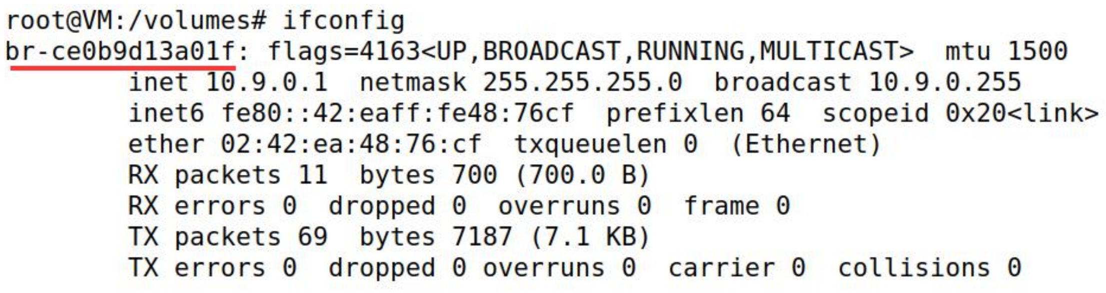
Packet Sniffifing and Spoofifing Lab

57118214 陈佳杰

Lab Task Set 1: Using Scapy to Sniff and Spoof Packets

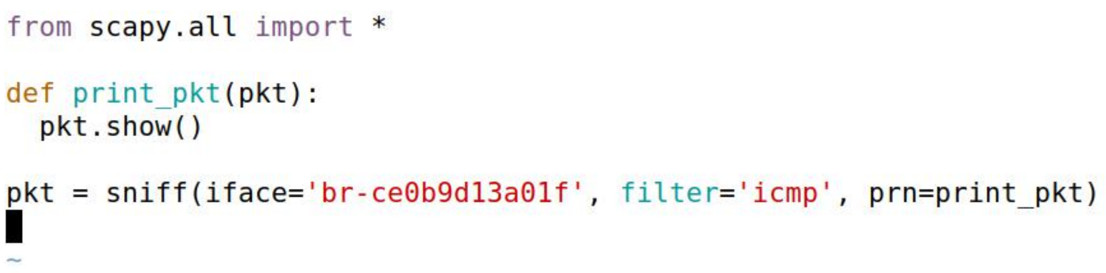
Task 1.1: Sniffifing Packets

获取接口名称



```
root@VM:/volumes# ifconfig
br-ce0b9d13a01f: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.1  netmask 255.255.255.0  broadcast 10.9.0.255
        inet6 fe80::42:eaff:fe48:76cf  prefixlen 64  scopeid 0x20<link>
        ether 02:42:ea:48:76:cf  txqueuelen 0  (Ethernet)
        RX packets 11  bytes 700 (700.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 69  bytes 7187 (7.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

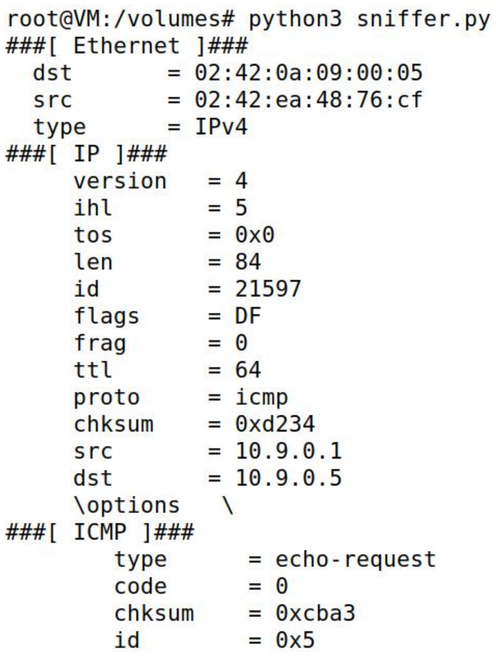
使用 Scapy 进行包嗅探的 Python 程序



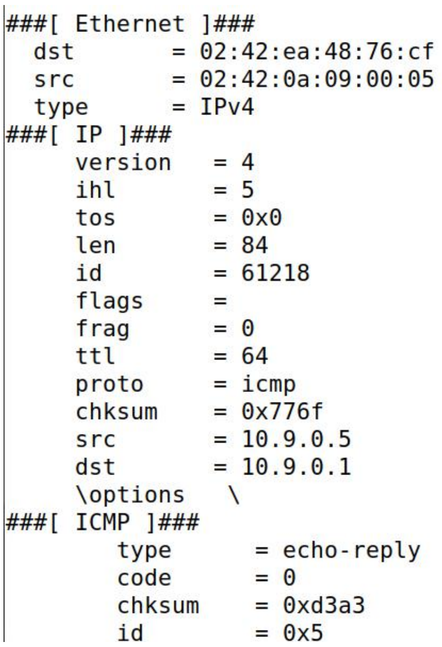
```
from scapy.all import *

def print_pkt(pkt):
  pkt.show()

pkt = sniff(iface='br-ce0b9d13a01f', filter='icmp', prn=print_pkt)

