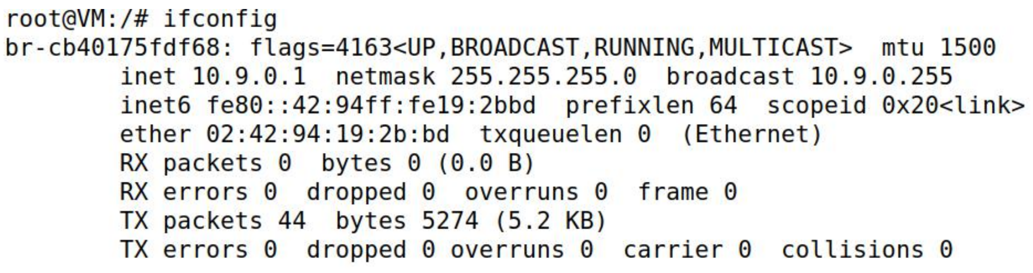
网络空间安全实训 实验报告

57118221 梅昊

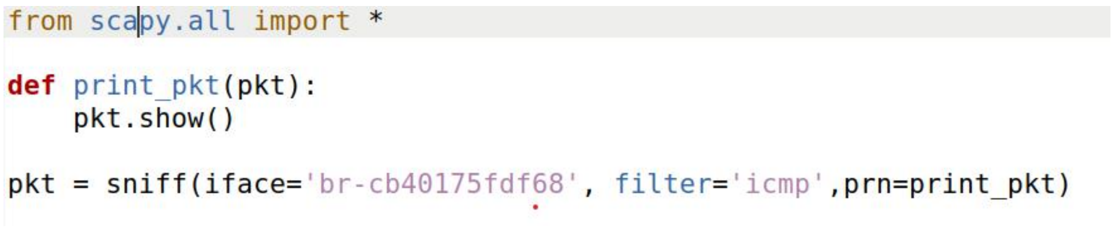
Task 1.1

1.首先获取网卡的名称



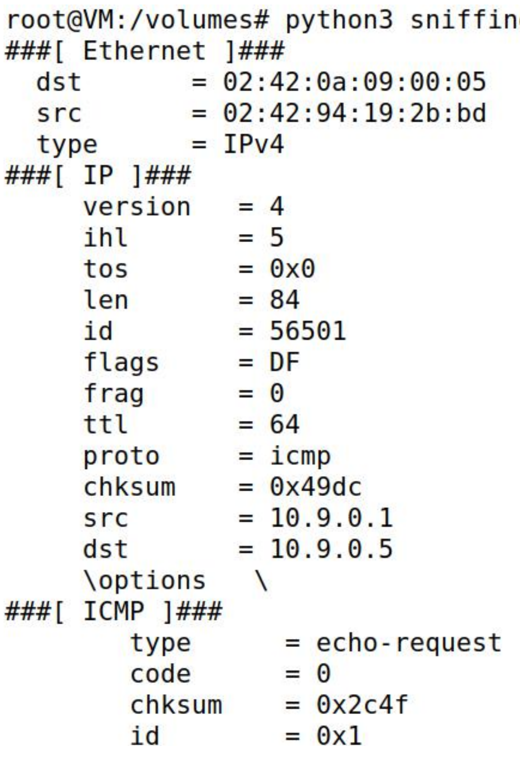
```
root@VM:/# ifconfig
br-cb40175fdf68: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.1  netmask 255.255.255.0  broadcast 10.9.0.255
        inet6 fe80::42:94ff:fe19:2bbd  prefixlen 64  scopeid 0x20<link>
        ether 02:42:94:19:2b:bd  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 44  bytes 5274 (5.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

2.编写包嗅探的程序



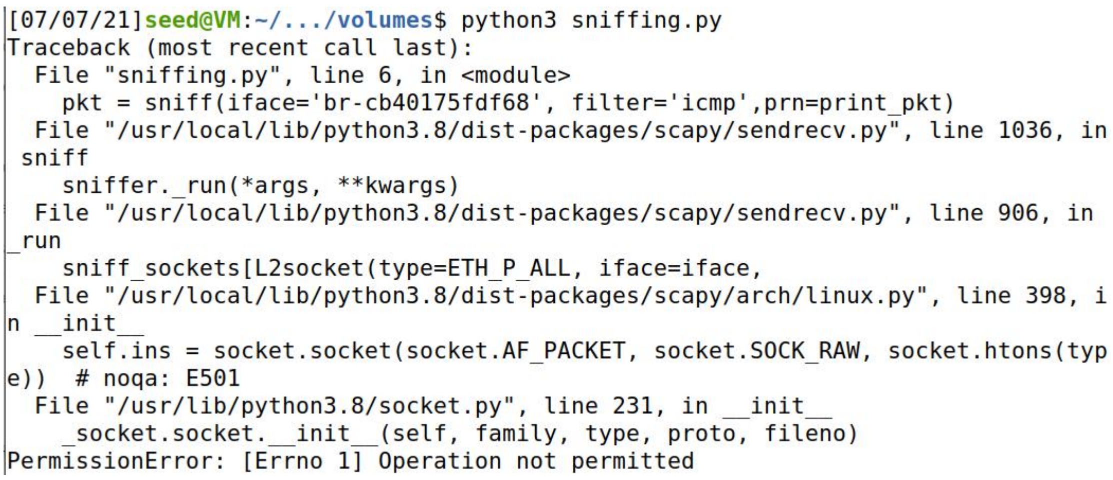
```python
from scapy.all import *

