Computer Network Assignment 2

Neerja Kasture
22110165

Anura Mantri
22110144

Team number: 13

Github Link : https://github.com/NeerjaKasture/CS331-Assignment-2/

Task-1: Comparison of congestion control protocols

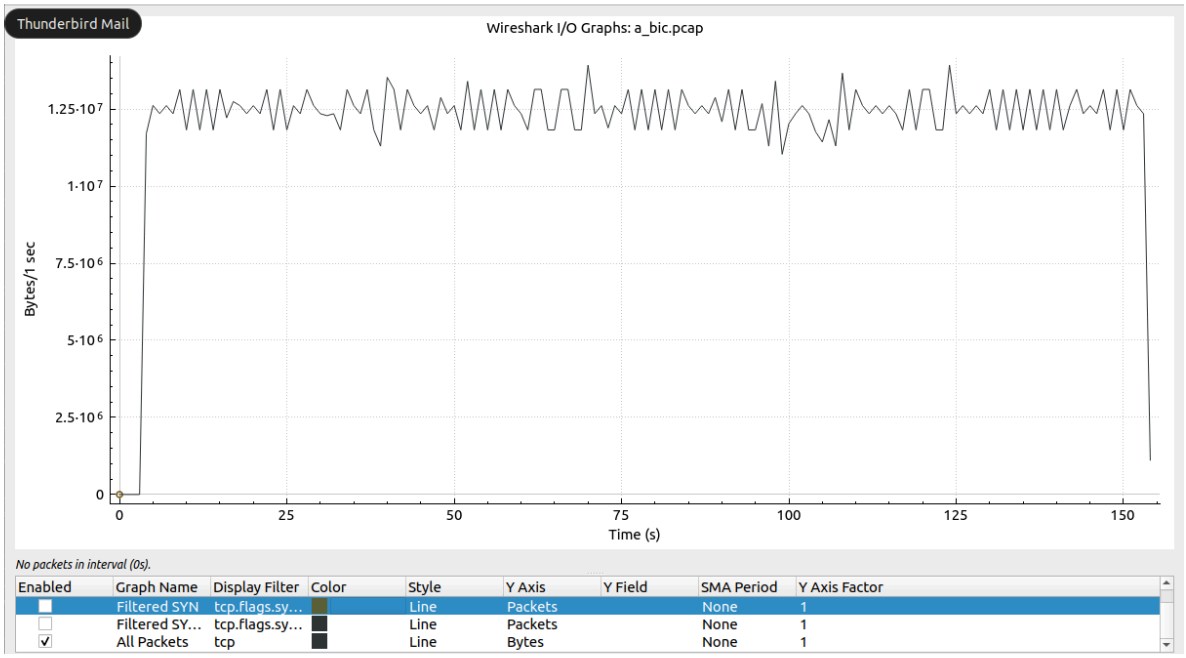
Protocols used:

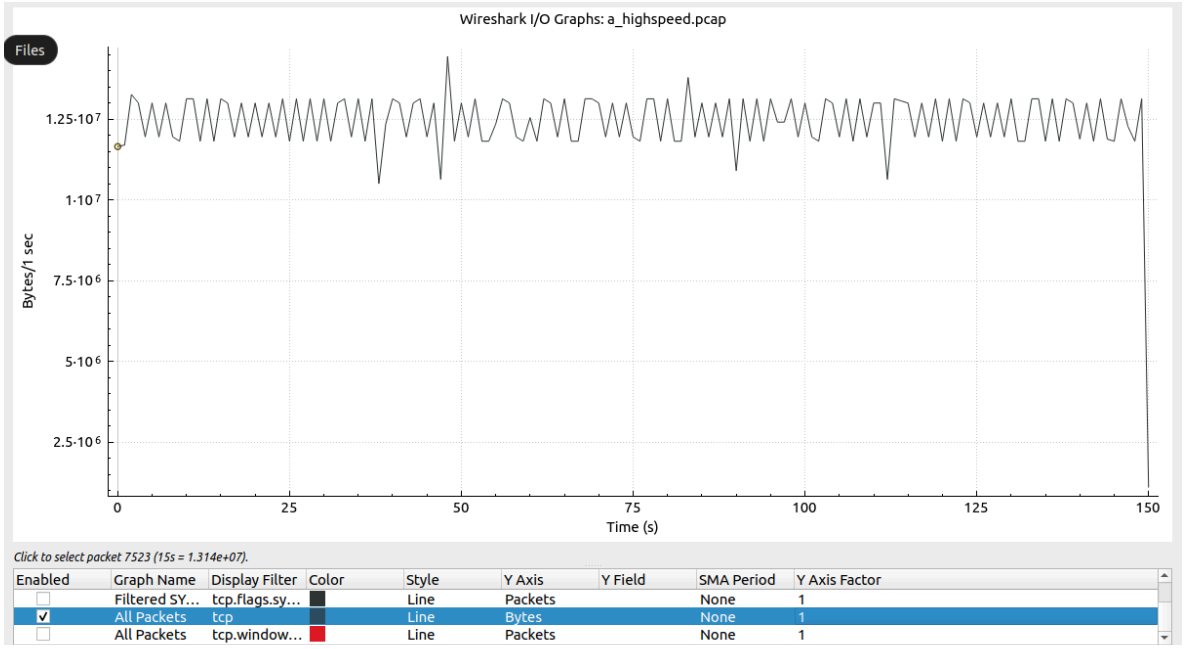
1. BIC
2. Highspeed
3. Yeah

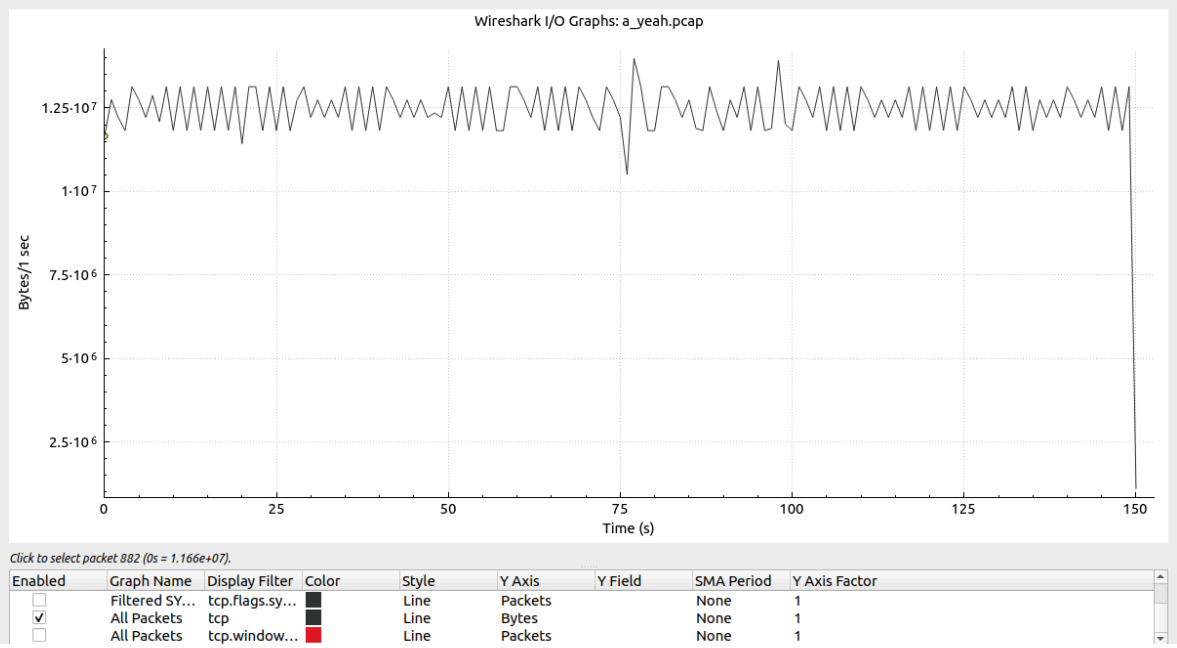
A.

```
neerja@Ubuntu:~/Documents/cn assignment 2$ sudo python3 q1.py --part a
Running iperf3 test from h1 to h7 with TCP bic
Packet capture saved to /tmp/a_bic.pcap
Metrics for TCP bic :
Throughput (KBps): 8608.96
Goodput (KBps): 8580.01
Packet Loss Rate: 0.10%
Max TCP Window Size: 43440 bytes
Running iperf3 test from h1 to h7 with TCP highspeed
Packet capture saved to /tmp/a_highspeed.pcap
Metrics for TCP highspeed :
Throughput (KBps): 12193.29
Goodput (KBps): 12166.22
Packet Loss Rate: 0.08%
Max TCP Window Size: 43440 bytes
Running iperf3 test from h1 to h7 with TCP yeah
Packet capture saved to /tmp/a_yeah.pcap
Metrics for TCP yeah :
Throughput (KBps): 12209.71
Goodput (KBps): 12181.98
Packet Loss Rate: 0.05%
Max TCP Window Size: 43440 bytes
```

