ELEC 331 Assignment 2

Wireshark UDP

```
No.     Time           Source                Destination           Protocol Length Info
      1 0.000000       192.168.1.70          192.168.1.255         UDP      305    54915 → 54915 Len=263
Frame 1: 305 bytes on wire (2440 bits), 305 bytes captured (2440 bits) on interface 0
Ethernet II, Src: IntelCor_85:bd:d5 (e0:94:67:85:bd:d5), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src: 192.168.1.70, Dst: 192.168.1.255
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 291
    Identification: 0xb2a6 (45734)
    Flags: 0x0000
    Time to live: 128
    Protocol: UDP (17)
    Header checksum: 0x028e [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.1.70
    Destination: 192.168.1.255
User Datagram Protocol, Src Port: 54915, Dst Port: 54915
    Source Port: 54915
    Destination Port: 54915
    Length: 271
    Checksum: 0x2012 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 0]
Data (263 bytes)
0000  00 44 45 53 4b 54 4f 50 2d 43 52 31 4c 39 50 49   .DESKTOP-CR1L9PI
0010  00 b7 cf 2e ce 00 00 00 00 00 00 00 00 00 00 00   ................
0020  33 27 00 00 00 00 00 00 50 a5 49 80 cb 02 00 00   3'......P.I.....
0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0040  00 00 00 00 00 00 00 00 7c 6a 06 76 00 00 00 00   ........|j.v....
0050  08 a5 b4 76 00 00 00 00 c9 bb cf 2e ce 00 00 00   ...v............
0060  00 00 00 00 00 00 00 00 c0 06 f3 ff cb 02 00 00   ................
0070  14 b8 cf 2e ce 00 00 00 30 b8 cf 2e ce 00 00 00   ........0.......
0080  18 56 00 7b 37 38 66 32 65 62 37 63 2d 30 38 39   .V.{78f2eb7c-089
0090  32 2d 34 35 36 65 2d 39 61 34 64 2d 30 32 39 34   2-456e-9a4d-0294
00a0  62 39 66 38 38 34 63 66 7d 00 46 fd cb 02 00 00   b9f884cf}.F.....
00b0  01 00 00 00 ce 00 00 00 10 b8 cf 2e ce 00 00 00   ................
00c0  50 a0 f2 80 cb 02 00 00 00 00 00 00 00 00 00 00   P...............
00d0  00 00 00 00 00 00 00 00 00 12 46 fd cb 02 00 00   ..........F.....
00e0  c0 0c 46 fd cb 02 00 00 39 00 00 00 00 00 00 00   ..F.....9.......
00f0  50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   P...............
0100  00 00 00 12 bf 77 78                             .....wx
    Data: 004445534b544f502d4352314c39504900b7cf2ece000000...
    [Length: 263]
```

