“Lab 12: The Firewall, Domain Name, and attacks that target DNS
Md Rony

For section 3.1 to 3.4, we use the /share/education/TCPSYNFlood_USC_ISI/synflood.ns file to create our topology.

Once the experiment is swapped in, we installed Apache on the server node by entering: ‘/share/education/TCPSYNFlood_USC_ISI/install-server’.

Then, installed the ‘flooder’ attack tool on the attacker node: ‘/share/education/TCPSYNFlood_USC_ISI/install-flooder’.
We get the following warnings and note:

```
Processing triggers for libc-bin (2.23-0ubuntu9) ...
g++ celp.c -o celp
celp.c: In function 'int main(int, char**)':
celp.c:181:28: warning: format '%d' expects argument of type 'int', but argument 4 has type 'long unsigned int' [-Wformat=]
        argv[0], sizeof(buf));
                            ^
g++  flooder.c -lnet -lm -lpcap -fpermissive -o flooder
flooder.c: In function 'int flood_process(FloodParameters*)':
flooder.c:640:27: warning: invalid conversion from 'unsigned char*' to 'char*' [-fpermissive]
      thislen = sprintf(p->payload, "%c%c%c%c%c%c%c%c%c%c%c%cjoe%cexample%ccom
                        ^
In file included from /usr/include/pcap/pcap.h:52:0,
                 from /usr/include/pcap.h:43,
                 from flooder.c:17:
/usr/include/stdio.h:364:12: note:   initializing argument 1 of 'int sprintf(char*, const char*, ...)'
 extern int sprintf (char *__restrict __s,
            ^
```

3.1 Generating Legitimate Traffic

```
#!/bin/bash

while true;

do wget http://5.6.7.8/index.html

sleep 1;

Done
```

With the above bash script (http://5.6.7.8 describes the server node’s IP address), we are able to download the index.html file once per second.

3.2 Turning off SYN Cookies

```
jjc400bj@server:~$ sudo sysctl net.ipv4.tcp_syncookies
net.ipv4.tcp_syncookies = 1
```

SYN Cookies are currently turned on, so we turn it off as shown in the following screenshot:

```
jjc400bj@server:~$ sudo sysctl -w net.ipv4.tcp_syncookies=0
net.ipv4.tcp_syncookies = 0
jjc400bj@server:~$ sudo sysctl net.ipv4.tcp_syncookies
net.ipv4.tcp_syncookies = 0
```

3.3 Generating attack traffic

We use the following command to generate attack traffic:
'Sudo flooder --dst 5.6.7.8 --srcmask 255.255.255.0'

3.4 Collecting traffic statistics

1. After entering 'ip route 5.6.7.8', we get:

```
jjc400bj@client:~$ ip route get 5.6.7.8
5.6.7.8 via 1.1.2.2 dev eth4  src 1.1.2.3
    cache
jjc400bj@client:~$
```

This tells us that the interface leading to 5.6.7.8 is eth4. To see the traffic flowing, we type: 'sudo tcpdump -nn -i eth4'.

```
jjc400bj@client:~$ sudo tcpdump -nn -i eth4
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth4, link-type EN10MB (Ethernet), capture size 262144 bytes
12:28:05.154579 LLDP, length 232: EHP17e
12:28:35.154693 LLDP, length 232: EHP17e
12:29:05.154761 LLDP, length 232: EHP17e
12:29:35.154878 LLDP, length 232: EHP17e
12:30:05.154946 LLDP, length 232: EHP17e
12:30:35.155114 LLDP, length 232: EHP17e
12:31:05.155182 LLDP, length 232: EHP17e
12:31:35.155300 LLDP, length 232: EHP17e
12:32:05.155317 LLDP, length 232: EHP17e
12:32:35.155435 LLDP, length 232: EHP17e
12:33:05.155502 LLDP, length 232: EHP17e
12:33:35.155670 LLDP, length 232: EHP17e
```

To see IP traffic and save traffic to a file, we can enter: 'sudo tcpdump -nn -v -i eth4 -w file.cap'. To see this file: 'sudo tcpdump -r file.cap'.

