**Lab[1]-report**

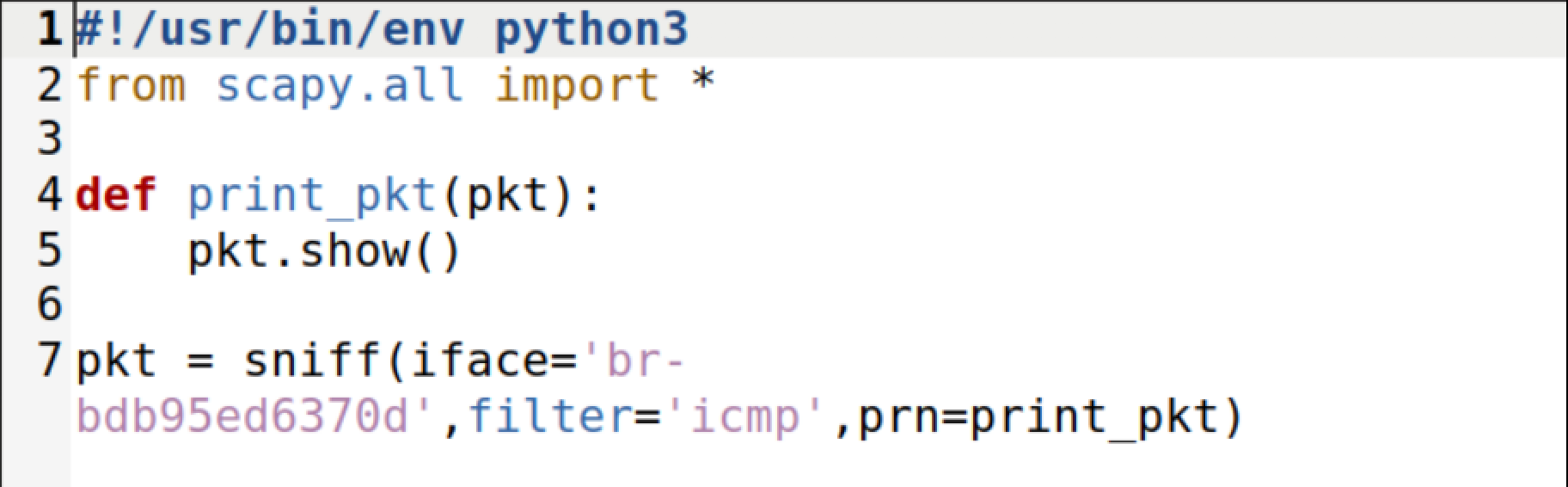
——57118207 魏宇萌

实验网络环境：网络地址转换（NAT）

**Task 1.1A**

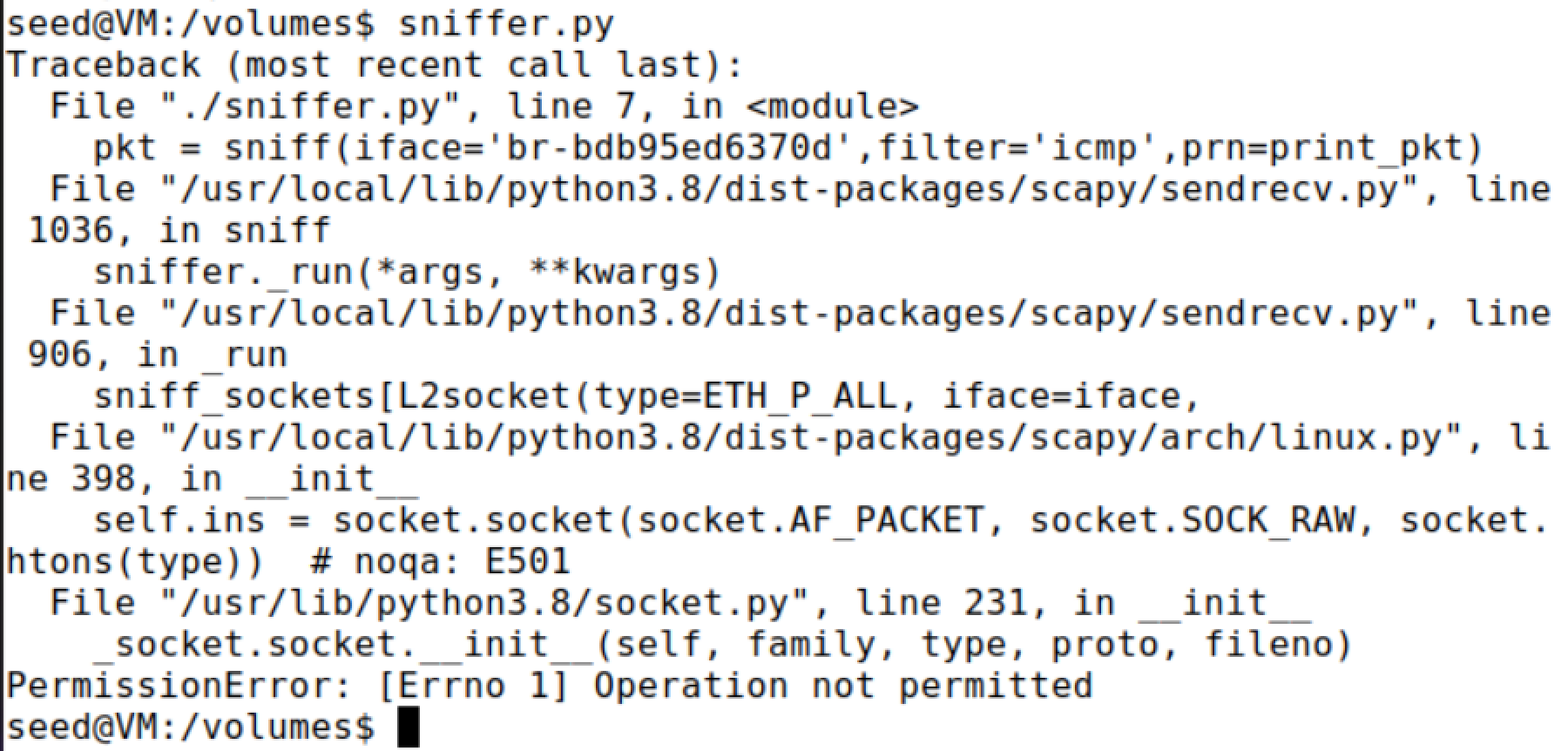
（1）实验内容：用Scapy编写数据报嗅探程序sniffer.py，观察sniffer.py在root和seed权限下的执行情况。

（2）代码：



sniffer.py

（3）程序执行情况：在seed权限下程序无法执行；在root权限下程序可以执行，然后在attacker中ping host，程序将嗅探到ICMP报文并打印相关信息。



