COMP 7003
Assignment 3
Report

Daryush Balsara
A01265967
Nov 2, 2024

Executive Summary.....3
Introduction.....3
1. Setting flags using hping3.....4
2. Start and End of a Session using netcat.....5
3. Same Data, Different Packets.....7
a. Within a Session.....7
IP.....9
TCP.....9
b. Across Sessions.....10
Key Differences and Explanations.....10
4. Website Download and TCP Analysis.....11
Throughput.....12
Window Size/Scale.....13
Window Congestion.....13
Retransmission in gray (There are none so the line stays at 0).....14
Bytes in fight in blue (if you click on the circle on the graph it highlights the get request to download.....14
Conclusion.....14

Executive Summary

- Tools Used: Wireshark, hping3, curl, ncat.

Session Management:

- Analyzed TCP session initiation and termination using SYN and FIN flags.
- Captured and explained the role of these flags in marking the start and end of a session.

Data Transmission Analysis:

- Comparing IP and TCP header fields by sending the same data within the same session and across different sessions.
- Identified changes in sequence numbers, acknowledgment numbers, and flags.

Network Performance:

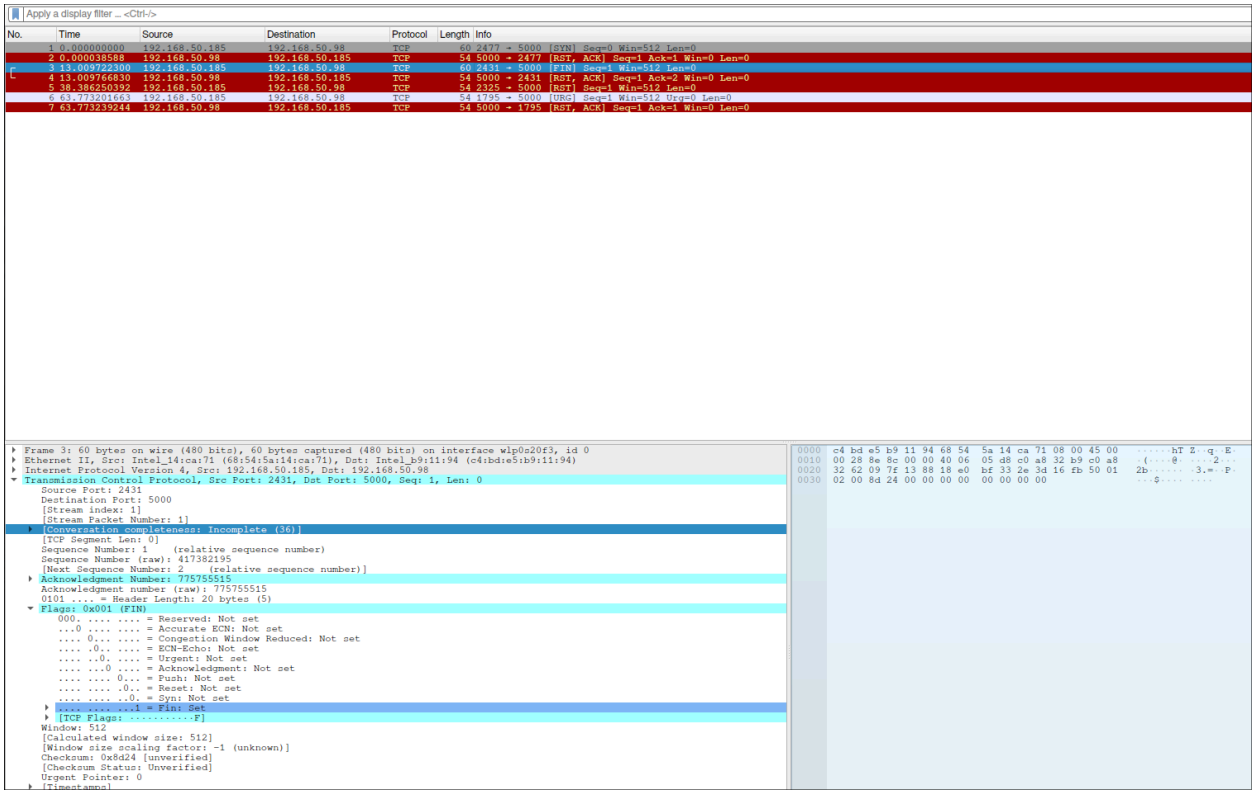
- Captured and analyzed network traffic during the download of a book from Project Gutenberg.

Introduction

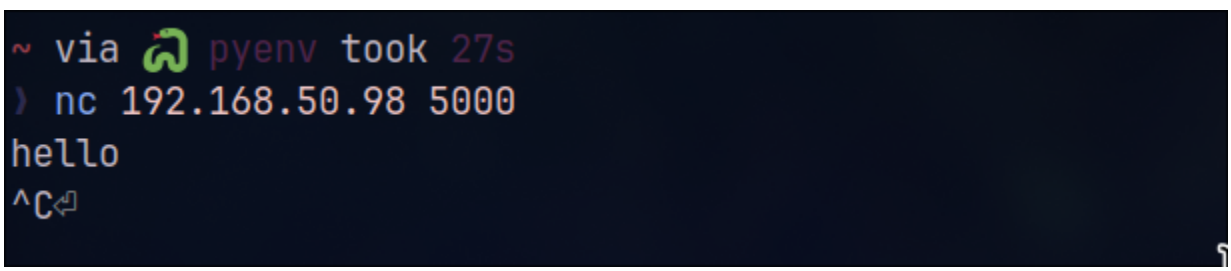
This assignment aims to provide a deep understanding of TCP protocol operation, focusing on session management, data identification, and performance optimization mechanisms. Through practical exercises involving tools like Wireshark, hping3, curl, and ncat, students will gain hands-on experience in analyzing and manipulating TCP traffic.

