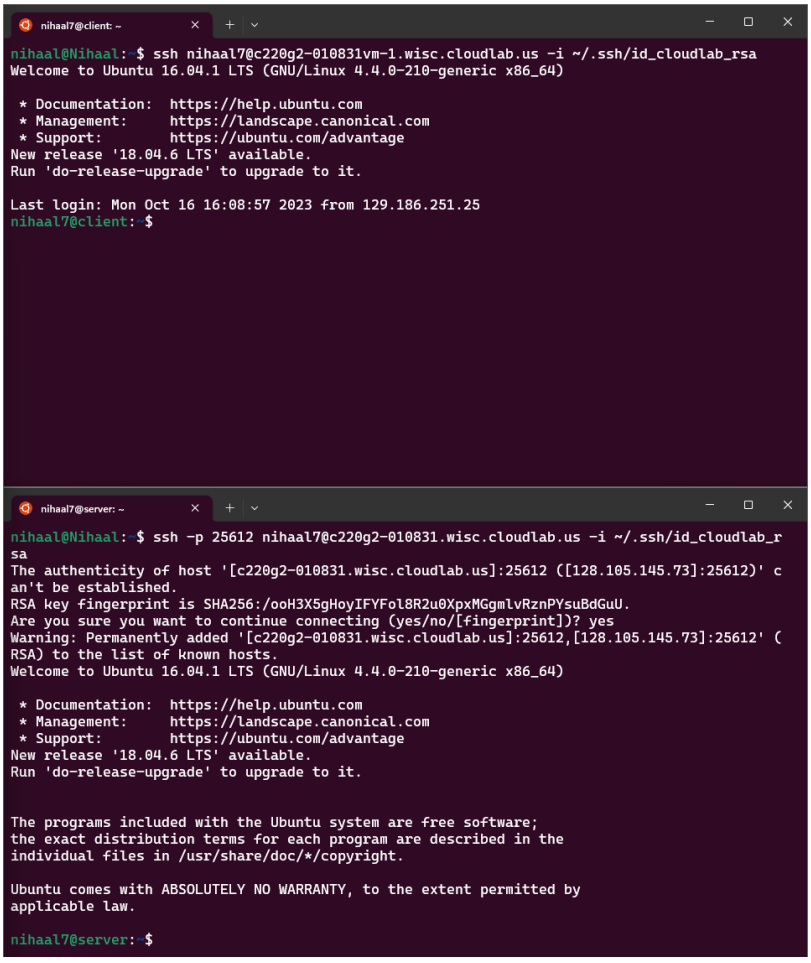
LAB 5

Summary

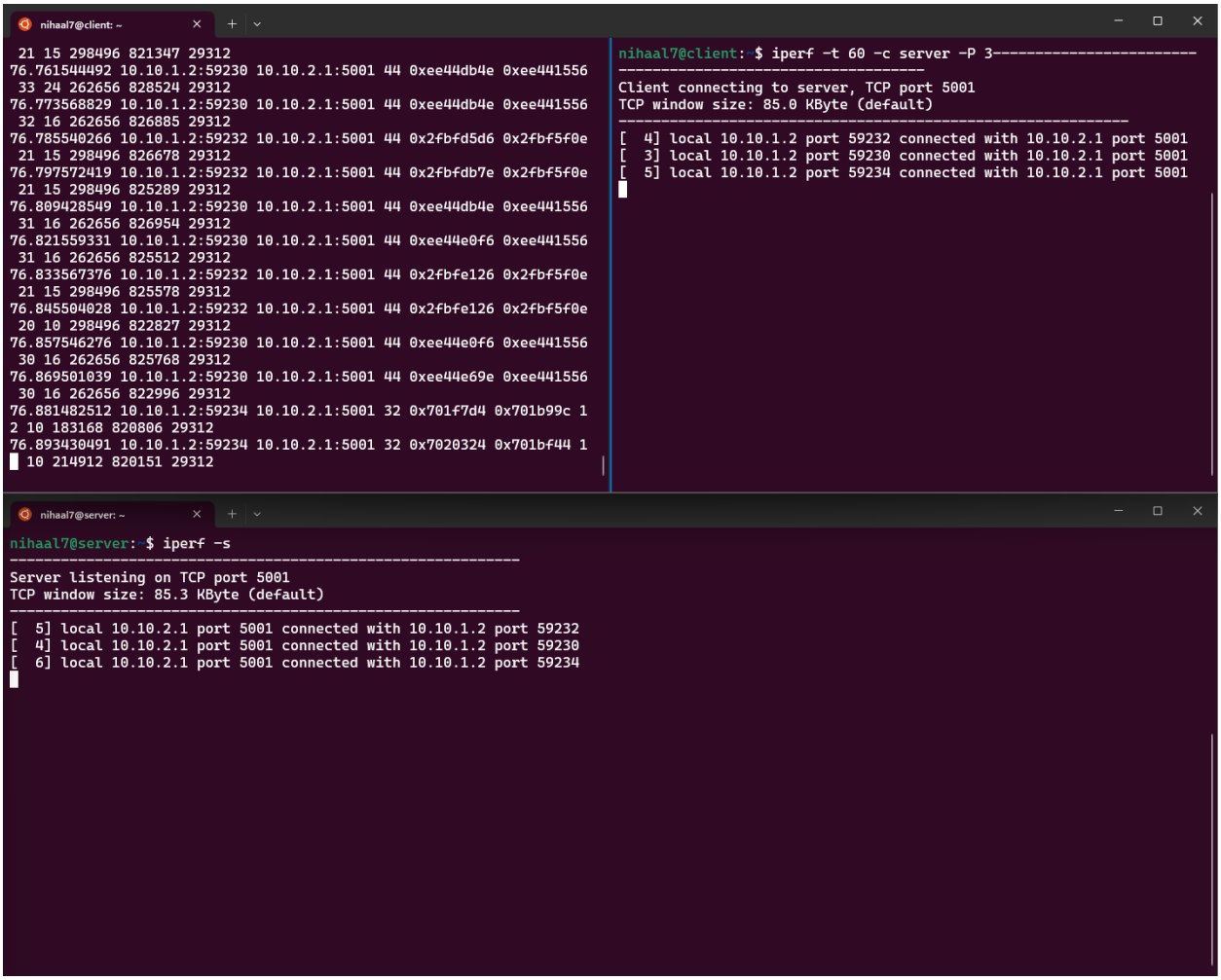
This lab focuses on evaluating TCP congestion protocol by collecting, visualizing, and analyzing data.

Firstly, we run iperf, and get the data in a tcpprobe.dat file. We then share this file from the cloud lab server to our personal computer using SCP. Then, using bash shell and scripting code, we convert this into a CSV file and extract 3 CSV files separated by source IPs. Using these we generate graphs to analyze data.