~
```

Task 1.1A.

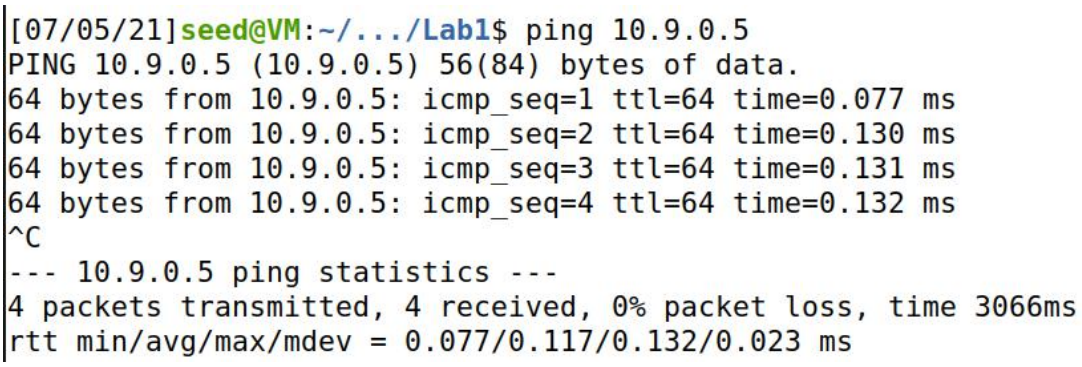
在 docker 中运行 sniffer.py，同时 ping 10.9.0.5，可以嗅探到数据包，结果如下所示。



```
root@VM:/volumes# python3 sniffer.py
###[ Ethernet ]###
  dst       = 02:42:0a:09:00:05
  src       = 02:42:ea:48:76:cf
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 21597
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0xd234
     src       = 10.9.0.1
     dst       = 10.9.0.5
     \options   \
###[ ICMP ]###
        type      = echo-request
        code      = 0
        chksum    = 0xcba3
        id        = 0x5
```

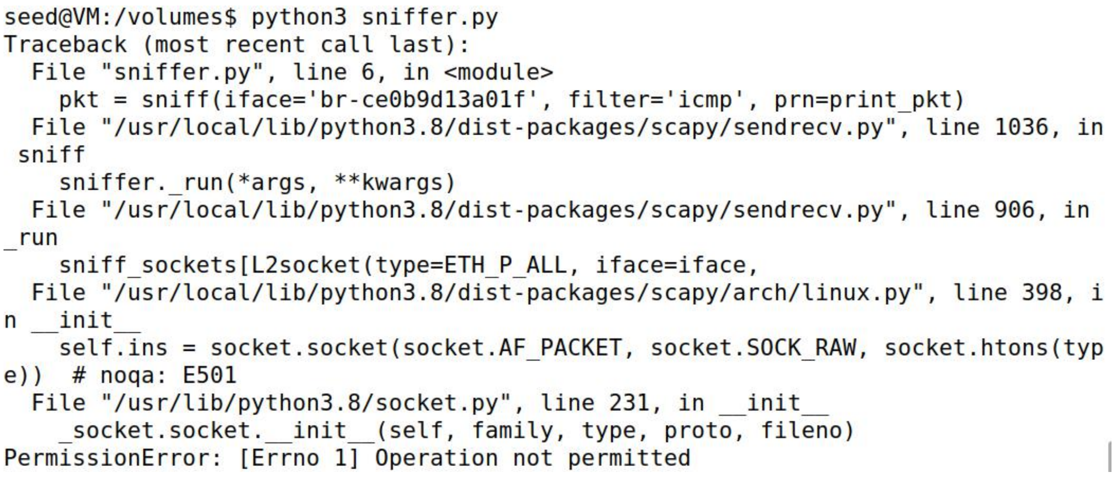


```
###[ Ethernet ]###
  dst       = 02:42:ea:48:76:cf
  src       = 02:42:0a:09:00:05
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 61218
     flags     =
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x776f
     src       = 10.9.0.5
     dst       = 10.9.0.1
     \options   \
###[ ICMP ]###
        type      = echo-reply
        code      = 0
        chksum    = 0xd3a3
        id        = 0x5
```



```
[07/05/21]seed@VM:~/.../Lab1$ ping 10.9.0.5
PING 10.9.0.5 (10.9.0.5) 56(84) bytes of data.
64 bytes from 10.9.0.5: icmp_seq=1 ttl=64 time=0.077 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=64 time=0.130 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=64 time=0.131 ms
64 bytes from 10.9.0.5: icmp_seq=4 ttl=64 time=0.132 ms
^C
--- 10.9.0.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3066ms
rtt min/avg/max/mdev = 0.077/0.117/0.132/0.023 ms
```

用 seed 用户来运行 sniffer.py，发现无法运行，因为 sniff 函数需要较高的权限才能运行。