seed权限下程序执行情况

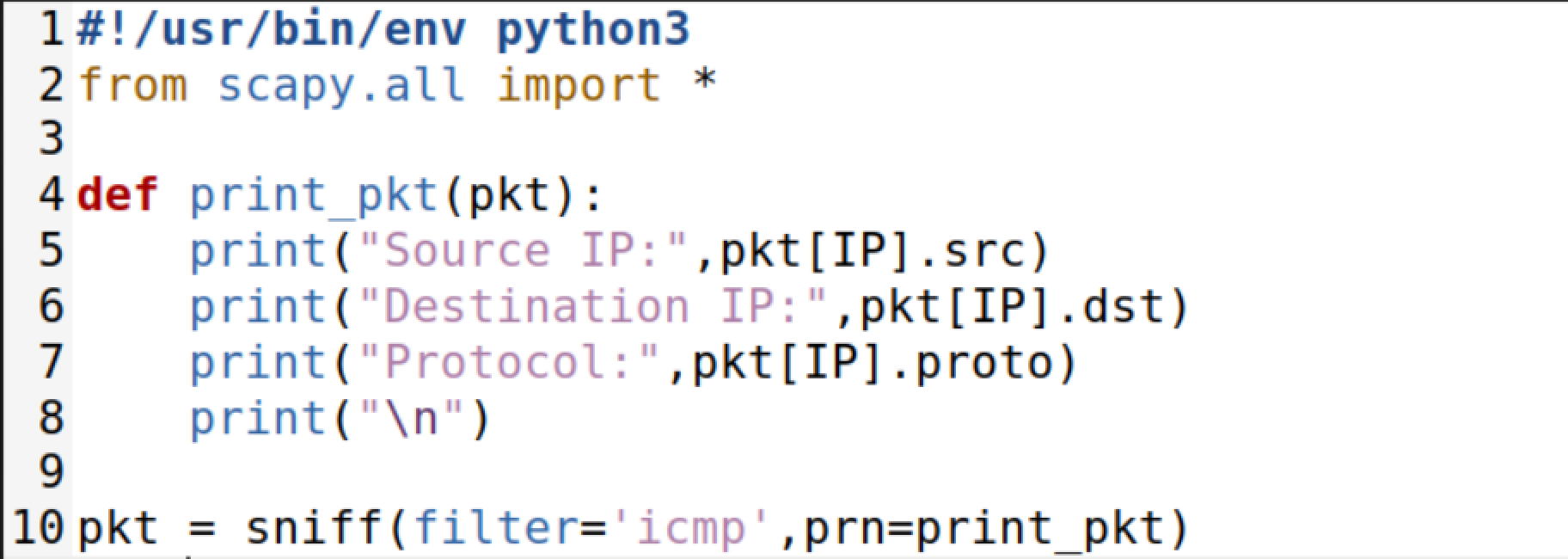


root权限下程序执行情况

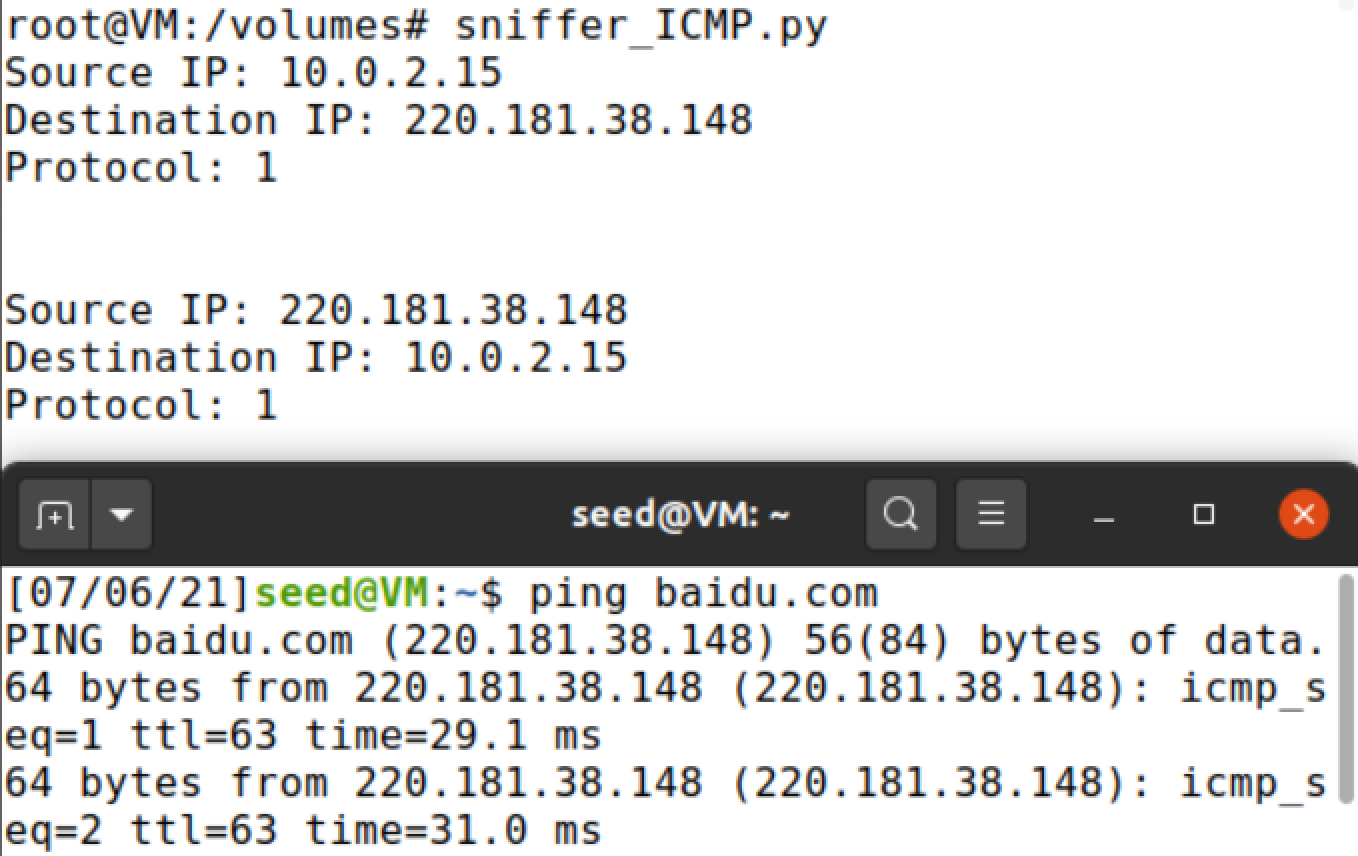
**Task 1.1B**

（1）实验内容:用Scapy filter嗅探特定报文

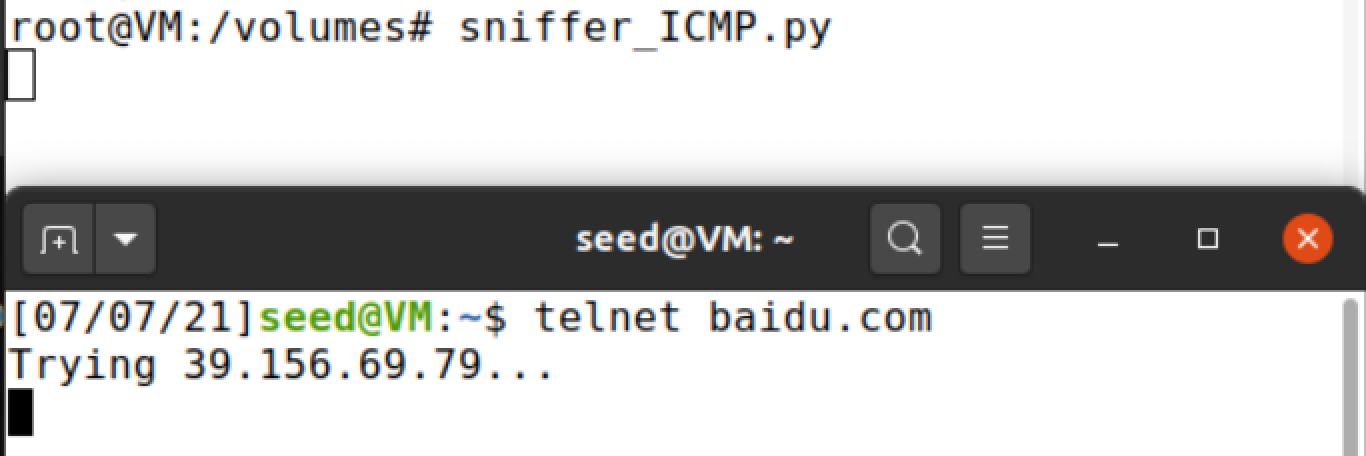
（2）代码与程序执行情况：  
（2.1）用Scapy filter嗅探ICMP报文，只有ICMP报文可被捕获，非ICMP报文不会被捕获。