2. Using a stopwatch, we perform the scenario:

We start tcpdump on the client node by entering: "sudo tcpdump -nn -v -i eth4 -w timedcapture-no-syn.txt"

   a. Start legitimate traffic (on client).

We run the bash script file on the client node, as described above.

   b. After 30 seconds start the attack.

On the attacker node, we enter “sudo flooder --dst 5.6.7.8 --srcmask 255.255.255.0”

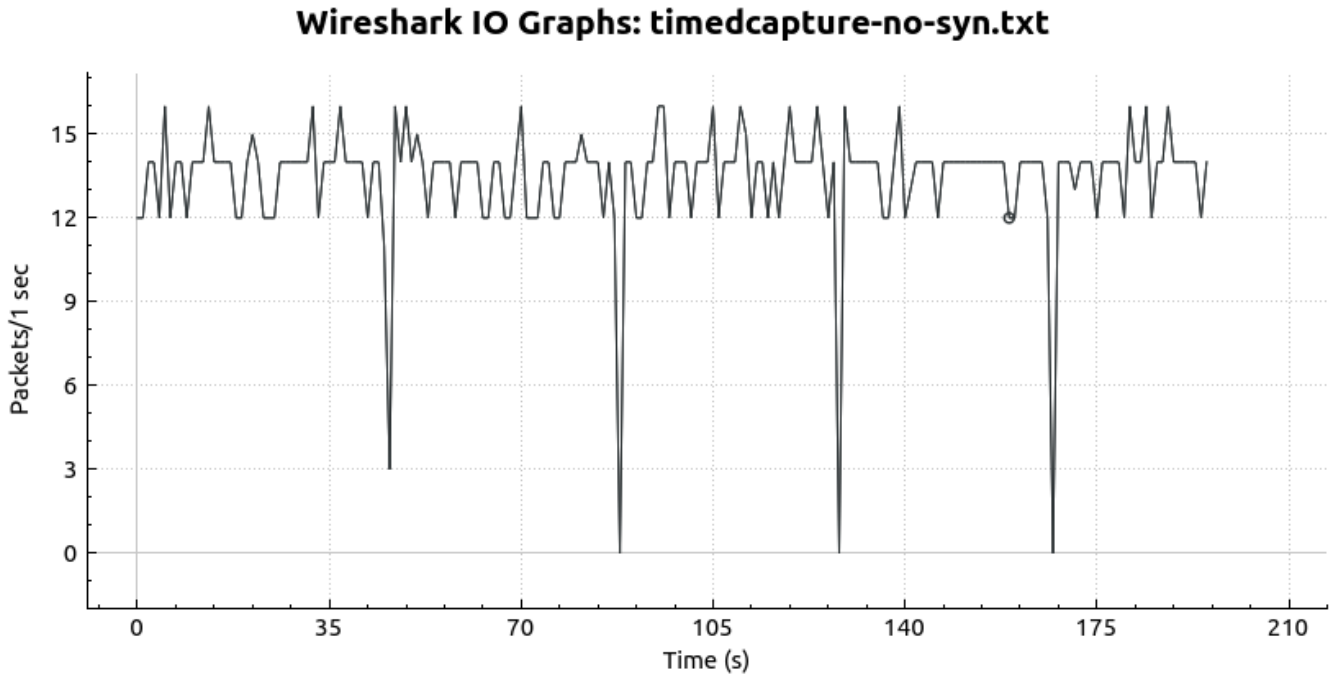
c. After 120 seconds stop the attack.
d. After 30 seconds stop the legitimate traffic
e. Stop the tcpdump on the client and save the file.