```
seed@VM:/volumes$ python3 sniffer.py
Traceback (most recent call last):
  File "sniffer.py", line 6, in <module>
    pkt = sniff(iface='br-ce0b9d13a01f', 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, i
n __init__
    self.ins = socket.socket(socket.AF_PACKET, socket.SOCK_RAW, socket.htons(typ
e))  # noqa: E501
  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
```

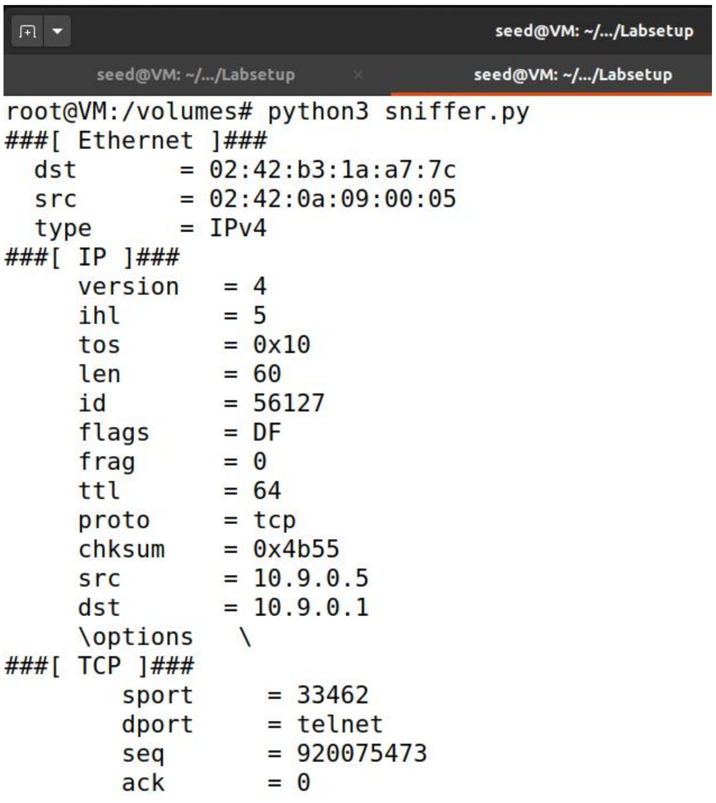
Task 1.1B.

①Capture only the ICMP packet

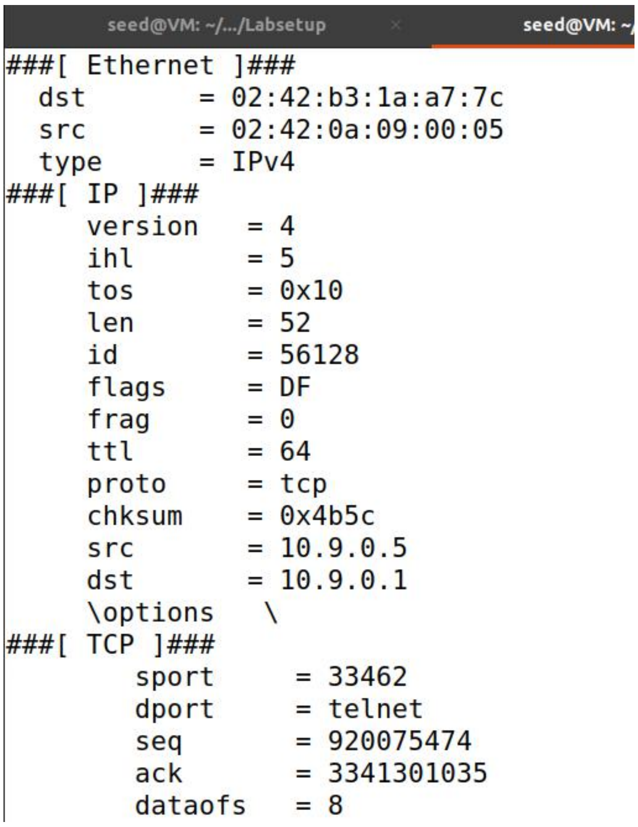
实验内容与 Task 1.1A.一样，filter='icmp'

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

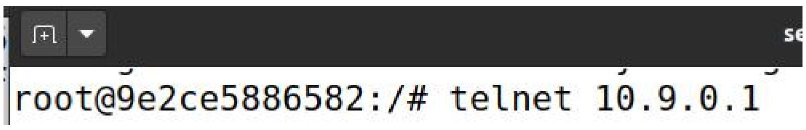
设置过滤器为 filter='src host 10.9.0.5 and tcp dst port 23'
运行 sniffer.py 后在 host 中 telnet 10.9.0.1 可获取以下报文



```
root@VM:/volumes# python3 sniffer.py
###[ Ethernet ]###
  dst       = 02:42:b3:1a:a7:7c
  src       = 02:42:0a:09:00:05
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x10
     len       = 60
     id        = 56127
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = tcp
     chksum    = 0x4b55
     src       = 10.9.0.5
     dst       = 10.9.0.1
     \options   \
###[ TCP ]###
        sport     = 33462
        dport     = telnet
        seq       = 920075473
        ack       = 0
```



