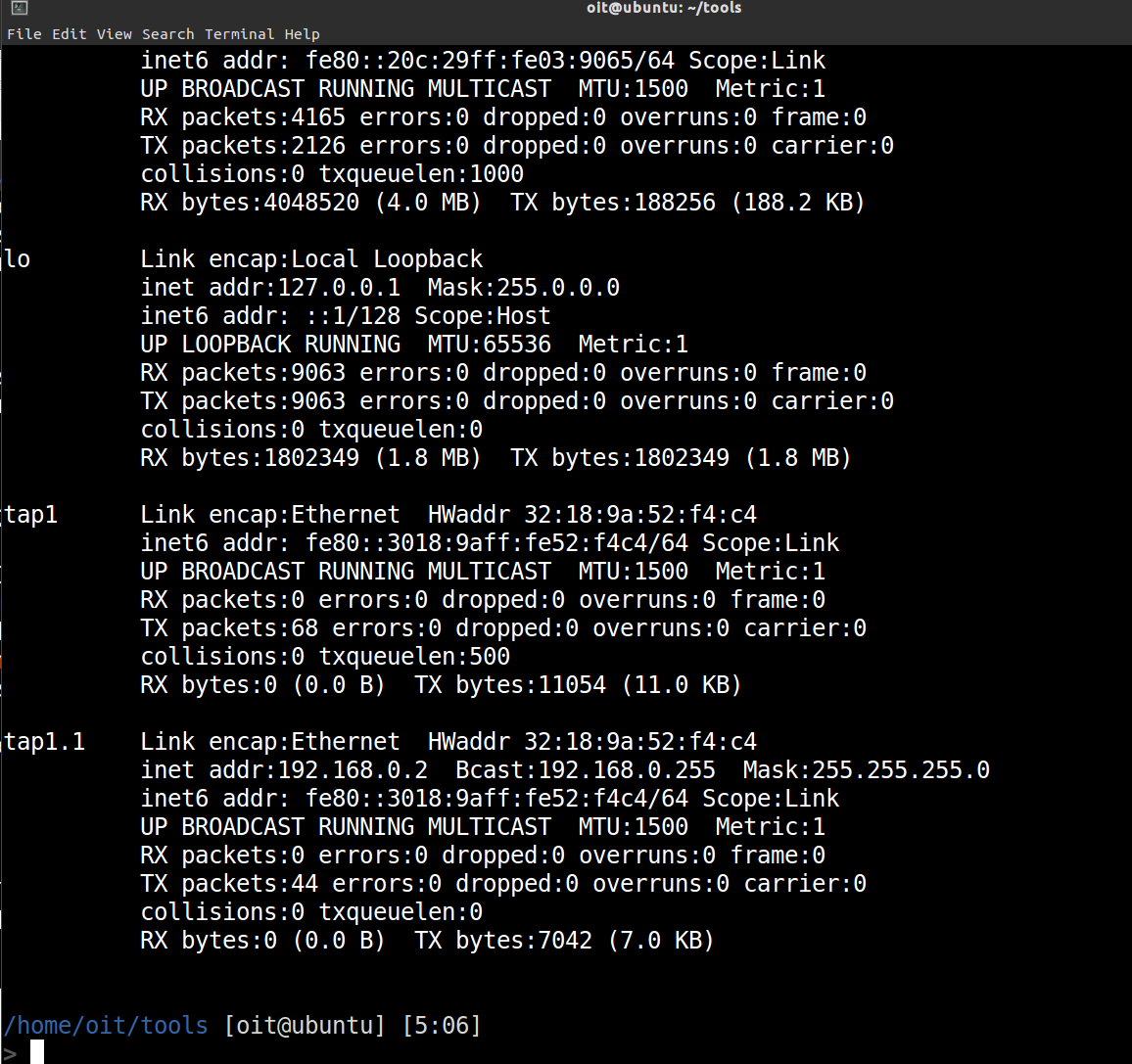
接下来在firmadyne的密码处输入firmadyne，在oit的密码处输入attify123