1. Setting flags using hping3

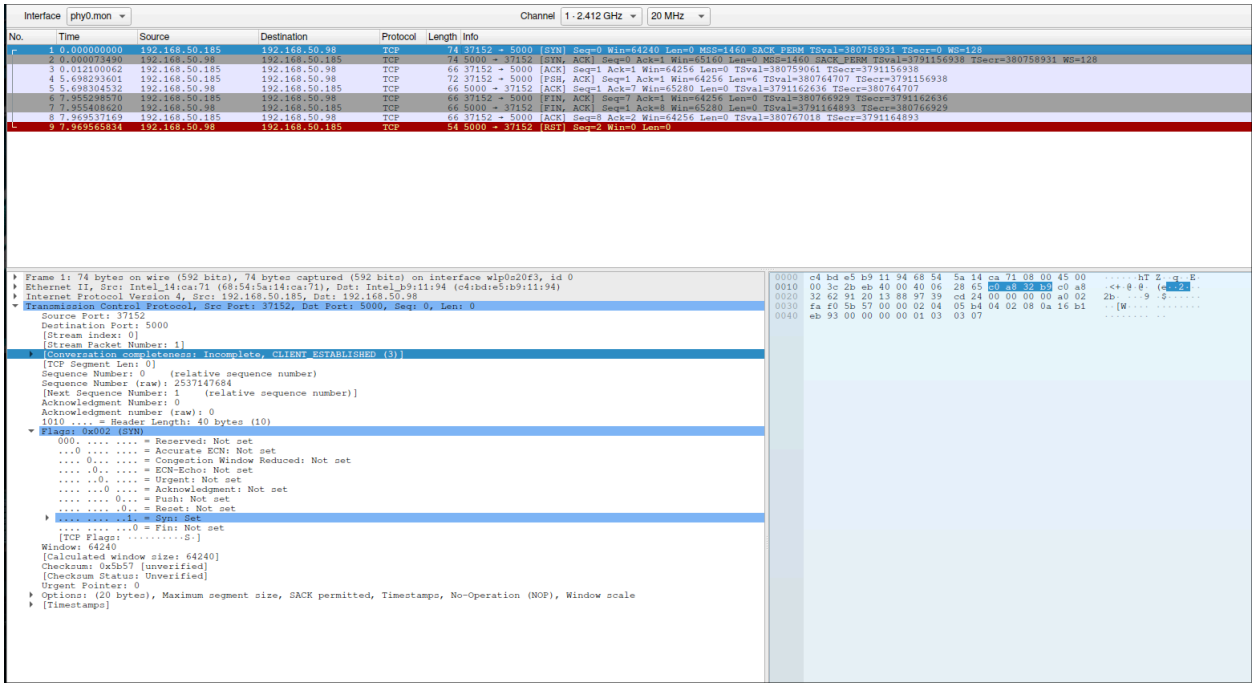
```
~ via pyenv took 3m17s
→ doas hping3 -S 192.168.50.98 -p 5000 -c 1
doas (daryush@JetFire) password:
HPING 192.168.50.98 (wlp8s0 192.168.50.98): S set, 40 headers + 0 data bytes
len=46 ip=192.168.50.98 ttl=64 DF id=0 sport=5000 flags=RA seq=0 win=0 rtt=76.5 ms

--- 192.168.50.98 hping statistic ---
1 packets tramitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 76.5/76.5/76.5 ms

~ via pyenv took 4s
→ doas hping3 -F 192.168.50.98 -p 5000 -c 1
doas (daryush@JetFire) password:
HPING 192.168.50.98 (wlp8s0 192.168.50.98): F set, 40 headers + 0 data bytes
len=46 ip=192.168.50.98 ttl=64 DF id=0 sport=5000 flags=RA seq=0 win=0 rtt=109.8 ms

--- 192.168.50.98 hping statistic ---
1 packets tramitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 109.8/109.8/109.8 ms

~ via pyenv took 4s
→ doas hping3 -R 192.168.50.98 -p 5000 -c 1
doas (daryush@JetFire) password:
HPING 192.168.50.98 (wlp8s0 192.168.50.98): R set, 40 headers + 0 data bytes

--- 192.168.50.98 hping statistic ---
1 packets tramitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms

~ via pyenv took 4s
) doas hping3 -U 192.168.50.98 -p 5000 -c 1
doas (daryush@JetFire) password:
doas: Authentication failed

~ via pyenv took 5s
) doas hping3 -U 192.168.50.98 -p 5000 -c 1
doas (daryush@JetFire) password:
HPING 192.168.50.98 (wlp8s0 192.168.50.98): U set, 40 headers + 0 data bytes
len=40 ip=192.168.50.98 ttl=64 DF id=0 sport=5000 flags=RA seq=0 win=0 rtt=119.8 ms

--- 192.168.50.98 hping statistic ---
1 packets tramitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 119.8/119.8/119.8 ms

~ via pyenv took 4s
→
```



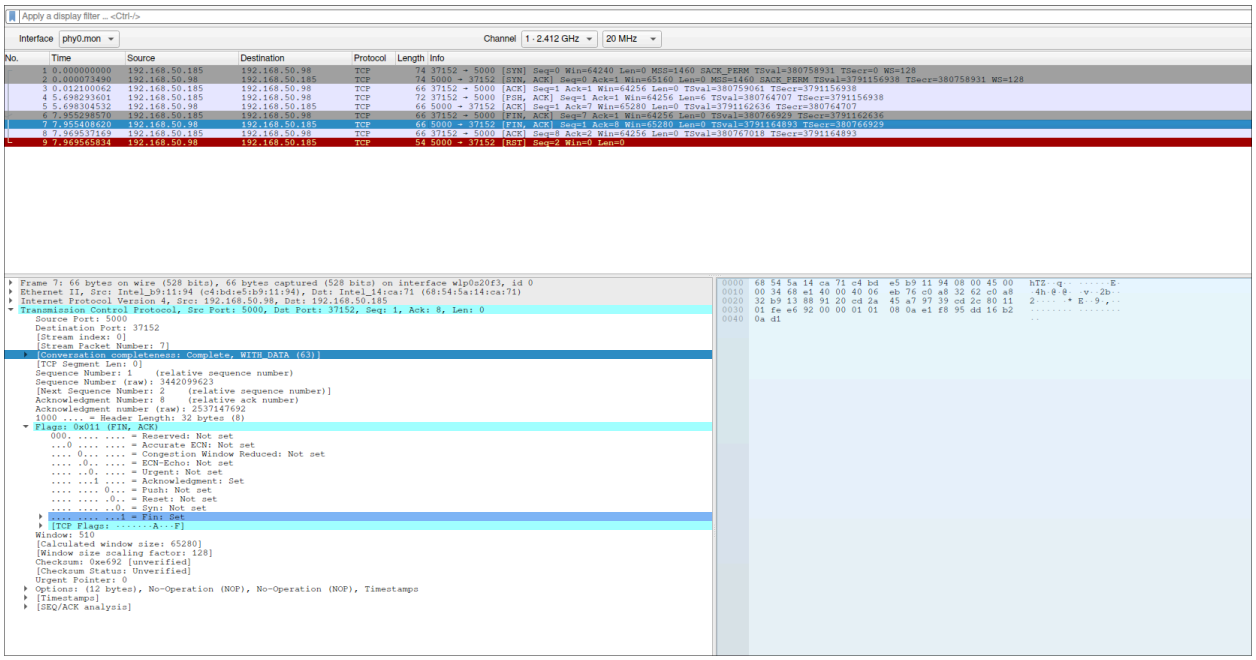
2. Start and End of a Session using netcat



