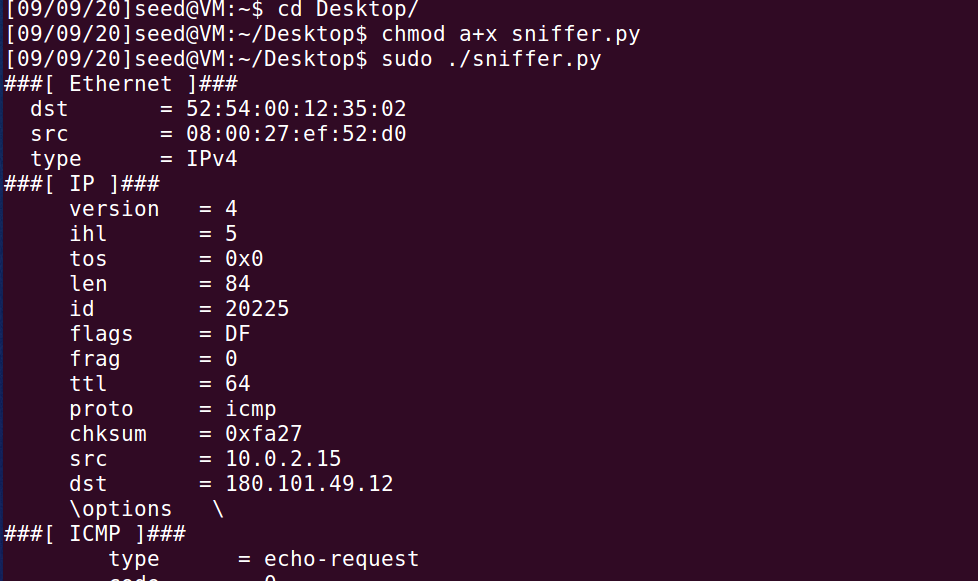
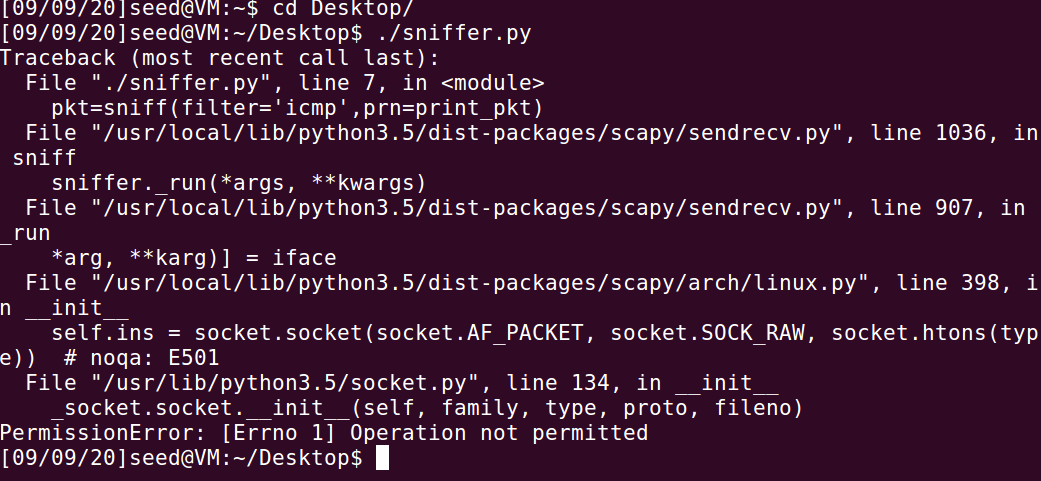
**Task1.1A**

使用root权限执行,运行成功

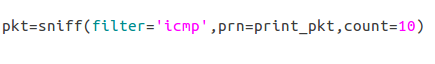


不使用root权限执行，失败



**Task1.1B**

（1）Capture only the ICMP packet



（2）Capture any TCP packet that comes from a particular IP and with a destination port number 23.