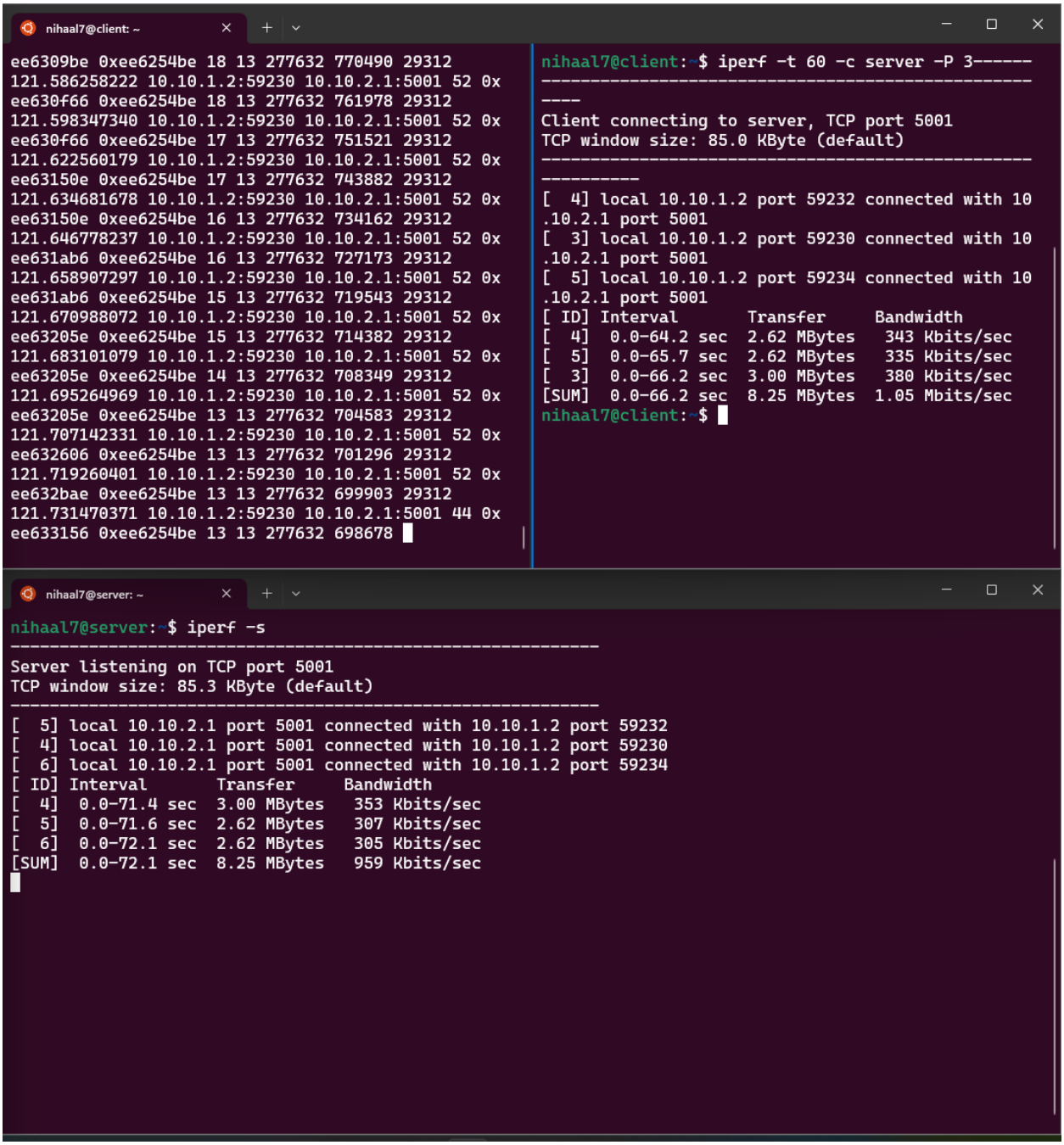
Exercise 1:



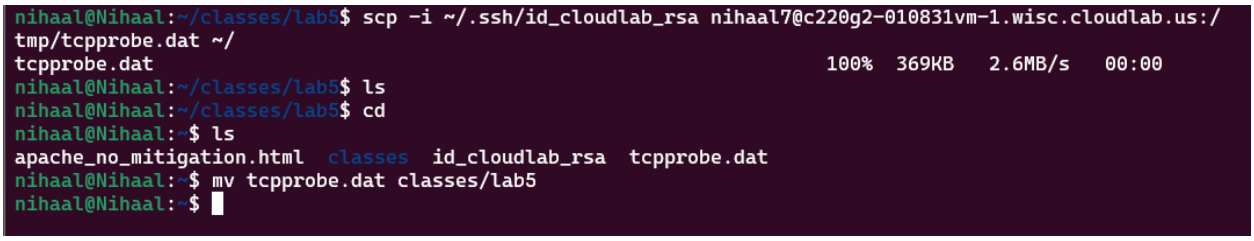
5. iperf running;



Iperf output;