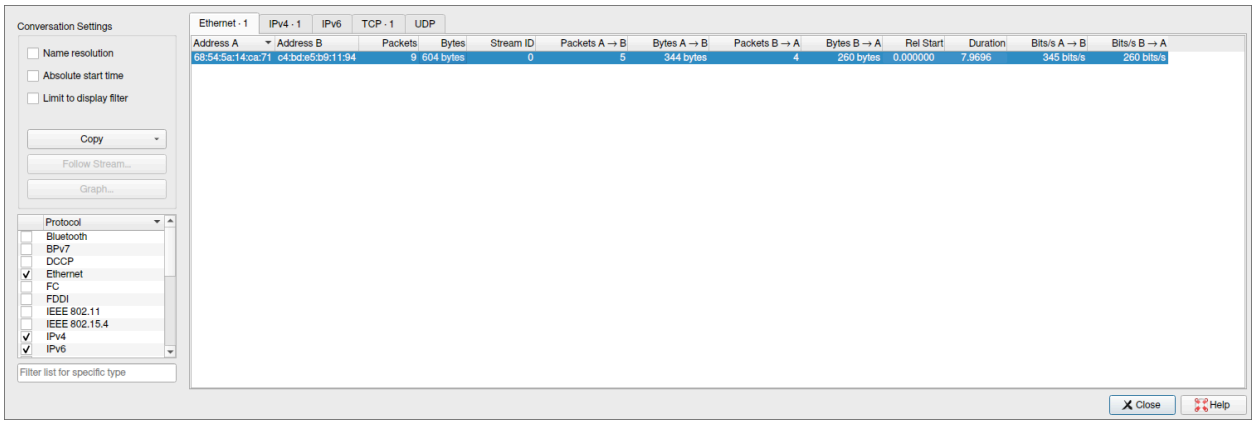
Purpose of the SYN packet allows for the start of the handshake. PC-1 (192.168.50.185) connects to PC-2 (192.168.50.98) while PC-2 listens on port 5000. When PC-1 sends the command for netcat to connect to the server at 192.168.50.98 it sends the SYN packet to establish the connection which it then receives the SYN-ACK and sends back a ACK (3 way handshakes). Once the connection is established PC-1 is allowed to communicate to PC-2.



Purpose of the FIN packet allows for the start of the teardown of the connection. Once PC-1 wants to terminate the connection after communicating to PC-2 it sends a FIN-ACK to start the teardown to PC-2. PC-2 will then send back a FIN-ACK and the connection will be terminated.



Amount of bytes being sent is 604. 344 bytes from my PC-1 (192.168.50.185) to PC-2 (192.168.50.98) and 260 bytes from my laptop to PC. If wireshark didn't show the bytes then if you add up all the lengths of each individual packet it equals the total bytes being sent which is 604 bytes.

