# Lab 1 Packet Sniffing and Spoofing Lab

57118115 陈烨

## Task 1.1 Sniffing Packets

### Task 1.1A

```
[07/05/21]seed@VM:~/.../Labsetup$ pwd
/home/seed/Desktop/Labs_20.04/Network Security/Packet Sniffing and Spoofing Lab/Labsetup
[07/05/21]seed@VM:~/.../Labsetup$ ls
docker-compose.yml  sniffer.py  volumes
```

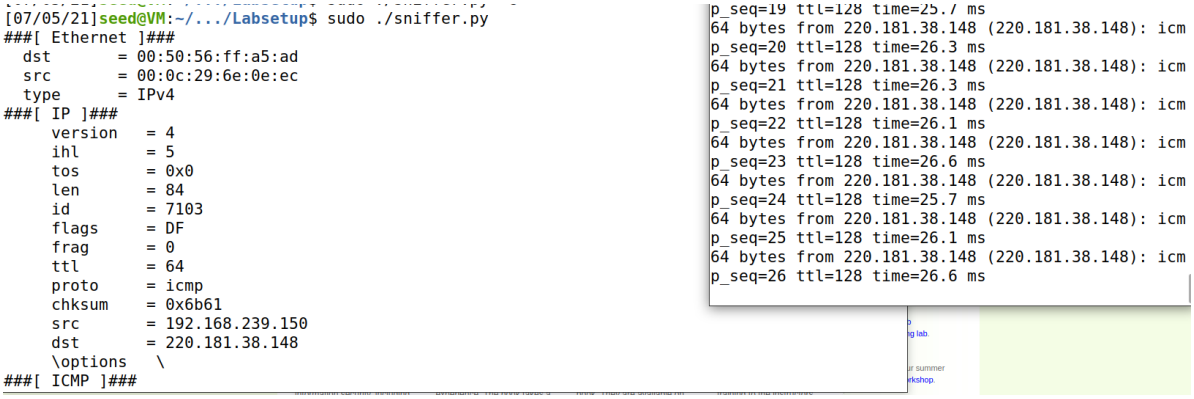
sniffer.py

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

def print_pkt(pkt):
  pkt.show()

#pkt = sniff(iface='br-68c809615df2',filter='icmp',prn=print_pkt)
#pkt = sniff(filter='icmp',prn=print_pkt)
#ens33
pkt = sniff(iface='ens33',filter='icmp',prn=print_pkt)

```

其中虚拟机采用NAT，网卡名称使用ifconfig查看

运行sniffer.py `seed@VM:~/.../Labsetup$ sudo ./sniffer.py`

使用另一terminal执行ping命令，sniffer.py获取报文




观察到root权限下可以正常抓包，普通用户权限下显示权限不够

