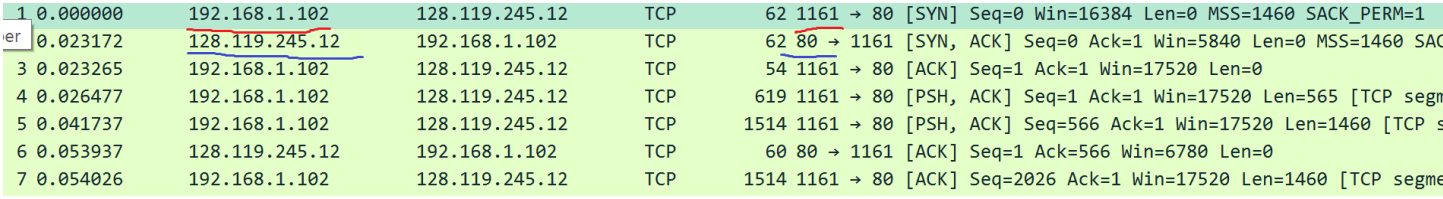
Kevin Pirabaharan
0946212

CIS 3210 Assignment 3 Report

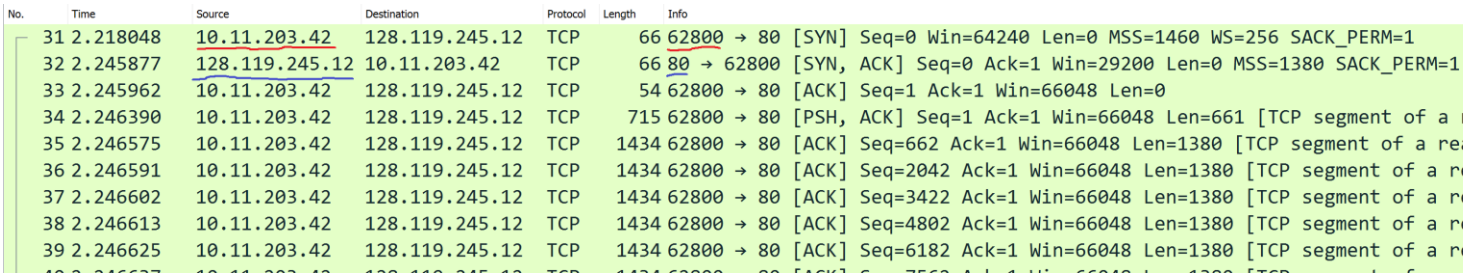
Part 1: Wireshark Lab: TCP v7.0



1	0.000000	192.168.1.102	128.119.245.12	TCP	62 1161 → 80 [SYN] Seq=0 Win=16384 Len=0 MSS=1460 SACK_PERM=1
2	0.023172	128.119.245.12	192.168.1.102	TCP	62 80 → 1161 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 SAC
3	0.023265	192.168.1.102	128.119.245.12	TCP	54 1161 → 80 [ACK] Seq=1 Ack=1 Win=17520 Len=0
4	0.026477	192.168.1.102	128.119.245.12	TCP	619 1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len=565 [TCP segm
5	0.041737	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [PSH, ACK] Seq=566 Ack=1 Win=17520 Len=1460 [TCP s
6	0.053937	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=1 Ack=566 Win=6780 Len=0
7	0.054026	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=2026 Ack=1 Win=17520 Len=1460 [TCP segme

Figure 1: Source (red) and destination (blue) IP Address and Port #

1. **What is the IP address and TCP port number used by the client computer (source) that is transferring the file to gaia.cs.umass.edu?**
   Source IP Address: 192.162.1.102
   Source Port Number: 1161
2. **What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?**
   Destination IP Address: 128.119.245.12
   Destination Port Number: 80