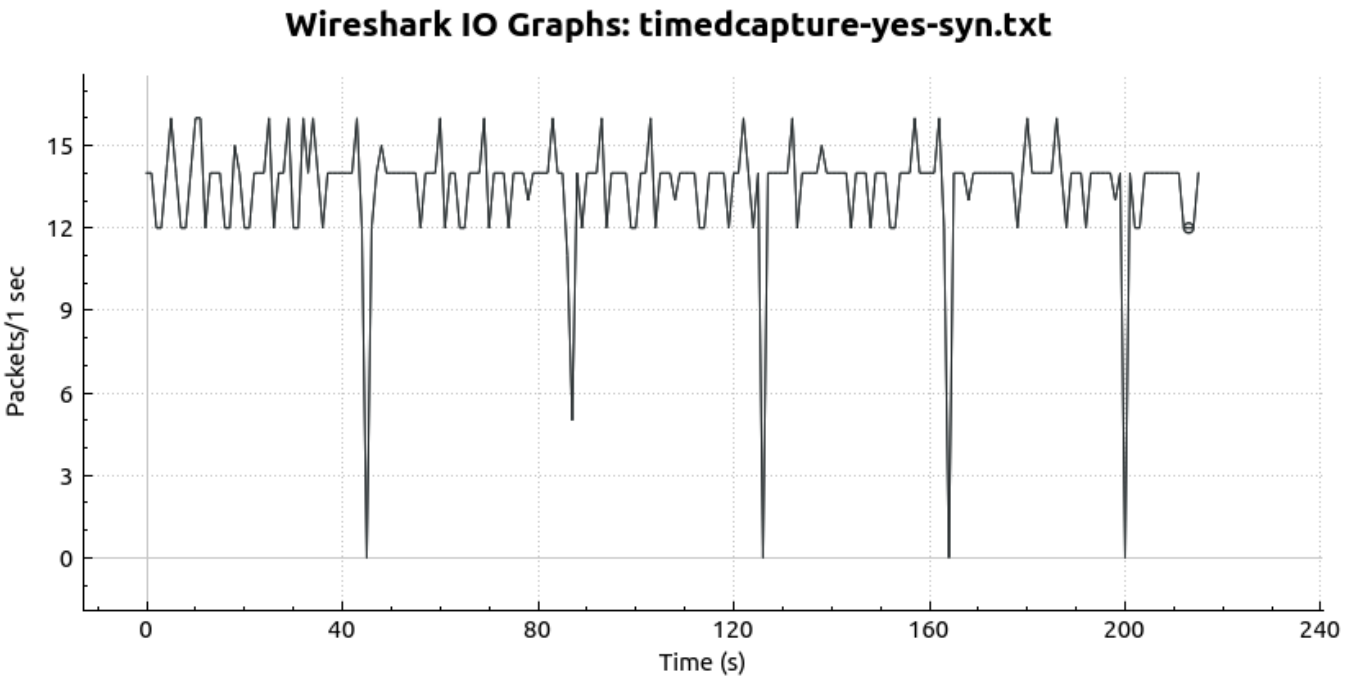
3. Turn SYN cookies on and repeat the above steps.

We turn SYN cookies on from the server node by entering: “sudo sysctl net.ipv4.tcp_syncookies=1”

4. Using the recorded traffic files and tcpdump to read them, process the output and calculate connection duration for each TCP connection seen in the files. Connection duration is the difference between the time of the first SYN and of the ACK following a FIN-ACK (or between the first SYN and the first RESET) on a connection. Remind yourself what uniquely identifies a TCP connection, i.e. how to detect packets that belong to the same connection? If a connection did not end with an ACK following a FIN-ACK assign to it the duration of 200 s. Include two graphs in your submission, showing connection duration vs connection start time for the case without and with SYN cookies. Label the graphs so they can be distinguished and indicate on each graph using vertical lines or arrows the start and the end of the attack.

To provide answers to this prompt, we use Wireshark to view the tcpdump files, and create their respective graphs.





3.5 No spoofing

What happens? It seams that by masking the IP address the user got more packets. This means the attack was successful.