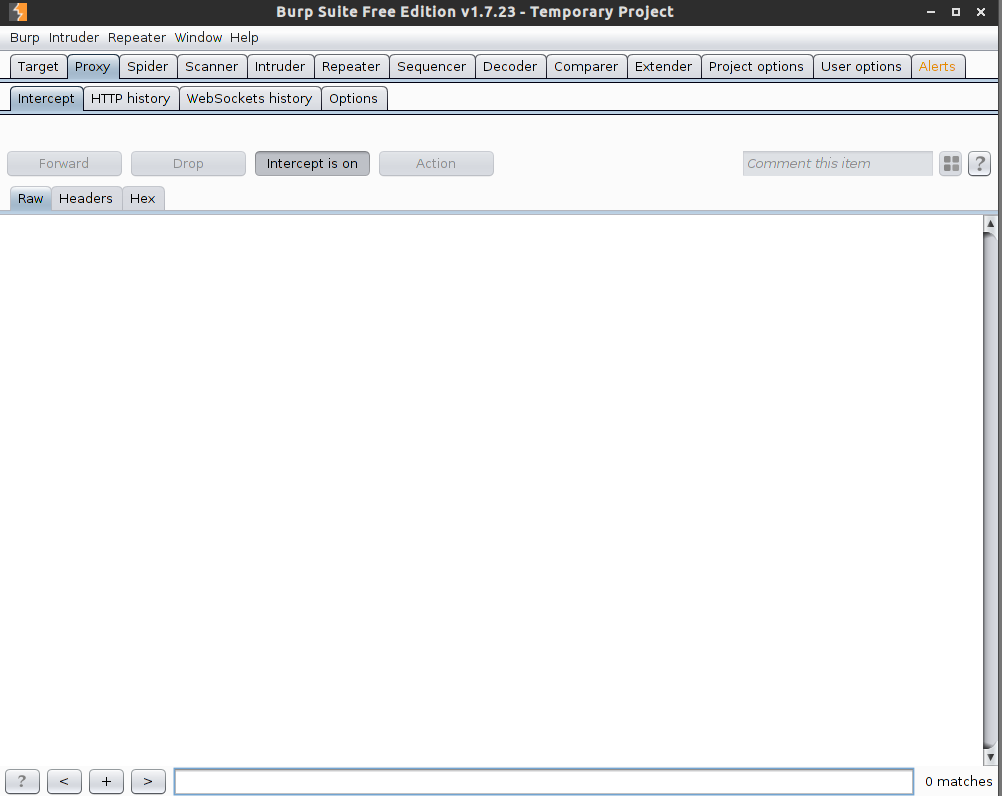
最终生成目标ip地址



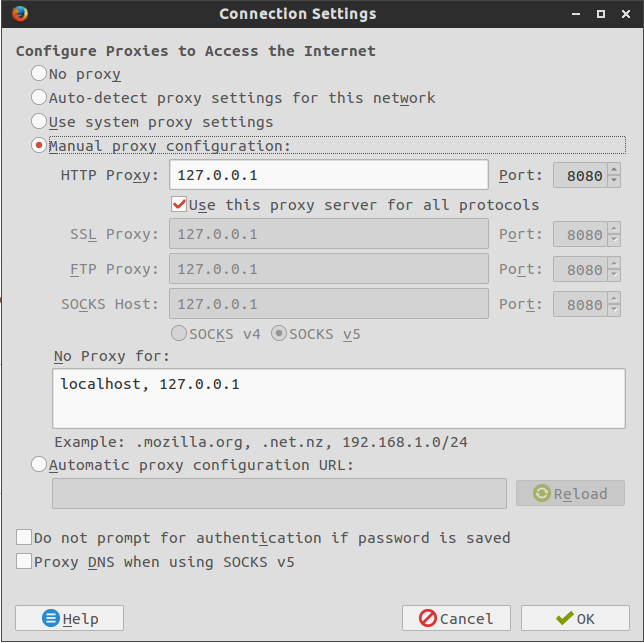
查看该虚拟机网卡



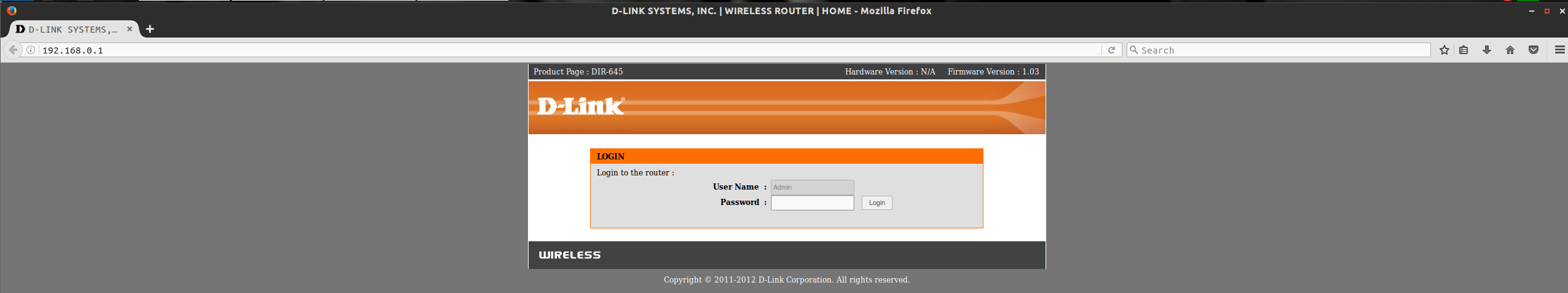
打开burp



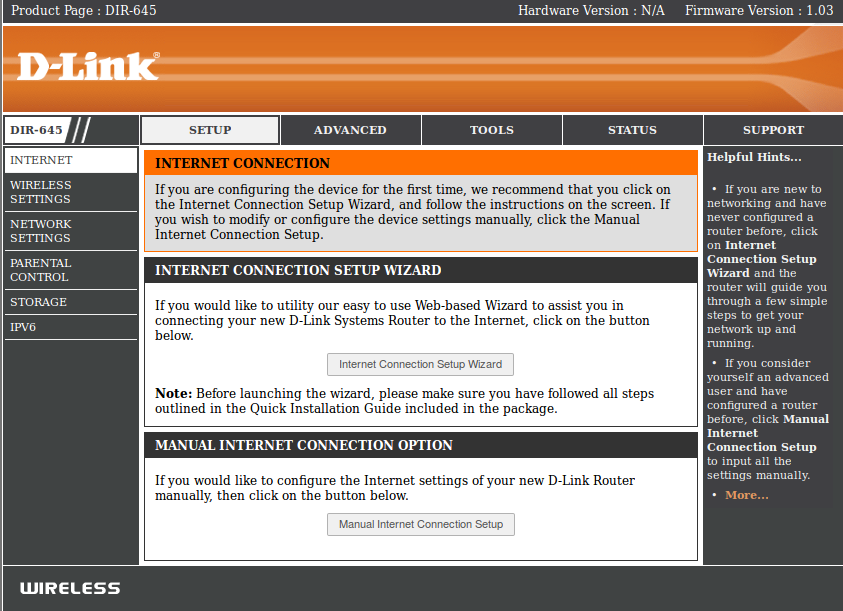
修改火狐浏览器代理