def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface='br-cb40175fdf68', filter='icmp',prn=print_pkt)
```

3.运行嗅探程序，同时 ping 10.9.0.5，观察嗅探到的数据包



```
root@VM:/volumes# python3 sniffing.py
###[ Ethernet ]###
  dst       = 02:42:0a:09:00:05
  src       = 02:42:94:19:2b:bd
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 56501
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x49dc
     src       = 10.9.0.1
     dst       = 10.9.0.5
     \options   \
###[ ICMP ]###
        type      = echo-request
        code      = 0
        chksum    = 0x2c4f
        id        = 0x1
```

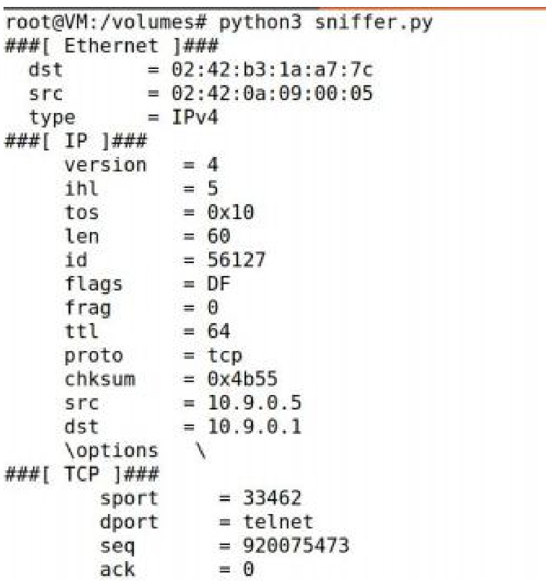
4.用 seed 用户，发现无法运行，需要更高的权限。



```
[07/07/21]seed@VM:~/.../volumes$ python3 sniffing.py
Traceback (most recent call last):
  File "sniffing.py", line 6, in <module>
    pkt = sniff(iface='br-cb40175fdf68', 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
```

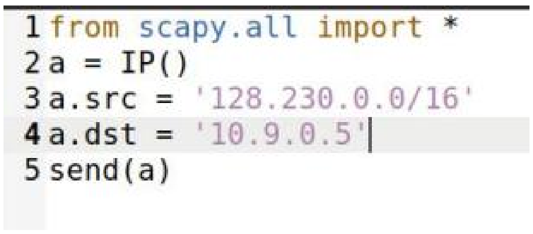
5. 设置过滤器，filter=’ src host 10.9.0.5 and tcp dst port 23’

可观察到如下报文

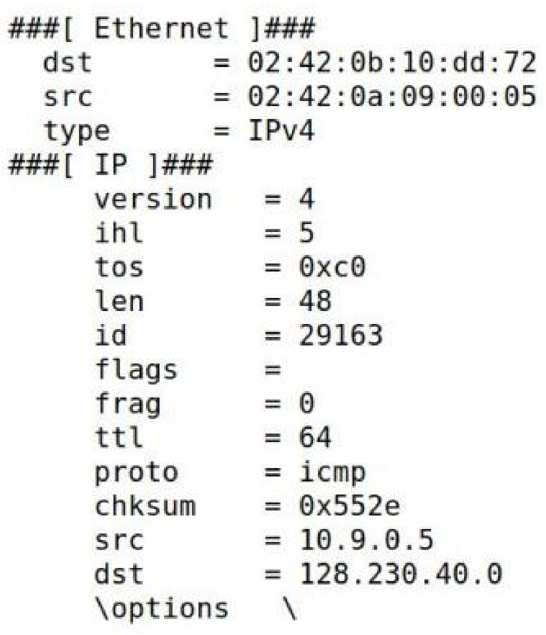