sniffer\_ICMP.py

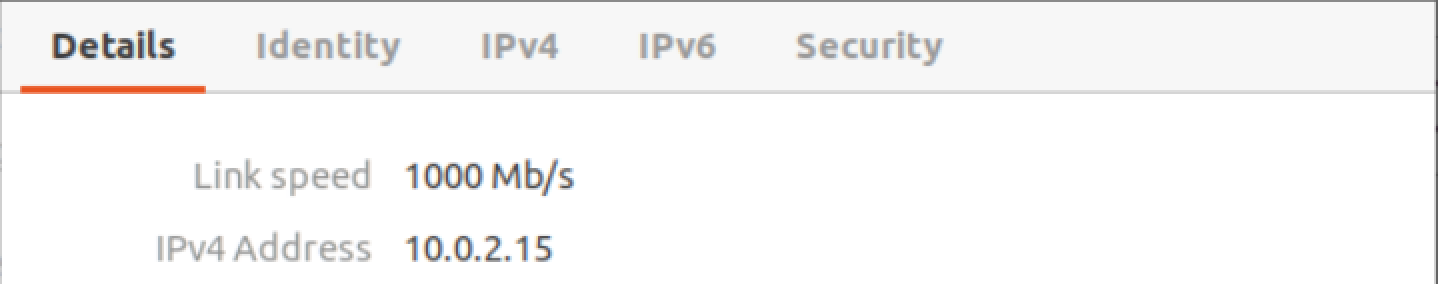


sniffer\_ICMP.py运行捕获了ICMP报文

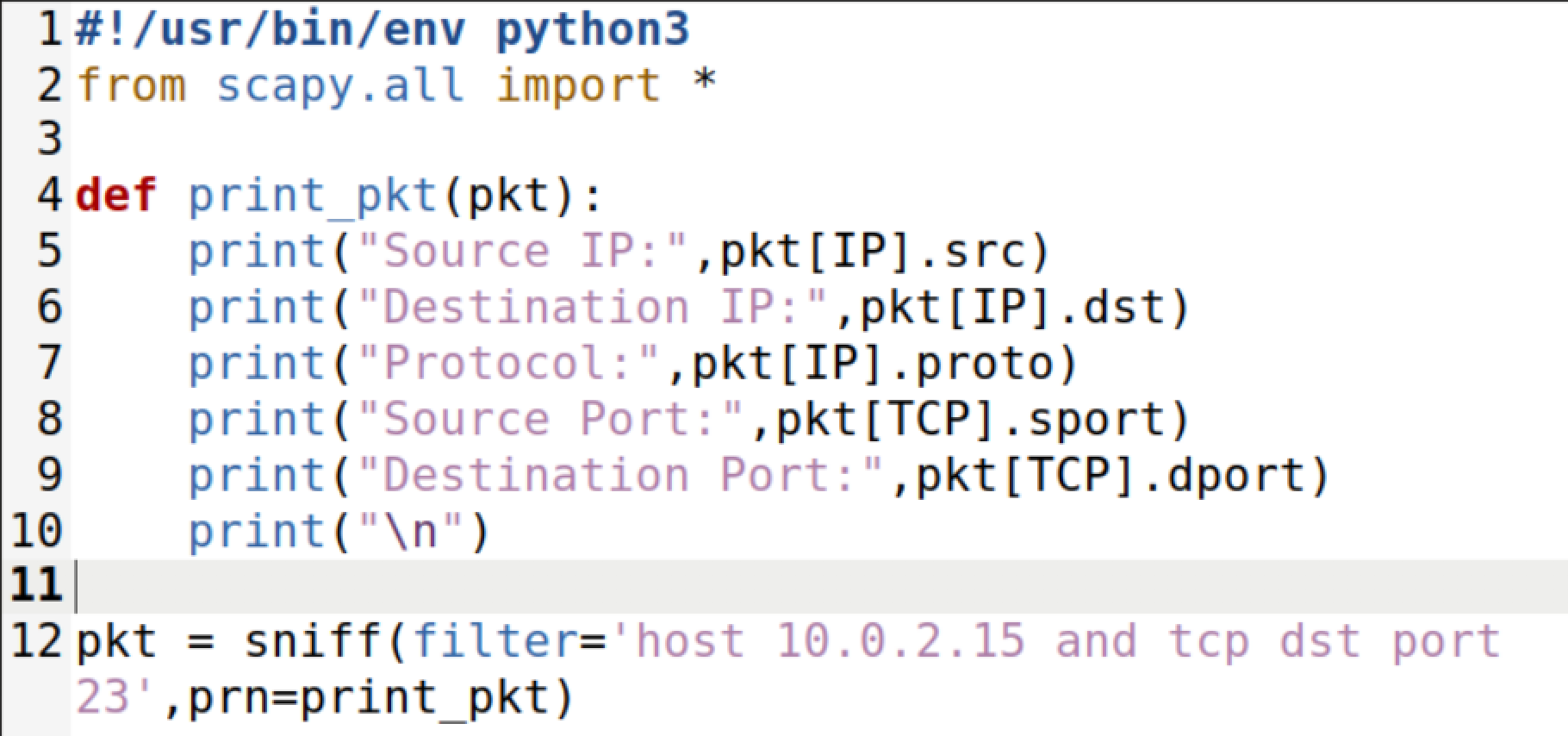


sniffer\_ICMP.py运行不会捕获非ICMP报文

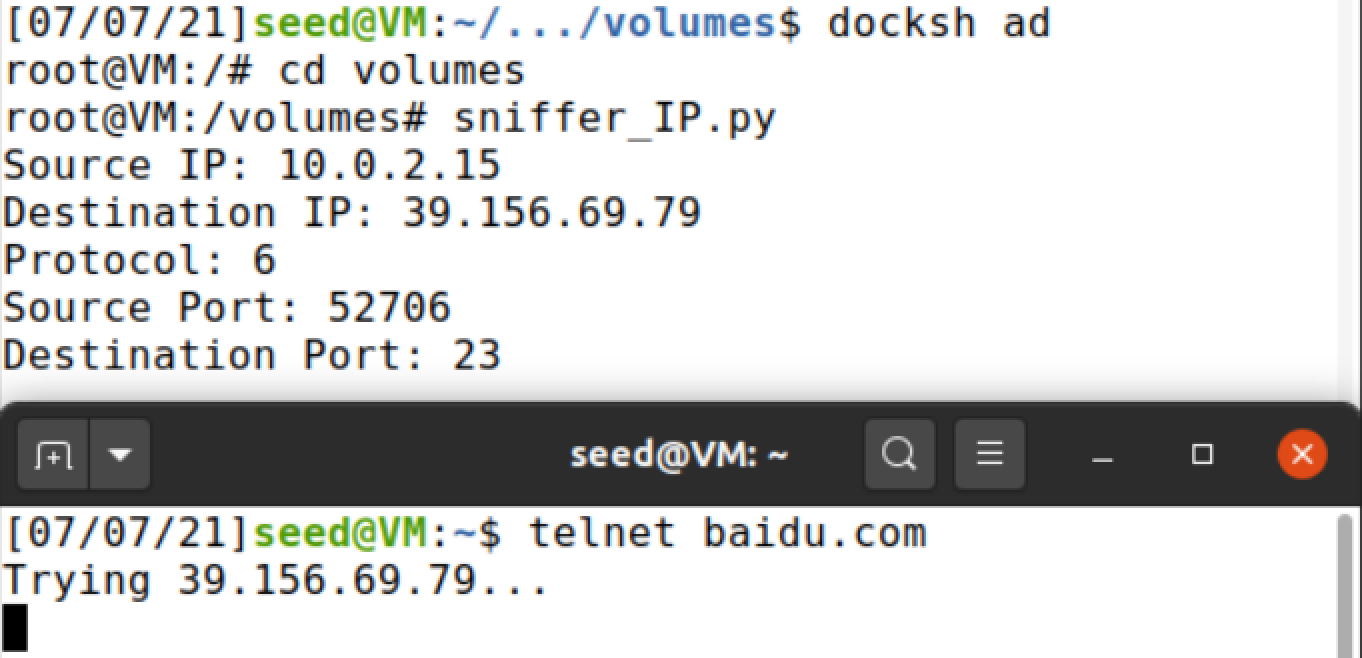
（2.2）用Scapy filter嗅探从特定IP地址发出且目的端口为23的TCP报文，只有从10.0.2.15发出的端口为23的TCP报文可被捕获，不符合条件的报文不会被捕获。