Figure #1: Overall details of the UDP packet

ELEC 331 Assignment 2

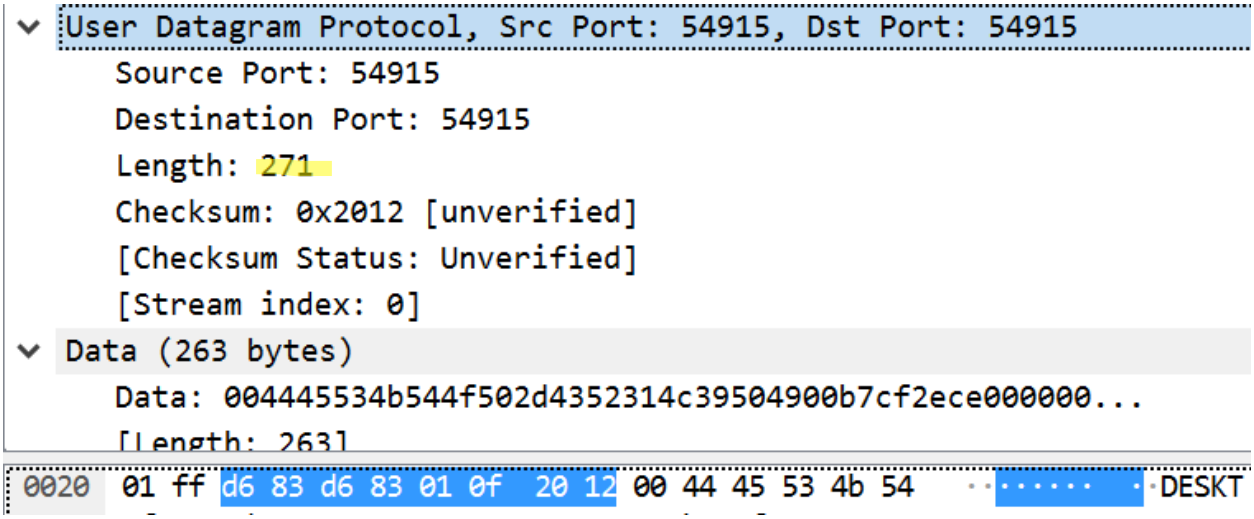
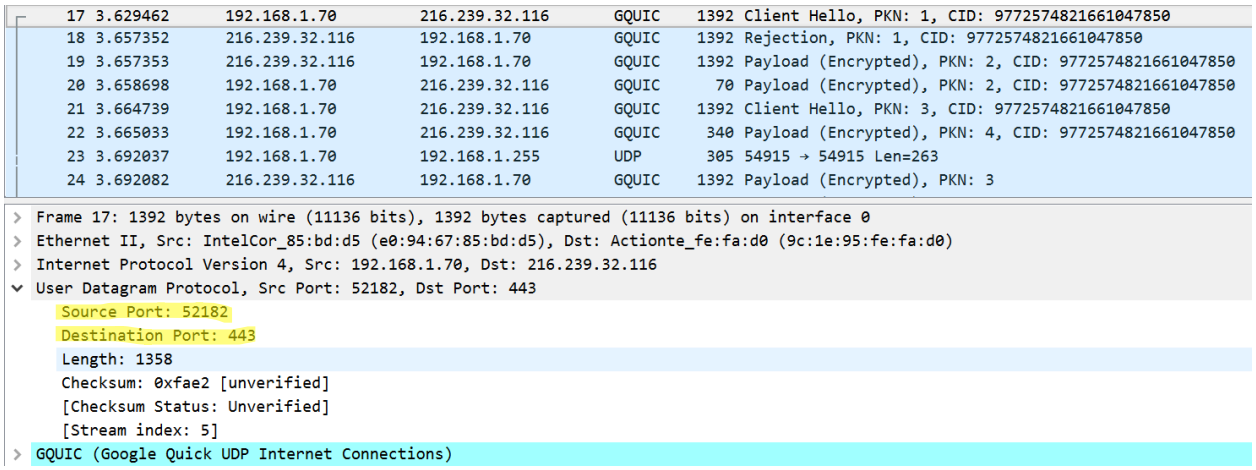


Figure #2: Headers of the UDP packet

1. There are 4 fields in the UDP header, specifically:
   a. Source Port
   b. Destination Port
   c. Length
   d. Checksum
2. For each header, their size is:
   a. Source Port: 2 bytes (Bytes 34 – 35)
   b. Destination Port: 2 bytes (Bytes 36 – 37)
   c. Length: 2 bytes (Bytes 38 – 39)
   d. Checksum: 2 bytes (Bytes 40 – 41)
3. The value of the length field is the length of the entire UDP segment, in this case being 271 bytes. To verify this, we must look at the payload, which Wireshark specifies to have 263 bytes of data. If we add the 8 bytes to it from UDP headers, we have exactly 271 bytes, the same as the value of the length field.
4. As the length field restricts the maximum size of the UDP segment, we know that the maximum is $2^{16} - 1 = 65535$ bytes. Considering headers, the maximum number of bytes for a UDP payload is 65527 bytes.
5. As mentioned above, the largest possible source port number is 65535.
6. The protocol number for UDP is 0x11 in hexadecimal and 17 in decimal.
7. As the second packet is being sent as a response to the first packet, the relationship is described in the table below:

Packet Number	Source Ports	Destination Ports
17	52182	443
18	443	52182

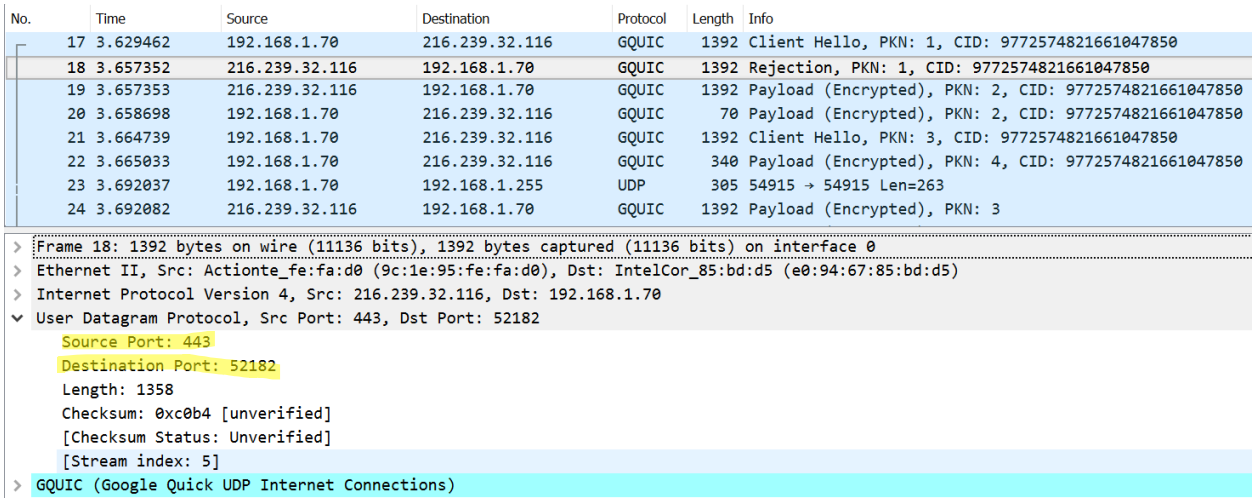
ELEC 331 Assignment 2



```
    17 3.629462    192.168.1.70      216.239.32.116    GQUIC    1392 Client Hello, PKN: 1, CID: 9772574821661047850
    18 3.657352    216.239.32.116    192.168.1.70      GQUIC    1392 Rejection, PKN: 1, CID: 9772574821661047850
    19 3.657353    216.239.32.116    192.168.1.70      GQUIC    1392 Payload (Encrypted), PKN: 2, CID: 9772574821661047850
    20 3.658698    192.168.1.70      216.239.32.116    GQUIC      70 Payload (Encrypted), PKN: 2, CID: 9772574821661047850
    21 3.664739    192.168.1.70      216.239.32.116    GQUIC    1392 Client Hello, PKN: 3, CID: 9772574821661047850
    22 3.665033    192.168.1.70      216.239.32.116    GQUIC     340 Payload (Encrypted), PKN: 4, CID: 9772574821661047850
    23 3.692037    192.168.1.70      192.168.1.255     UDP       305 54915 → 54915 Len=263
    24 3.692082    216.239.32.116    192.168.1.70      GQUIC    1392 Payload (Encrypted), PKN: 3

> Frame 17: 1392 bytes on wire (11136 bits), 1392 bytes captured (11136 bits) on interface 0
> Ethernet II, Src: IntelCor_85:bd:d5 (e0:94:67:85:bd:d5), Dst: Actionte_fe:fa:d0 (9c:1e:95:fe:fa:d0)
> Internet Protocol Version 4, Src: 192.168.1.70, Dst: 216.239.32.116
v User Datagram Protocol, Src Port: 52182, Dst Port: 443
    Source Port: 52182
    Destination Port: 443
    Length: 1358
    Checksum: 0xfae2 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 5]
> GQUIC (Google Quick UDP Internet Connections)
```