nihaal7@client: ~
ee6309be 0xee6254be 18 13 277632 770490 29312
121.586258222 10.10.1.2:59230 10.10.2.1:5001 52 0x
ee630f66 0xee6254be 18 13 277632 761978 29312
121.598347340 10.10.1.2:59230 10.10.2.1:5001 52 0x
ee630f66 0xee6254be 17 13 277632 751521 29312
121.622560179 10.10.1.2:59230 10.10.2.1:5001 52 0x
ee63150e 0xee6254be 17 13 277632 743882 29312
121.634681678 10.10.1.2:59230 10.10.2.1:5001 52 0x
ee63150e 0xee6254be 16 13 277632 734162 29312
121.646778237 10.10.1.2:59230 10.10.2.1:5001 52 0x
ee631ab6 0xee6254be 16 13 277632 727173 29312
121.658907297 10.10.1.2:59230 10.10.2.1:5001 52 0x
ee631ab6 0xee6254be 15 13 277632 719543 29312
121.670988072 10.10.1.2:59230 10.10.2.1:5001 52 0x
ee63205e 0xee6254be 15 13 277632 714382 29312
121.683101079 10.10.1.2:59230 10.10.2.1:5001 52 0x
ee63205e 0xee6254be 14 13 277632 708349 29312
121.695264969 10.10.1.2:59230 10.10.2.1:5001 52 0x
ee63205e 0xee6254be 13 13 277632 704583 29312
121.707142331 10.10.1.2:59230 10.10.2.1:5001 52 0x
ee632606 0xee6254be 13 13 277632 701296 29312
121.719260401 10.10.1.2:59230 10.10.2.1:5001 52 0x
ee632bae 0xee6254be 13 13 277632 699903 29312
121.731470371 10.10.1.2:59230 10.10.2.1:5001 44 0x
ee633156 0xee6254be 13 13 277632 698678
nihaal7@client:~$ iperf -t 60 -c server -P 3------
Client connecting to server, TCP port 5001
TCP window size: 85.0 KByte (default)
[4] local 10.10.1.2 port 59232 connected with 10
.10.2.1 port 5001
[3] local 10.10.1.2 port 59230 connected with 10
.10.2.1 port 5001
[5] local 10.10.1.2 port 59234 connected with 10
.10.2.1 port 5001
[ID] Interval Transfer Bandwidth
[4] 0.0-64.2 sec 2.62 MBytes 343 Kbits/sec
[5] 0.0-65.7 sec 2.62 MBytes 335 Kbits/sec
[3] 0.0-66.2 sec 3.00 MBytes 380 Kbits/sec
[SUM] 0.0-66.2 sec 8.25 MBytes 1.05 Mbits/sec
nihaal7@client:~$
nihaal7@server: ~
nihaal7@server:~$ iperf -s
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
[5] local 10.10.2.1 port 5001 connected with 10.10.1.2 port 59232
[4] local 10.10.2.1 port 5001 connected with 10.10.1.2 port 59230
[6] local 10.10.2.1 port 5001 connected with 10.10.1.2 port 59234
[ID] Interval Transfer Bandwidth
[4] 0.0-71.4 sec 3.00 MBytes 353 Kbits/sec
[5] 0.0-71.6 sec 2.62 MBytes 307 Kbits/sec
[6] 0.0-72.1 sec 2.62 MBytes 305 Kbits/sec
[SUM] 0.0-72.1 sec 8.25 MBytes 959 Kbits/sec

SCP:


nihaal@Nihaal:~/classes/lab5$ scp -i ~/.ssh/id_cloudlab_rsa nihaal7@c220g2-010831vm-1.wisc.cloudlab.us:/
tmp/tcpprobe.dat ~/
tcpprobe.dat 100% 369KB 2.6MB/s 00:00
nihaal@Nihaal:~/classes/lab5$ ls
nihaal@Nihaal:~/classes/lab5$ cd
nihaal@Nihaal:~$ ls
apache_no_mitigation.html classes id_cloudlab_rsa tcpprobe.dat
nihaal@Nihaal:~$ mv tcpprobe.dat classes/lab5
nihaal@Nihaal:~$