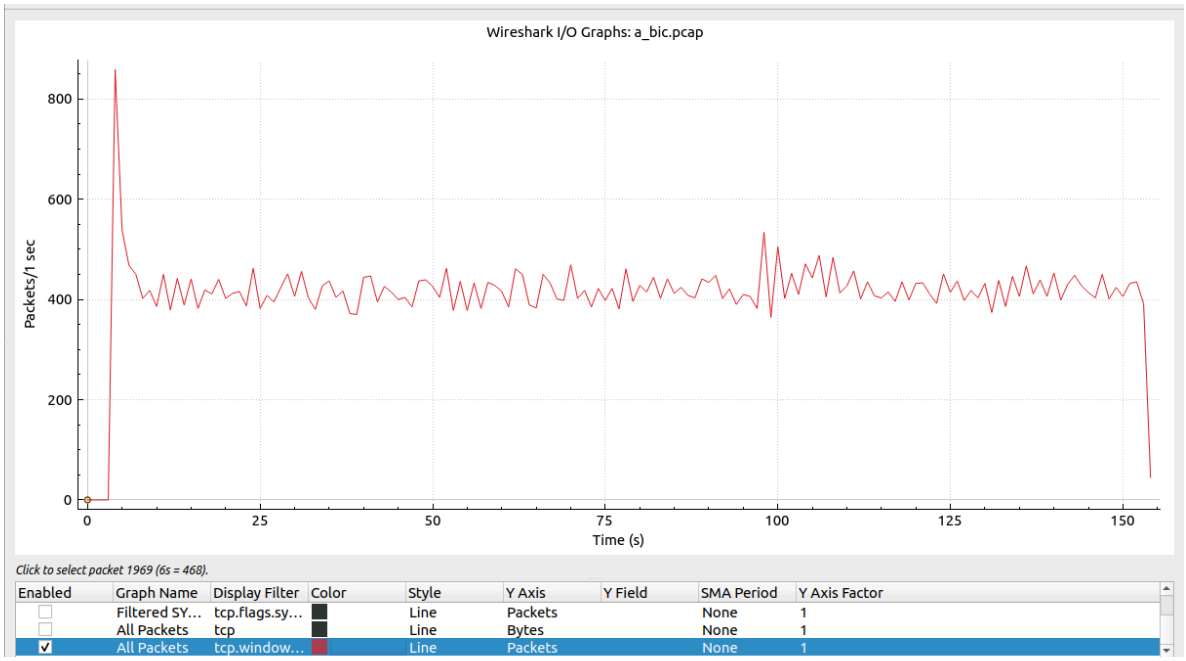
Graphs for throughput:

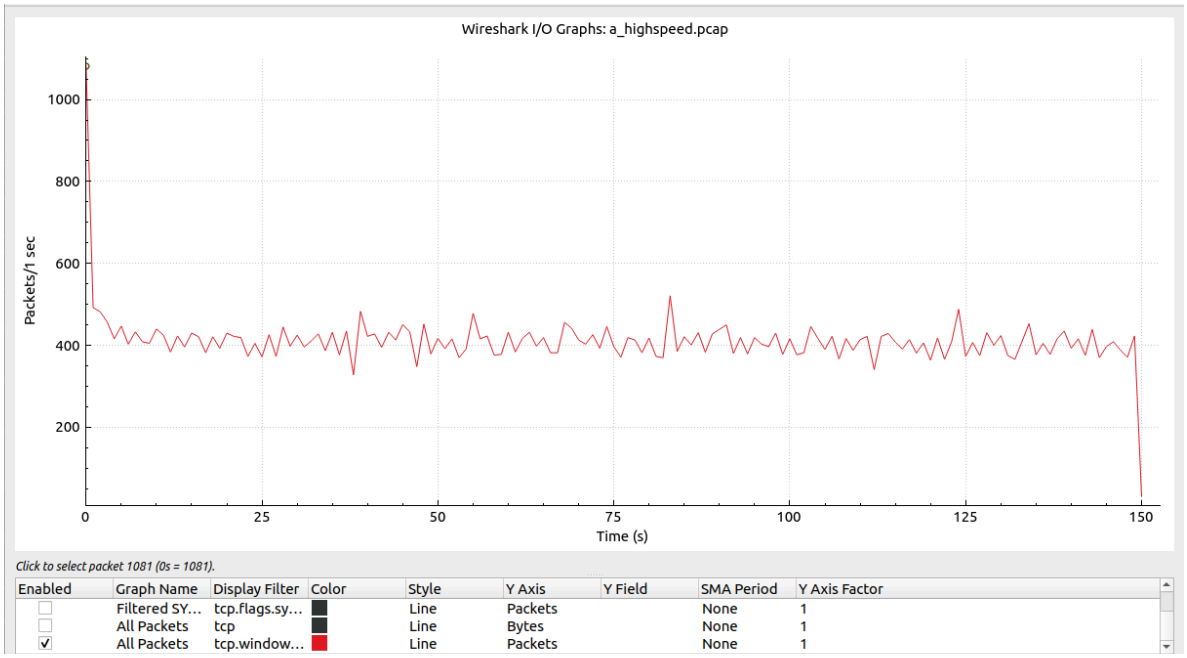

Thunderbird Mail
Wireshark I/O Graphs: a_bic.pcap
Bytes/1 sec
1.25·10⁷
1·10⁷
7.5·10⁶
5·10⁶
2.5·10⁶
0
0
25
50
75
100
125
150
Time (s)
No packets in interval (0s).
Enabled | Graph Name | Display Filter | Color | Style | Y Axis | Y Field | SMA Period | Y Axis Factor
Filtered SYN | tcp.flags.sy… | Line | Packets | None | 1
Filtered SY… | tcp.flags.sy… | Line | Packets | None | 1
All Packets | tcp | Line | Bytes | None | 1