```
###[ Ethernet ]###
  dst       = 02:42:b3:1a:a7:7c
  src       = 02:42:0a:09:00:05
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x10
     len       = 52
     id        = 56128
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = tcp
     chksum    = 0x4b5c
     src       = 10.9.0.5
     dst       = 10.9.0.1
     \options   \
###[ TCP ]###
        sport     = 33462
        dport     = telnet
        seq       = 920075474
        ack       = 3341301035
        dataofs   = 8
```



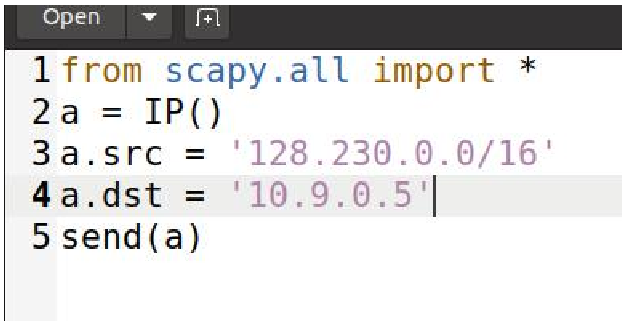
```
root@9e2ce5886582:/# telnet 10.9.0.1
```

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.

设置过滤器为 filter='net 128.230.0.0/16'

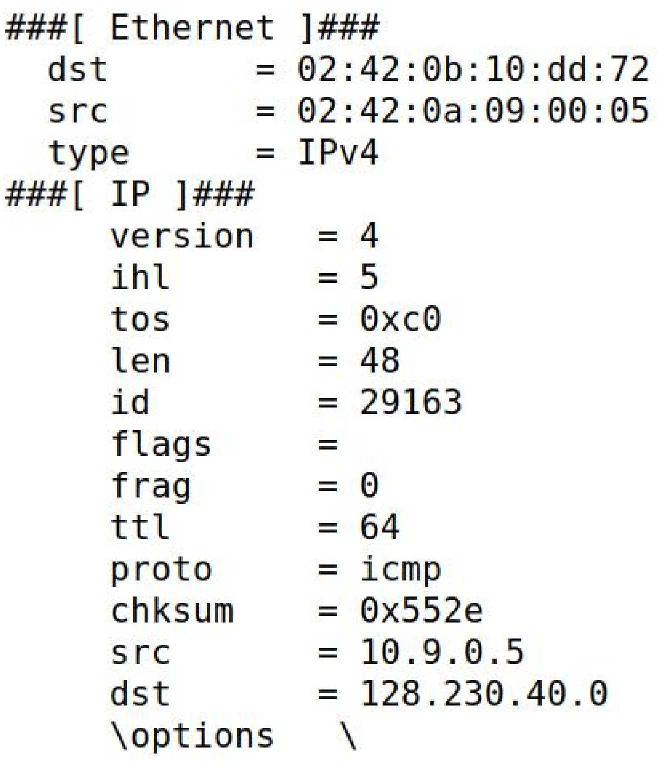
能捕获来自或转到 128.230.0.0/16 这个子网的数据包

构造发包程序



