1.1A：

对于每个捕获的数据包，使用调用回调函数打印 pkt()，打印出有关该数据包的一些信息。分别使用根权限和不使用根权限运行该程序，描述和解释观察结果。

测试程序：

```
#!/usr/bin/env python3

from scapy.all import *

def print_pkt(pkt):
    pkt.show()
pkt = sniff(iface='br-1a1624b97234', filter='net 10.9.0.0/16', prn=print_pkt)

~
```

有根权限时：

```
root@ad02046fb9cb:/# ping 10.9.0.1
PING 10.9.0.1 (10.9.0.1) 56(84) bytes of data.
64 bytes from 10.9.0.1: icmp_seq=1 ttl=64 time=0.251 ms
64 bytes from 10.9.0.1: icmp_seq=2 ttl=64 time=0.124 ms
^Z
[2]+  Stopped                 ping 10.9.0.1
root@ad02046fb9cb:/#
```

```
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup   seed@VM: ~/.../Labsetup   seed@VM: ~/.../volumes   seed@VM: ~/.../Labsetup
root@VM:/volumes# python3 mycode.py
###[ Ethernet ]###
  dst       = 02:42:90:69:cd:89
  src       = 02:42:0a:09:00:05
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 84
     id        = 8658
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x4c0
     src       = 10.9.0.5
     dst       = 10.9.0.1
     \options   \
###[ ICMP ]###
        type      = echo-request
        code      = 0
        chksum    = 0x5060
        id        = 0x11
```

```
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = icmp
     chksum    = 0x4c0
     src       = 10.9.0.5
     dst       = 10.9.0.1
     \options   \
###[ ICMP ]###
        type      = echo-request
        code      = 0
        chksum    = 0x5060
        id        = 0x11
        seq       = 0x1
###[ Raw ]###
           load      = '\xdf-\xe3`\x00\x00\x00\x00\x1c,\n\x00\x00\x00\x00\x00\x1
0\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f !"#$%&\'()*+,-./01
234567'

###[ Ethernet ]###
  dst       = 02:42:0a:09:00:05
  src       = 02:42:90:69:cd:89
  type      = IPv4
###[ IP ]###
```

无根权限时：

```
^Z
[4]+  Stopped                 python3 mycode.py
root@VM:/volumes# su seed
seed@VM:/volumes$ python mycode.py
bash: python: command not found
seed@VM:/volumes$ python3 mycode.py
Traceback (most recent call last):
  File "mycode.py", line 7, in <module>
    pkt = sniff(iface='br-1a1624b97234', 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
seed@VM:/volumes$
```

使用根权限运行该程序，证明 root 用户确实可以捕获数据包。再次运行程序，但不使用根权限，程序就会报错，证明 seed 用户不能捕获数据包。

1.1B：

通过更改 1.1A 程序中的 filter 的内容实现不同的过滤效果。

（1）filter='icmp'

1.1A 的程序中的 filter = ‘icmp’，因此上述结果就是只获取 icmp 报文的结果。

（2）filter='src host 10.9.0.5 and tcp dst port 23'

```
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup   seed@VM: ~/.../Labsetup   seed@VM: ~/.../volumes   seed@VM: ~/.../Labsetup
^Z
[1]+  Stopped                 python3 mycode.py
root@VM:/volumes# python3 mycode.py
###[ Ethernet ]###
  dst       = 02:42:90:69:cd:89
  src       = 02:42:0a:09:00:05
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x10
     len       = 60
     id        = 41373
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = tcp
     chksum    = 0x84f7
     src       = 10.9.0.5
     dst       = 10.9.0.1
     \options   \
###[ TCP ]###
        sport     = 38814
        dport     = telnet
```