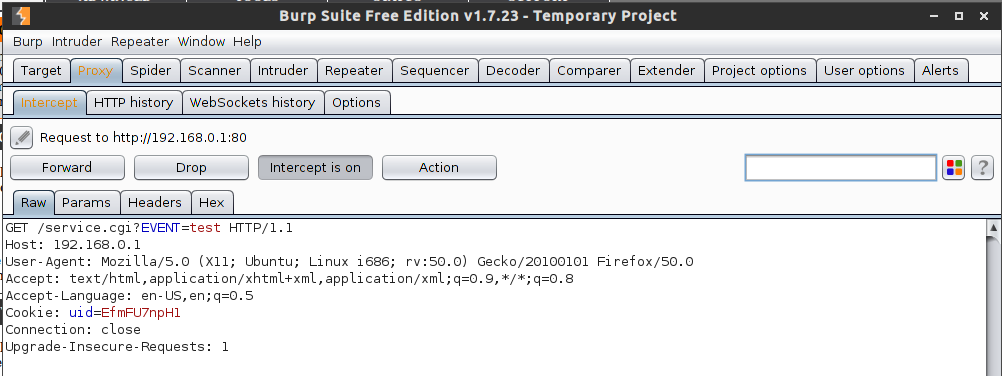
之后访问终端给出的ip：192.168.0.1成功实现固件仿真



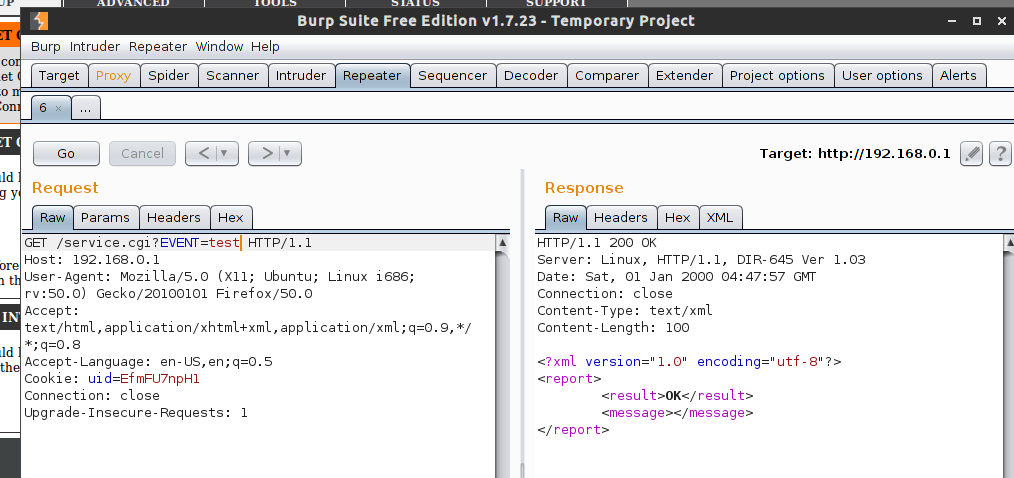
随即不用输入密码直接点击login，登陆成功之后成功劫持到cookie



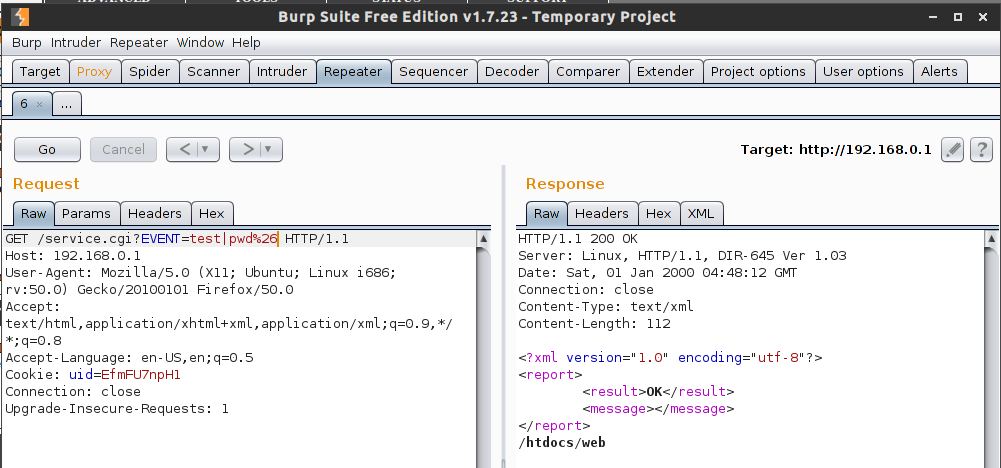
接下来使用火狐浏览器访问192.168.0.1/service.cgi?EVENT=test,抓包结果如下