Figure #3: First packet of UDP pair



```
No.  Time       Source            Destination       Protocol Length Info
    17 3.629462    192.168.1.70      216.239.32.116    GQUIC    1392 Client Hello, PKN: 1, CID: 9772574821661047850
    18 3.657352    216.239.32.116    192.168.1.70      GQUIC    1392 Rejection, PKN: 1, CID: 9772574821661047850
    19 3.657353    216.239.32.116    192.168.1.70      GQUIC    1392 Payload (Encrypted), PKN: 2, CID: 9772574821661047850
    20 3.658698    192.168.1.70      216.239.32.116    GQUIC      70 Payload (Encrypted), PKN: 2, CID: 9772574821661047850
    21 3.664739    192.168.1.70      216.239.32.116    GQUIC    1392 Client Hello, PKN: 3, CID: 9772574821661047850
    22 3.665033    192.168.1.70      216.239.32.116    GQUIC     340 Payload (Encrypted), PKN: 4, CID: 9772574821661047850
    23 3.692037    192.168.1.70      192.168.1.255     UDP       305 54915 → 54915 Len=263
    24 3.692082    216.239.32.116    192.168.1.70      GQUIC    1392 Payload (Encrypted), PKN: 3

> Frame 18: 1392 bytes on wire (11136 bits), 1392 bytes captured (11136 bits) on interface 0
> Ethernet II, Src: Actionte_fe:fa:d0 (9c:1e:95:fe:fa:d0), Dst: IntelCor_85:bd:d5 (e0:94:67:85:bd:d5)
> Internet Protocol Version 4, Src: 216.239.32.116, Dst: 192.168.1.70
v User Datagram Protocol, Src Port: 443, Dst Port: 52182
    Source Port: 443
    Destination Port: 52182
    Length: 1358
    Checksum: 0xc0b4 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 5]
> GQUIC (Google Quick UDP Internet Connections)
```

Figure #4: Second packet of UDP pair

ELEC 331 Assignment 2

Wireshark TCP

1. The IP address and TCP port number for the client computer is: 192.168.1.102 and 1161.
2. The IP address and TCP port number for gaia.cs.umass.edu is: 128.119.245.12 and 80.