No.	Time	Source	Destination	Protocol	Length	Info
31	2.218048	10.11.203.42	128.119.245.12	TCP	66	62800 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
32	2.245877	128.119.245.12	10.11.203.42	TCP	66	80 → 62800 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1380 SACK_PERM=1
33	2.245962	10.11.203.42	128.119.245.12	TCP	54	62800 → 80 [ACK] Seq=1 Ack=1 Win=66048 Len=0
34	2.246390	10.11.203.42	128.119.245.12	TCP	715	62800 → 80 [PSH, ACK] Seq=1 Ack=1 Win=66048 Len=661 [TCP segment of a
35	2.246575	10.11.203.42	128.119.245.12	TCP	1434	62800 → 80 [ACK] Seq=662 Ack=1 Win=66048 Len=1380 [TCP segment of a re
36	2.246591	10.11.203.42	128.119.245.12	TCP	1434	62800 → 80 [ACK] Seq=2042 Ack=1 Win=66048 Len=1380 [TCP segment of a r
37	2.246602	10.11.203.42	128.119.245.12	TCP	1434	62800 → 80 [ACK] Seq=3422 Ack=1 Win=66048 Len=1380 [TCP segment of a r
38	2.246613	10.11.203.42	128.119.245.12	TCP	1434	62800 → 80 [ACK] Seq=4802 Ack=1 Win=66048 Len=1380 [TCP segment of a r
39	2.246625	10.11.203.42	128.119.245.12	TCP	1434	62800 → 80 [ACK] Seq=6182 Ack=1 Win=66048 Len=1380 [TCP segment of a r

Figure 2 Source (red) and Destination (blue) IP Address and Port #

3. **What is the IP address and TCP port number used by your client computer (source) to transfer the file to gaia.cs.umass.edu?**
   Source IP Address: 10.11.203.42
   Source Port Number: 62800
   Destination IP Address: 128.119.245.12
   Destination Port Number: 80