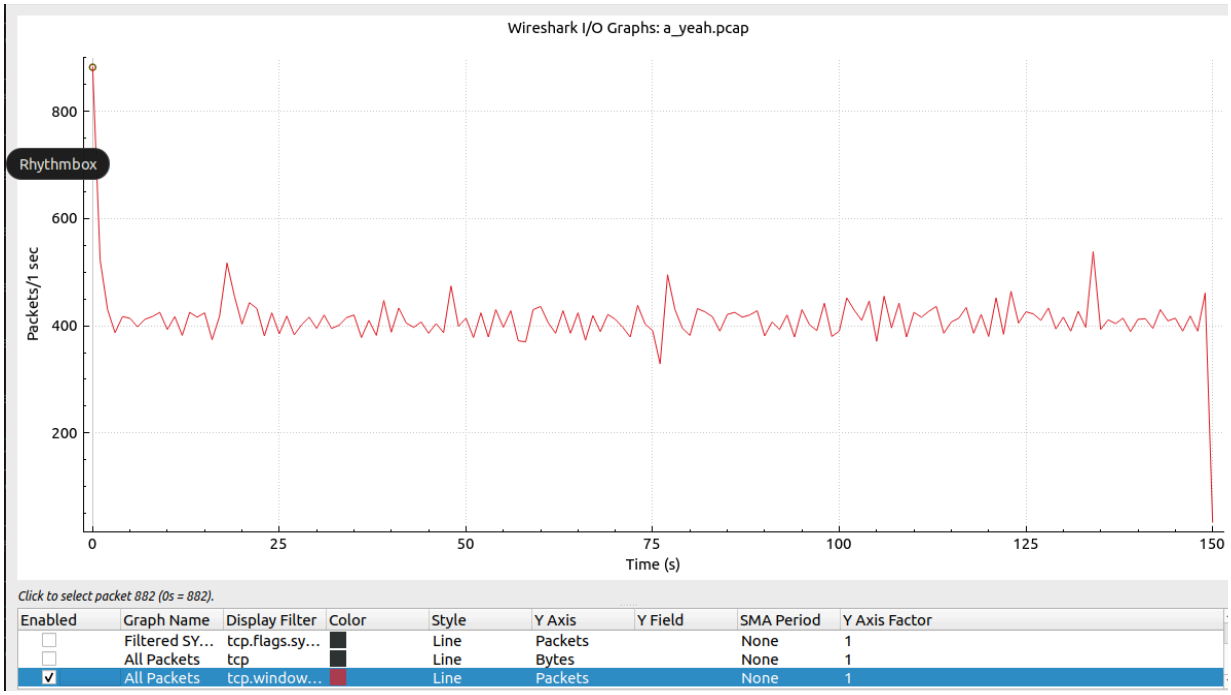

Files
Wireshark I/O Graphs: a_highspeed.pcap
Bytes/1 sec
1.25·10⁷
1·10⁷
7.5·10⁶
5·10⁶
2.5·10⁶
0
25
50
75
100
125
150
Time (s)
Click to select packet 7523 (15s = 1.314e+07).
Enabled | Graph Name | Display Filter | Color | Style | Y Axis | Y Field | SMA Period | Y Axis Factor
Filtered SY… | tcp.flags.sy… | Line | Packets | None | 1
All Packets | tcp | Line | Bytes | None | 1
All Packets | tcp.window… | Line | Packets | None | 1



Graph for window size:







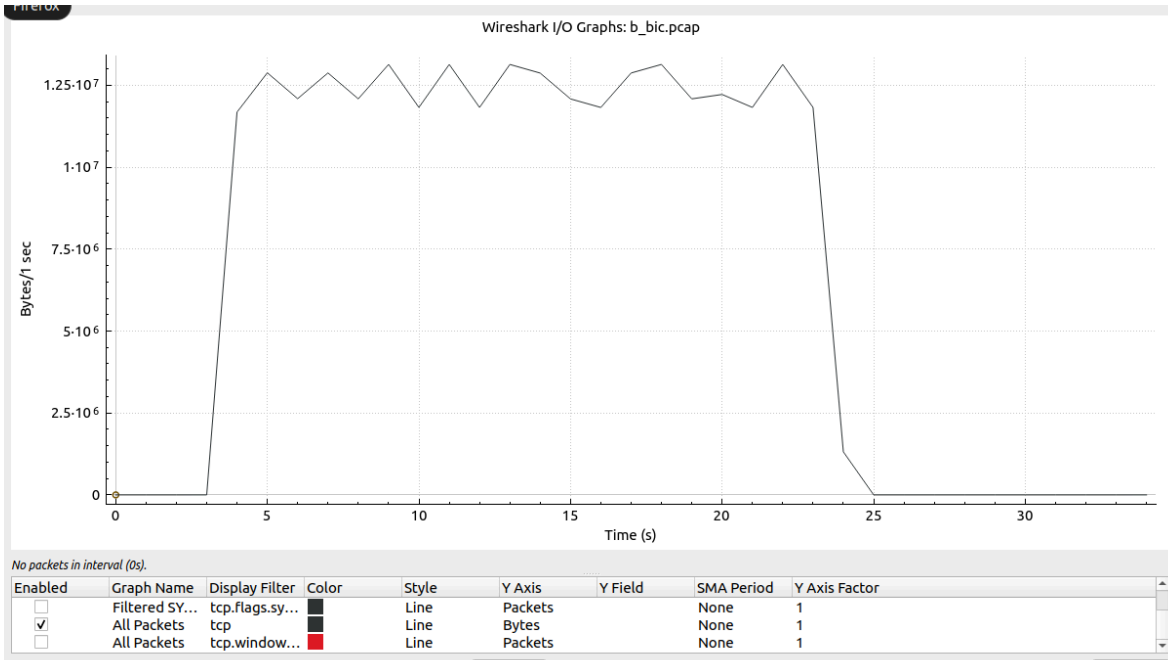
Observations:

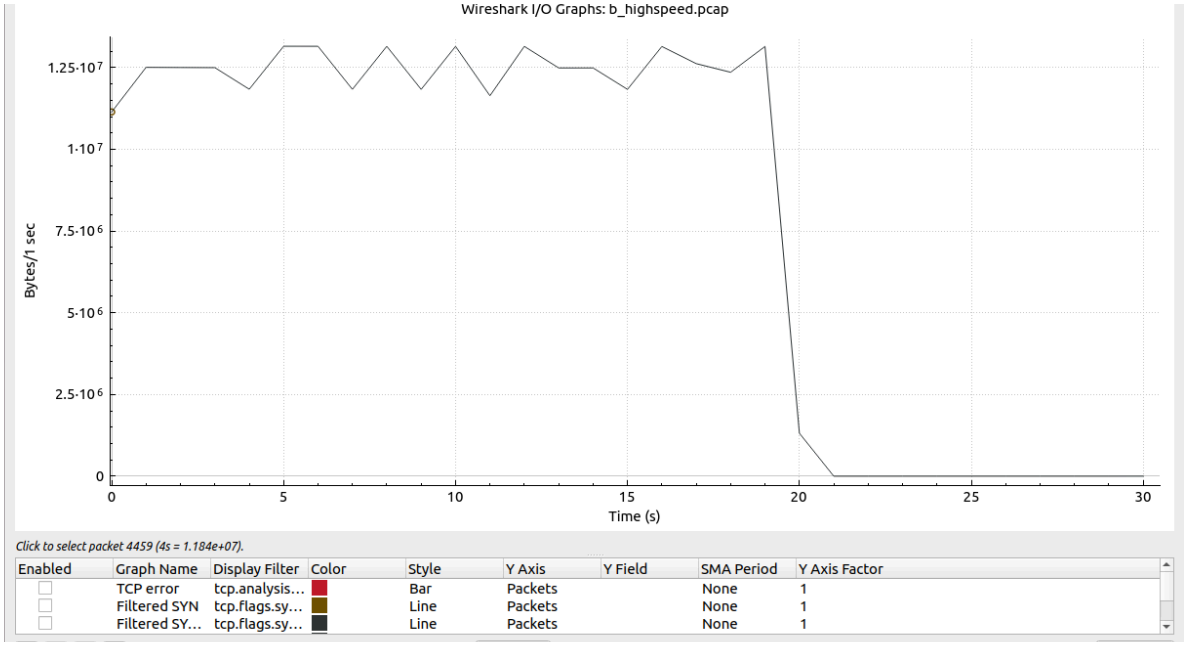
- The throughput order is Yeah, Highspeed, BIC
- The packet loss is highest for BIC, but all three have low packet loss
- The goodput is only slightly lower than the throughput in all cases.
- The window size varies around the same range for all.