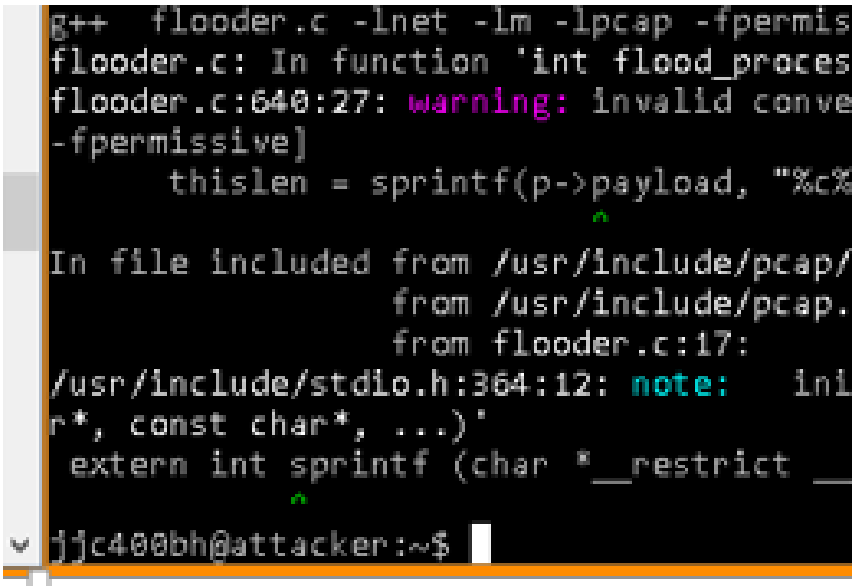
Can you explain why this happens? by masking the IP address we got TCP packets and without only UDP.

Can you modify the attack so that it is effective without spoofing and how would you do this?

This attack could could be effective if we were able to have the server send more packets.

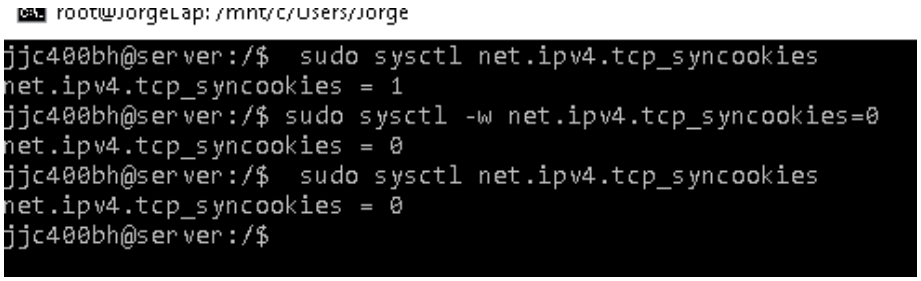
The process for this section was to do everything in 3.1 to 3.4 except that in 3.3 we had to skip spoofing the IP address. The process was easy as the work and commands were already done by the first team member.

Screenshot of the installation of Apache2 in the server



Screenshot of flooder installation

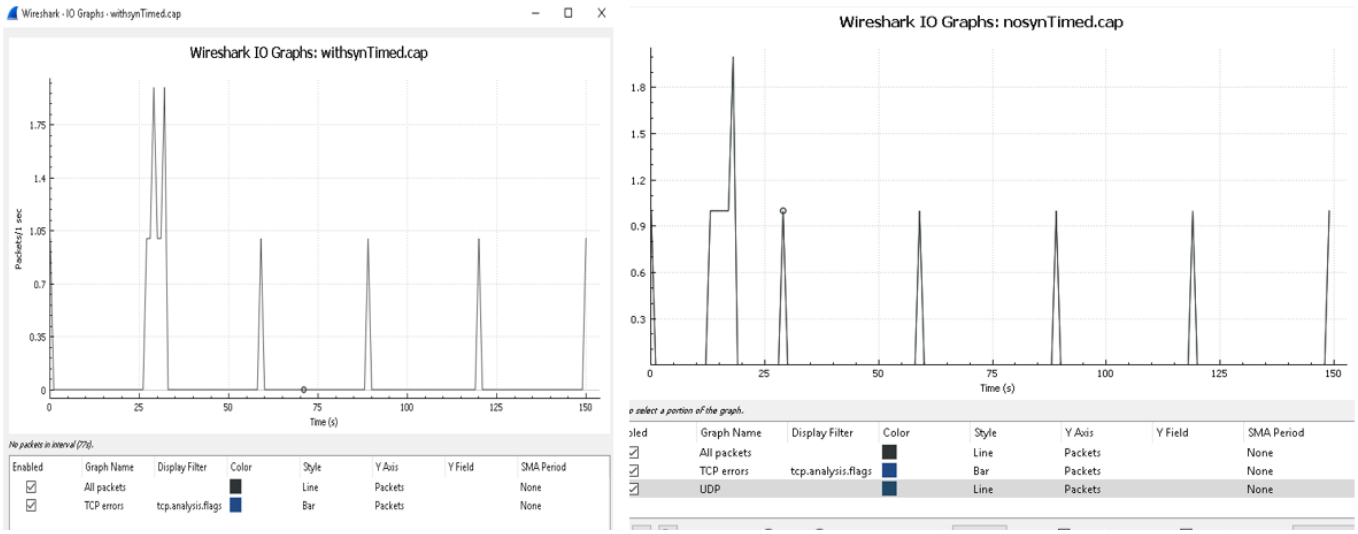
3.1 We used the same code as in the first part, implemented in nano editor.