```
root@VM:/volumes# python3 sniffer.py
###[ Ethernet ]###
  dst       = 02:42:b3:1a:a7:7c
  src       = 02:42:0a:09:00:05
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x10
     len       = 60
     id        = 56127
     flags     = DF
     frag      = 0
     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
```

6. 设置过滤器，filter=’net 128.230.0.0/16’



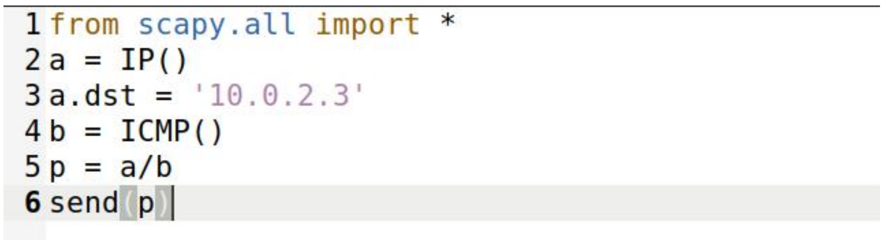
```
from scapy.all import *
a = IP()
a.src = '128.230.0.0/16'
a.dst = '10.9.0.5'
send(a)
```



```
###[ 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   \
```

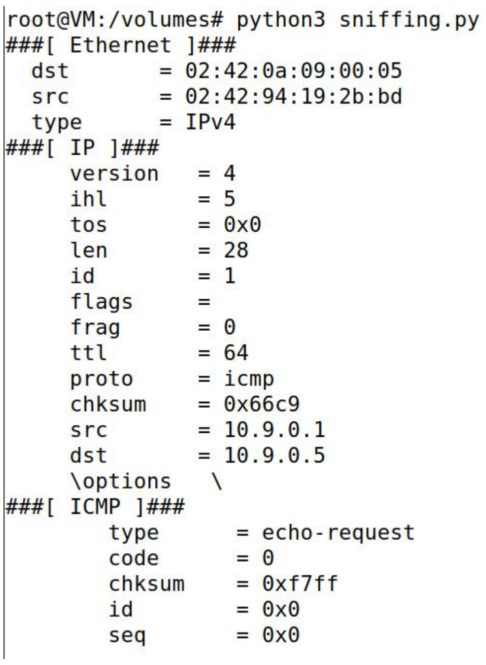
Task 1.2

编写发送数据包的程序



```python
from scapy.all import *
a = IP()
a.dst = '10.0.2.3'
b = ICMP()
p = a/b
send(p)
```

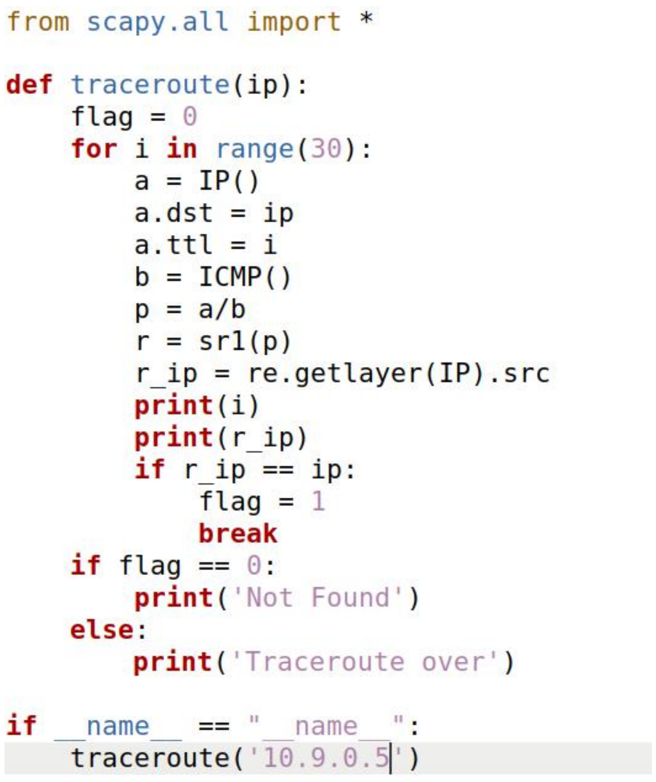
运行后能嗅探到