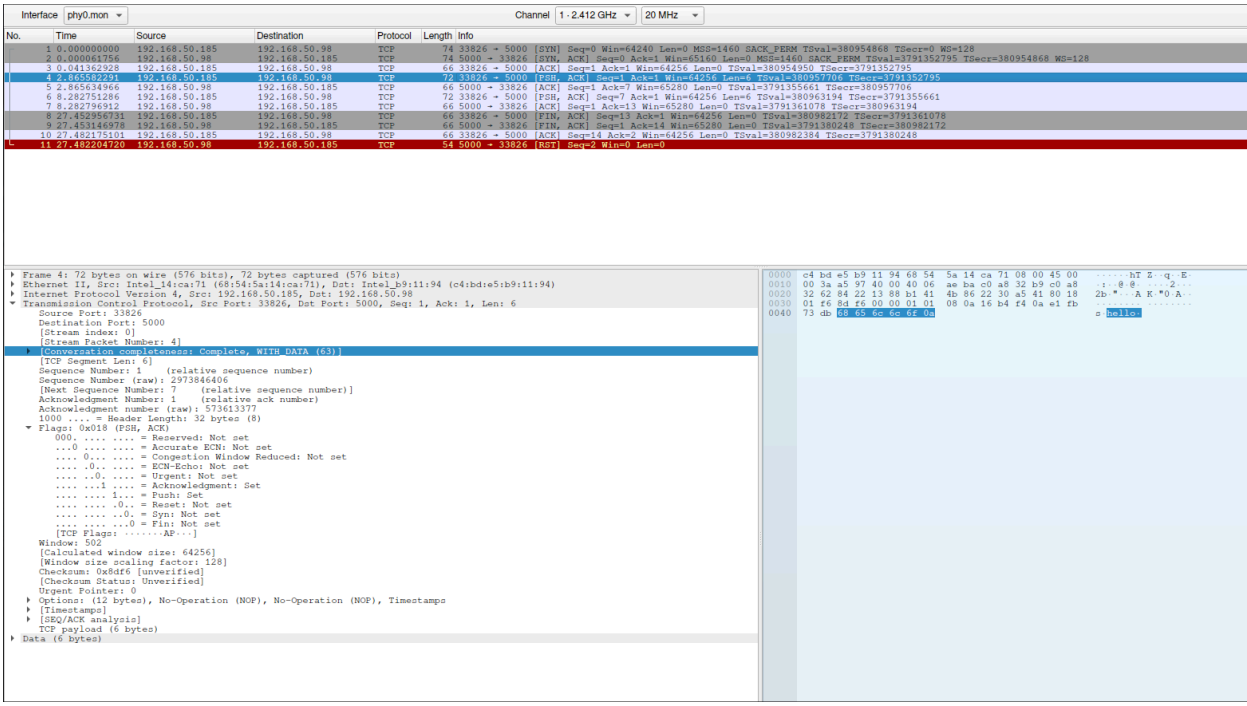


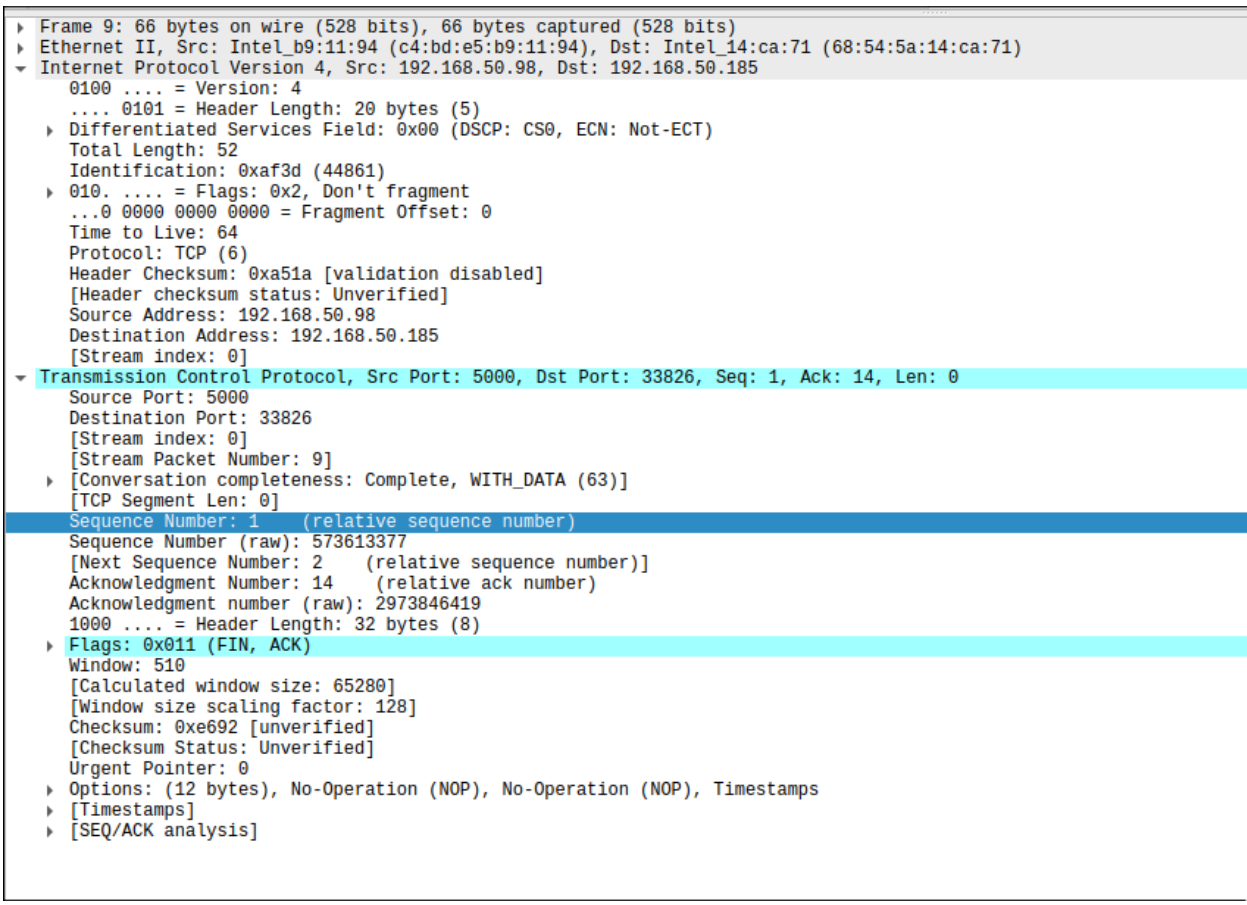
3. Same Data, Different Packets

a. Within a Session

```
~ via 🐍 pyenv took 8s
❯ nc 192.168.50.98 5000
hello
hello
^C⏎
```

```
~ took 17s
❯ nc -l -p 5000
hello
hello
```


Interface phy0.mon
Channel 1 · 2.412 GHz
20 MHz
No. Time Source Destination Protocol Length Info
1 0.000000000 192.168.50.185 192.168.50.98 TCP 74 33826 → 5000 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=380954868 TSecr=0 WS=128
2 0.000061756 192.168.50.98 192.168.50.185 TCP 74 5000 → 33826 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=3791352795 TSecr=380954868 WS=128
3 0.041362928 192.168.50.185 192.168.50.98 TCP 66 33826 → 5000 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=380954950 TSecr=3791352795
4 2.865582291 192.168.50.185 192.168.50.98 TCP 72 33826 → 5000 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=6 TSval=380957706 TSecr=3791352795
5 2.865634966 192.168.50.98 192.168.50.185 TCP 66 5000 → 33826 [ACK] Seq=1 Ack=7 Win=65280 Len=0 TSval=3791355661 TSecr=380957706
6 8.282751286 192.168.50.185 192.168.50.98 TCP 72 33826 → 5000 [PSH, ACK] Seq=7 Ack=1 Win=64256 Len=6 TSval=380963194 TSecr=3791355661
7 8.282796912 192.168.50.98 192.168.50.185 TCP 66 5000 → 33826 [ACK] Seq=1 Ack=13 Win=65280 Len=0 TSval=3791361078 TSecr=380963194
8 27.452956734 192.168.50.185 192.168.50.98 TCP 66 33826 → 5000 [FIN, ACK] Seq=13 Ack=1 Win=64256 Len=0 TSval=380982172 TSecr=3791361078
9 27.453146978 192.168.50.98 192.168.50.185 TCP 66 5000 → 33826 [FIN, ACK] Seq=1 Ack=14 Win=65280 Len=0 TSval=3791380248 TSecr=380982172
10 27.482175101 192.168.50.185 192.168.50.98 TCP 66 33826 → 5000 [ACK] Seq=14 Ack=2 Win=64256 Len=0 TSval=380982384 TSecr=3791380248
11 27.482204720 192.168.50.98 192.168.50.185 TCP 54 5000 → 33826 [RST] Seq=2 Win=0 Len=0
Frame 4: 72 bytes on wire (576 bits), 72 bytes captured (576 bits)
Ethernet II, Src: Intel_14:ca:71 (68:54:5a:14:ca:71), Dst: Intel_b9:11:94 (c4:bd:e5:b9:11:94)
Internet Protocol Version 4, Src: 192.168.50.185, Dst: 192.168.50.98
Transmission Control Protocol, Src Port: 33826, Dst Port: 5000, Seq: 1, Ack: 1, Len: 6
Source Port: 33826
Destination Port: 5000
[Stream index: 0]
[Stream Packet Number: 4]
[Conversation completeness: Complete, WITH_DATA (63)]
[TCP Segment Len: 6]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 2973846406
[Next Sequence Number: 7 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 573613377
1000 .... = Header Length: 32 bytes (8)
Flags: 0x018 (PSH, ACK)
000. .... = Reserved: Not set
...0 .... = Accurate ECN: Not set
.... 0... .... = Congestion Window Reduced: Not set
.... .0.. .... = ECN-Echo: Not set
.... ..0. .... = Urgent: Not set
....1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
....0. = Syn: Not set
....0 = Fin: Not set
[TCP Flags: ·····AP···]
Window: 502
[Calculated window size: 64256]
[Window size scaling factor: 128]
Checksum: 0x8df6 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
[Timestamps]
[SEQ/ACK analysis]
TCP payload (6 bytes)
Data (6 bytes)
0000 c4 bd e5 b9 11 94 68 54 5a 14 ca 71 08 00 45 00
0010 00 3a a5 97 40 00 40 06 ae ba c0 a8 32 b9 c0 a8
0020 32 62 84 22 13 88 b1 41 4b 86 22 30 a5 41 80 18
0030 01 f6 8d f6 00 00 01 01 08 0a 16 b4 f4 0a e1 fb
0040 73 db 68 65 6c 6c 6f 0a
s hello·