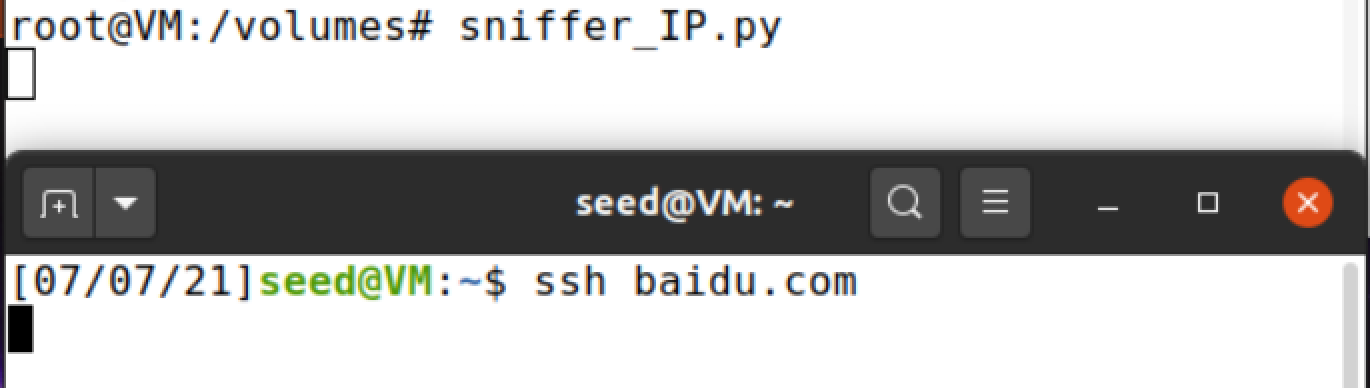
VM的IP地址分配



sniffer\_IP.py

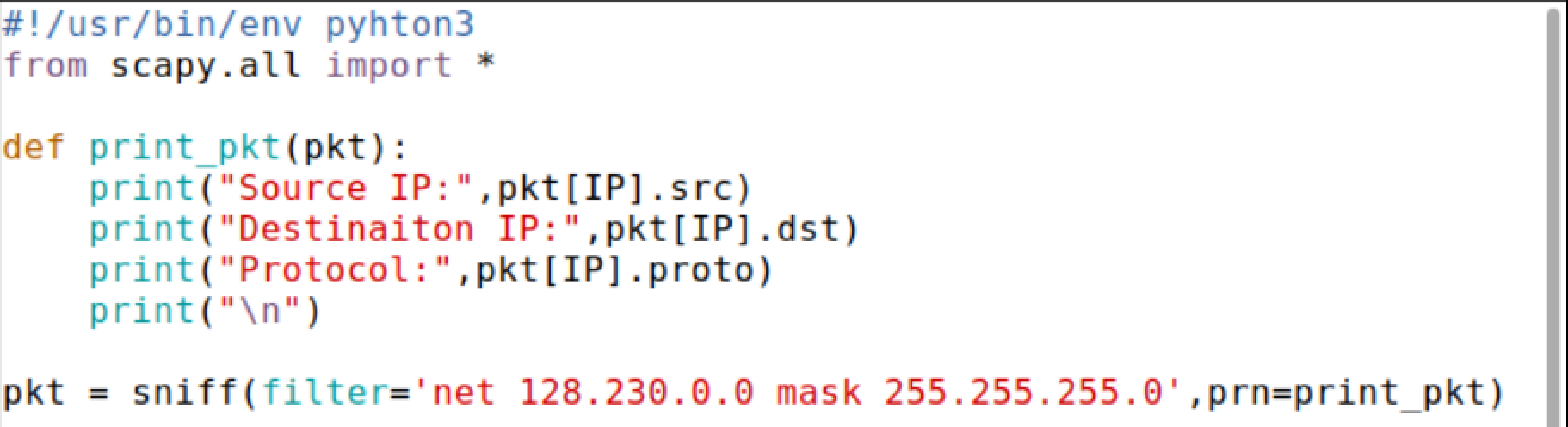


sniffer\_IP.py运行捕获符合条件的TCP报文

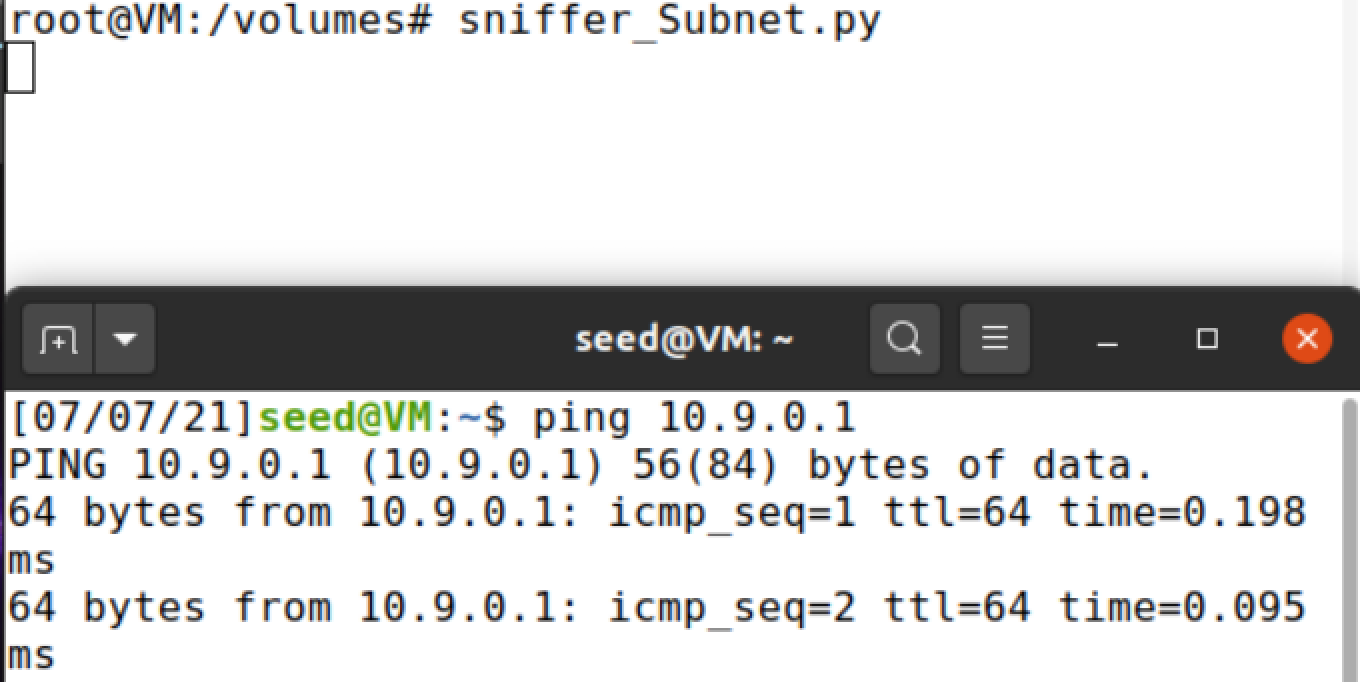


sniffer\_IP.py运行不会捕获不符合条件的报文

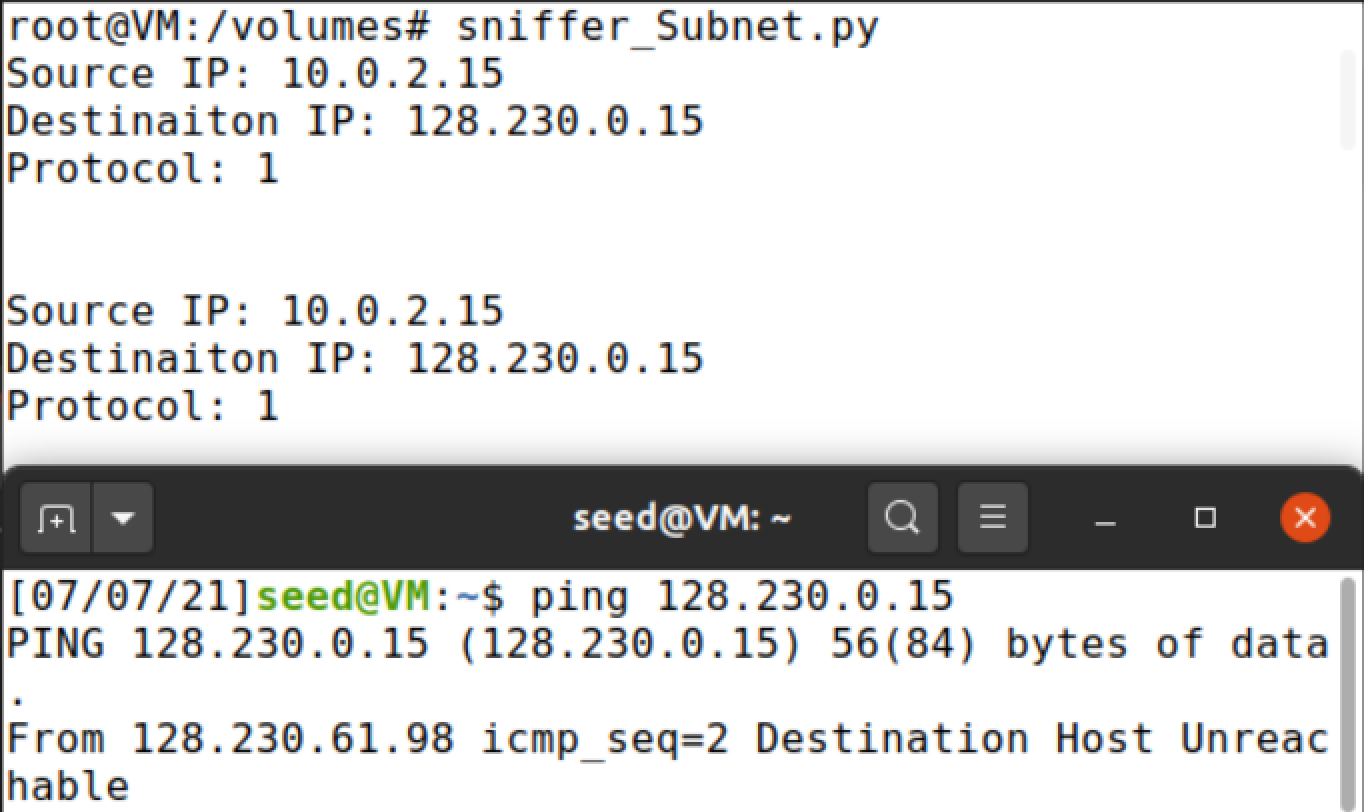
（2.3）用Scapy filter嗅探来自或发往某一子网的报文，只有属于该子网的报文可被捕获，不属于该子网的报文不会被捕获。