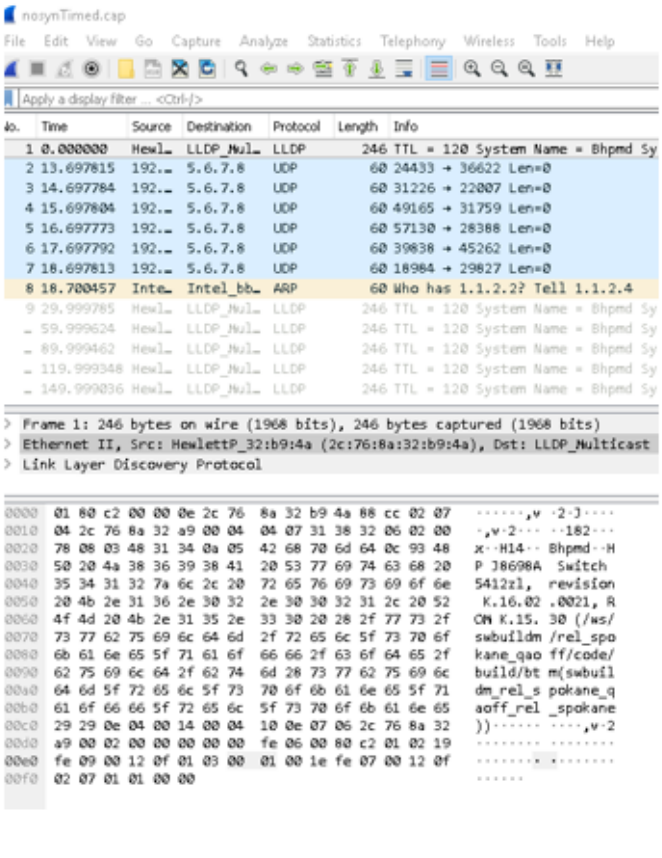
3.2 Command for turning off SYN cookies

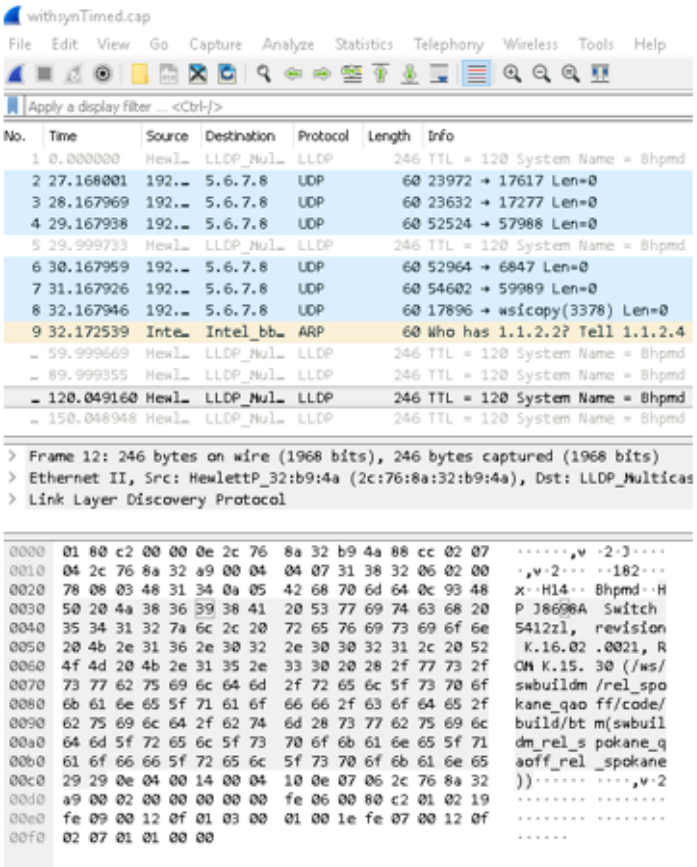
3.3 No need to mask IP address.