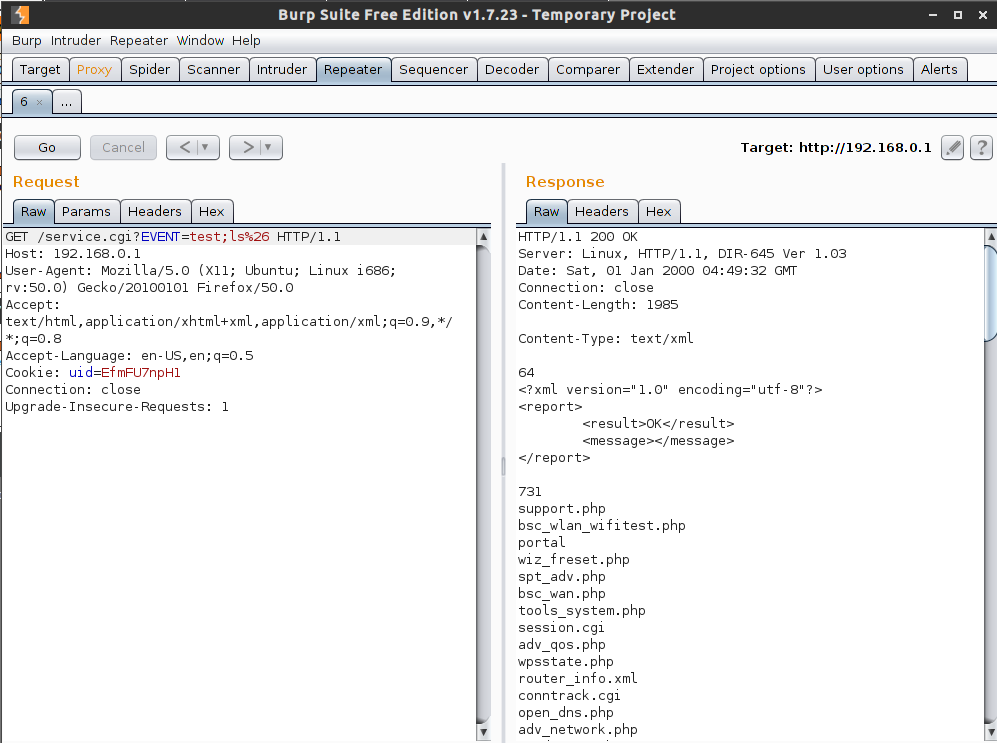
将抓取的流量包发送给重放器



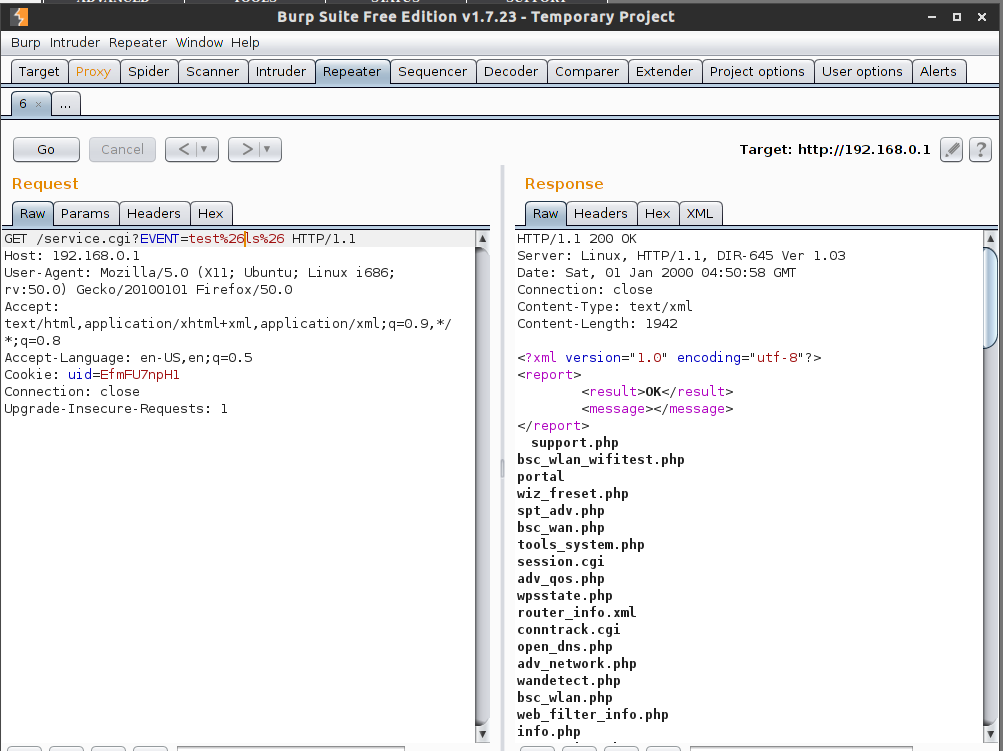
修改EVENT的内容为test | pwd%26可以发现response返回了pwd命令的执行结果，这里我们使用的截断方式为|