```python
from scapy.all import *
a = IP()
a.src = '128.230.0.0/16'
a.dst = '10.9.0.5'
send(a)
```

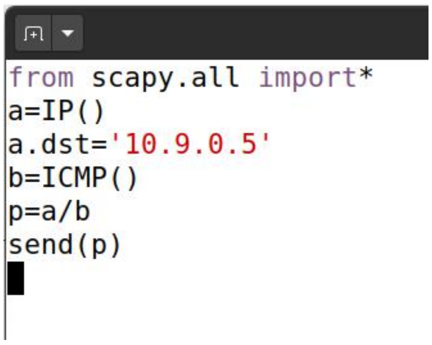
运行 sniffer.py 的结果



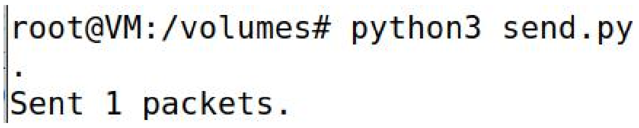
```
###[ Ethernet ]###
  dst       = 02:42:0b:10:dd:72
  src       = 02:42:0a:09:00:05
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0xc0
     len       = 48
     id        = 29163
     flags     =
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x552e
     src       = 10.9.0.5
     dst       = 128.230.40.0
     \options   \
```

Task 1.2: Spoofifing ICMP Packets

写发送数据包的程序 sned.py 并运行



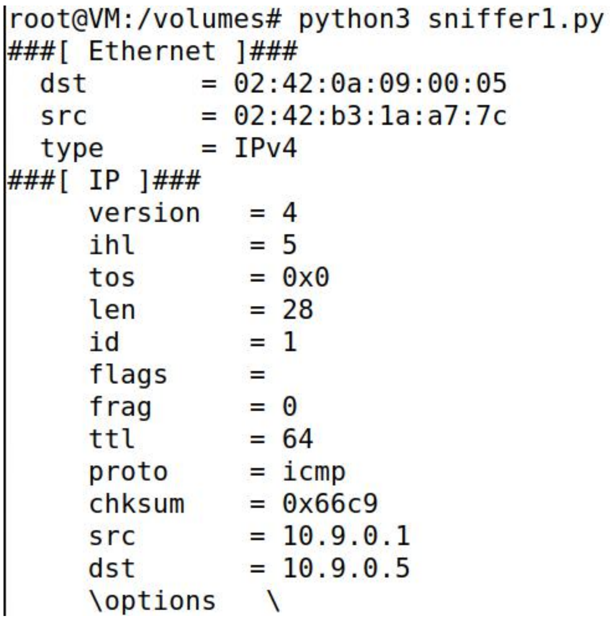
```
from scapy.all import*
a=IP()
a.dst='10.9.0.5'
b=ICMP()
p=a/b
send(p)
```



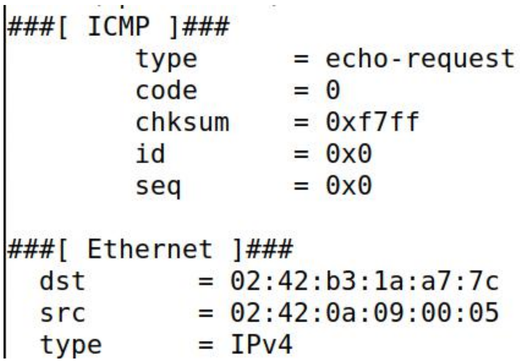
```
root@VM:/volumes# python3 send.py
.
Sent 1 packets.
```

此时发送一个 ICMP 数据包到 10.9.0.5

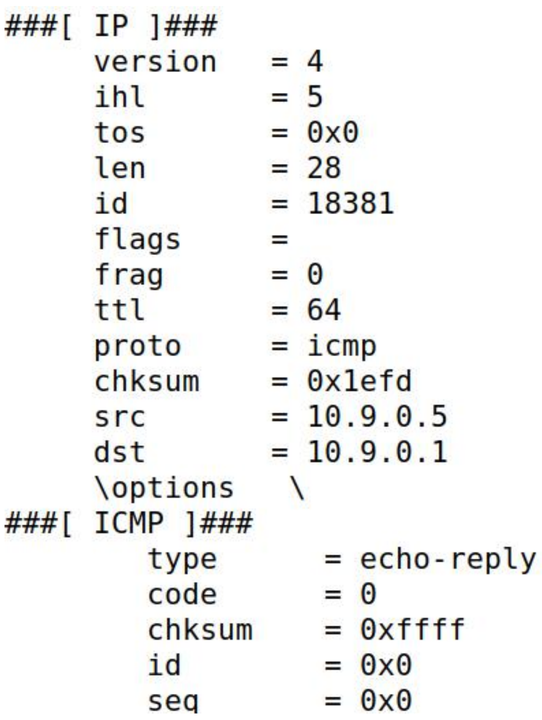
攻击者运行的嗅探程序能捕获到这个数据包



```
root@VM:/volumes# python3 sniffer1.py
###[ Ethernet ]###
  dst       = 02:42:0a:09:00:05
  src       = 02:42:b3:1a:a7:7c
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 28
     id        = 1
     flags     =
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x66c9
     src       = 10.9.0.1
     dst       = 10.9.0.5
     \options   \
```



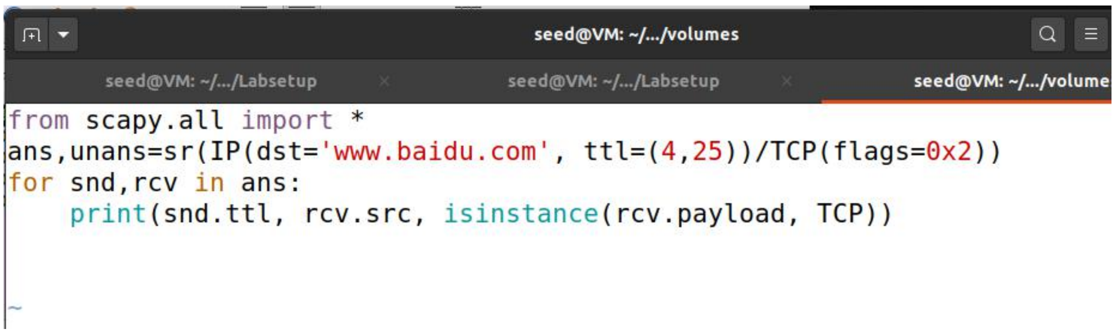
```
###[ ICMP ]###
        type      = echo-request
        code      = 0
        chksum    = 0xf7ff
        id        = 0x0
        seq       = 0x0