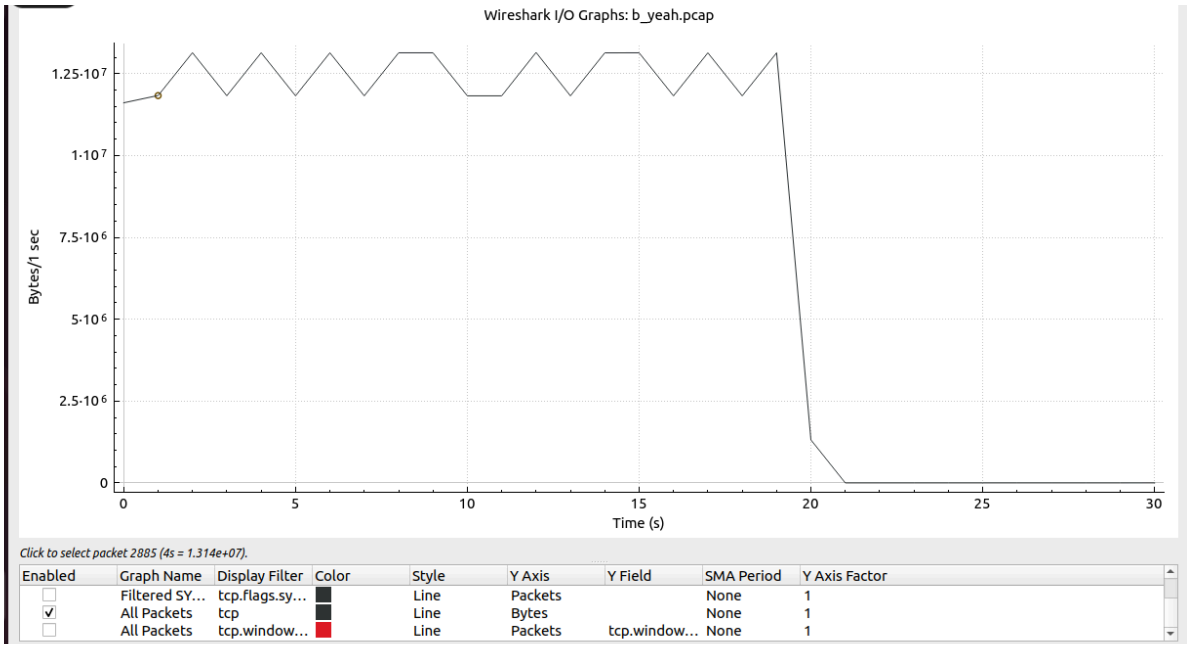
B.

```
neerja@Ubuntu:~/Documents/cn assignment 2$ sudo python3 q1.py --part b
Running tests with bic congestion control scheme...
Starting H1 at T=0s...
Starting H3 at T=15s...
Starting H4 at T=30s...
Metrics for TCP bic :
Throughput (KBps): 1755.43
Goodput (KBps): 1750.62
Packet Loss Rate: 0.13%
Max TCP Window Size: 43440 bytes

Running tests with highspeed congestion control scheme...
Starting H1 at T=0s...
Starting H3 at T=15s...
Starting H4 at T=30s...
Metrics for TCP highspeed :
Throughput (KBps): 2445.59
Goodput (KBps): 2436.51
Packet Loss Rate: 0.09%
Max TCP Window Size: 43440 bytes

Running tests with yeah congestion control scheme...
Starting H1 at T=0s...
Starting H3 at T=15s...
Starting H4 at T=30s...
Metrics for TCP yeah :
Throughput (KBps): 1525.25
Goodput (KBps): 1521.24
Packet Loss Rate: 0.05%
Max TCP Window Size: 43440 bytes
```

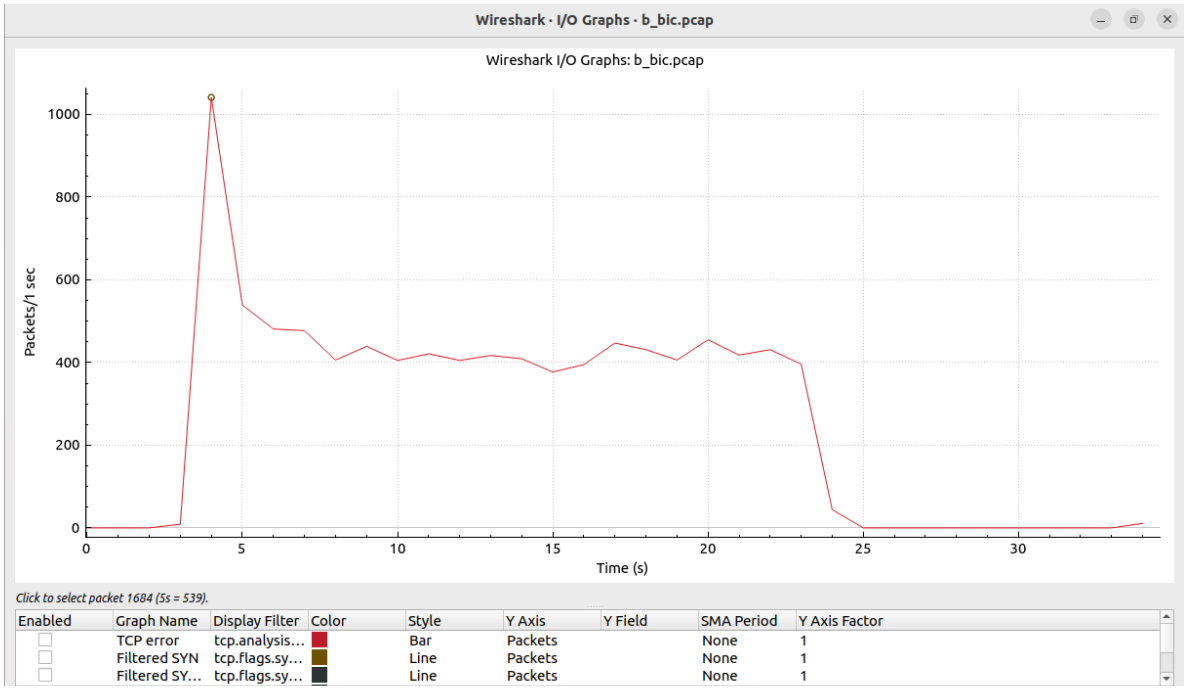
Graphs for throughput:

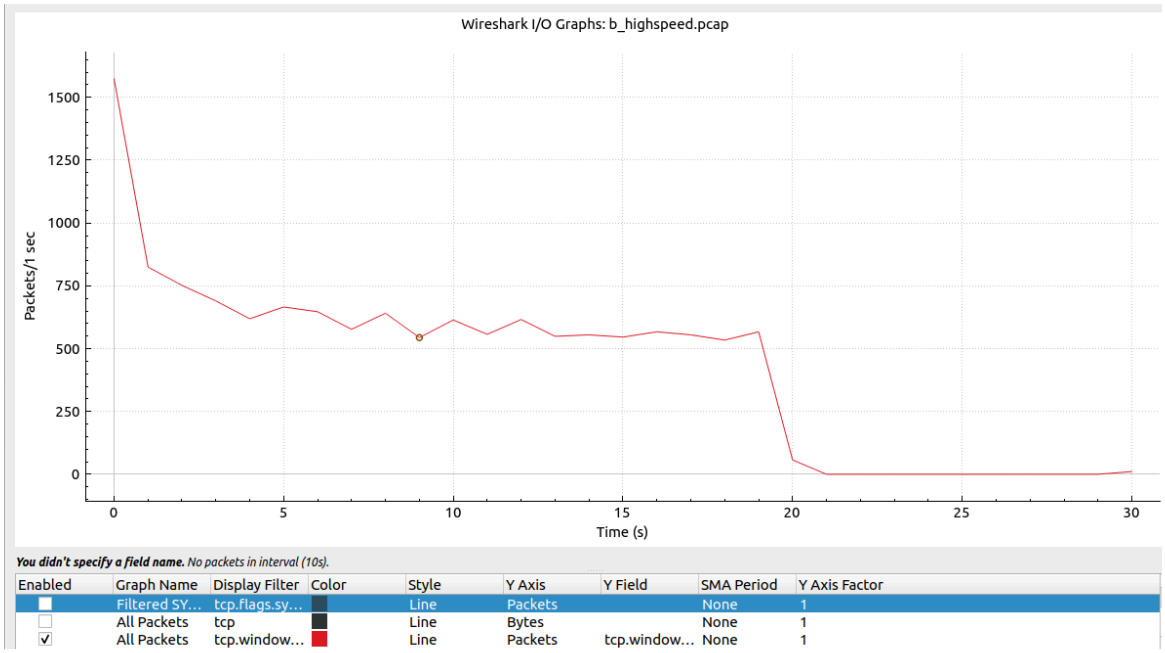


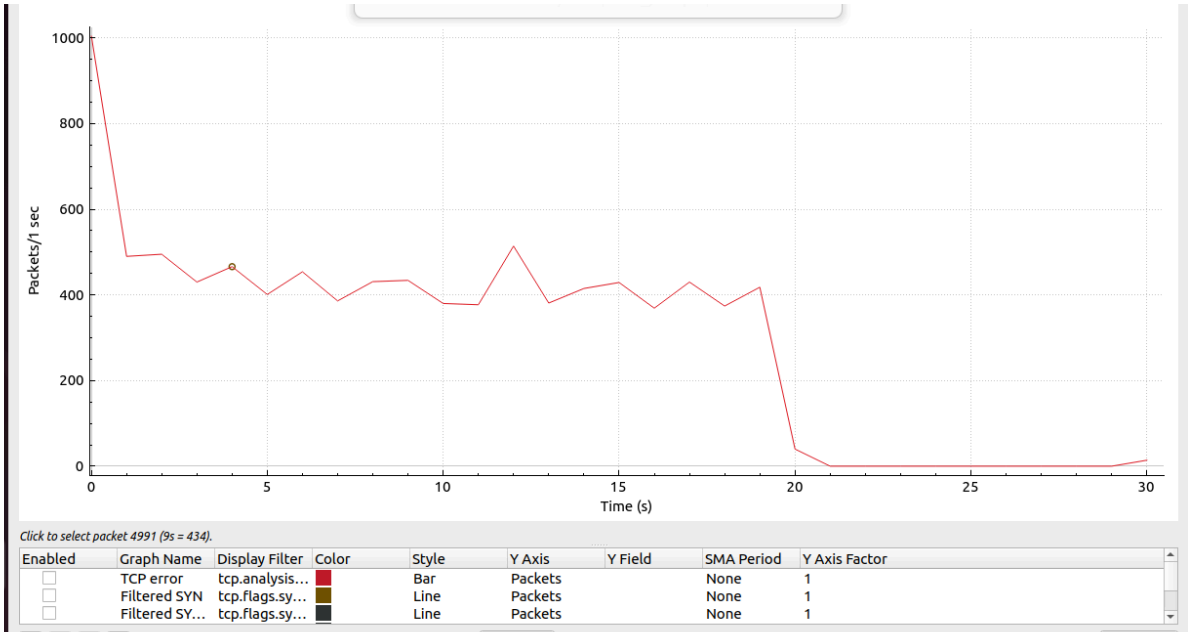




Graphs for window size:


Wireshark · I/O Graphs · b_bic.pcap
Wireshark I/O Graphs: b_bic.pcap
Packets/1 sec
Time (s)
0
200
400
600
800
1000
0
5
10
15
20
25
30
Click to select packet 1684 (5s = 539).
Enabled Graph Name Display Filter Color Style Y Axis Y Field SMA Period Y Axis Factor
TCP error tcp.analysis… Bar Packets None 1
Filtered SYN tcp.flags.sy… Line Packets None 1
Filtered SY… tcp.flags.sy… Line Packets None 1


Wireshark I/O Graphs: b_highspeed.pcap
Packets/1 sec
Time (s)
0
250
500
750
1000
1250
1500
0
5
10
15
20
25
30
You didn't specify a field name. No packets in interval (10s).
Enabled Graph Name Display Filter Color Style Y Axis Y Field SMA Period Y Axis Factor
Filtered SY… tcp.flags.sy… Line Packets None 1
All Packets tcp Line Bytes None 1
All Packets tcp.window… Line Packets tcp.window… None 1



Observations:

- The throughput order is Highspeed, BIC, Yeah
- However difference between throughput and goodput is higher for Highspeed than the other two.
- The window size range is also higher for highspeed than the other two.
- The packet loss is highest for BIC, packet loss is higher than in a for all cases