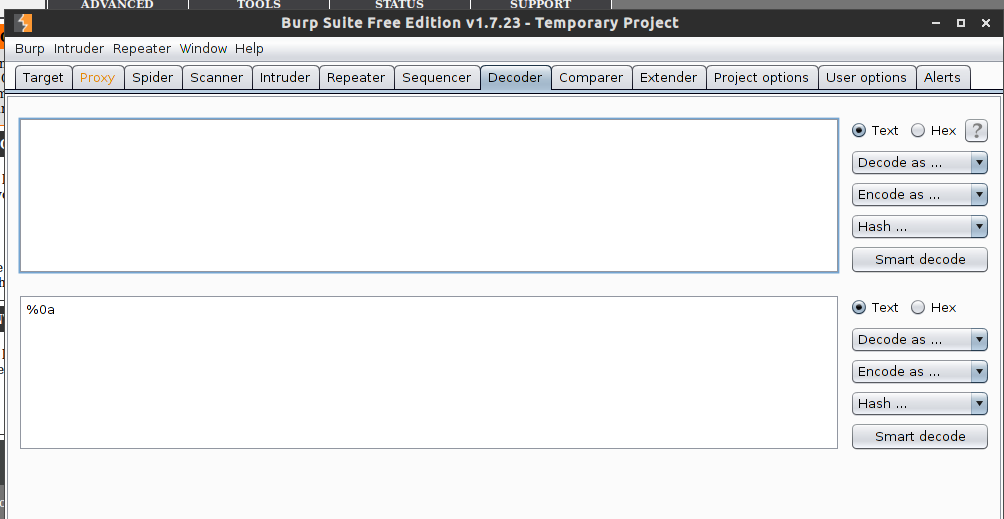
接下来尝试使用 ; 进行截断构造流量包如下所示，执行ls命令后成功返回命令执行结果