sniffer\_Subnet.py



sniffer\_Subnet.py运行不会捕获非128.230.0.0/24子网的报文

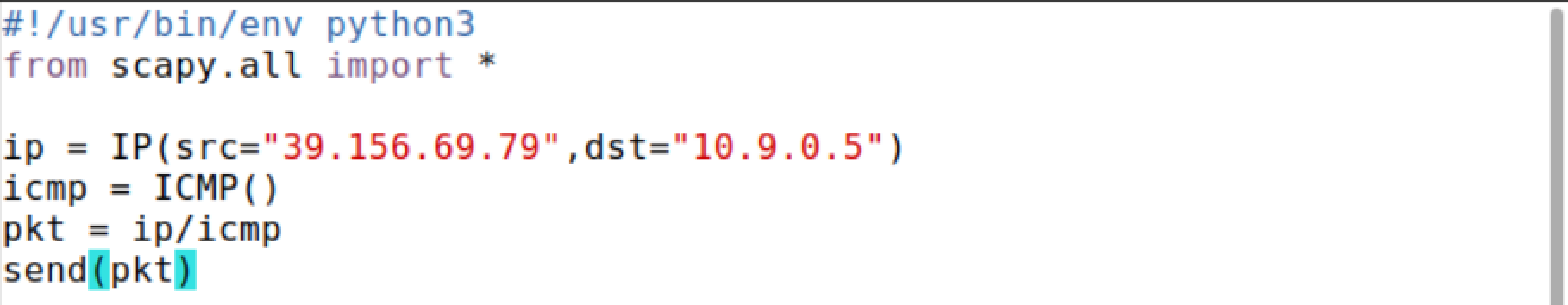


sniffer\_Subnet.py运行捕128.230.0.0/24子网的报文

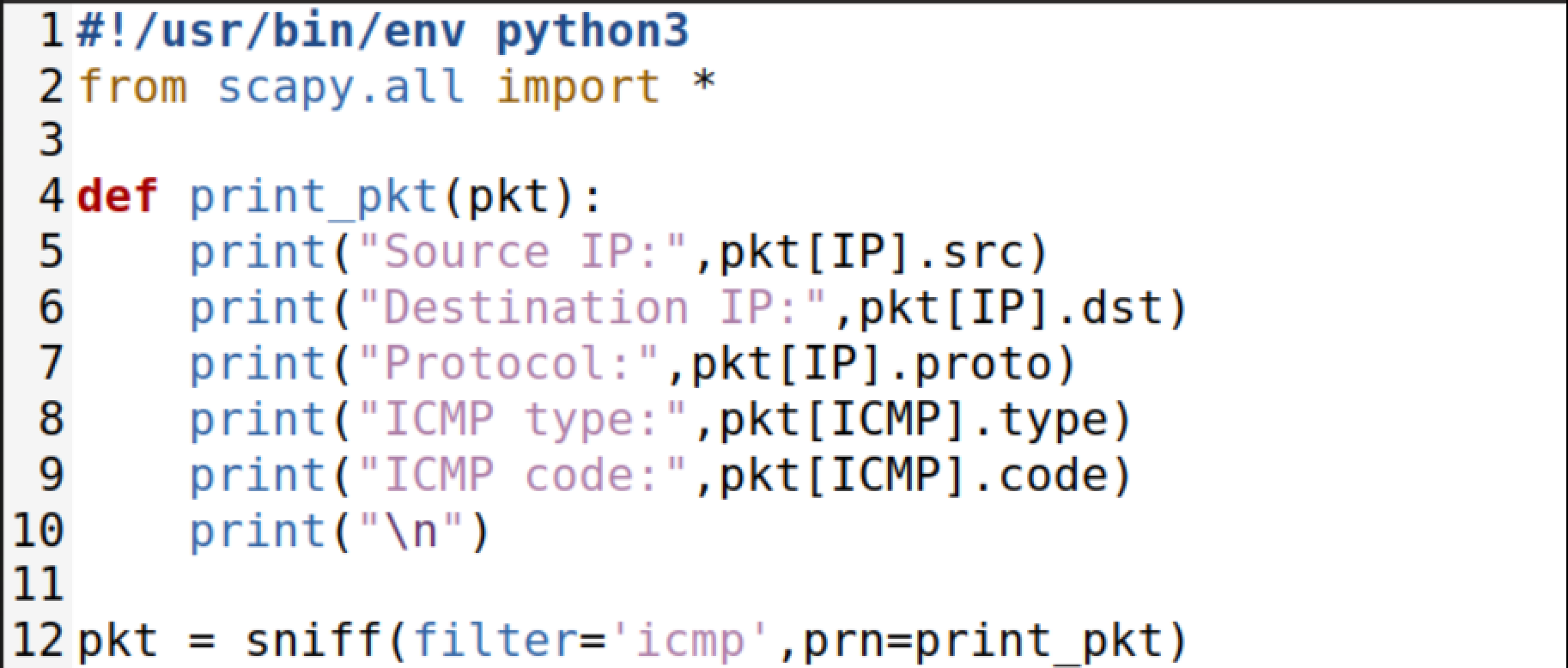
**Task 1.2**

（1）实验内容：伪造任意源IP地址的ICMP请求报文

（2）代码：伪造源IP地址为百度网址的ICMP请求报文，用Scapy filter嗅探ICMP报文。

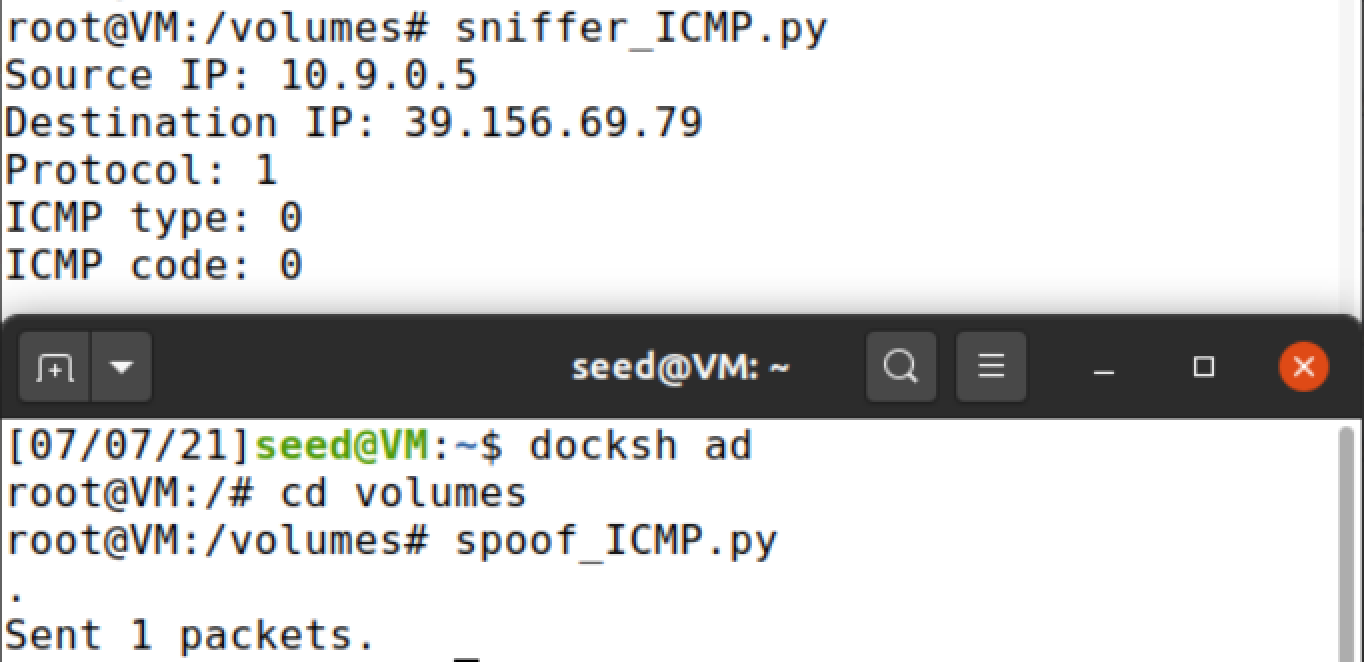


spoof\_ICMP.py

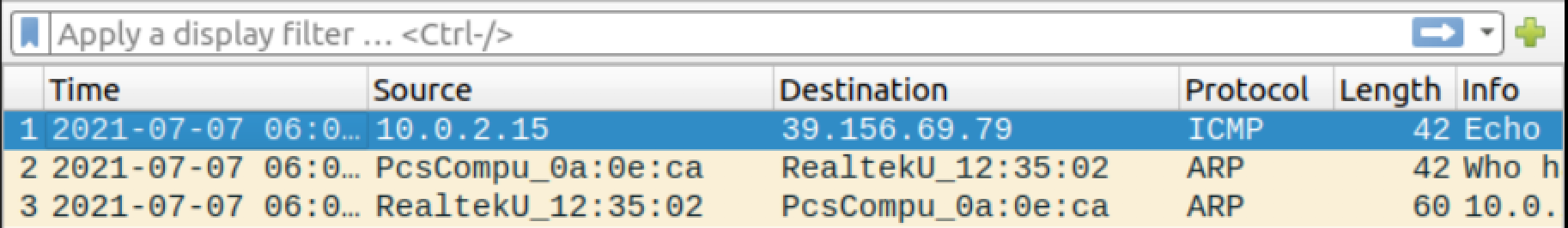


修改后的sniff\_ICMP.py

（3）程序执行情况：先运行sniff\_ICMP.py嗅探ICMP报文，然后运行spoof\_ICMP.py发送伪造源IP地址的ICMP报文。sniffer\_ICMP.py并没有嗅探到发送的伪造ICMP请求报文，但嗅探到了回复该伪造报文的ICMP响应报文（type为0，code为0）。用WireShark捕获报文得到的结果与之相同。



sniffer\_ICMP.py运行捕获报文的结果



Wireshark捕获报文的结果

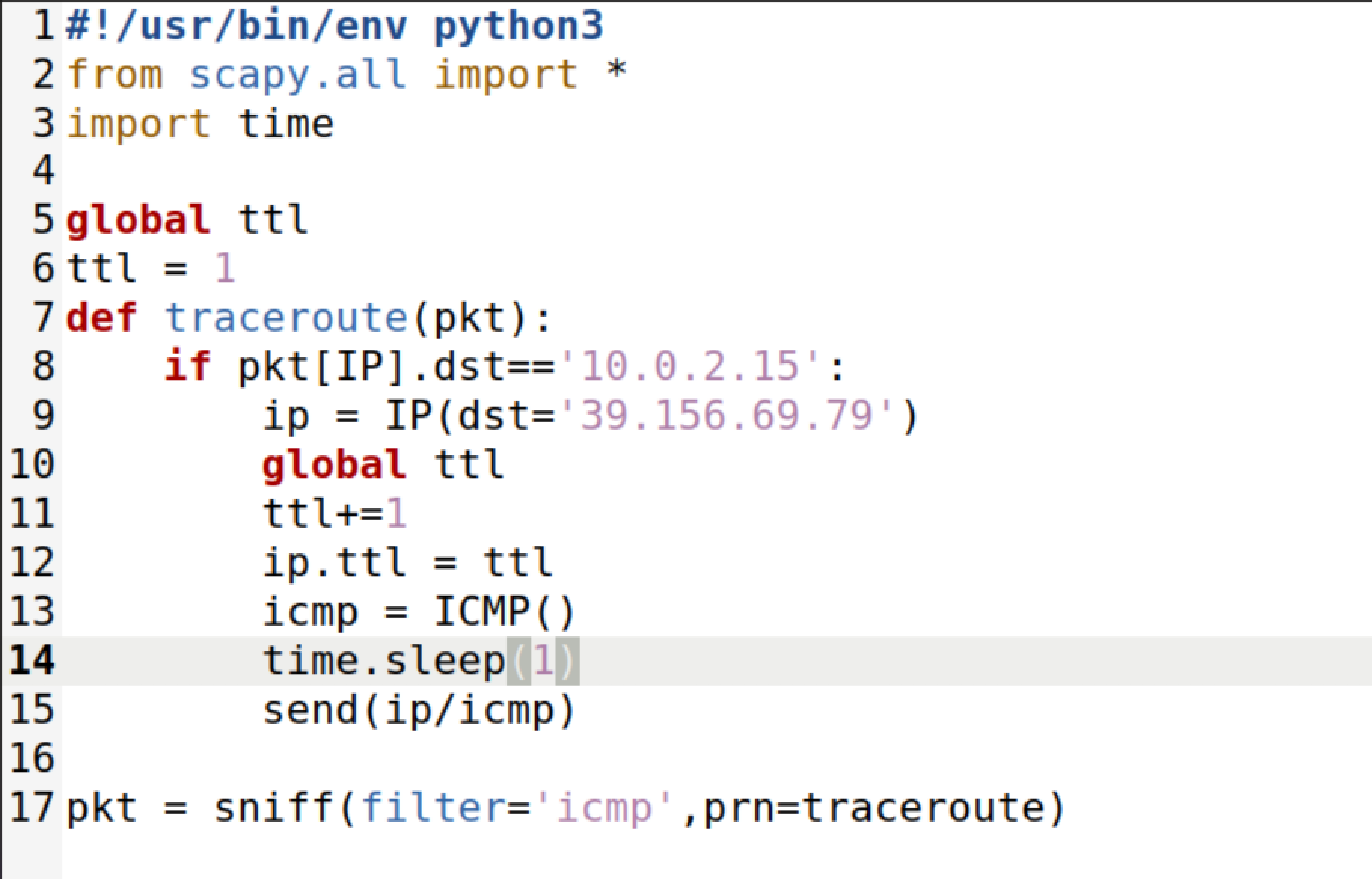
**Task 1.3**

（1）实验内容：用Scapy进行traceroute得到路由情况

（2）代码：用一个sniff程序和一个spoof程序完成traceroute，sniff程序嗅探ICMP请求报文，在spoof程序中先发送ttl=1的第一个traceroute报文后，sniff嗅探到ICMP响应报文便发送之ttl=2的第二个traceroute报文，如此反复，直到traceroute结束。即每次收到ICMP响应报文时再发出下一个ICMP请求报文。