Frame 9: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
Ethernet II, Src: Intel_b9:11:94 (c4:bd:e5:b9:11:94), Dst: Intel_14:ca:71 (68:54:5a:14:ca:71)
Internet Protocol Version 4, Src: 192.168.50.98, Dst: 192.168.50.185
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 52
Identification: 0xaf3d (44861)
010. .... = Flags: 0x2, Don't fragment
...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 64
Protocol: TCP (6)
Header Checksum: 0xa51a [validation disabled]
[Header checksum status: Unverified]
Source Address: 192.168.50.98
Destination Address: 192.168.50.185
[Stream index: 0]
Transmission Control Protocol, Src Port: 5000, Dst Port: 33826, Seq: 1, Ack: 14, Len: 0
Source Port: 5000
Destination Port: 33826
[Stream index: 0]
[Stream Packet Number: 9]
[Conversation completeness: Complete, WITH_DATA (63)]
[TCP Segment Len: 0]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 573613377
[Next Sequence Number: 2 (relative sequence number)]
Acknowledgment Number: 14 (relative ack number)
Acknowledgment number (raw): 2973846419
1000 .... = Header Length: 32 bytes (8)
Flags: 0x011 (FIN, ACK)
Window: 510
[Calculated window size: 65280]
[Window size scaling factor: 128]
Checksum: 0xe692 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
[Timestamps]
[SEQ/ACK analysis]

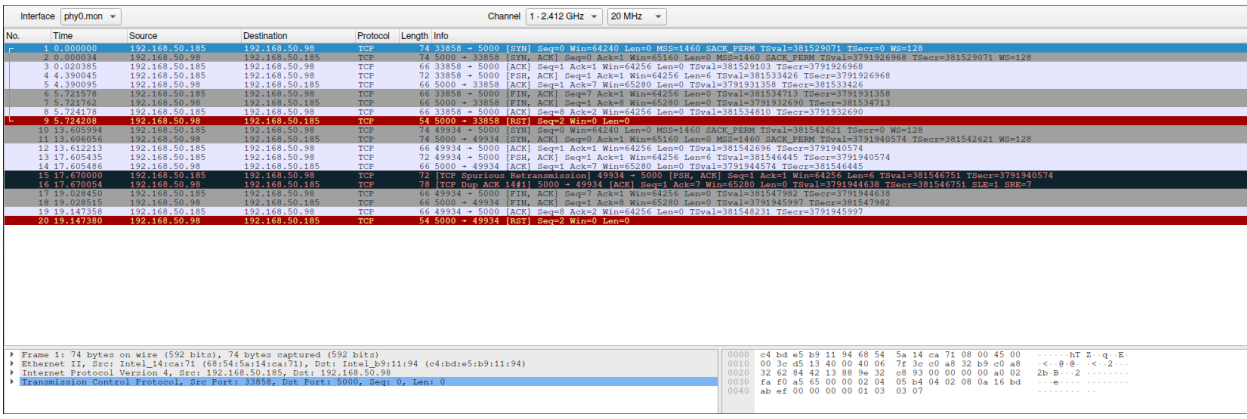
IP

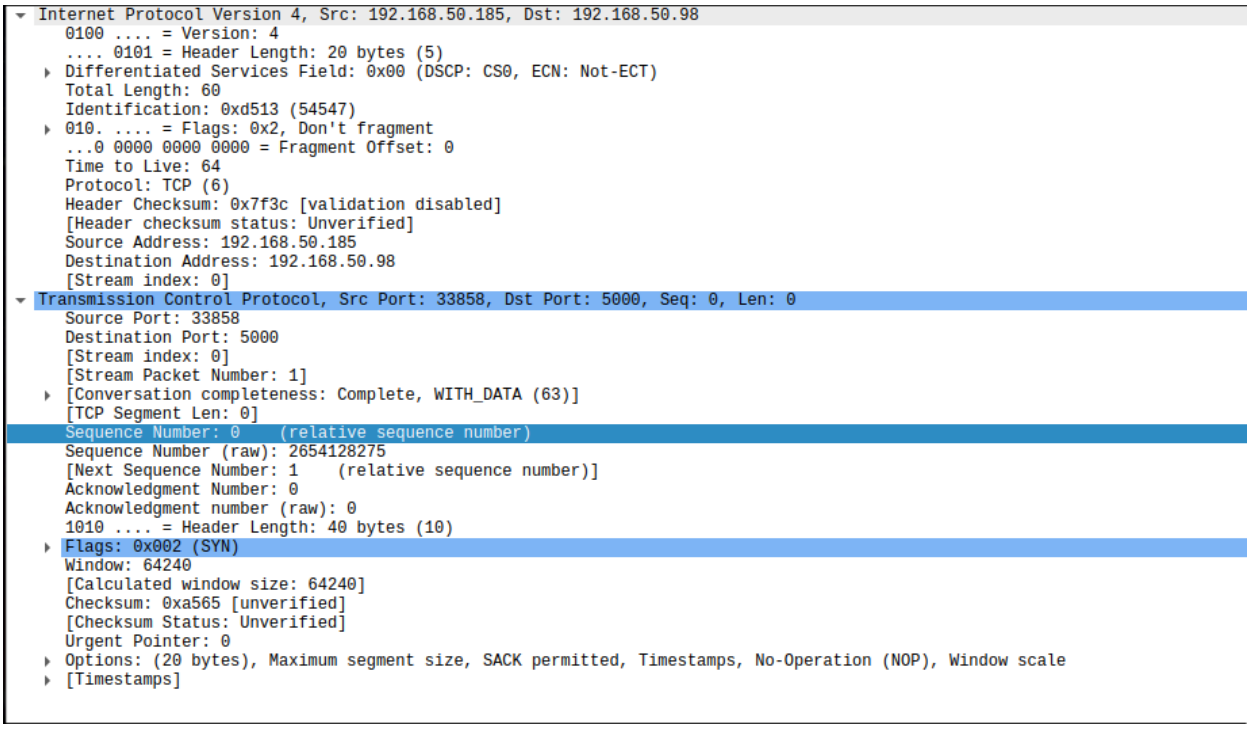
Version - Specifies the IP version (4 for IPv4).
IHL - Internet Header Length, specifies the header length in 32-bit words.
Type of Service - Defines the quality of service for the packet.
Total Length - Total length of the IP packet (header + data).
Identification - Identifies fragments of a packet if fragmentation occurs.
Flags - Controls fragmentation.
Fragment Offset - Specifies the position of a fragment relative to the start of the packet.
Time to Live (TTL) - Limits the packet's lifespan (prevents endless looping).
Protocol 8 Specifies the protocol encapsulated in the packet's payload (e.g., TCP, UDP).
Header Checksum - Provides error-checking for the header to ensure its integrity.
Source IP Address - The IP address of the sender.
Destination IP Address - The IP address of the recipient.
Options Variable Optional settings for special routing purposes (optional field).