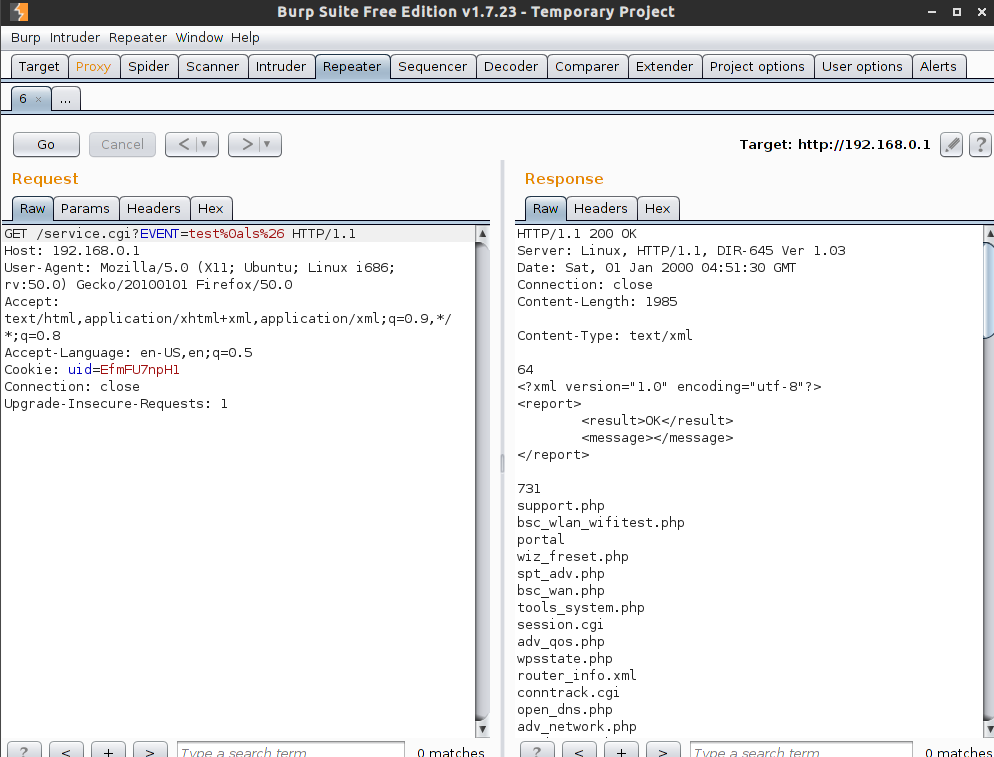
接下来尝试使用 %26 进行截断，发现依然能够成功执行ls命令



在Decoder中查看换行符的URL编码为%0a

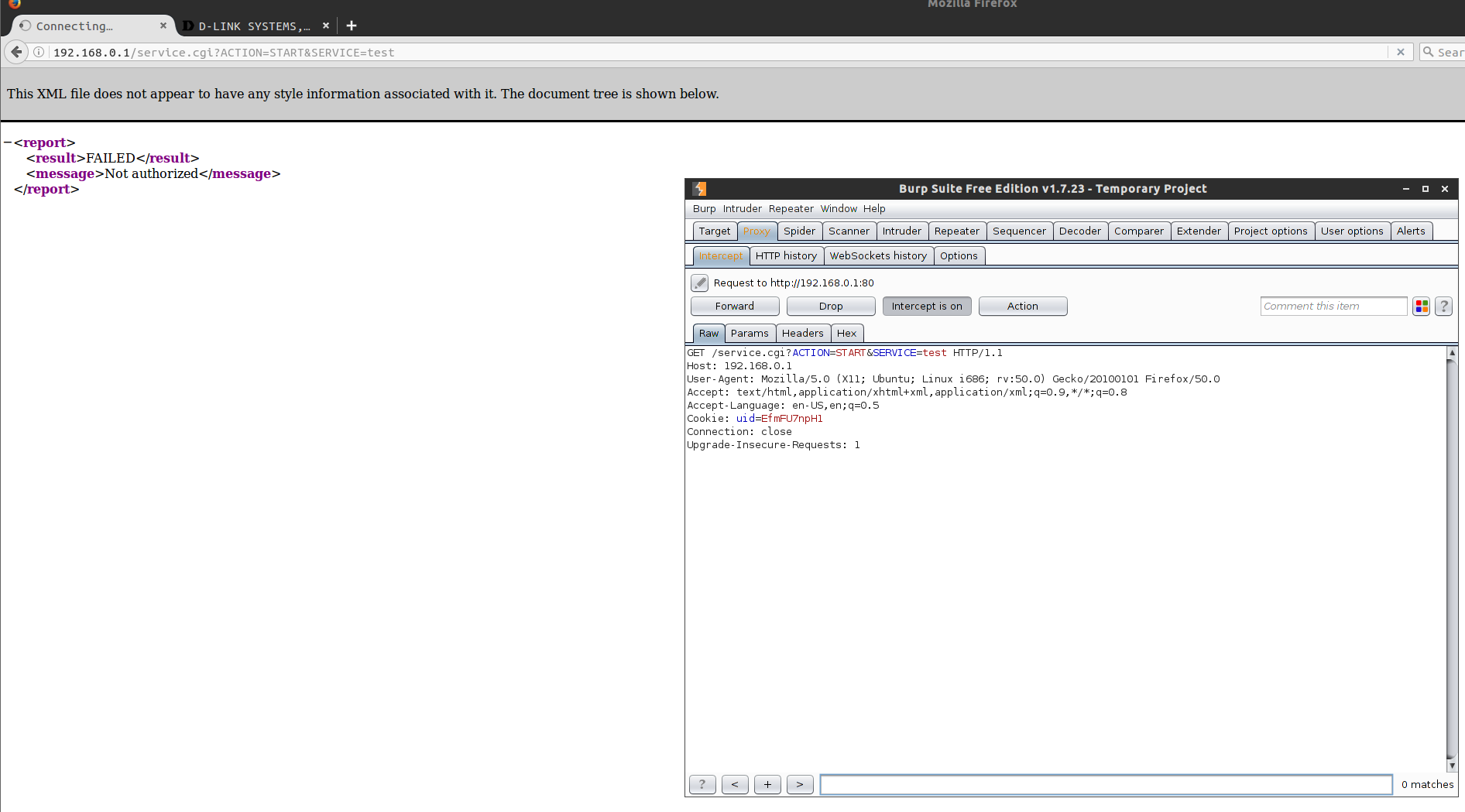


使用%0a进行截断也能够成功执行ls命令

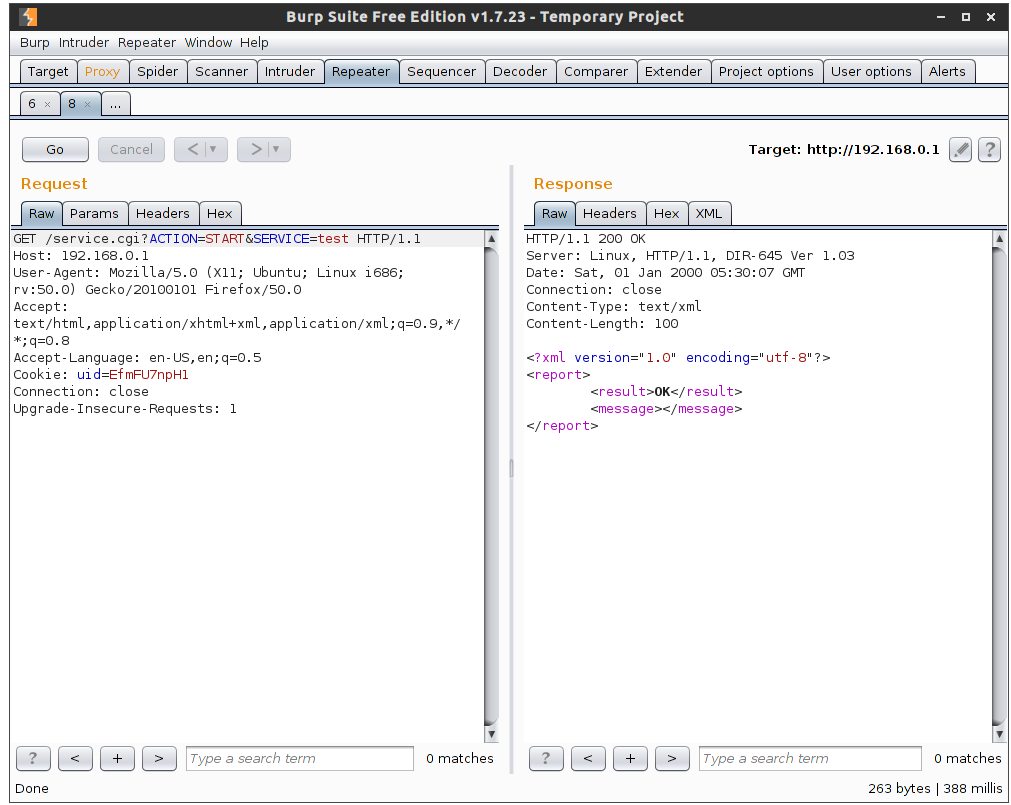