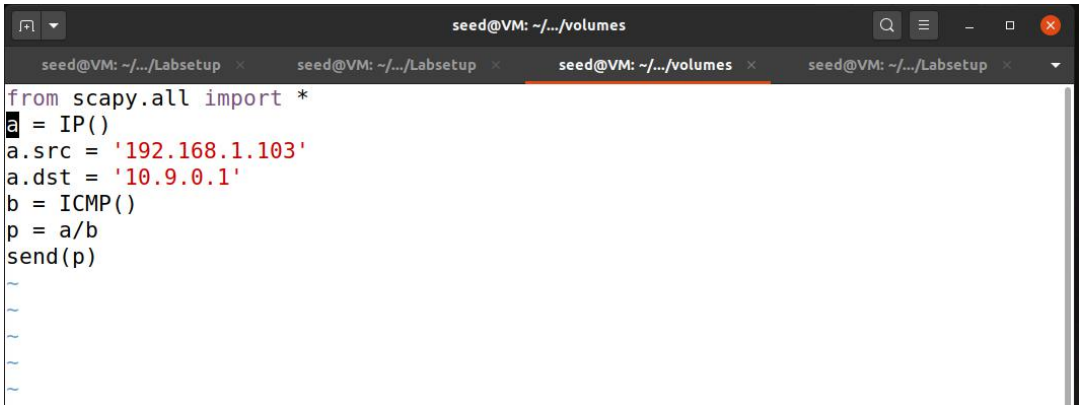
```
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup   seed@VM: ~/.../Labsetup   seed@VM: ~/.../volumes   seed@VM: ~/.../Labsetup
     ttl       = 64
     proto     = tcp
     chksum    = 0x84f7
     src       = 10.9.0.5
     dst       = 10.9.0.1
     \options   \
###[ TCP ]###
        sport     = 38814
        dport     = telnet
        seq       = 3860377642
        ack       = 0
        dataofs   = 10
        reserved  = 0
        flags     = S
        window    = 64240
        chksum    = 0x1446
        urgptr    = 0
        options   = [('MSS', 1460), ('SAckOK', b''), ('Timestamp', (2986312306,
0)), ('NOP', None), ('WScale', 7)]

###[ Ethernet ]###
  dst       = 02:42:90:69:cd:89
  src       = 02:42:0a:09:00:05
  type      = IPv4
```

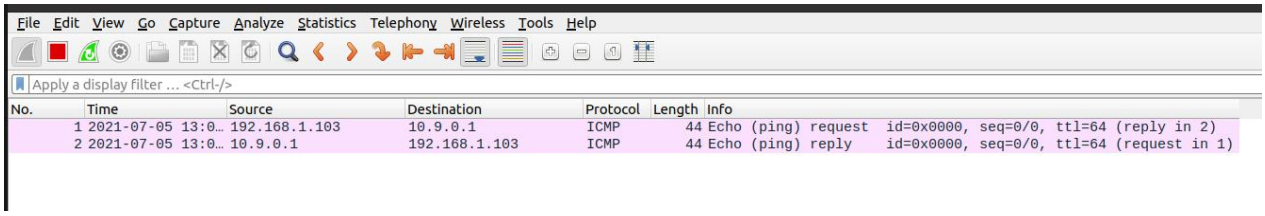
（3）filter='net 128.230.0.0/16'

使用 filter='net 10.9.0.0/16'测试成功，但是题目要求不能使用虚拟机所在子网，其他子网无法测试。

1.2：

通过程序进行数据包欺骗，将源地址改为自己设定的地址 192.168.1.103，向目标为 10.9.0.1 的地址发送数据包，通过 wireshark 观察结果。





从 wireshark 的观测结果可以看到，成功使用 192.168.1.103 的欺骗 ip 发送报文，伪造成功。

1.3：

此任务的目标是使用 Scapy 来估计虚拟机与选定目标之间的路由器数量之间的距离。

测试程序：

```
from scapy.all import *

a = IP()
a.dst = '202.108.22.5'
a.ttl = 17
b = ICMP()
send(a/b)
```

通过更改 ttl 的值进行测试，每经过一个路由 ttl 值都会-1，ttl 变为 0 的路由将向我们发送一个 ICMP 错误消息，告诉我们它的生存时间已经超过了运行时间。如果我们收到了正确的 replay 报文，意味着此时设定的 ttl 可以让报文传送至目标 ip。

当 ttl=17 时：

No.	Time	Source	Destination	Protocol	Length	Info
1	2021-07-05 16:5…	VMware_bf:76:55		ARP	44	Who has 192.168.43.1? Tell 192.168.43.247
2	2021-07-05 16:5…	HuaweiTe_c4:4d:a1		ARP	62	192.168.43.1 is at 00:be:3b:c4:4d:a1
3	2021-07-05 16:5…	192.168.43.247	202.108.22.5	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=17 (no response …
4	2021-07-05 16:5…	10.166.1.44	192.168.43.247	ICMP	72	Time-to-live exceeded (Time to live exceeded in transit)