C.

```
neerja@Ubuntu:~/Documents/cn assignment 2$ sudo python3 q1.py --part c
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
1. Link S2-S4 is active with client on H3 and server on H7.
Packet capture saved to /tmp/c_bic_case1.pcap
Metrics for TCP bic (case1):
Throughput (KBps): 609.91
Goodput (KBps): 597.30
Packet Loss Rate: 0.02%
Max TCP Window Size: 43440 bytes

2.Link S1-S4 is active with:
a. Running client on H1 and H2 and server on H7
Packet capture saved to /tmp/c_bic_case2a.pcap
Metrics for TCP bic (case2a):
Throughput (KBps): 0.07
Goodput (KBps): 0.00
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

b. Running client on H1 and H3 and server on H7
Packet capture saved to /tmp/c_bic_case2b.pcap
Metrics for TCP bic (case2b):
Throughput (KBps): 1.32
Goodput (KBps): 0.11
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

c. Running client on H1, H3 and H4 and server on H7
Packet capture saved to /tmp/c_bic_case2c.pcap
Metrics for TCP bic (case2c):
Throughput (KBps): 0.08
Goodput (KBps): 0.00
Packet Loss Rate: 1.77%
Max TCP Window Size: 43440 bytes
```

```
1. Link S2-S4 is active with client on H3 and server on H7.
Packet capture saved to /tmp/c_highspeed_case1.pcap
Metrics for TCP highspeed (case1):
Throughput (KBps): 4422.69
Goodput (KBps): 4369.68
Packet Loss Rate: 0.01%
Max TCP Window Size: 43440 bytes

2.Link S1-S4 is active with:
a. Running client on H1 and H2 and server on H7
Packet capture saved to /tmp/c_highspeed_case2a.pcap
Metrics for TCP highspeed (case2a):
Throughput (KBps): 0.06
Goodput (KBps): 0.00
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

b. Running client on H1 and H3 and server on H7
Packet capture saved to /tmp/c_highspeed_case2b.pcap
Metrics for TCP highspeed (case2b):
Throughput (KBps): 0.07
Goodput (KBps): 0.00
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

c. Running client on H1, H3 and H4 and server on H7
Packet capture saved to /tmp/c_highspeed_case2c.pcap
Metrics for TCP highspeed (case2c):
Throughput (KBps): 0.07
Goodput (KBps): 0.00
Packet Loss Rate: 0.92%
Max TCP Window Size: 43440 bytes

1. Link S2-S4 is active with client on H3 and server on H7.
Packet capture saved to /tmp/c_yeah_case1.pcap
Metrics for TCP yeah (case1):
Throughput (KBps): 4126.76
Goodput (KBps): 4081.42
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes
```

```
2.Link S1-S4 is active with:
a. Running client on H1 and H2 and server on H7
Packet capture saved to /tmp/c_yeah_case2a.pcap
Metrics for TCP yeah (case2a):
Throughput (KBps): 1.38
Goodput (KBps): 0.11
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

b. Running client on H1 and H3 and server on H7
Packet capture saved to /tmp/c_yeah_case2b.pcap
Metrics for TCP yeah (case2b):
Throughput (KBps): 1.51
Goodput (KBps): 0.11
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

c. Running client on H1, H3 and H4 and server on H7
Packet capture saved to /tmp/c_yeah_case2c.pcap
Metrics for TCP yeah (case2c):
Throughput (KBps): 1.54
Goodput (KBps): 0.12
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes
```

Graphs for throughput and window size : [link]

Observations:

- YEAH TCP is the best performer, with high throughput, high goodput, and low packet loss across all cases.
- BIC TCP performs well in single-client scenarios but struggles with multi-client setups, showing higher packet loss.
- HighSpeed TCP shows the worst performance, especially in single-client tests where its goodput is nearly zero.

D.

(the pcap files have the name c_* since I didnt change them in the code but they are for part d)

```
neerja@Ubuntu:~/Documents/cn assignment 2$ sudo python3 q1.py --part d
S2-S3 loss is 1%
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
1. Link S2-S4 is active with client on H3 and server on H7.
Packet capture saved to /tmp/c_bic_case1.pcap
Metrics for TCP bic (case1):
Throughput (KBps): 677.65
Goodput (KBps): 663.31
Packet Loss Rate: 0.01%
Max TCP Window Size: 43440 bytes

2.Link S1-S4 is active with:
a. Running client on H1 and H2 and server on H7
Packet capture saved to /tmp/c_bic_case2a.pcap
Metrics for TCP bic (case2a):
Throughput (KBps): 0.07
Goodput (KBps): 0.00
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

b. Running client on H1 and H3 and server on H7
Packet capture saved to /tmp/c_bic_case2b.pcap
Metrics for TCP bic (case2b):
Throughput (KBps): 0.06
Goodput (KBps): 0.00
Packet Loss Rate: 0.93%
Max TCP Window Size: 43440 bytes
```

```
c. Running client on H1, H3 and H4 and server on H7
Packet capture saved to /tmp/c_bic_case2c.pcap
Metrics for TCP bic (case2c):
Throughput (KBps): 0.09
Goodput (KBps): 0.01
Packet Loss Rate: 0.90%
Max TCP Window Size: 43440 bytes

1. Link S2-S4 is active with client on H3 and server on H7.
Packet capture saved to /tmp/c_highspeed_case1.pcap
Metrics for TCP highspeed (case1):
Throughput (KBps): 9.58
Goodput (KBps): 0.00
Packet Loss Rate: 0.00%
Max TCP Window Size: 0 bytes

2.Link S1-S4 is active with:
a. Running client on H1 and H2 and server on H7
Packet capture saved to /tmp/c_highspeed_case2a.pcap
Metrics for TCP highspeed (case2a):
Throughput (KBps): 0.10
Goodput (KBps): 0.00
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

b. Running client on H1 and H3 and server on H7
Packet capture saved to /tmp/c_highspeed_case2b.pcap
Metrics for TCP highspeed (case2b):
Throughput (KBps): 0.08
Goodput (KBps): 0.01
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

c. Running client on H1, H3 and H4 and server on H7
Packet capture saved to /tmp/c_highspeed_case2c.pcap
Metrics for TCP highspeed (case2c):
Throughput (KBps): 0.09
Goodput (KBps): 0.01
```

```
Throughput (KBps): 0.09
Goodput (KBps): 0.01
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

1. Link S2-S4 is active with client on H3 and server on H7.
Packet capture saved to /tmp/c_yeah_case1.pcap
Metrics for TCP yeah (case1):
Throughput (KBps): 2032.54
Goodput (KBps): 1994.32
Packet Loss Rate: 0.01%
Max TCP Window Size: 43440 bytes

2.Link S1-S4 is active with:
a. Running client on H1 and H2 and server on H7
[ saved to /tmp/c_yeah_case2a.pcap
Ubuntu Software
yeah (case2a):
Throughput (KBps): 1.55
Goodput (KBps): 0.11
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

b. Running client on H1 and H3 and server on H7
Packet capture saved to /tmp/c_yeah_case2b.pcap
Metrics for TCP yeah (case2b):
Throughput (KBps): 1.49
Goodput (KBps): 0.11
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

c. Running client on H1, H3 and H4 and server on H7
Packet capture saved to /tmp/c_yeah_case2c.pcap
Metrics for TCP yeah (case2c):
Throughput (KBps): 0.07
Goodput (KBps): 0.00
Packet Loss Rate: 0.82%
Max TCP Window Size: 43440 bytes
```

```
S2-S3 loss is 5%
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
*** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
1. Link S2-S4 is active with client on H3 and server on H7.
Packet capture saved to /tmp/c_bic_case1.pcap
Metrics for TCP bic (case1):
Throughput (KBps): 677.65
Goodput (KBps): 663.31
Packet Loss Rate: 0.01%
Max TCP Window Size: 43440 bytes

2.Link S1-S4 is active with:
a. Running client on H1 and H2 and server on H7
Packet capture saved to /tmp/c_bic_case2a.pcap
Metrics for TCP bic (case2a):
Throughput (KBps): 0.07
Goodput (KBps): 0.00
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

b. Running client on H1 and H3 and server on H7
Packet capture saved to /tmp/c_bic_case2b.pcap
Metrics for TCP bic (case2b):
Throughput (KBps): 0.06
Goodput (KBps): 0.00
Packet Loss Rate: 0.93%
Max TCP Window Size: 43440 bytes

c. Running client on H1, H3 and H4 and server on H7
Packet capture saved to /tmp/c_bic_case2c.pcap
Metrics for TCP bic (case2c):
Throughput (KBps): 0.09
Goodput (KBps): 0.01
Packet Loss Rate: 0.90%
```

```
Thunderbird Mail
s active with client on H3 and server on H7.
saved to /tmp/c_highspeed_case1.pcap
Metrics for TCP highspeed (case1):
Throughput (KBps): 9.58
Goodput (KBps): 0.00
Packet Loss Rate: 0.00%
Max TCP Window Size: 0 bytes

2.Link S1-S4 is active with:
a. Running client on H1 and H2 and server on H7
Packet capture saved to /tmp/c_highspeed_case2a.pcap
Metrics for TCP highspeed (case2a):
Throughput (KBps): 0.10
Goodput (KBps): 0.00
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

b. Running client on H1 and H3 and server on H7
Packet capture saved to /tmp/c_highspeed_case2b.pcap
Metrics for TCP highspeed (case2b):
Throughput (KBps): 0.08
Goodput (KBps): 0.01
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

c. Running client on H1, H3 and H4 and server on H7
Packet capture saved to /tmp/c_highspeed_case2c.pcap
Metrics for TCP highspeed (case2c):
Throughput (KBps): 0.09
Goodput (KBps): 0.01
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

1. Link S2-S4 is active with client on H3 and server on H7.
Packet capture saved to /tmp/c_yeah_case1.pcap
Metrics for TCP yeah (case1):
Throughput (KBps): 2032.54
Goodput (KBps): 1994.32
```

```
Throughput (KBps): 0.09
Goodput (KBps): 0.01
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

1. Link S2-S4 is active with client on H3 and server on H7.
Packet capture saved to /tmp/c_yeah_case1.pcap
Metrics for TCP yeah (case1):
Rhythmbox (KBps): 2032.54
Goodput (KBps): 1994.32
Packet Loss Rate: 0.01%
Max TCP Window Size: 43440 bytes

2.Link S1-S4 is active with:
a. Running client on H1 and H2 and server on H7
Packet capture saved to /tmp/c_yeah_case2a.pcap
Metrics for TCP yeah (case2a):
Throughput (KBps): 1.55
Goodput (KBps): 0.11
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

b. Running client on H1 and H3 and server on H7
Packet capture saved to /tmp/c_yeah_case2b.pcap
Metrics for TCP yeah (case2b):
Throughput (KBps): 1.49
Goodput (KBps): 0.11
Packet Loss Rate: 0.00%
Max TCP Window Size: 43440 bytes

c. Running client on H1, H3 and H4 and server on H7
Packet capture saved to /tmp/c_yeah_case2c.pcap
Metrics for TCP yeah (case2c):
Throughput (KBps): 0.07
Goodput (KBps): 0.00
Packet Loss Rate: 0.82%
Max TCP Window Size: 43440 bytes
```