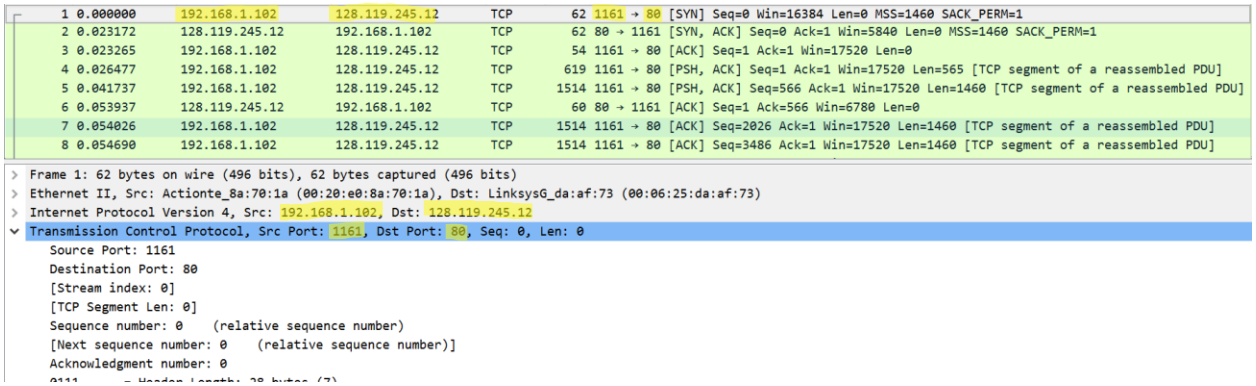


Figure #5: Ethereal SYN Segment

3. The IP address and TCP port number for my client computer is: 192.168.1.70 and 50221.

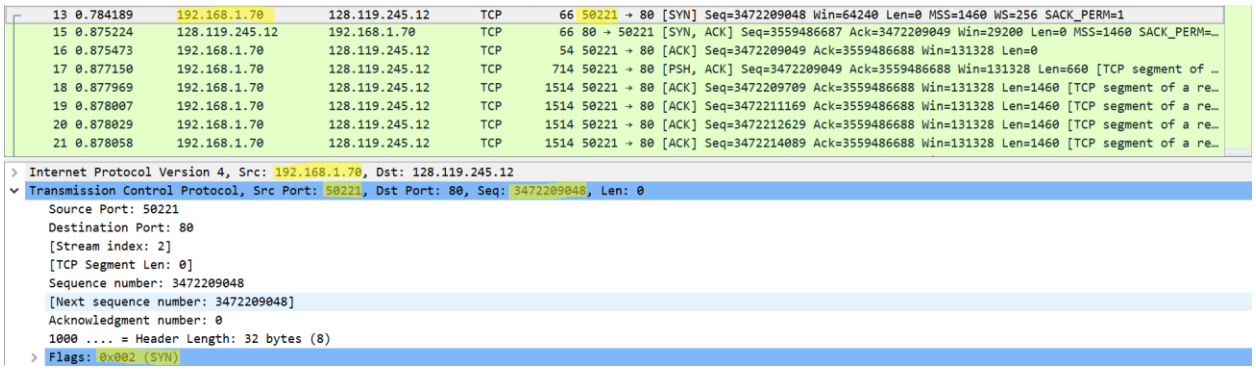


Figure #6: My own SYN Segment

4. Using Figure #6, we see that the sequence number is $X = 3472209048$ (relative 0). In the segment the SYN flag was set to true, indicating the start of the TCP connection.
5. The sequence number of the SYNACK segment is $Y = 3559486687$ (relative 0). The value of the Acknowledgement field in the SYNACK segment is $X + 1 = 3472209049$, which gaia determined by adding 1 to the sequence number of the SYN segment it previously received. In the segment the SYN and ACK flags were set to true, indicating it as a SYNACK segment.