当 ttl=18 时：

No.	Time	Source	Destination	Protocol	Length	Info
1	2021-07-05 16:5…	VMware_bf:76:55		ARP	44	Who has 192.168.43.1? Tell 192.168.43.247
2	2021-07-05 16:5…	HuaweiTe_c4:4d:a1		ARP	62	192.168.43.1 is at 00:be:3b:c4:4d:a1
3	2021-07-05 16:5…	192.168.43.247	202.108.22.5	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=18 (reply in 4)
4	2021-07-05 16:5…	202.108.22.5	192.168.43.247	ICMP	62	Echo (ping) reply id=0x0000, seq=0/0, ttl=47 (request in 3)

通过对比可以得知，从 10.9.0.1 到 202.108.22.5 总共有 18 个路由。

1.4：

首先捕获 icmp 报文，然后使用该报文的源地址作为目的地址，目的地址作为源地址发送报文，设定类型为 replay，进行欺骗返回一个 replay、

测试程序：

```
from scapy.all import *

def spoof_pkt(pkt):
    if 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(iface='br-ae85ac59e183',filter='icmp', prn=spoof_pkt)
~
```

ping 1.2.3.4 时：

在不进行 ip 欺骗时：

```
root@6f402becdcaa:/# ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
```

可以看到 ping 命令没有回应，对于一个不存在的主机 ip 是 ping 不通的。

进行 ip 欺骗之后：

```
seed@VM: ~/.../Lab
        TX errors 0  dropped 0 overruns 0  c

root@VM:/volumes# python3 lab1_4.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
```

```
seed@VM: ~/.../Labsetup    seed@VM: ~/.../volumes    seed@
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
^Z
[8]+  Stopped                 ping 1.2.3.4
root@6f402becdcaa:/# ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
^Z
[9]+  Stopped                 ping 1.2.3.4
root@6f402becdcaa:/# 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=60.3 ms
64 bytes from 1.2.3.4: icmp_seq=2 ttl=64 time=19.4 ms
64 bytes from 1.2.3.4: icmp_seq=3 ttl=64 time=24.6 ms
64 bytes from 1.2.3.4: icmp_seq=4 ttl=64 time=19.2 ms
64 bytes from 1.2.3.4: icmp_seq=5 ttl=64 time=18.2 ms
64 bytes from 1.2.3.4: icmp_seq=6 ttl=64 time=30.3 ms
64 bytes from 1.2.3.4: icmp_seq=7 ttl=64 time=24.8 ms
64 bytes from 1.2.3.4: icmp_seq=8 ttl=64 time=22.7 ms
64 bytes from 1.2.3.4: icmp_seq=9 ttl=64 time=21.8 ms
64 bytes from 1.2.3.4: icmp_seq=10 ttl=64 time=25.9 ms
64 bytes from 1.2.3.4: icmp_seq=11 ttl=64 time=24.0 ms
64 bytes from 1.2.3.4: icmp_seq=12 ttl=64 time=19.4 ms
^Z
[10]+  Stopped                 ping 1.2.3.4
root@6f402becdcaa:/#
```

在执行程序的容器一方显示成功发回了 replay 的数据包，而执行 ping 命令的一方也显示收到了 replay 的报文，证明程序成功进行了欺骗。

ping 10.9.0.99 时，因为这个主机 ip 在 LAN 上是不可达的，因此会返回不可达的错误提示：

```
root@d387bbd2d679:/# 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
From 10.9.0.5 icmp_seq=10 Destination Host Unreachable
From 10.9.0.5 icmp_seq=11 Destination Host Unreachable
From 10.9.0.5 icmp_seq=12 Destination Host Unreachable
```

ping 8.8.8.8 时，因为该主机 ip 是存在的，因此可以接受到 replay：

```
root@d387bbd2d679:/# 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=108 time=68.3 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=108 time=73.6 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=108 time=72.0 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=108 time=78.1 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=108 time=78.3 ms
```