```
[07/05/21]seed@VM:~/.../Labsetup$  ./sniffer.py
Traceback (most recent call last):
  File "./sniffer.py", line 10, in <module>
    pkt = sniff(iface='ens33',filter='icmp',prn=print_pkt)
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 1036, in sniff
    sniffer._run(*args, **kwargs)
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 906, in _run
    sniff_sockets[L2socket(type=ETH_P_ALL, iface=iface,
  File "/usr/local/lib/python3.8/dist-packages/scapy/arch/linux.py", line 398, in __init__
    self.ins = socket.socket(socket.AF_PACKET, socket.SOCK_RAW, socket.htons(type))  # noqa: E
501
  File "/usr/lib/python3.8/socket.py", line 231, in __init__
    _socket.socket.__init__(self, family, type, proto, fileno)
PermissionError: [Errno 1] Operation not permitted
```

## Task 1.1B BPF

应用BPF语法来对scapy抓包进行过滤，分别满足

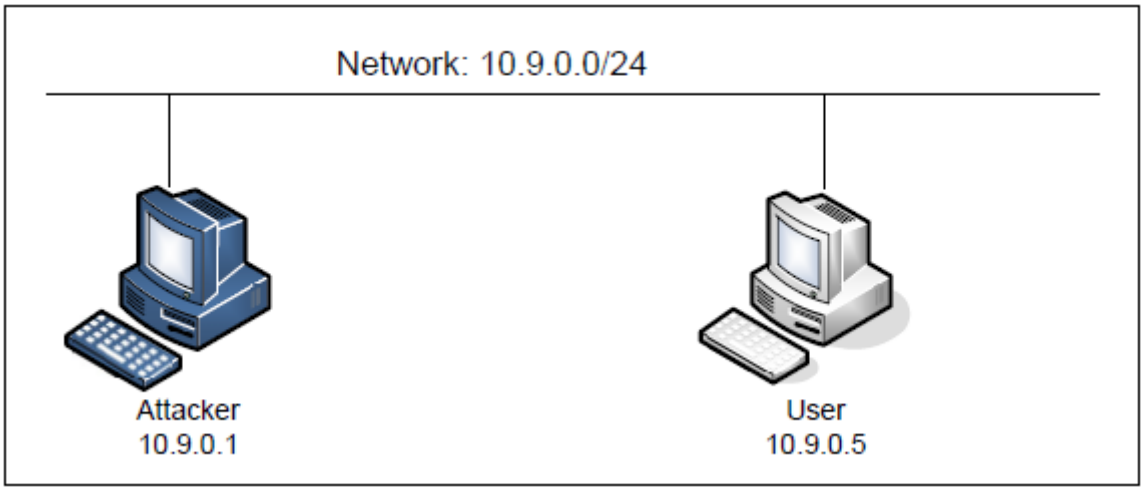
- **Capture only the ICMP packet**

filter与原代码一致，在这里使用docker

```
[07/05/21]seed@VM:~/.../Labsetup$ ls
docker-compose.yml  sniffer.py  volumes
[07/05/21]seed@VM:~/.../Labsetup$ dockps
[07/05/21]seed@VM:~/.../Labsetup$ dcbuild
attacker uses an image, skipping
host uses an image, skipping
[07/05/21]seed@VM:~/.../Labsetup$ dcup
Starting seed-attacker ... done
Starting host-10.9.0.5 ... done
Attaching to seed-attacker, host-10.9.0.5
```

```
[07/05/21]seed@VM:~/.../Packet Sniffing and Spoofing Lab$ cd Labs
etup/
[07/05/21]seed@VM:~/.../Labsetup$ ls
docker-compose.yml  sniffer.py  volumes
[07/05/21]seed@VM:~/.../Labsetup$ dockps
362310dfaef7  host-10.9.0.5
4ae6a2c8d91b  seed-attacker
[07/05/21]seed@VM:~/.../Labsetup$ docksh 4a
root@VM:/# ping 10.9.0.5
PING 10.9.0.5 (10.9.0.5) 56(84) bytes of data.
^C
--- 10.9.0.5 ping statistics ---
12 packets transmitted, 0 received, 100% packet loss, time 11273m
s

root@VM:/#
```

开启host和attacker并在attacker上ping主机host的IP




sniffer.py获取报文如下：

```
###[ Ethernet ]###
  dst       = 00:50:56:ff:a5:ad
  src       = 00:0c:29:6e:0e:ec
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 12878
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x4e0e
     src       = 192.168.239.150
     dst       = 10.9.0.5
     \options   \
###[ ICMP ]###
        type      = echo-request
        code      = 0
        chksum    = 0xd05c
        id        = 0x5
        seq       = 0x6
###[ Raw ]###
           load      = 'X\xba\xe2`\x00\x00\x00\x00#\xaa\n\x00\x00\x00\x00\x00\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f !"#$%&\'()*+,-./01234567'

```

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

将filter写为