3.4 Using a watch and having command prompts connected to the server, client and attacker conducted transmission and attack.



After using tcpdump we used wireshark to provide IO graphs to compare with and without syncookies





4. Word Problem:

1. Explain How TCP SYN flood attack works

SYN flooding Attack is a form of DOS Attack. A client uses many False IP addresses to send synchronizing (SYN) packets to a targeted port of server . The false IP addresses act like multiple user to the server and server interprets packets multiplely. Server sends Acknowledge (ACK) back for each IP but in reality all one person .Thus it consume/attacks server's resources.

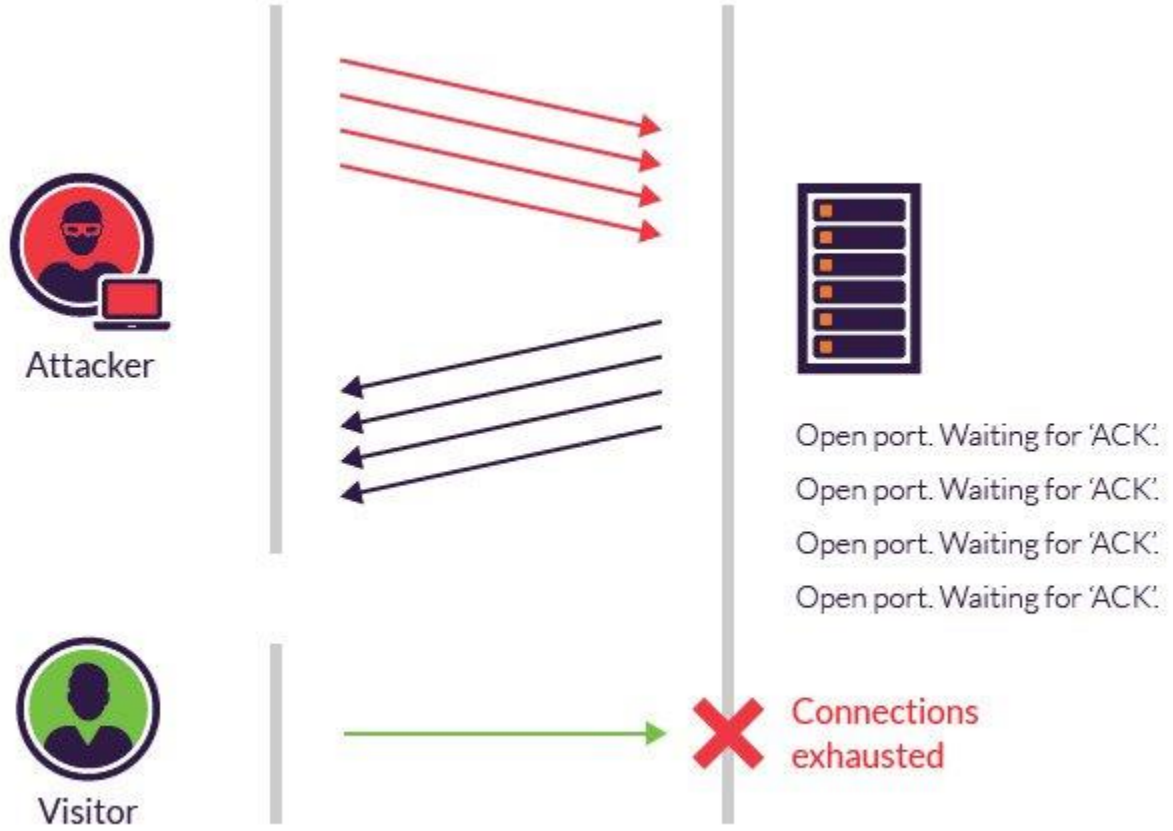
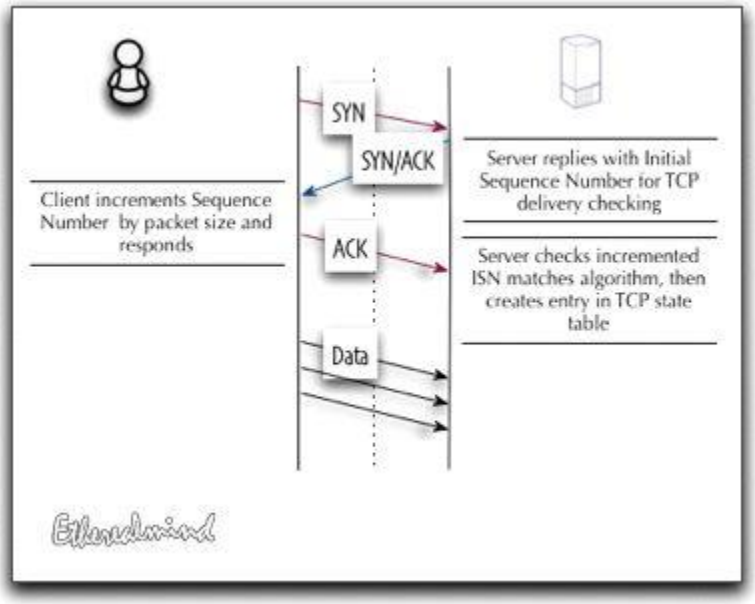