```
root@VM:/volumes# python3 sniffing.py
###[ Ethernet ]###
  dst       = 02:42:0a:09:00:05
  src       = 02:42:94:19:2b:bd
  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
```

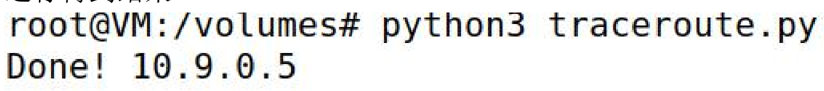
Task 1.3

首先编写 TraceRoute 相关代码



```
from scapy.all import *

def traceroute(ip):
    flag = 0
    for i in range(30):
        a = IP()
        a.dst = ip
        a.ttl = i
        b = ICMP()
        p = a/b
        r = sr1(p)
        r_ip = re.getlayer(IP).src
        print(i)
        print(r_ip)
        if r_ip == ip:
            flag = 1
            break
    if flag == 0:
        print('Not Found')
    else:
        print('Traceroute over')

if __name__ == "__name__":
    traceroute('10.9.0.5')
```

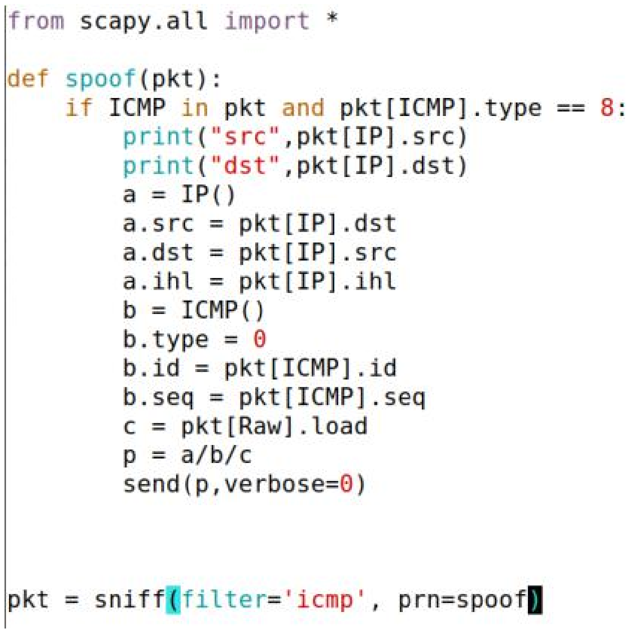
运行得到结果



```
root@VM:/volumes# python3 traceroute.py
Done! 10.9.0.5
```

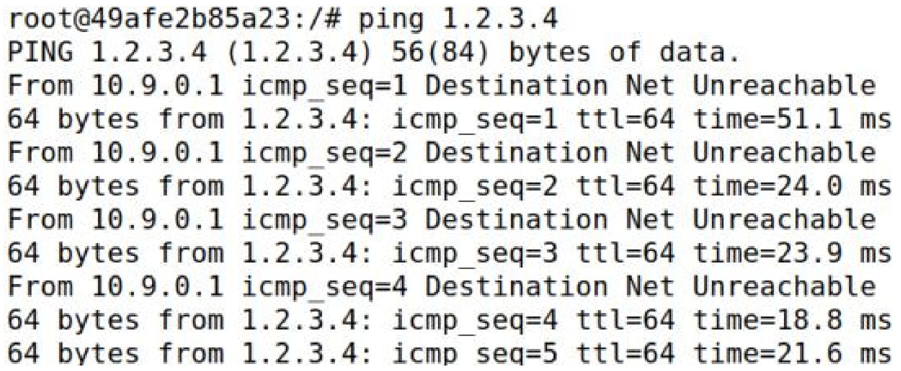
Task 1.4

首先编写相关欺骗代码



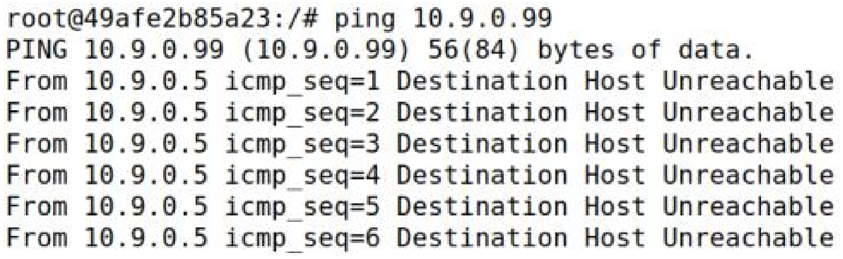
```
from scapy.all import *