```
pkt = sniff(iface='ens33',filter='src host 192.168.10.2 and tcp dst port 23',prn=print_pkt)
```

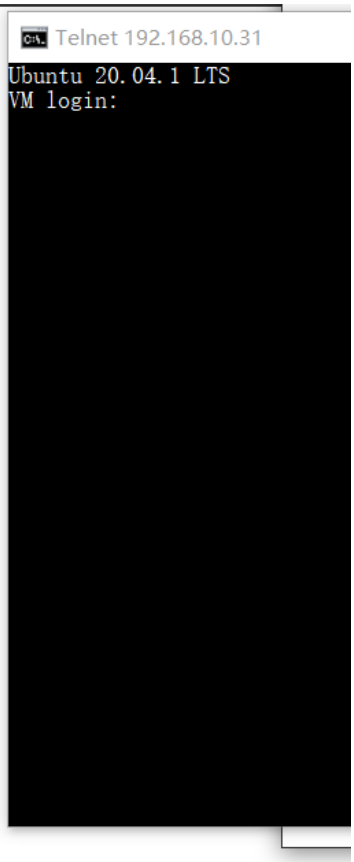
```
无线局域网适配器 WLAN:

   连接特定的 DNS 后缀 . . . . . : wifi
   本地链接 IPv6 地址. . . . . : fe80::38d0:ba3d:5ab7:4ce3%19
   IPv4 地址 . . . . . : 192.168.10.2
   子网掩码 . . . . . : 255.255.255.0
   默认网关. . . . . : 192.168.10.1
```

虚拟机改为桥接，IP为

```
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.10.31  netmask 255.255.255.0  broadcast 192.
168.10.255
        inet6 fe80::22ff:218e:cee2:425d  prefixlen 64  scopeid 0x
20<link>
        ether 00:0c:29:6e:0e:ec  txqueuelen 1000  (Ethernet)
```

尝试telnet连接，sniff.py抓到流量

```
           load      = "\xff\xfa'\x00\xff\xf0"

###[ Ethernet ]###
  dst       = 00:0c:29:6e:0e:ec
  src       = e8:6f:38:1a:09:67
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 50
     id        = 12685
     flags     = DF
     frag      = 0
     ttl       = 128
     proto     = tcp
     chksum    = 0x33c7
     src       = 192.168.10.2
     dst       = 192.168.10.31
     \options   \
###[ TCP ]###
        sport     = 65058
        dport     = telnet
        seq       = 3015402091
```