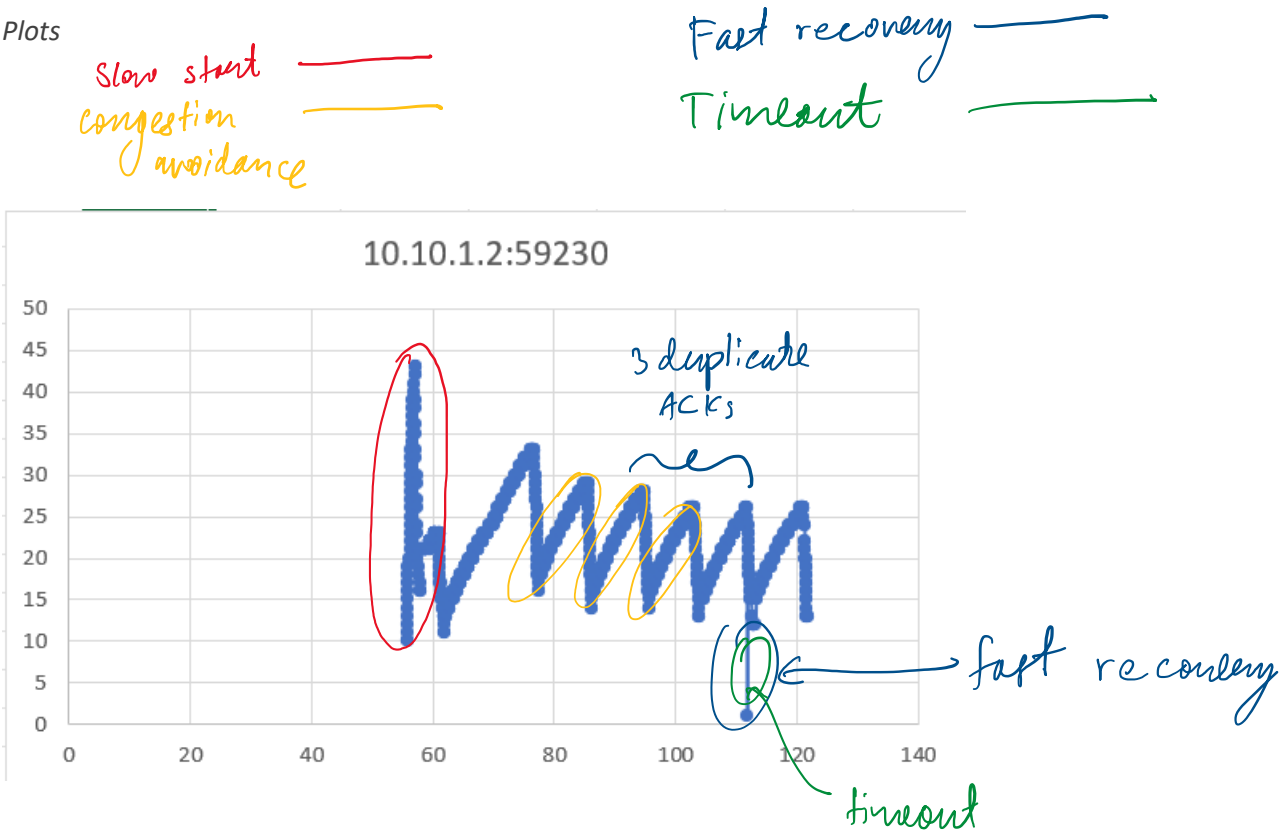
Exercise 2

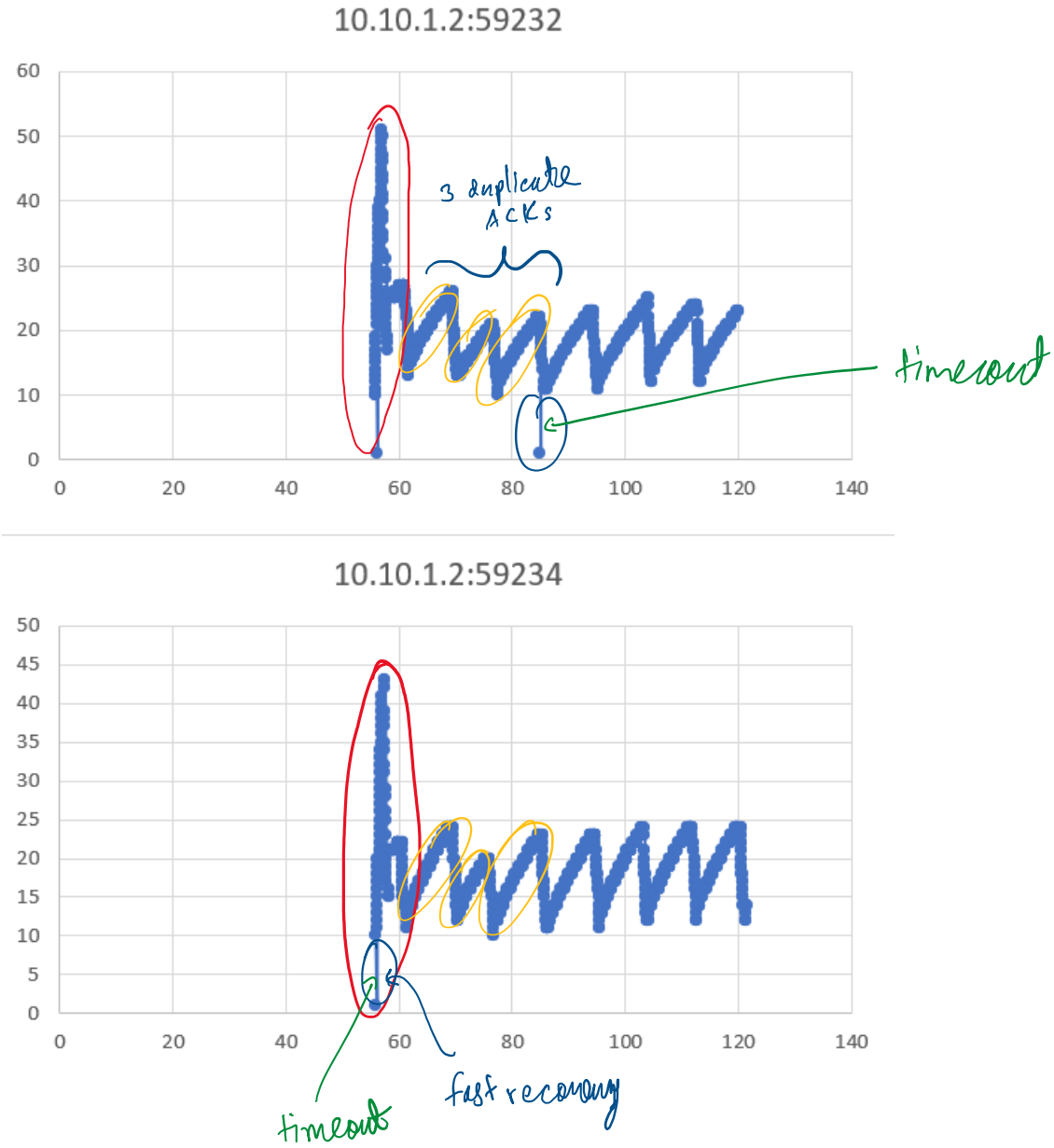
MINGW64:/c/Users/nihaa/OneDrive - Iowa State University/#Subjects/Sem 7/C...

```
#!/bin/bash

# Input CSV file path
input_file="tcpprobe.csv"

# Output CSV file path for matched lines
output_file="output.csv"

# Check if input file exists
if [ -e "$input_file" ]; then
    # Use awk to extract lines where the second column matches the IP address an
d port
    awk -F, '$2 == "10.10.1.2:59230"' "$input_file" > "$output_file"

    echo "Matching lines extracted and saved to $output_file"
else
    echo "Error: Input file not found."
fi

~
~
~
test.sh [unix] (14:15 19/10/2023)                                    12,29 All
"test.sh" [unix] 18L, 456B
```

This is the shell script I used to extract data from specific IP addresses from the tcpprobe.csv file. The tcpprobe.csv file was converted to a CSV file from the tcpprobe.dat file by removing white spaces and adding commas in its place. This was done by using awk in bash shell.

Plots





Annotation:

Explanation of behavior of TCP different from Slow Start and Congestion Avoidance

In slow start, because there isn't any congestion initially, TCP aggressively increases the sending rate, which can lead to rapid network congestion if not controlled.

Congestion avoidance on the other hand employs a more gradual approach, increasing the congestion window linearly. This prevents a sudden spike in network traffic and ensures a more stable data transfer process.

Explanation of what happens to congestion window and slow start threshold when 3 duplicate ACKs are received.

When 3 duplicate ACKs are received, TCP assumes that a packet has been lost and enters fast recovery. In this phase, the sender reduces its congestion window to a fraction of its previous value, thereby avoiding further congestion.

Exercise 3:

Termination