由此我们成功实现利用EVENT获取目标shell

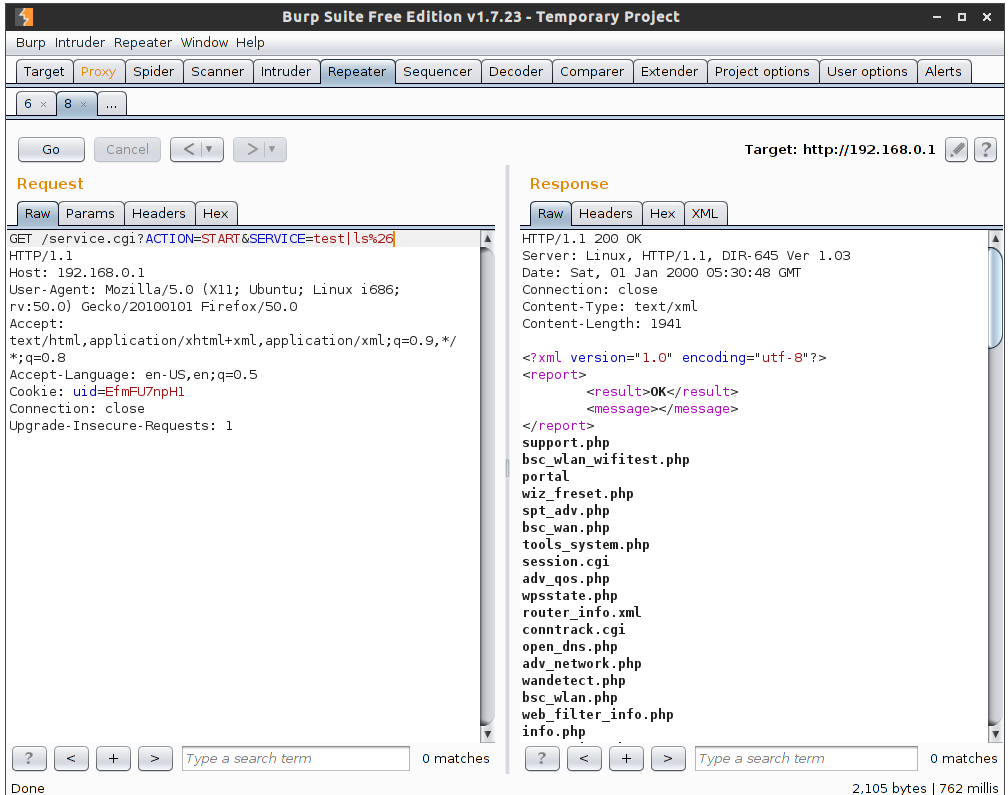
因为在分析过程中我们知道可以使用EVENT也可以使用SERVICE所以我们继续尝试service，访问http://192.168.0.1/service.cgi?ACTION=START&SERVICE=test之后济进行抓包