Graphs for throughput and window size : [link]

Observations:

- The throughput is very low for 1% loss but increases for 5%, relative observations are same as in C.

Task-2 : Implementation and mitigation of SYN flood attack

A. Implementation of SYN flood attack

On the server:

```
neerja@UbuntuVictim:~$ su -
Password:
root@UbuntuVictim:~# sysctl -w net.ipv4.tcp_max_syn_backlog=65535
sysctl -w net.ipv4.tcp_syncookies=0
sysctl -w net.ipv4.tcp_synack_retries=1
net.ipv4.tcp_max_syn_backlog = 65535
net.ipv4.tcp_syncookies = 0
net.ipv4.tcp_synack_retries = 1
root@UbuntuVictim:~# exit
logout
neerja@UbuntuVictim:~$ nc -l -p 8080
```

On the client:

```
neerja@Ubuntu:~$ sudo tcpdump -i enp0s8 -w attack.pcap
[sudo] password for neerja:
tcpdump: listening on enp0s8, link-type EN10MB (Ethernet), snapshot length 262144 bytes
S
```

```
neerja@Ubuntu:~$ while true; do nc 192.168.1.100 8080 < /dev/null; sleep 2; done
```

```
neerja@Ubuntu:~$ sudo hping3 -S --flood -p 8080 192.168.1.100
HPING 192.168.1.100 (enp0s8 192.168.1.100): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
```

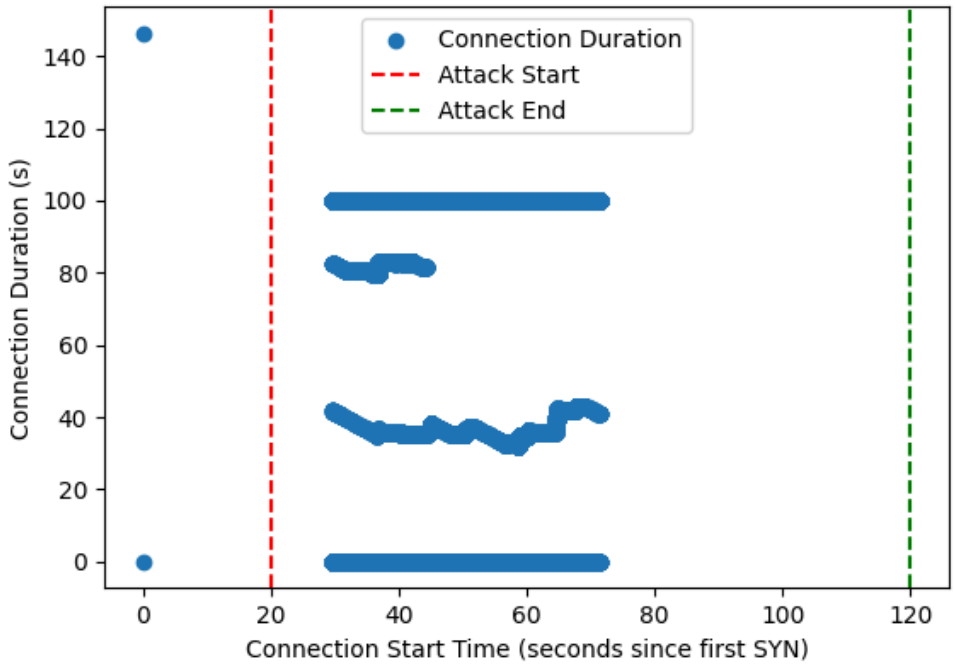
Commands used to extract connection details:

tshark -r attack.pcap -Y "tcp.flags & 0x04 || tcp.flags & 0x01" -T fields -e frame.time_epoch -e ip.src -e ip.dst -e tcp.srcport -e tcp.dstport -e tcp.flags > fin_reset_connections2.txt

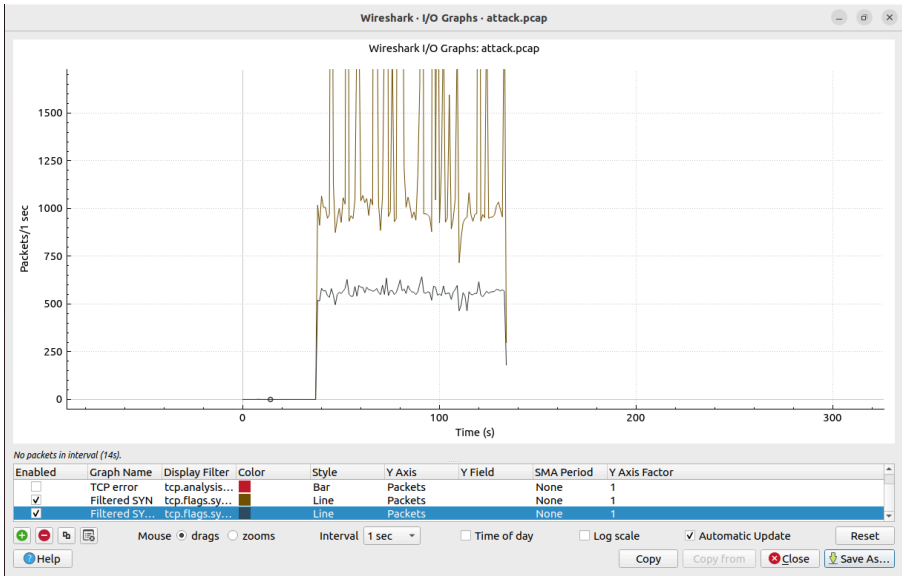
tshark -r attack.pcap -Y "tcp.flags.syn == 1 && tcp.flags.ack == 0" -T fields -e frame.time_epoch -e ip.src -e ip.dst -e tcp.srcport -e tcp.dstport > syn_connections2.txt

tshark -r attack.pcap -Y "tcp.flags.ack == 1 && tcp.flags.fin == 0" -T fields -e frame.time_epoch -e ip.src -e ip.dst -e tcp.srcport -e tcp.dstport > ack_packets.txt

Here a great majority of the connections have duration 100 which means they are the connections that remained half open.