Photo source : https://www.incapsula.com/ddos/attack-glossary/syn-flood.html

2. Explain how SYN cookies work to prevent denial-of-service effect from SYN flood Attacks.

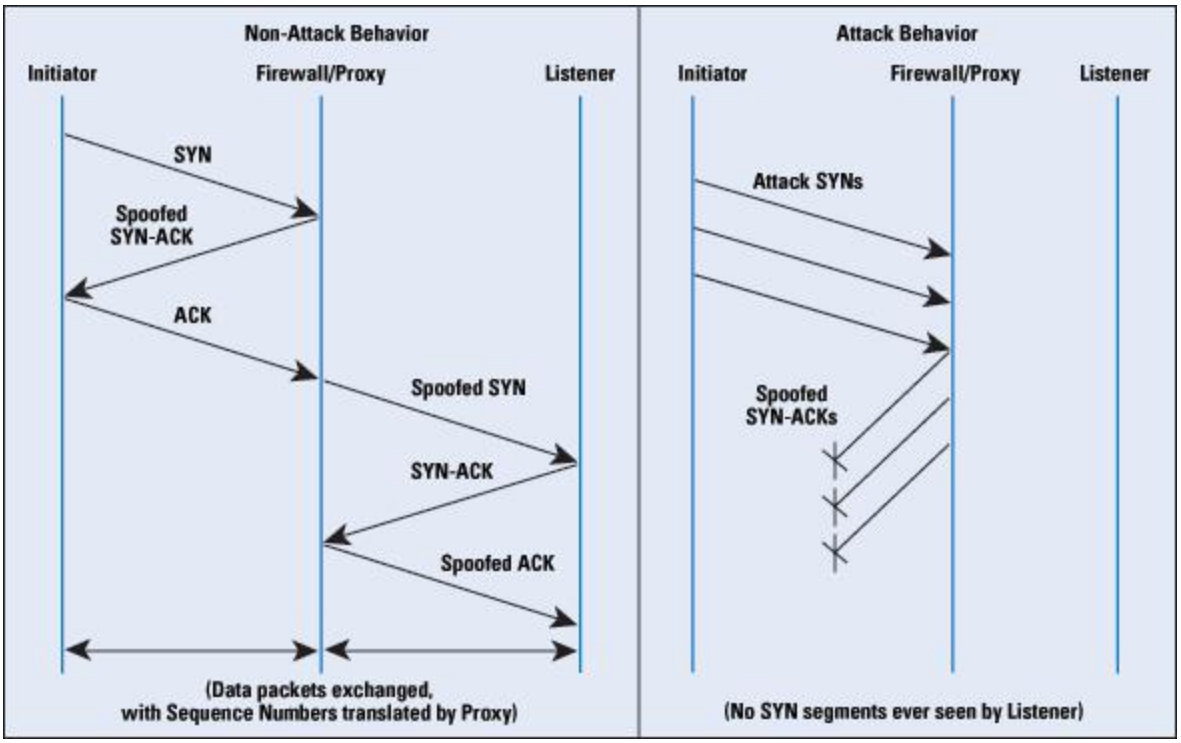
SYN cookies are a specific choice of initial TCP sequence numbers by the TCP software which prevent SYN flood attacks. When A client sends a SYN and the server replies with an SYN And ACK- response, Then the server will hold the informations in the TCP stack while it waits for the client's acknowledgement ACK-message. The SYN flood will then operate the SYN packets which, in return, will consume all TCP memory. Since the state table has limitation of memory , the server can not store any new TCP network connections which cause it will be a failed , denial or even worse response to the attacker.