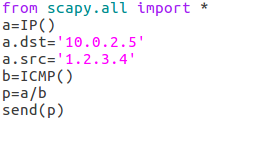


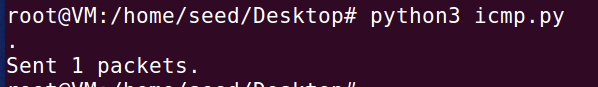
（3）Capture packets comes from or to go to a particular subnet. You can pick any subnet, such as 128.230.0.0/16; you should not pick the subnet that your VM is attached to



**Task1.2: Spooﬁng ICMP Packets**

将原地址ip设置为‘1.2.3.4’，并执行icmp.py





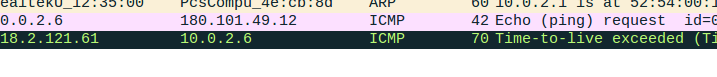
使用Wireshark观察，可以看到目的主机向伪造的ip地址发送了应答响应数据包。这里只要把1.2.3.4改成任意ip地址，都会达成同样的效果。



**Task1.3: Traceroute**

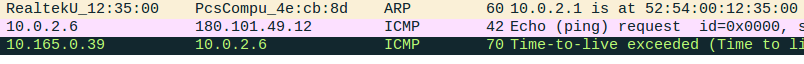
选择百度的ip地址作为实验对象，

TTL=4



不断的修改TTL的值进行逼近

TTL=11



TTL=12



所以，计算出路由距离为12

Task1: ARP Cache Poisoning

场景：

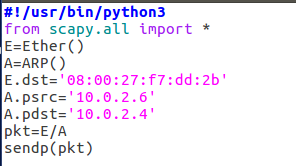
seed1：ip-mac 10.0.2.4- 08:00:27:f7:dd:2b

seed2：ip-mac 10.0.2.6- 08:00:27:b9:77:de

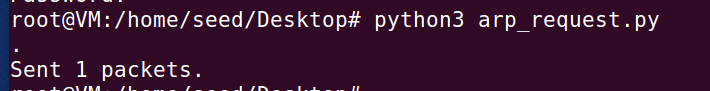
seed3：ip-mac 10.0.2.5 -08:00:27:ad:75:94（攻击者）