```
###[ Ethernet ]###
  dst       = 00:0c:29:6e:0e:ec
  src       = e8:6f:38:1a:09:67
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 50
     id        = 12685
     flags     = DF
     frag      = 0
     ttl       = 128
     proto     = tcp
     chksum    = 0x33c7
```

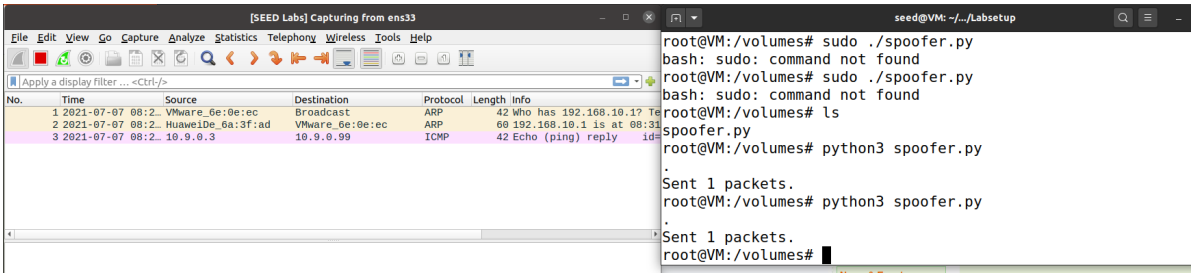
```
     src       = 192.168.10.2
     dst       = 192.168.10.31
     \options   \
###[ TCP ]###
        sport     = 65058
        dport     = telnet
        seq       = 3015402091
        ack       = 887869522
        dataofs   = 5
        reserved  = 0
        flags     = PA
        window    = 4106
        chksum    = 0x4424
        urgptr    = 0
        options   = []
###[ Raw ]###
           load      = '\xff\xfa\x18\x00ANSI\xff\xf0'

```

- **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.**

  当抓取SEED所在子网时，能够抓取到包, filter为"net 192.168.10" ，


## Task 1.2: Spoofing ICMP Packets

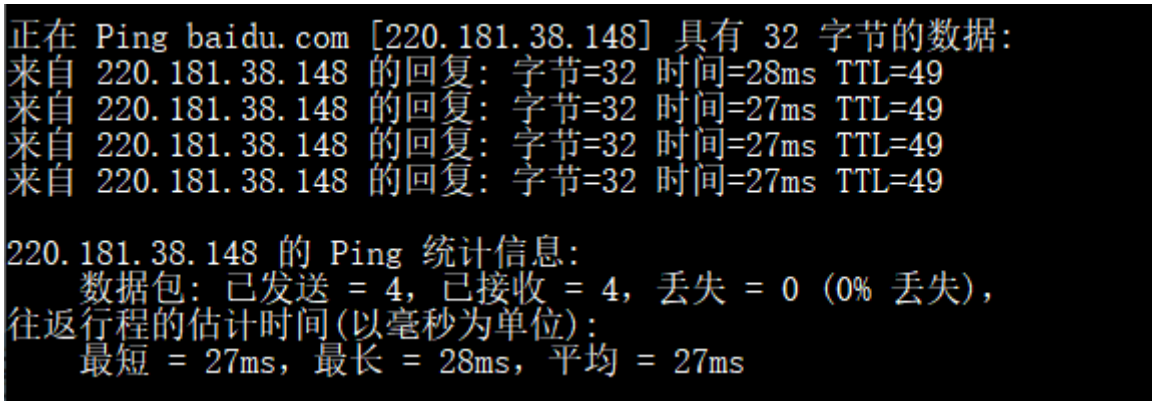
```
#!/usr/bin/python3

from scapy.all import *

a = IP()
a.src = '10.9.0.3'
a.dst = '10.9.0.99'
b = ICMP(type=0)
p = a/b
send(p)

```

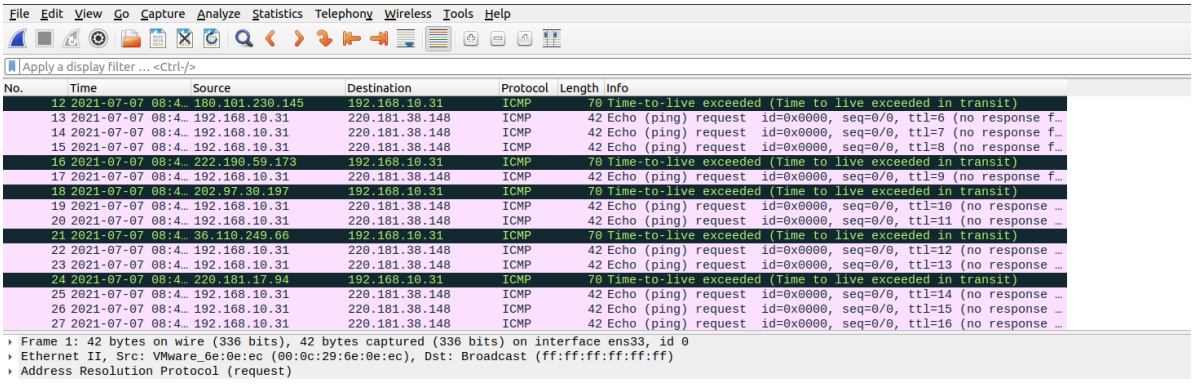



## Task 1.3: Traceroute

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

ttl = 1
while True:
  a = IP()
  a.dst = '220.181.38.148'
  a.ttl = ttl
  b = ICMP()
  send(a/b)
  ttl += 1
```