ELEC 331 Assignment 2

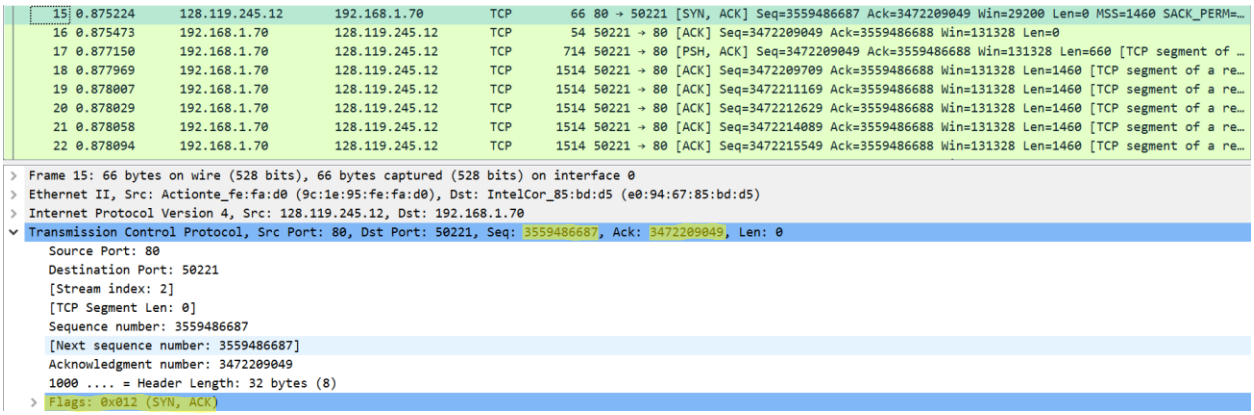


Figure #7: My own SYNACK segment

6. The sequence number of the segment that contains the POST is $X = 3472209049$

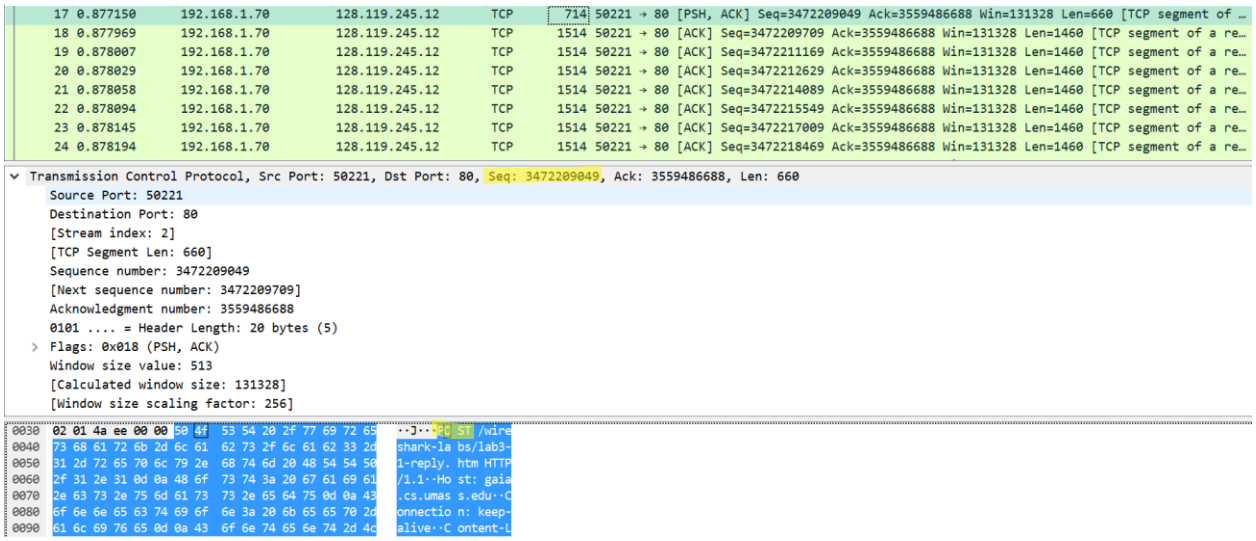


Figure #8: POST segment

7. Assuming that the POST segment to be segment 1 (relative numbering):

Segment	Sequence	Time Sent (s)	ACK Received (s)	RTT (s)	EstimatedRTT (s)
1	1	0.877150	0.967594	0.090444	0.090444
2	661	0.877969	0.967594	0.089625	0.090342
3	2121	0.878007	0.967597	0.08959	0.090248
4	3581	0.878029	0.967598	0.089569	0.090163
5	5041	0.878058	0.967598	0.08954	0.090085
6	6501	0.878094	0.967599	0.089505	0.090012

Table #1: RTT and Estimated RTTs for the 6 sent TCP segments

ELEC 331 Assignment 2

	Time	Source	Destination	Protocol	Length	Info
17	0.877150	192.168.1.70	128.119.245.12	TCP	714	50221 → 80 [PSH, ACK] Seq=1 Ack=1 Win=131328 Len=660 [TCP segment of a reassembled PDU]
18	0.877969	192.168.1.70	128.119.245.12	TCP	1514	50221 → 80 [ACK] Seq=661 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
19	0.878007	192.168.1.70	128.119.245.12	TCP	1514	50221 → 80 [ACK] Seq=2121 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
20	0.878029	192.168.1.70	128.119.245.12	TCP	1514	50221 → 80 [ACK] Seq=3581 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
21	0.878058	192.168.1.70	128.119.245.12	TCP	1514	50221 → 80 [ACK] Seq=5041 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
22	0.878094	192.168.1.70	128.119.245.12	TCP	1514	50221 → 80 [ACK] Seq=6501 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]

Figure #9: TCP Send

27	0.967594	128.119.245.12	192.168.1.70	TCP	60	80 → 50221 [ACK] Seq=1 Ack=661 Win=30592 Len=0
28	0.967597	128.119.245.12	192.168.1.70	TCP	60	80 → 50221 [ACK] Seq=1 Ack=2121 Win=33536 Len=0
29	0.967598	128.119.245.12	192.168.1.70	TCP	60	80 → 50221 [ACK] Seq=1 Ack=5041 Win=39296 Len=0
30	0.967599	128.119.245.12	192.168.1.70	TCP	60	80 → 50221 [ACK] Seq=1 Ack=7961 Win=45184 Len=0

Figure #10: TCP ACK Responses

8. The POST segment had a length of 660 bytes while the other 5 have a length of 1460 bytes.
9. Referring to Figure #9, we see that the advertised window size is 131328 for all 6 segments. If we look at the Time column, it seems that a lack of buffer space didn't throttle the sender.
10. It does not look like there were any retransmits in my trace file. This was achieved by looking at the sequence and acknowledgement numbers for both the sender and receiver. I was unable to find any duplicates or loss and thus no retransmits were made.
11. Typically, the receiver ACKs 2920 bytes of data, which is "every other" segment as each segment is typically 1460 bytes. If we refer to Figure #10, we see that packet number 28 only ACKed 1460 bytes.