spoof程序

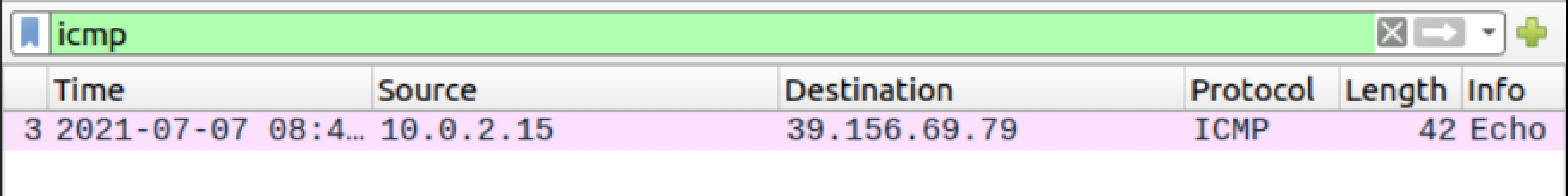


sniff程序

（3）程序执行情况：用Wireshark捕获报文，先执行sniff程序，再执行spoof程序，在Wireshark中可以观察到traceroute的情况，黑色部分的Source即为途径的路由地址，但目的主机并未返回ICMP响应报文。通过直接向目的主机发送ICMP请求报文发现确实未响应，推测目的主机应该拒绝被ICMP请求报文发现。