###[ Ethernet ]###
  dst       = 02:42:b3:1a:a7:7c
  src       = 02:42:0a:09:00:05
  type      = IPv4
```



```
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 28
     id        = 18381
     flags     =
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x1efd
     src       = 10.9.0.5
     dst       = 10.9.0.1
     \options   \
###[ ICMP ]###
        type      = echo-reply
        code      = 0
        chksum    = 0xffff
        id        = 0x0
        seq       = 0x0
```

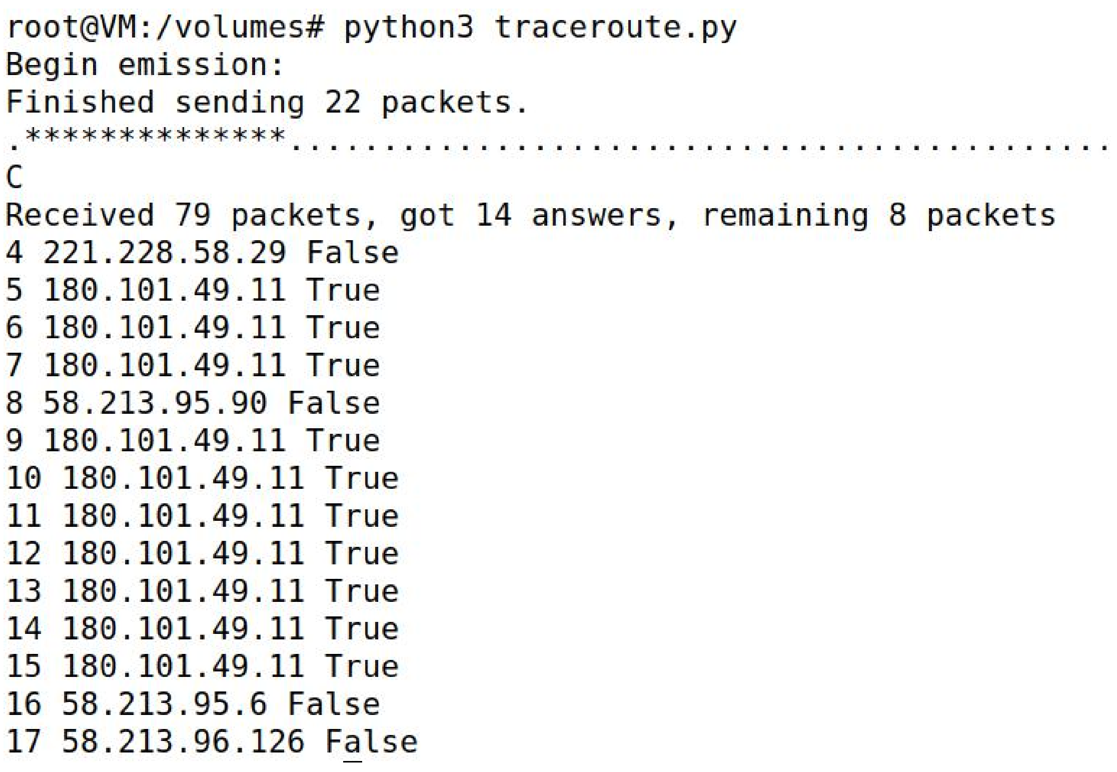
Task 1.3: Traceroute

写一个脚本来发送数据包



```
from scapy.all import *
ans,unans=sr(IP(dst='www.baidu.com', ttl=(4,25))/TCP(flags=0x2))
for snd,rcv in ans:
    print(snd.ttl, rcv.src, isinstance(rcv.payload, TCP))
```

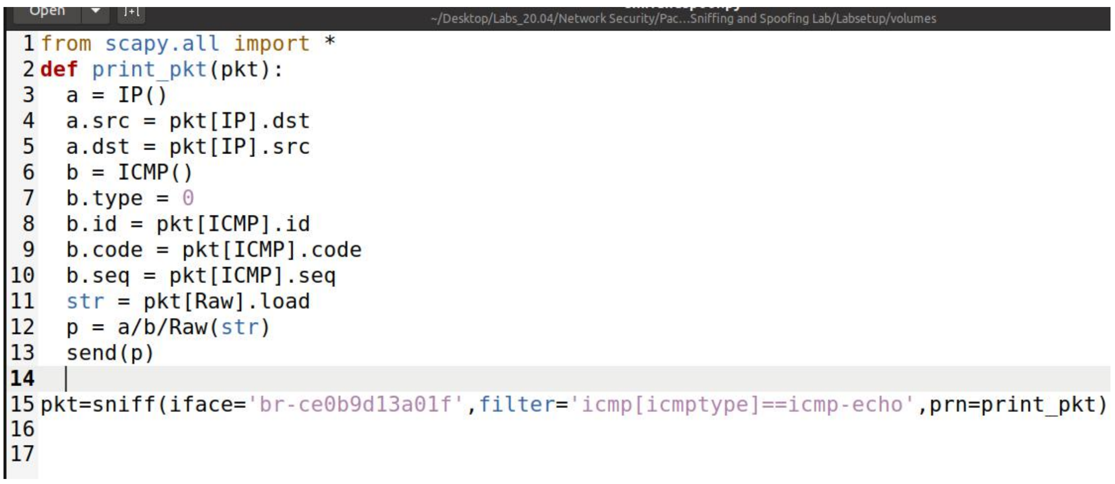
运行后得到的结果如下所示