52	1.057982	128.119.245.12	192.168.1.70	TCP	60	80 → 50221 [ACK] Seq=1 Ack=15261 Win=59776 Len=0
53	1.057985	128.119.245.12	192.168.1.70	TCP	60	80 → 50221 [ACK] Seq=1 Ack=18181 Win=65664 Len=0
54	1.057987	128.119.245.12	192.168.1.70	TCP	60	80 → 50221 [ACK] Seq=1 Ack=21101 Win=71424 Len=0
55	1.057988	128.119.245.12	192.168.1.70	TCP	60	80 → 50221 [ACK] Seq=1 Ack=24021 Win=77312 Len=0
56	1.057989	128.119.245.12	192.168.1.70	TCP	60	80 → 50221 [ACK] Seq=1 Ack=26941 Win=83200 Len=0
57	1.057990	128.119.245.12	192.168.1.70	TCP	60	80 → 50221 [ACK] Seq=1 Ack=29861 Win=88960 Len=0

Figure #11: Sample ACKs

12. Throughput was calculated using how many bytes were transmit over a period. In our case:

16	0.875473	192.168.1.70	128.119.245.12	TCP	54	50221 → 80 [ACK] Seq=1 Ack=1 Win=131328 Len=0

Figure #12: Start of Transmit

202	6.330326	192.168.1.70	128.119.245.12	TCP	54	50221 → 80 [ACK] Seq=152982 Ack=779 Win=130560 Len=0

Figure #13: End of Transmit

$$Throughout = \frac{Number\ of\ Bytes}{Time}$$

$$Throughput = \frac{152982}{6.330326 - 0.875473}$$

$$Throughput = 28045.1187227 \frac{\text{bytes}}{\text{sec}} = 28.045 \frac{kbytes}{sec}$$

The File Size (152982) came from the sequence number with relative 0. Could use 150,000.

ELEC 331 Assignment 2

13. For Figure #14, it looks like TCP slowstart phase occurs between 0 and 0.305 seconds, with congestion avoidance occurring from 0.305 seconds to the end. The graph differs from the idealized behaviour of TCP as exactly 6 packets are sent each time instead of sending one packet at a time, linearly increasing the window size.

    In this trace, the advertised receive window has a size of 17520, which can hold 12 packets, each of length 1460 bytes. As the rate is not restricted by flow control (receive window), it must be restricted by congestion control (congestion window) from the network side.