TCP

Source port - The port number of the sending application
Destination port - The port number of the receiving application
Sequence Number - The position of the first byte of data in the current segment
Acknowledgement Number - The next byte the sender expects to receive from the receiver
Data offset - Specifies the size of the TCP header in 32-bit words
Reserved - Reserved for future use, must be set to 0
Flags - Control flags that manage the connection
Window size - A flow control mechanism that specifies the size of the receive buffer
Checksum - Error-checking for the header and payload
Urgent pointer - Points to the last urgent byte in the data (if the URG flag is set)
Options - Optional fields for special settings, such as Maximum Segment Size (MSS)

b. Across Sessions





Key Differences and Explanations

Source/Destination Ports:

- In the IP header it does not include the port info and only deals with routing the packet to the correct destination IP. The TCP header includes

the source and destination port which is used to send the packet to the correct app.

Sequence/Acknowledgment Numbers:

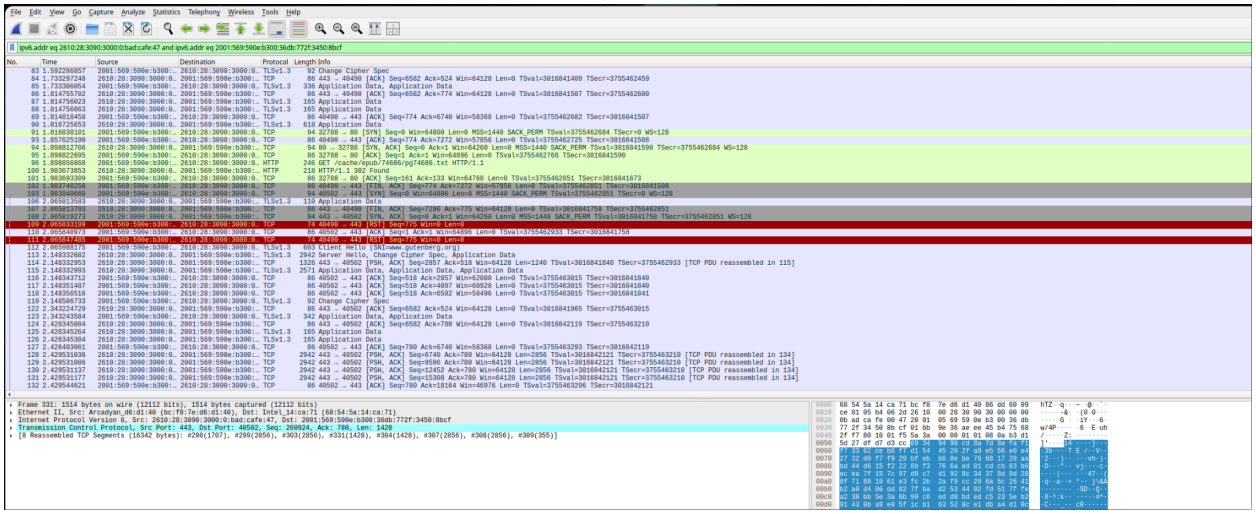
- In IP the header does not include the sequence or acknowledgement numbers and it's only for the delivery of packets. The TCP header has the sequence and acknowledgement number which helps the data be received in order and without errors. The sequence numbers help when reconstructing the data stream and the acknowledgement numbers confirm the sent data.

Flags:

- The IP header includes flags for the fragmentation. They are Don't Fragment, and More Fragments. The TCP has flags for fragmentation; they are SYN, ACK, FIN, RST, PSH and URG. They are used for connection establishment/termination and data transfer.

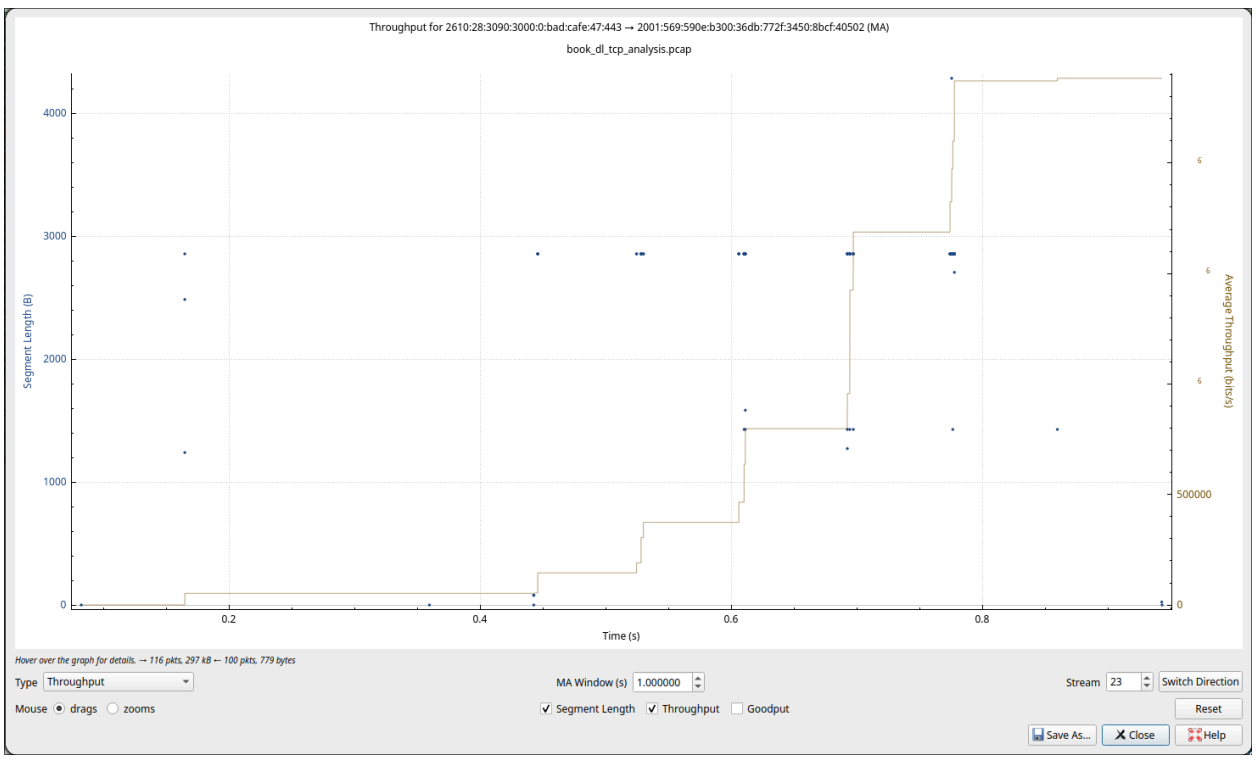
4. Website Download and TCP Analysis

```
~ via 🐍 pyenv
→ wget https://www.gutenberg.org/ebooks/74686.txt.utf-8
--2024-11-05 17:40:21--  https://www.gutenberg.org/ebooks/74686.txt.utf-8
Loaded CA certificate '/etc/ssl/certs/ca-certificates.crt'
Resolving www.gutenberg.org (www.gutenberg.org)... 2610:28:3090:3000:0:bad:cafe:47, 152.19.134.47
Connecting to www.gutenberg.org (www.gutenberg.org)|2610:28:3090:3000:0:bad:cafe:47|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: http://www.gutenberg.org/cache/epub/74686/pg74686.txt [following]
--2024-11-05 17:40:21--  http://www.gutenberg.org/cache/epub/74686/pg74686.txt
Connecting to www.gutenberg.org (www.gutenberg.org)|2610:28:3090:3000:0:bad:cafe:47|:80... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://www.gutenberg.org/cache/epub/74686/pg74686.txt [following]
--2024-11-05 17:40:21--  https://www.gutenberg.org/cache/epub/74686/pg74686.txt
Connecting to www.gutenberg.org (www.gutenberg.org)|2610:28:3090:3000:0:bad:cafe:47|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 290517 (284K) [text/plain]
Saving to: '74686.txt.utf-8'

74686.txt.utf-8             100%[===========================================>] 283.71K  1.16MB/s    in 0.2s

2024-11-05 17:40:22 (1.16 MB/s) - '74686.txt.utf-8' saved [290517/290517]
```