def spoof(pkt):
    if ICMP in pkt and pkt[ICMP].type == 8:
        print("src",pkt[IP].src)
        print("dst",pkt[IP].dst)
        a = IP()
        a.src = pkt[IP].dst
        a.dst = pkt[IP].src
        a.ihl = pkt[IP].ihl
        b = ICMP()
        b.type = 0
        b.id = pkt[ICMP].id
        b.seq = pkt[ICMP].seq
        c = pkt[Raw].load
        p = a/b/c
        send(p,verbose=0)



pkt = sniff(filter='icmp', prn=spoof)
```

首先 ping 原本无法 ping 通的 1.2.3.4，运行欺骗代码后，可以得到结果：



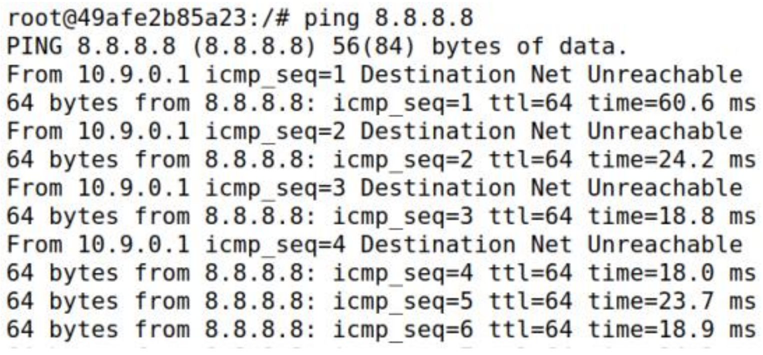
```
root@49afe2b85a23:/# ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
From 10.9.0.1 icmp_seq=1 Destination Net Unreachable
64 bytes from 1.2.3.4: icmp_seq=1 ttl=64 time=51.1 ms
From 10.9.0.1 icmp_seq=2 Destination Net Unreachable
64 bytes from 1.2.3.4: icmp_seq=2 ttl=64 time=24.0 ms
From 10.9.0.1 icmp_seq=3 Destination Net Unreachable
64 bytes from 1.2.3.4: icmp_seq=3 ttl=64 time=23.9 ms
From 10.9.0.1 icmp_seq=4 Destination Net Unreachable
64 bytes from 1.2.3.4: icmp_seq=4 ttl=64 time=18.8 ms
64 bytes from 1.2.3.4: icmp_seq=5 ttl=64 time=21.6 ms
```

对内网不存在的地址进行 ping，无法 ping 通。因为在同一网段内，报文并不经过攻击者，所以无法进行欺骗。



```
root@49afe2b85a23:/# 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
```

对外网存在的地址 8.8.8.8 进行 ping，仍可以 ping 通：



```
root@49afe2b85a23:/# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
From 10.9.0.1 icmp_seq=1 Destination Net Unreachable
64 bytes from 8.8.8.8: icmp_seq=1 ttl=64 time=60.6 ms
From 10.9.0.1 icmp_seq=2 Destination Net Unreachable
64 bytes from 8.8.8.8: icmp_seq=2 ttl=64 time=24.2 ms
From 10.9.0.1 icmp_seq=3 Destination Net Unreachable
64 bytes from 8.8.8.8: icmp_seq=3 ttl=64 time=18.8 ms
From 10.9.0.1 icmp_seq=4 Destination Net Unreachable
64 bytes from 8.8.8.8: icmp_seq=4 ttl=64 time=18.0 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=64 time=23.7 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=64 time=18.9 ms
```