将抓取的流量包发送给重放器

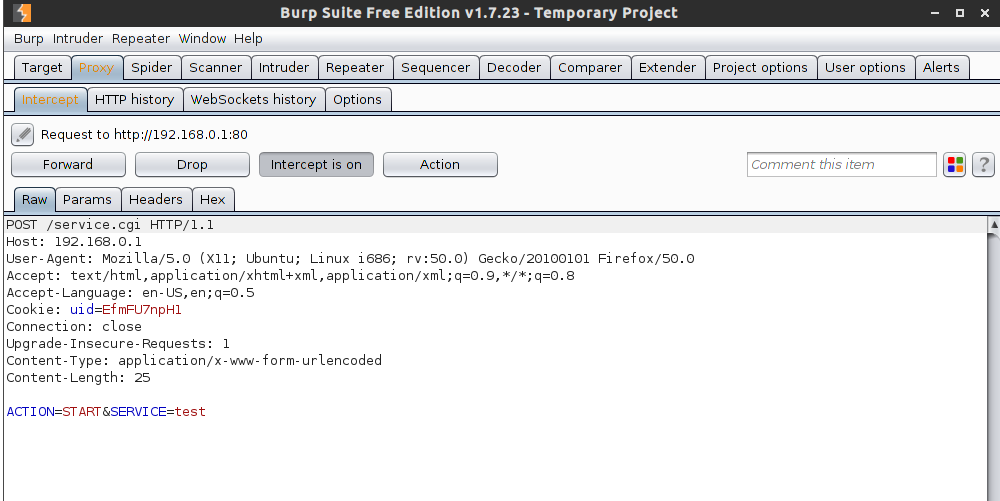


修改流量包内容，使用|截断来执行ls命令，成功返回根目录文件

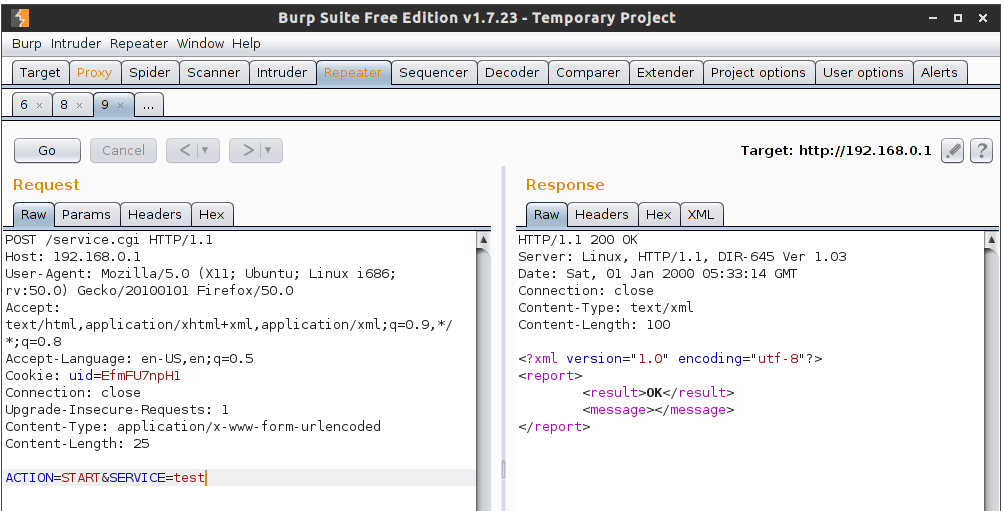


因为在之前IDA的分析中我们可以知道GET与POST两种请求命令所执行的函数相同，所以在这里尝试使用POST请求触发漏洞

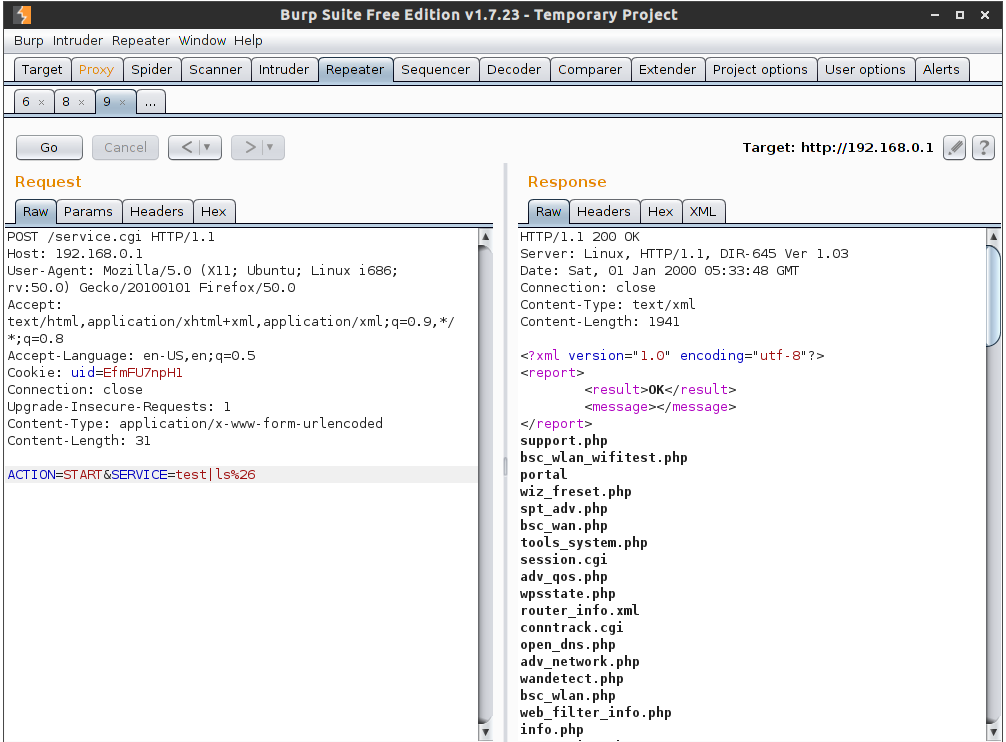
在bp中将GET请求更改为POST



然后与之前的步骤一样将流量包发送给重放器



之后同样使用|截断执行ls命令，同样能够执行成功



综上所述，成功实现获取目标路由器固件的shell。