**Task 1A (using ARP request)**

arp\_request.py



在seed3上发送给seed1



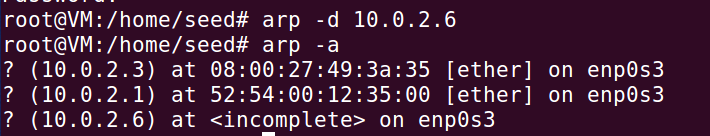
查看seed1的arp表



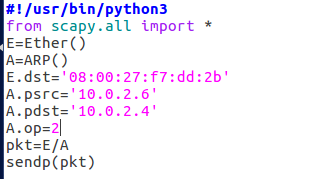
seed2的ip10.0.2.6对应的是seed3的mac地址，攻击成功。

**Task 1B (using ARP reply)**

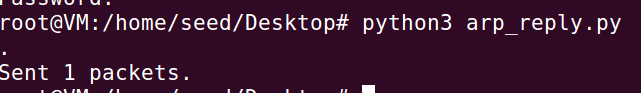
先清除arp表中的相关项



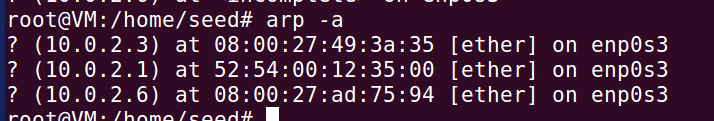
arp\_reply.py

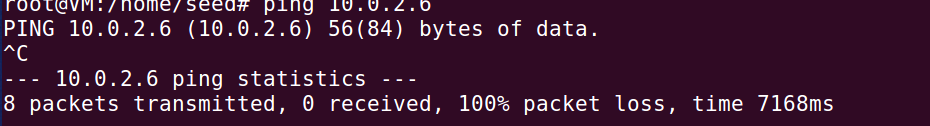


在seed3上发送给seed1



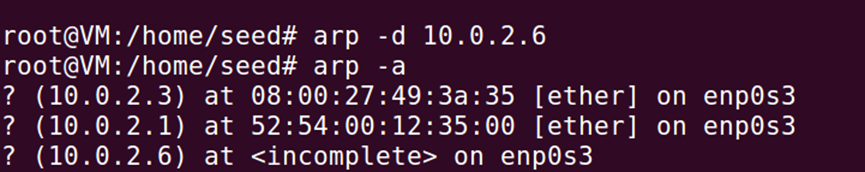
再次查看arp表，发现攻击成功。



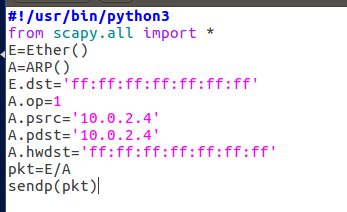


**Task 1C (using ARP gratuitous message)**

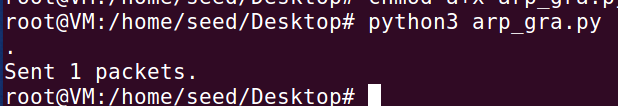
清除arp表相关项

****

arp\_gra.py



发送ARP gratuitous message



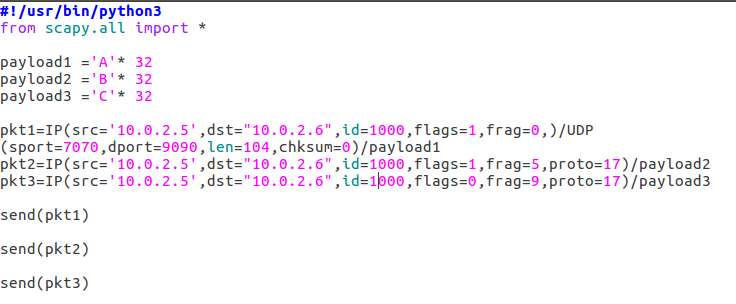
查看seed1的arp表，攻击成功



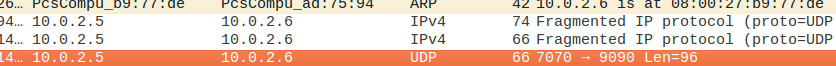
**Tasks1: IP Fragmentation**

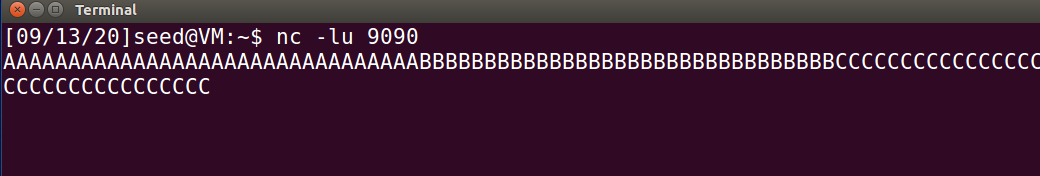
**Task1.a: Conducting IP Fragmentation**

udp1.py

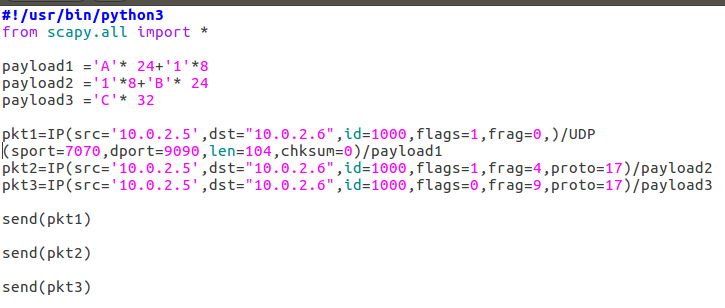


wireshark：





**Task1.b: IP Fragments with Overlapping Contents**



**K=8**

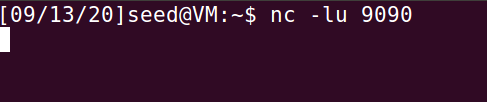
这样，payload1的后8个字节将与payload2的前8个字节具有相同的偏移量4



nc-lu 9090没反应看不出来



nc命令依旧没反应



nc命令总是失灵，一个上午只成功输出一次。