We can observe below that number of SYN packets is very large and SYN ACKS is lower in Wireshark IO graph which indicates a successful SYN flood attack



B. Apply SYN flood attack mitigation

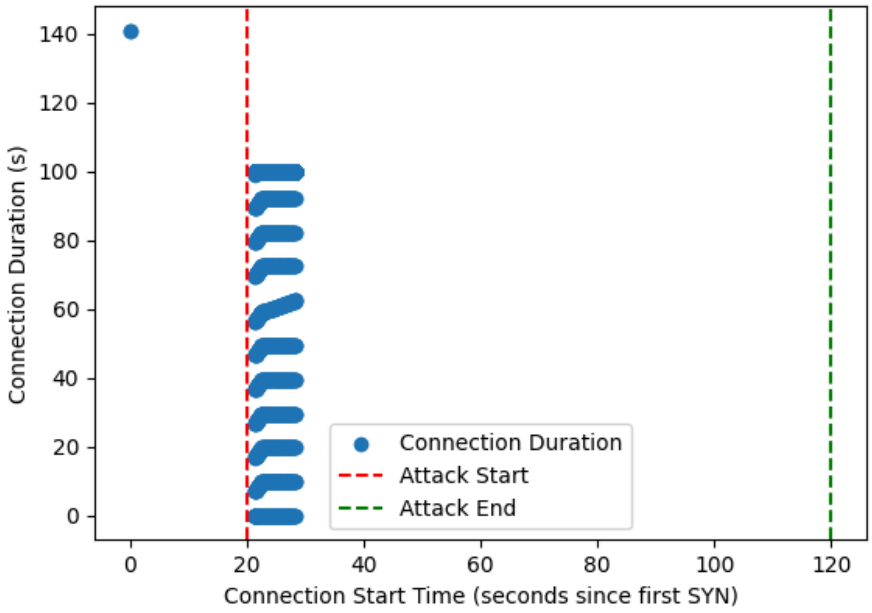
Strategy used:

- SYN cookies enable legitimate connections to go through, even under attack.
- Limit syn backlog to avoid exhaustion
- Increase the SYN-ACK retries to allow the legitimate connections
- Use iptables to drop excessive syn packets from the same source.

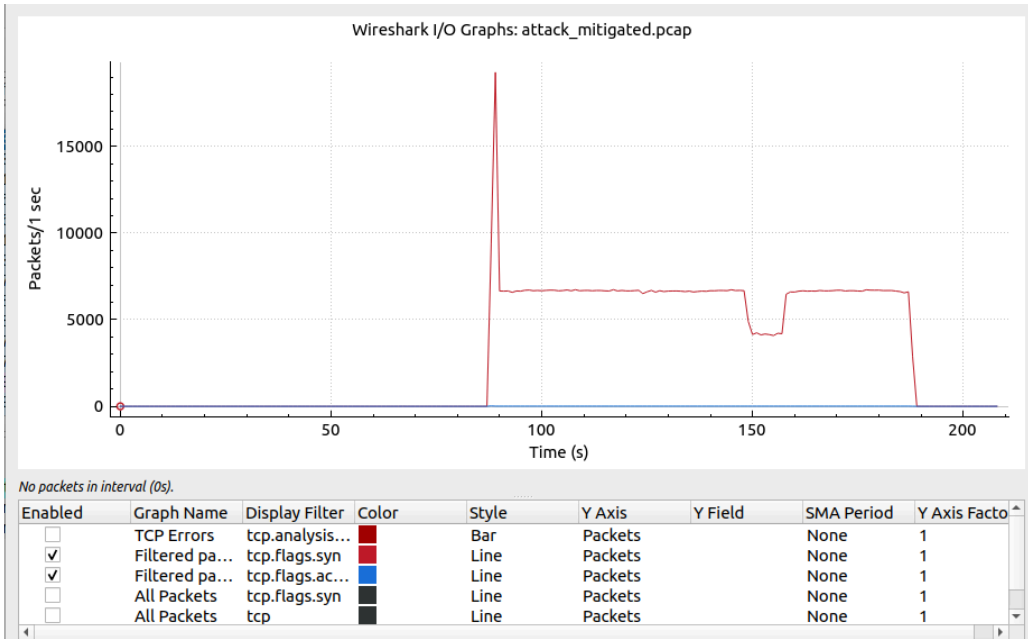
```
neerja@UbuntuVictim:~$ sudo sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1
neerja@UbuntuVictim:~$ sudo sysctl -w net.ipv4.tcp_synack_retries=5
net.ipv4.tcp_synack_retries = 5
neerja@UbuntuVictim:~$ sudo sysctl -w net.ipv4.tcp_max_syn_backlog=4096
net.ipv4.tcp_max_syn_backlog = 4096
neerja@UbuntuVictim:~$ sudo iptables -A INPUT -p tcp --syn -m limit --limit 10/s
 --limit-burst 20 -j ACCEPT
sudo iptables -A INPUT -p tcp --syn -j DROP
neerja@UbuntuVictim:~$
```

Repeat the same commands on the client side.

We can observe that the number of connections with duration 100 has decreased and most now have a reasonably small duration.



In the below graph we observe the initial spike associated with syn flood attack, followed by a stabilization which means that mitigation worked.



Task-3: Analyze the effect of Nagle's algorithm on TCP/IP performance.

Capture pcap file using:

```
neerja@Ubuntu:~/Documents/cn assignment 2$ sudo tcpdump -i lo port 12345 -w capture_01.pcap
[sudo] password for neerja:
tcpdump: listening on lo, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C212 packets captured
424 packets received by filter
0 packets dropped by kernel
```

1. Nagle enabled, Delayed ACK enabled

```
neerja@Ubuntu:~/Documents/cn assignment 2$ python3 analyze_pcap.py
Throughput (Bps): 175.29
Goodput (Bps): 39.66
Packet Loss Rate: 0.00%
Max Packet Size: 78 bytes
```

2. Nagle enabled, Delayed Ack disabled

```
neerja@Ubuntu:~/Documents/cn assignment 2$ python3 analyze_pcap.py
Throughput (Bps): 175.19
Goodput (Bps): 39.64
Packet Loss Rate: 0.00%
Max Packet Size: 78 bytes
```

3. Nagle disabled, Delayed Ack enabled

```
neerja@Ubuntu:~/Documents/cn assignment 2$ python3 analyze_pcap.py
Throughput (Bps): 175.32
Goodput (Bps): 39.67
Packet Loss Rate: 0.00%
Max Packet Size: 78 bytes
```

4. Nagle disabled, Delayed ACK disabled

```
neerja@Ubuntu:~/Documents/cn assignment 2$ python3 analyze_pcap.py
Throughput (Bps): 175.03
Goodput (Bps): 39.60
Packet Loss Rate: 0.00%
Max Packet Size: 78 bytes
```

Observations:

- Enabling delayed ack improves throughput since there is less overhead.
- Enabling nagle increases the throughput with delayed ack disabled but when delayed ack is enabled it masks the nagle effect thus reducing the throughput comparatively.