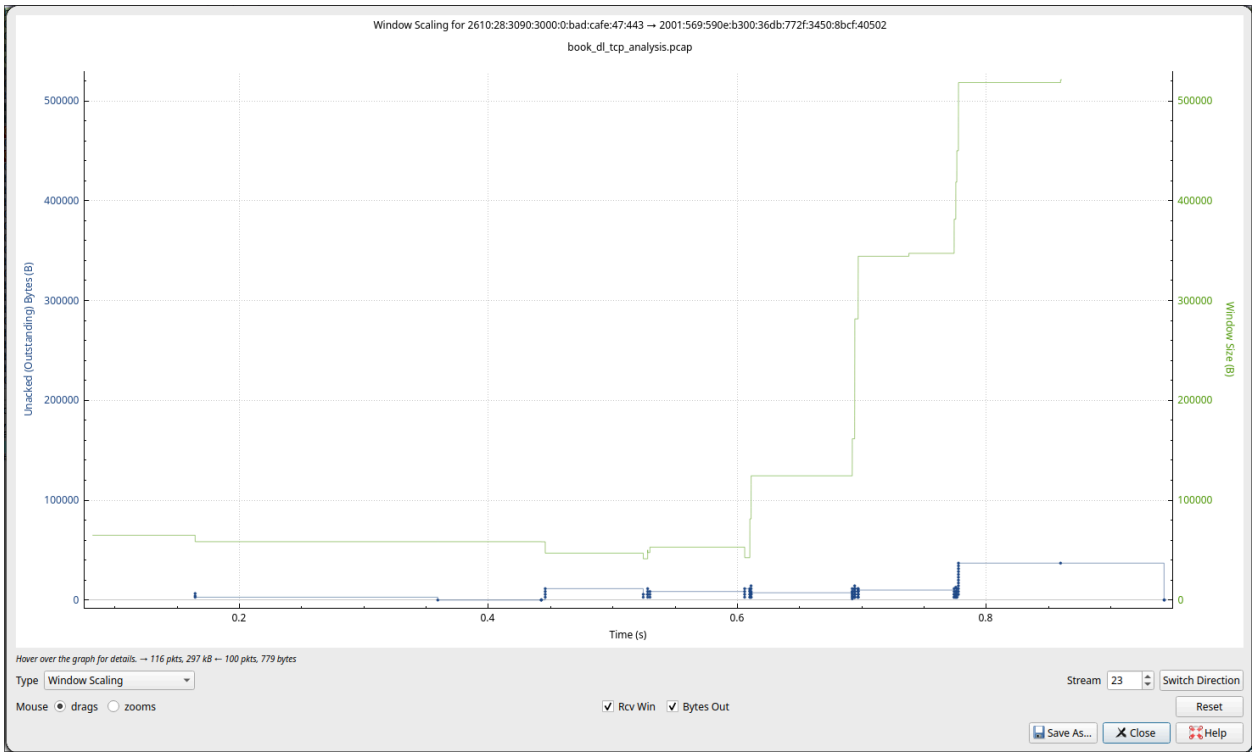


“ipv6.addr eq 2610:28:3090:3000:0:bad:cafe:47 and ipv6.addr eq 2001:569:590e:b300:36db:772f:3450:8bcf” is the filter i used to generate the graphs.