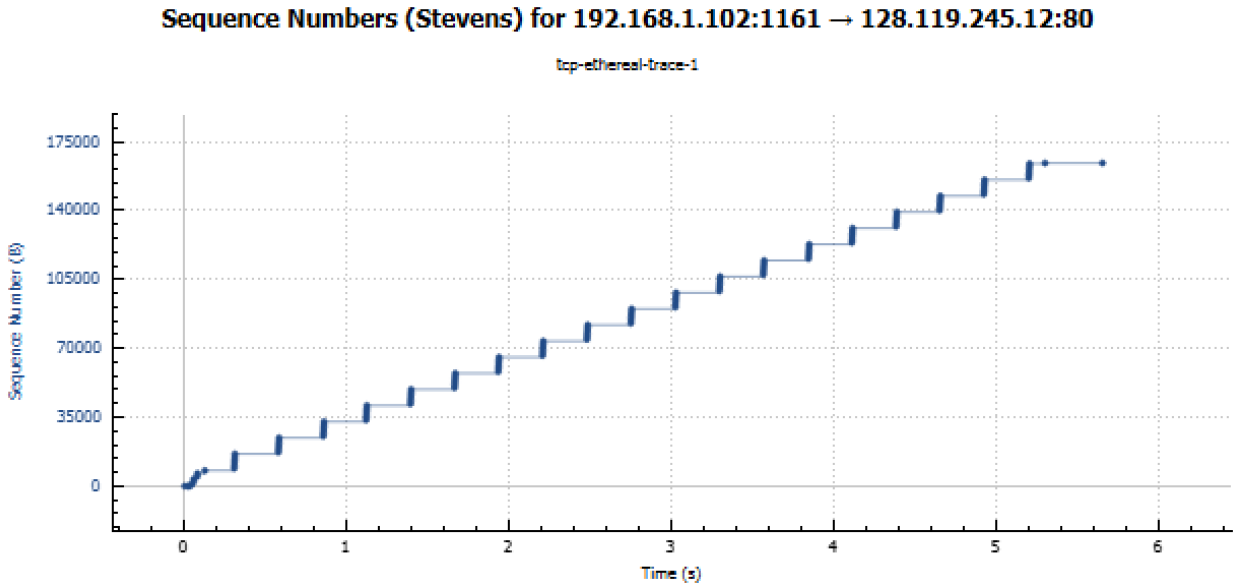


Figure #14: Sequence Number Graph for tcp-ethereal-trace-1

ELEC 331 Assignment 2

14. For Figure #15, to determine the times for slowstart and congestion avoidance, a little zooming in is required. Inspecting the sequence numbers and packet numbers, I see that the last two chunks of packets contain exactly 40 packets each. Therefore, it seems that slowstart occurs between 0 and 0.36 seconds, with congestion occurring from 0.36 seconds to the end. As before, there is no linear growth as we are sending multiple packets before acknowledgements come in.

    In this trace, the advertised receive window has a size of 131328, which can hold 90 packets, each of length 1460 bytes. The largest chunks of packets only contained a maximum of 40 packets. As the rate is not restricted by flow control (receive window), it must be restricted by congestion control (congestion window) from the network side.

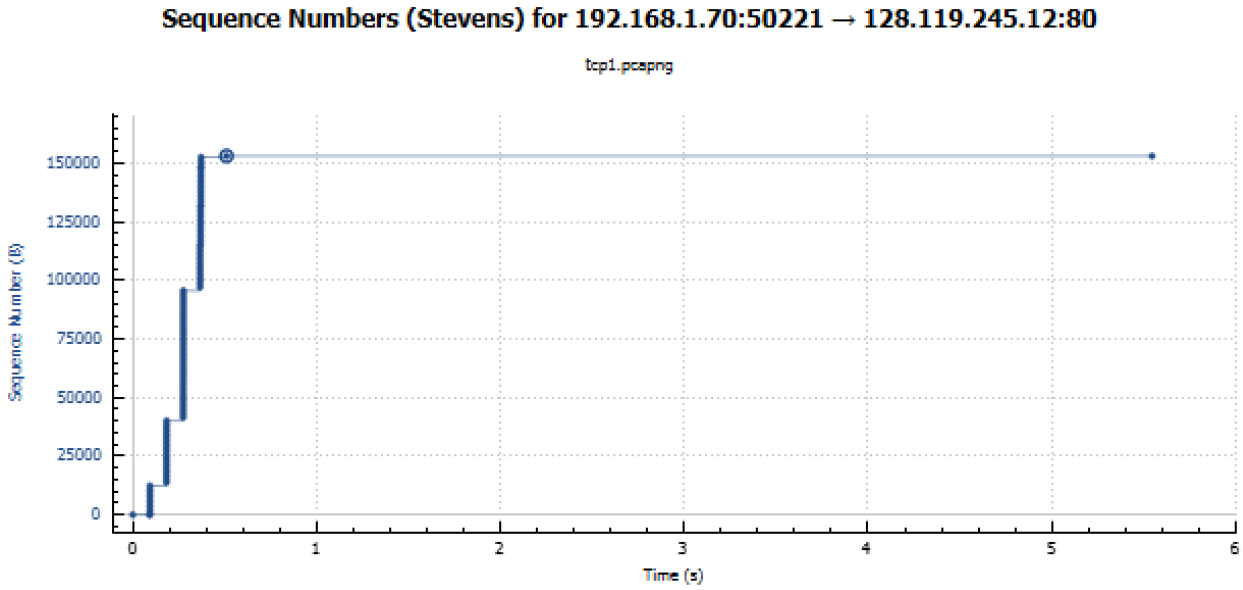


Figure #15: Sequence Number Graph for my own TCP trace