https://etherealmind.com/tcp-syn-cookies-ddos-defence/

3. Would changing the network buffers in the OS (e.g. as available FreeBSD) have any impact on attack effectiveness?

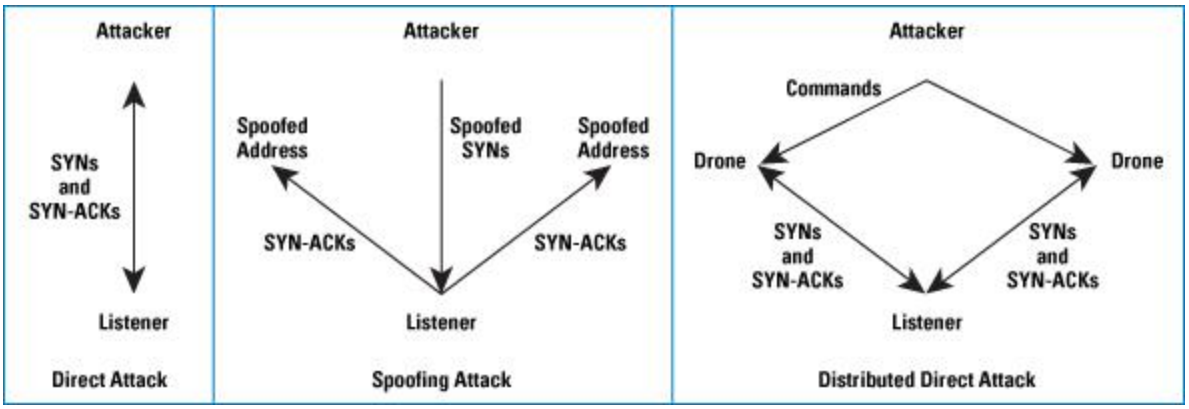
Yes, By changing the network buffer in OS (e.g. as available FreeBSD) could build an effective attack creating additional loopholes that a client get it hardship to crack.Exploiting the behavior of the buffer overflow become secure the OS.



https://www.cisco.com/c/dam/en_us/about/ac123/ac147/images/ipj/ipj_9-4/94_syn_fig6_lg.jpg

4. How would you defend against these attacks other than with SYN cookies?

a) Hardening tcp/IP stack against SYN floods
b) Changing the network buffer in OS as question 3.
c) Implanting proper firewall policies are certainly first line of defense
d) however the Linux kernel can also be hardened against these types of attacks.
e) A spoofing attack when executed confuses the client/attacker with false or “spoofed” IP addresses that resemble the legitimate address. It is often used to overload network and initial devices to appear as the real deal, when in fact, it is a clone or similar to the real thing but not quite it.



https://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-34/syn-flooding-attacks.html

”