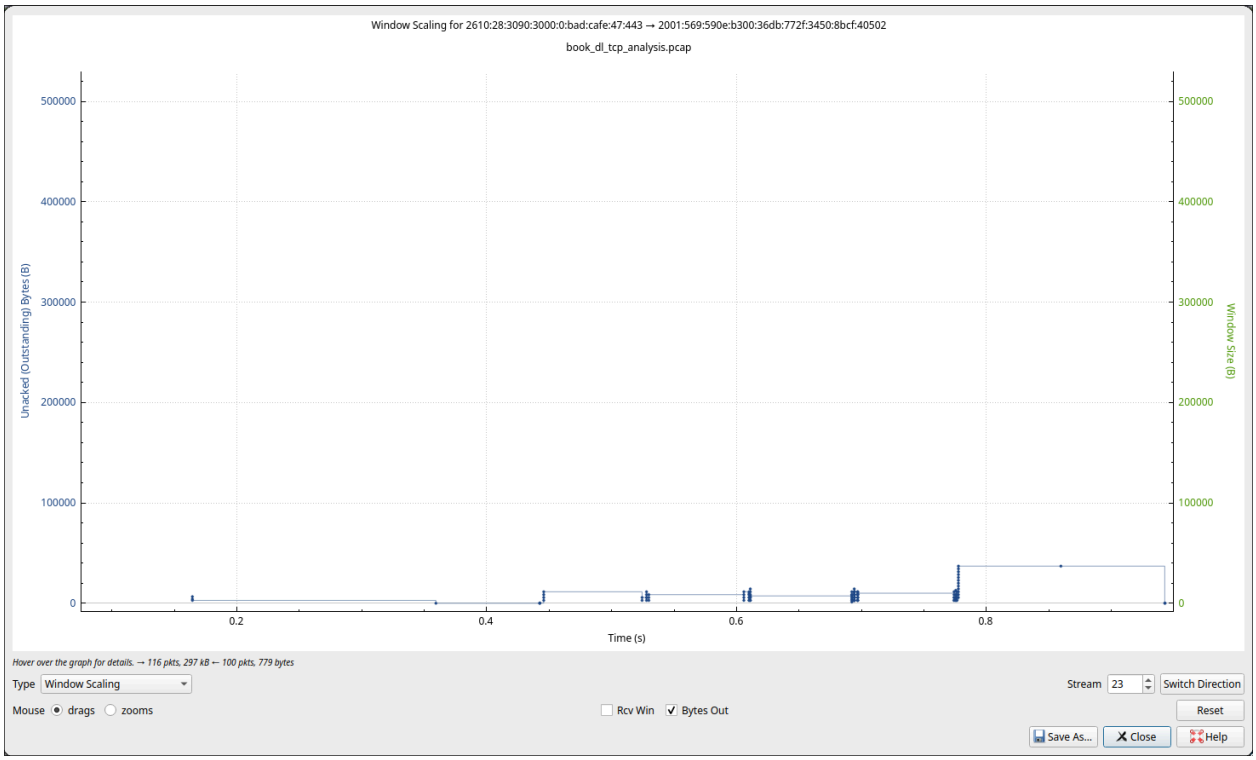
Throughput



Window Size/Scale

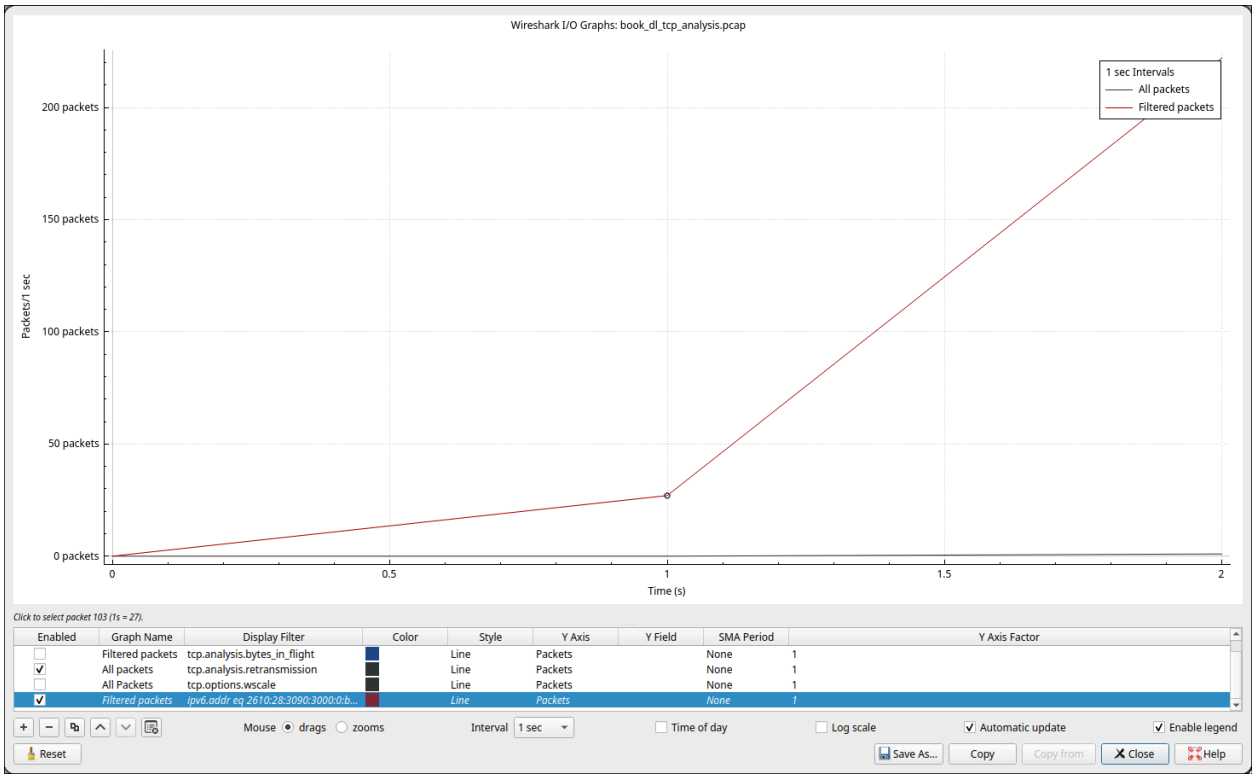


Window Congestion



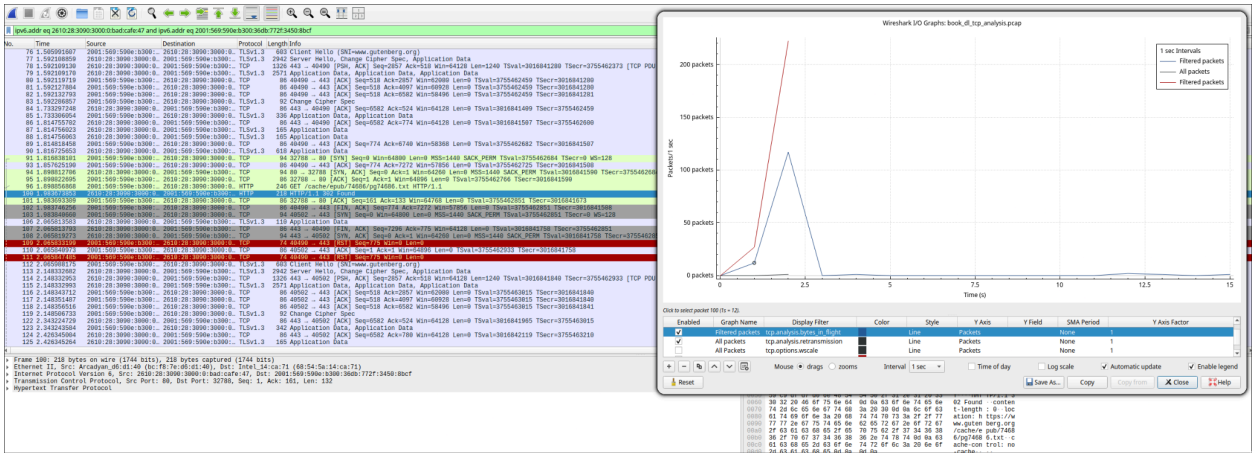
Retransmission

(The line in gray. There are none so the line stays at 0)



Bytes in fight in blue

(The line in blue. if you click on the circle on the graph it highlights the get request to download



Conclusion

This assignment provided an in-depth exploration of the TCP protocol, emphasizing session management, data identification, and performance optimization mechanisms. Through practical exercises using tools like hping3, ncat, and Wireshark, we were able to dissect and analyze various aspects of TCP operations.