traceroute路由结果

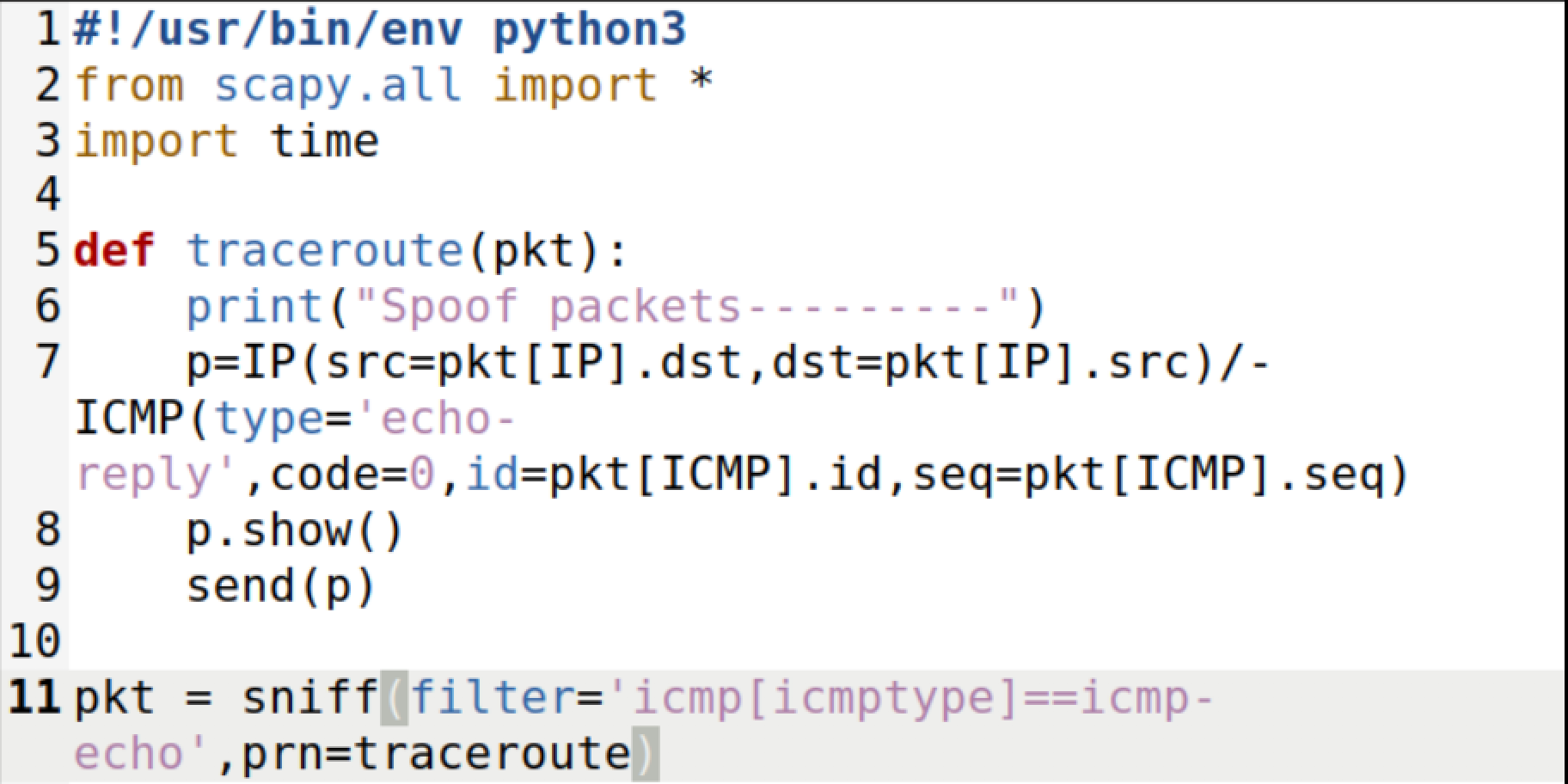


目的主机不响应ICMP请求报文

**Task 1.4**

（1）实验内容：嗅探ICMP请求报文，一旦发现ICMP请求报文便伪造ICMP响应报文发回主机。

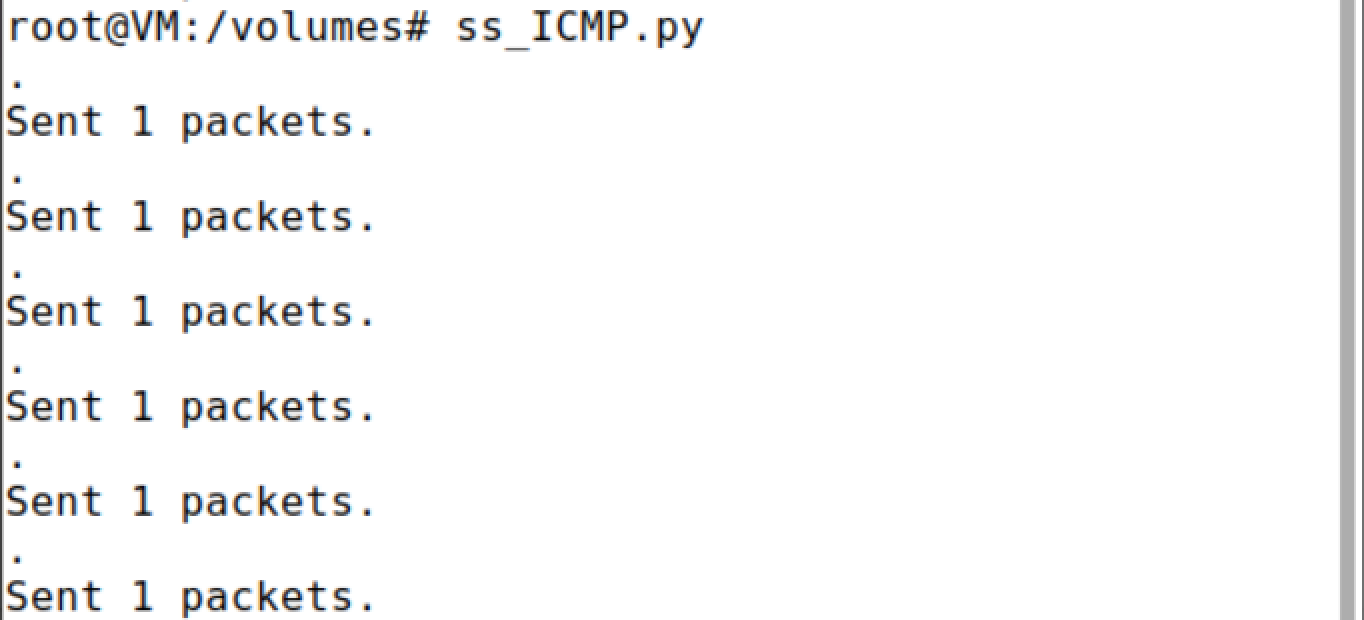
（2）代码：



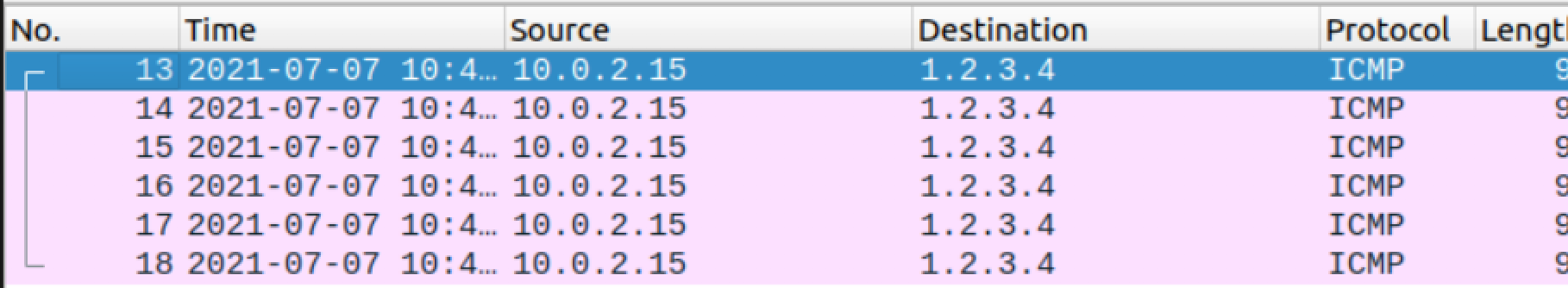
ss\_ICMP.py（中途做过修改）

（3）程序执行情况：

（3.1）ping一个外网中不存在的IP地址，如1.2.3.4，发现进程嗅探到了ICMP请求报文并发送ICMP响应报文，这是因为主机ping外网IP地址，直接发送ICMP请求报文于路由器，所以进程嗅探到了ICMP请求报文。但通过Wireshark发现，伪造的ICMP响应报文并没有被捕获。

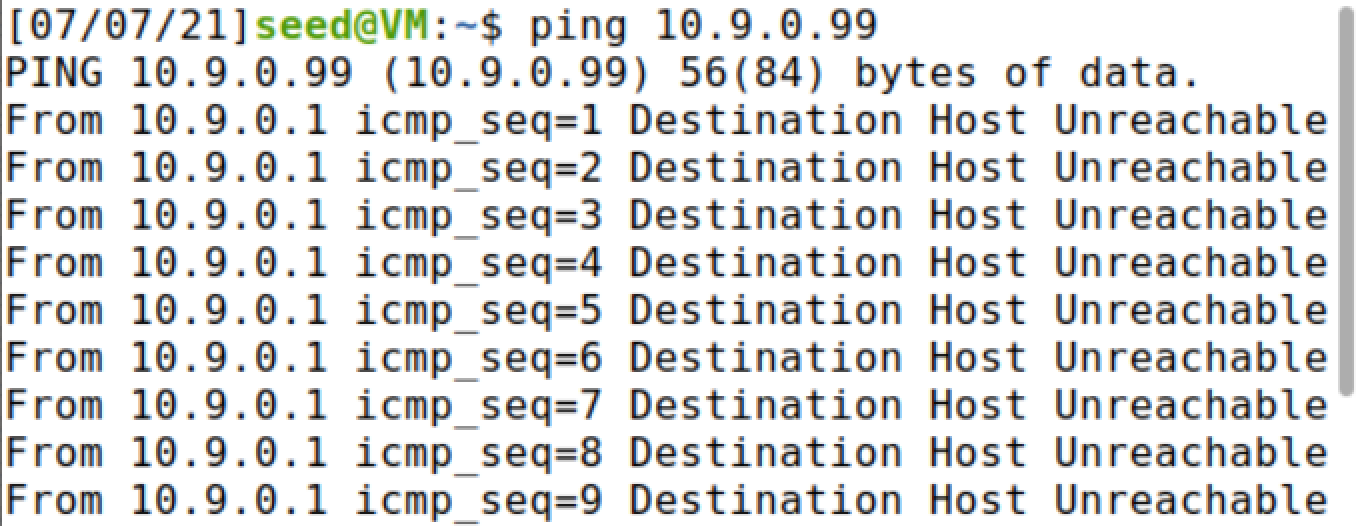


ping 1.2.3.4后发送伪造的ICMP响应报文

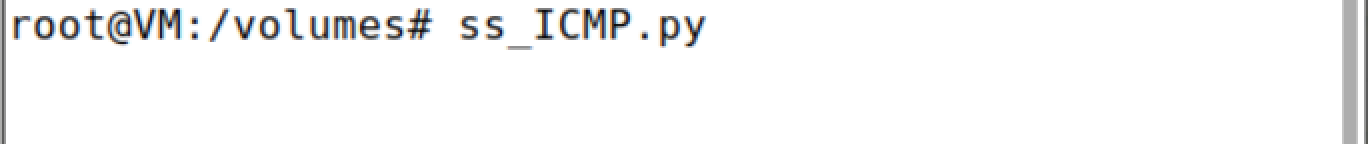


