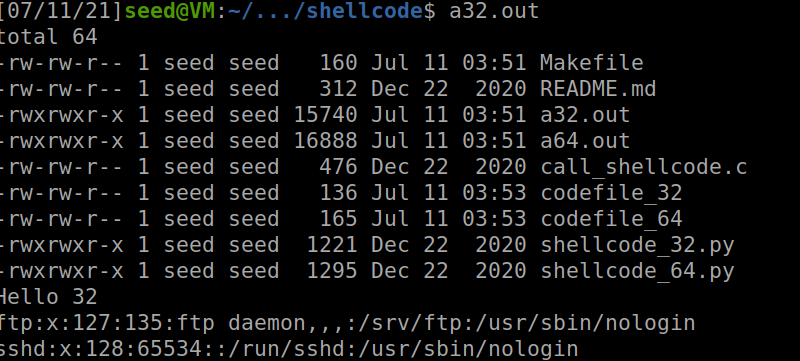
**操作系统实验二实验报告**

**学号57119123 姓名高昂 时间7 11**

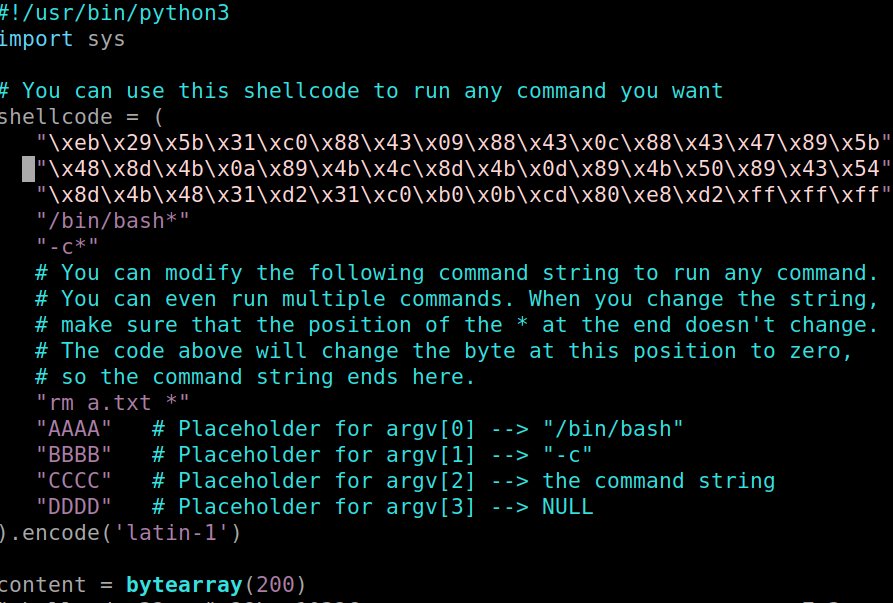
**Task 1: Get Familiar with the Shellcode**

**测试：**原始的shellcode

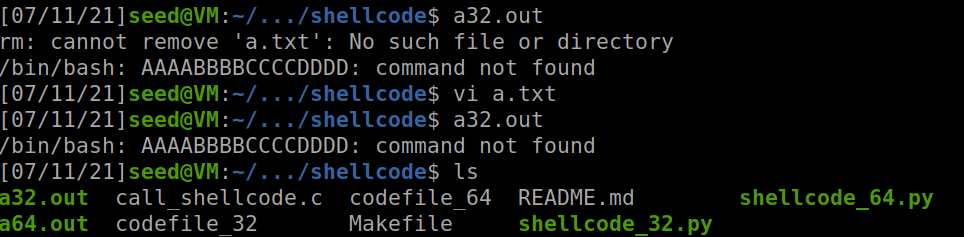
****

shellcode功能改为删除某一文件:

**修改：**



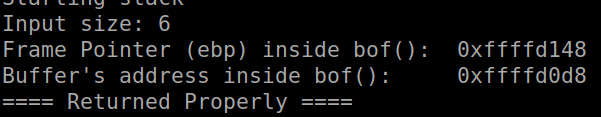
可以看到a.txt被删除



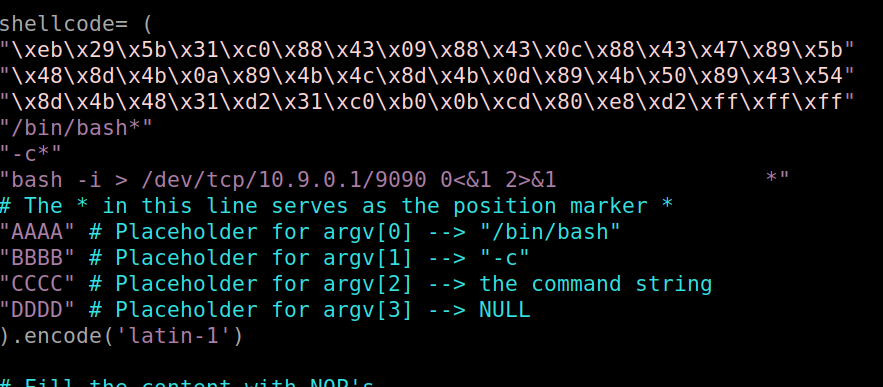
**Task 2: Level-1 Attack**

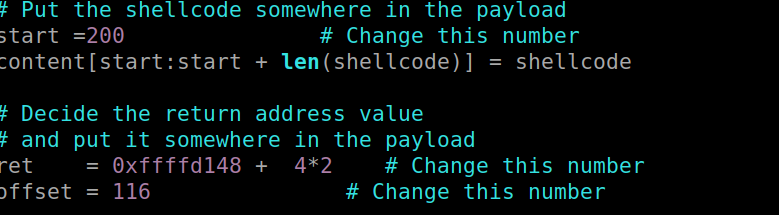
**测试：**

服务器：

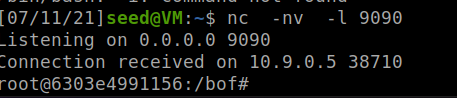


修改 exploit.py 文件：





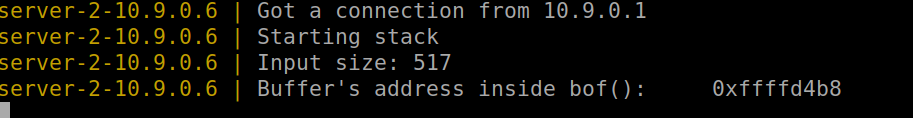
在本地客户端上监听：



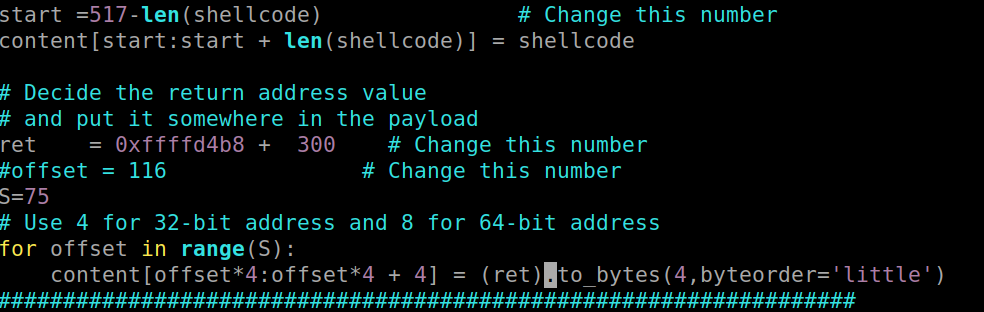
已经可以在攻击端上执行服务器的 shell 了。

**Task 3: Level-2 Attack**

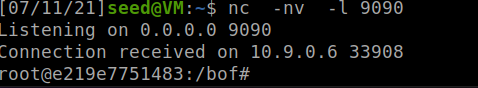
正常链接服务器获取其 Buffer 地址：



改写攻击程序： 利用循环在我们插入的 shellcode 前的每一个位置都输入其地址，总会有一个覆盖到返回地址。



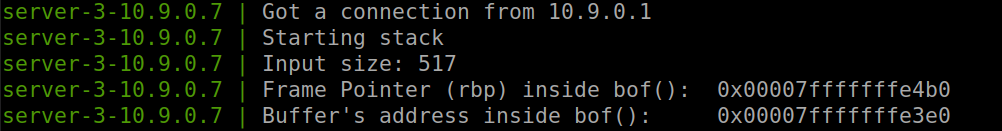
监听端：