这里选取的是baidu.com的IP地址




在wireshark中查看得到虚拟机到目标IP地址的路由跳数为13

## Task 1.4: Sniffing and-then Spoofing

通过捕获ICMP报文，并将其源宿地址对调，并设置ICMP类型为Reply，再发出后，就可以伪造ICMP的reply。

```python
from scapy.all import *

def spoof_pkt(pkt):
  if ICMP in pkt and pkt[ICMP].type == 8:
    ip = IP(src=pkt[IP].dst, dst=pkt[IP].src, ihl=pkt[IP].ihl)
    icmp = ICMP(type=0, id=pkt[ICMP].id, seq=pkt[ICMP].seq)
    data = pkt[Raw].load
    newpkt = ip/icmp/data
    send(newpkt)

pkt = sniff(filter='icmp', prn=spoof_pkt)

```

启动程序之前执行ping

```
root@VM:/volumes# [07/07/21]seed@VM:~/.../Labsetup$ cd
[07/07/21]seed@VM:~$ ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
^C
--- 1.2.3.4 ping statistics ---
8 packets transmitted, 0 received, 100% packet loss, time 7164ms

[07/07/21]seed@VM:~$ ping 10.9.0.99
PING 10.9.0.99 (10.9.0.99) 56(84) bytes of data.
^C
--- 10.9.0.99 ping statistics ---
9 packets transmitted, 0 received, 100% packet loss, time 8180ms

[07/07/21]seed@VM:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=112 time=37.6 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=112 time=37.7 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=112 time=38.0 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=112 time=37.6 ms
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3008ms
rtt min/avg/max/mdev = 37.577/37.722/38.039/0.185 ms
```

启动程序之后：

```
[07/07/21]seed@VM:~$ ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
64 bytes from 1.2.3.4: icmp_seq=1 ttl=64 time=23.3 ms
64 bytes from 1.2.3.4: icmp_seq=2 ttl=64 time=21.4 ms
64 bytes from 1.2.3.4: icmp_seq=3 ttl=64 time=17.1 ms
64 bytes from 1.2.3.4: icmp_seq=4 ttl=64 time=28.7 ms
```

```
. .
root@362310dfaef7:/# ping 10.9.0.99
PING 10.9.0.99 (10.9.0.99) 56(84) bytes of data.
From 10.9.0.5 icmp_seq=1 Destination Host Unreachable
From 10.9.0.5 icmp_seq=2 Destination Host Unreachable
From 10.9.0.5 icmp_seq=3 Destination Host Unreachable
From 10.9.0.5 icmp_seq=4 Destination Host Unreachable
From 10.9.0.5 icmp_seq=5 Destination Host Unreachable
From 10.9.0.5 icmp_seq=6 Destination Host Unreachable
From 10.9.0.5 icmp_seq=7 Destination Host Unreachable
From 10.9.0.5 icmp_seq=8 Destination Host Unreachable
From 10.9.0.5 icmp_seq=9 Destination Host Unreachable
```

```
[07/07/21]seed@VM:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=64 time=15.8 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=112 time=59.5 ms (DUP!)
64 bytes from 8.8.8.8: icmp_seq=2 ttl=64 time=16.0 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=112 time=146 ms (DUP!)
64 bytes from 8.8.8.8: icmp_seq=3 ttl=64 time=26.3 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=64 time=26.4 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=64 time=27.6 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=64 time=28.7 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=64 time=20.5 ms
```

观察到程序在发ICMP报文：

```
[07/07/21]seed@VM:~/.../volumes$ sudo python3 sniff_spoof_icmp.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

对于1.2.3.4，在互联网中不存在，该程序能够欺骗容器10.9.0.5，伪造返回的报文；

对于10.9.0.99，该程序无法完成欺骗，10.9.0.5不会通过网关来寻址，而是使用ARP广播

对于Internet上真实存在的主机8.8.8.8，该程序能够产生作用，同时接受到真正的返回报文和虚假的返回报文，且两者时延具有较大的区别。