Wireshark没有捕获伪造的ICMP响应报文

（3.2）ping一个内网中不存在的IP地址，如10.9.0.99，发现进程并没有嗅探到ICMP请求报文。通过Wireshark发现，确实没有ICMP请求报文发出，发出的都是ARP报文，这是因为ping内网IP地址，不需要经过路由，需要知道目的主机的MAC地址，即可将报文发送给内网主机。所以先是ARP报文，再是ICMP请求报文，ARP报文无响应，则无法发出ICMP请求报文，则进程就不会嗅探到ICMP请求报文。



不存在的内网IP地址

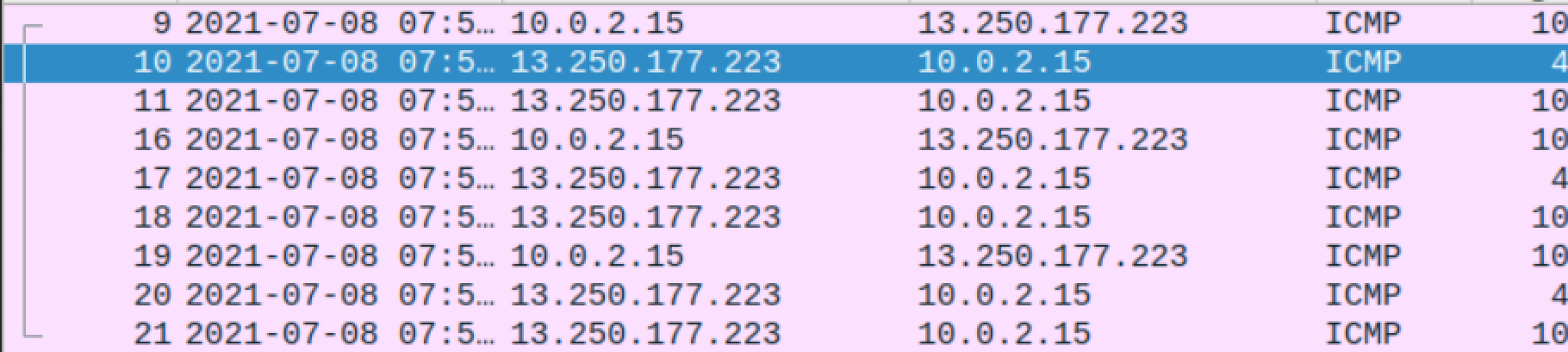


ping 10.9.0.99没有嗅探到ICMP请求报文



ping 10.9.0.99后一直发送ARP报文

（3.3）ping一个外网中存在的IP地址，如github.com，发现进程嗅探到了ICMP请求报文并发送ICMP响应报文，这是因为主机ping外网IP地址，直接发送ICMP请求报文于路由器，所以进程嗅探到了ICMP请求报文。通过Wireshark发现，一个ICMP请求报文有两个ICMP响应报文，一个是伪造的ICMP响应报文，一个是真实的ICMP报文。



ping 外网存在的地址