```
root@VM:/volumes# python3 traceroute.py
Begin emission:
Finished sending 22 packets.
.**************..............................................
C
Received 79 packets, got 14 answers, remaining 8 packets
4 221.228.58.29 False
5 180.101.49.11 True
6 180.101.49.11 True
7 180.101.49.11 True
8 58.213.95.90 False
9 180.101.49.11 True
10 180.101.49.11 True
11 180.101.49.11 True
12 180.101.49.11 True
13 180.101.49.11 True
14 180.101.49.11 True
15 180.101.49.11 True
16 58.213.95.6 False
17 58.213.96.126 False
```

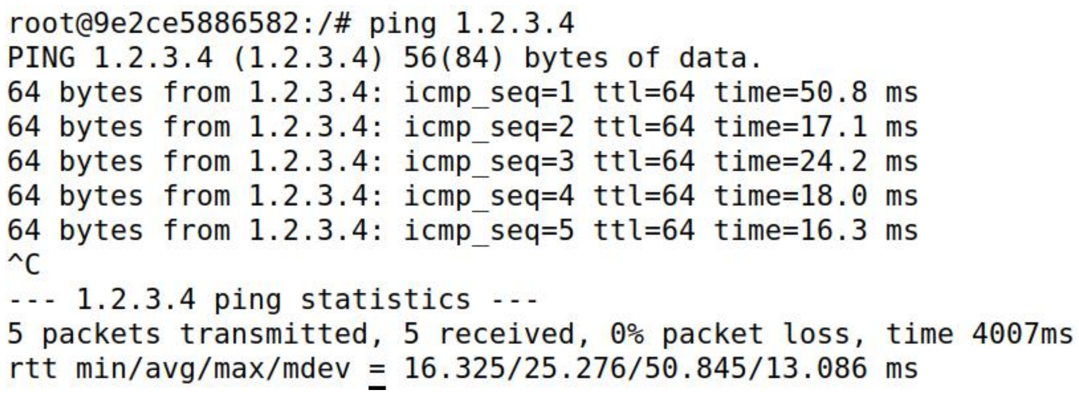
Task 1.4: Sniffifing and-then Spoofifing

编写如下程序


Open
~/Desktop/Labs_20.04/Network Security/Pac...Sniffing and Spoofing Lab/Labsetup/volumes

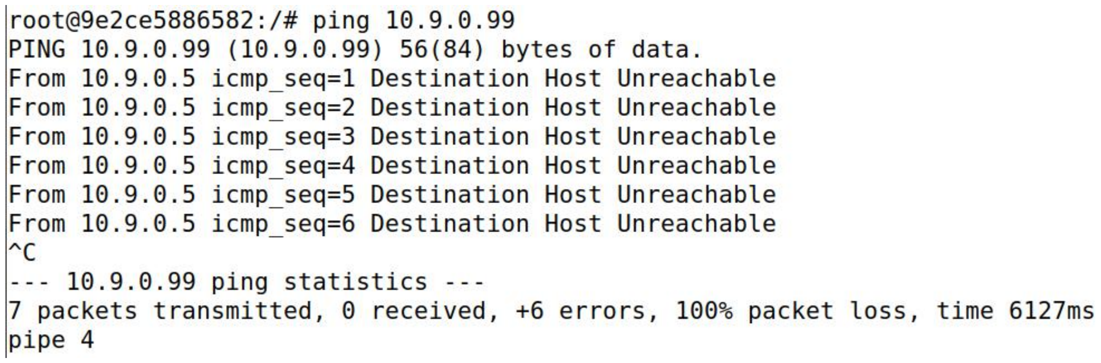
```
from scapy.all import *
def print_pkt(pkt):
  a = IP()
  a.src = pkt[IP].dst
  a.dst = pkt[IP].src
  b = ICMP()
  b.type = 0
  b.id = pkt[ICMP].id
  b.code = pkt[ICMP].code
  b.seq = pkt[ICMP].seq
  str = pkt[Raw].load
  p = a/b/Raw(str)
  send(p)