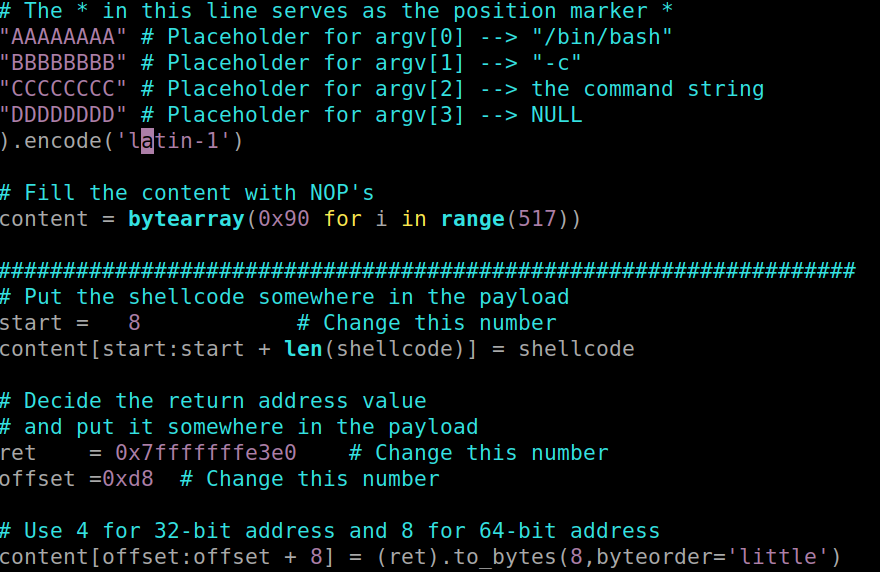
Ip 地址为服务器的地址，说明入侵成功

**Task 4: Level-3 Attack**

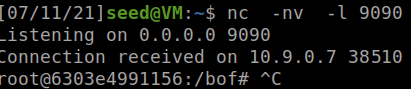
正常链接服务器获取相关信息：



修改攻击代码，由于 64 位地址 0 的存在，攻击程序在刚刚写到返回地址时便会停止，不再进行后面的写入。把攻击程序写到 buffer 内，，同时修改返回地址为 buffer 的入口地址，这样经过指令跳转可以执行到攻击程序。



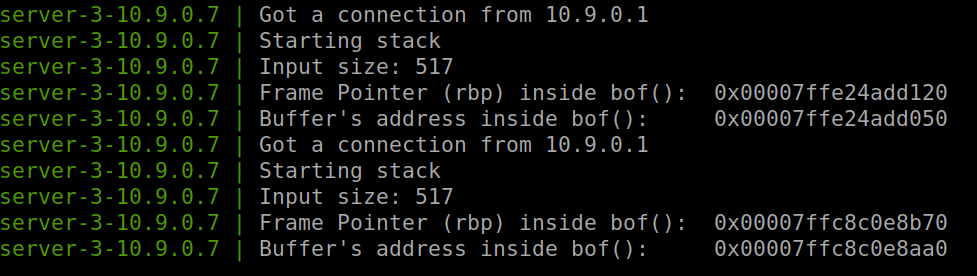
监听端



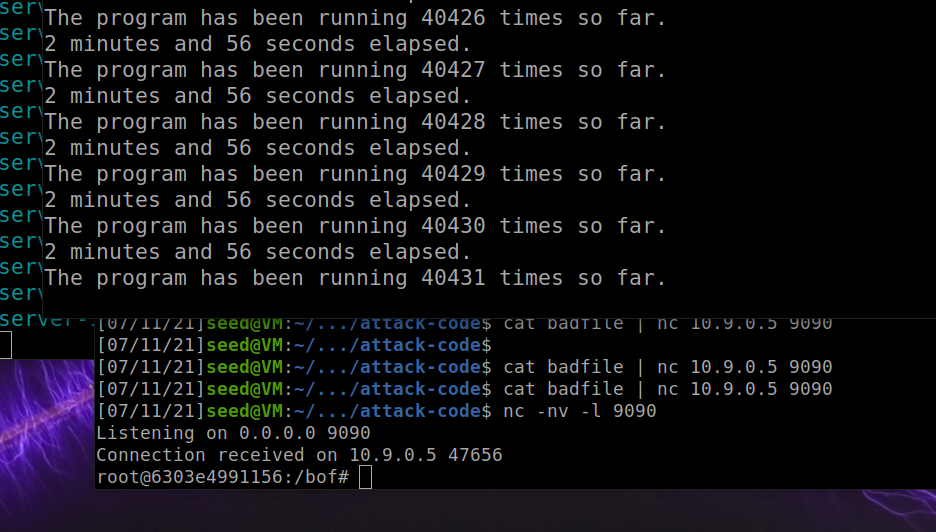
攻击成功

**Task 6: Experimenting with the Address Randomization**

打开防御 观察栈地址变化。



使用 shell 脚本暴力攻击。



尝试40431次后攻击成功

**实验体会：**

本次实验通过不检查越界的函数的复制溢出，把代码复制到shell内，并通过计算来修改其函数的返回位置为我们插入的代码地址，在返回函数时返回到了我们插入的代码位置，并以其高权限来执行我们的代码，从而进行攻击。

防御方法：栈地址随机化，每次在其前加入一段随机的地址。检测栈是否越界。