pkt=sniff(iface='br-ce0b9d13a01f',filter='icmp[icmptype]==icmp-echo',prn=print_pkt)
```

在运行程序的情况下用 HOST 依次 ping 1.2.3.4；10.9.0.99；8.8.8.8 三个 ip 得到的结果如下：



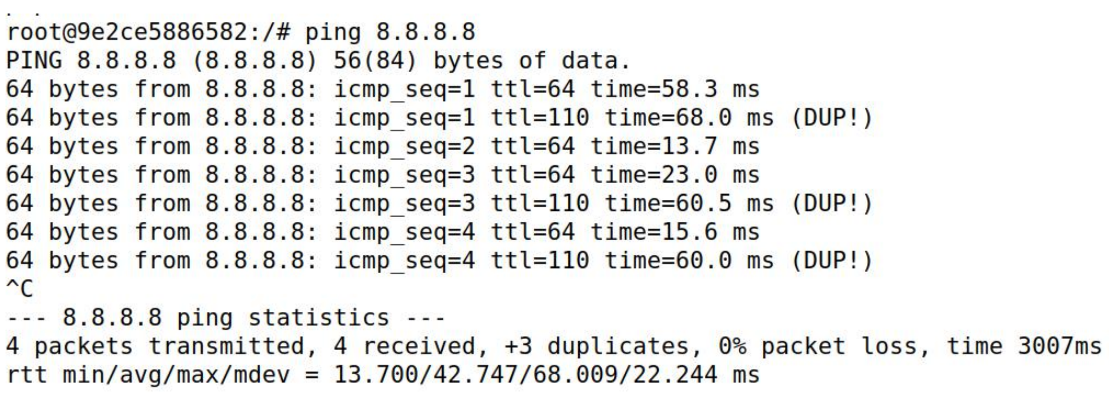
```
root@9e2ce5886582:/# ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
64 bytes from 1.2.3.4: icmp_seq=1 ttl=64 time=50.8 ms
64 bytes from 1.2.3.4: icmp_seq=2 ttl=64 time=17.1 ms
64 bytes from 1.2.3.4: icmp_seq=3 ttl=64 time=24.2 ms
64 bytes from 1.2.3.4: icmp_seq=4 ttl=64 time=18.0 ms
64 bytes from 1.2.3.4: icmp_seq=5 ttl=64 time=16.3 ms
^C
--- 1.2.3.4 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 16.325/25.276/50.845/13.086 ms
```

对不存在的地址伪造了报文并发送回去



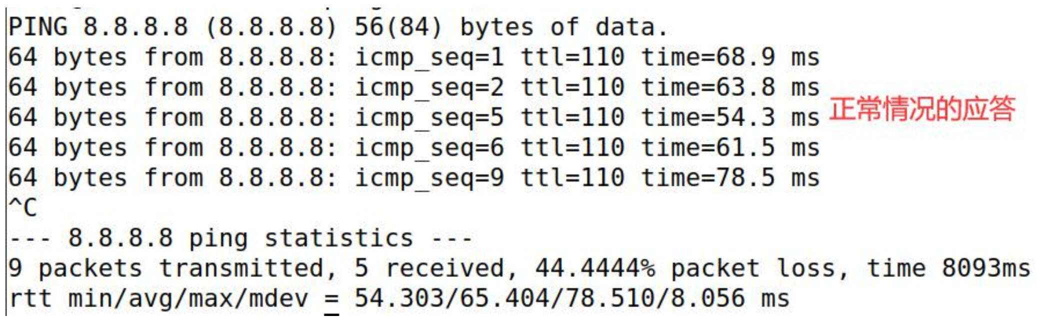
```
root@9e2ce5886582:/# ping 10.9.0.99
PING 10.9.0.99 (10.9.0.99) 56(84) bytes of data.
From 10.9.0.5 icmp_seq=1 Destination Host Unreachable
From 10.9.0.5 icmp_seq=2 Destination Host Unreachable
From 10.9.0.5 icmp_seq=3 Destination Host Unreachable
From 10.9.0.5 icmp_seq=4 Destination Host Unreachable
From 10.9.0.5 icmp_seq=5 Destination Host Unreachable
From 10.9.0.5 icmp_seq=6 Destination Host Unreachable
^C
--- 10.9.0.99 ping statistics ---
7 packets transmitted, 0 received, +6 errors, 100% packet loss, time 6127ms
pipe 4
```

在同一网段内不存在的地址没有 ICMP 回应报文



```
root@9e2ce5886582:/# 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=58.3 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=110 time=68.0 ms (DUP!)
64 bytes from 8.8.8.8: icmp_seq=2 ttl=64 time=13.7 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=64 time=23.0 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=110 time=60.5 ms (DUP!)
64 bytes from 8.8.8.8: icmp_seq=4 ttl=64 time=15.6 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=110 time=60.0 ms (DUP!)
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, +3 duplicates, 0% packet loss, time 3007ms
rtt min/avg/max/mdev = 13.700/42.747/68.009/22.244 ms
```

对一个存在的地址应答速度明显快于正常情况



```
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=110 time=68.9 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=110 time=63.8 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=110 time=54.3 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=110 time=61.5 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=110 time=78.5 ms
^C
--- 8.8.8.8 ping statistics ---
9 packets transmitted, 5 received, 44.4444% packet loss, time 8093ms
rtt min/avg/max/mdev = 54.303/65.404/78.510/8.056 ms
```