Kevin Pirabaharan
0946212

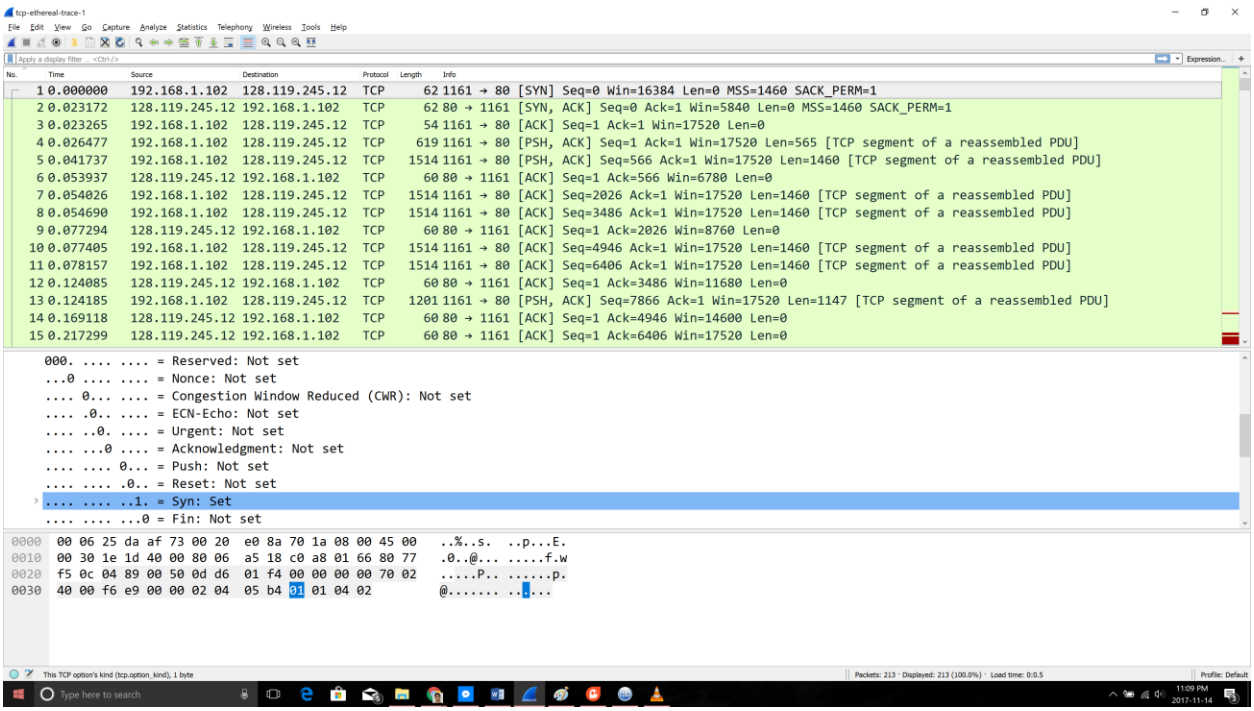


Figure 3 For Question 4

4. **What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? What is it in the segment that identifies the segment as a SYN segment?**
   The sequence number used to initiate the TCP connection is 0. The SYN flag is set to 1 which means that the segment is a SYN segment.

Kevin Pirabaharan
0946212

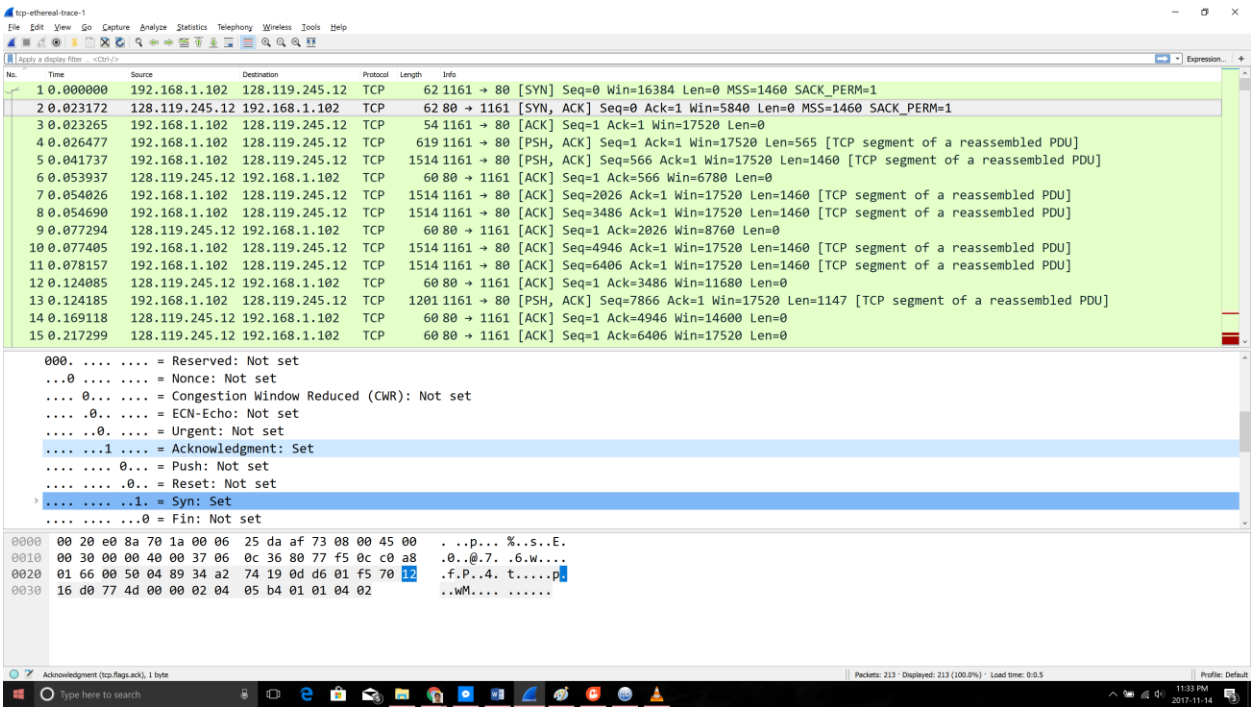


Figure 4 For Question 5

5. **What is the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN? What is the value of the Acknowledgement field in the SYNACK segment? How did gaia.cs.umass.edu determine that value? What is it in the segment that identifies the segment as a SYNACK segment?**

   Sequence number of the SYNACK segment is 0. The value of the Acknowledgement field in the SYNACK segment is 1, it's value is determined by adding 1 to the initial value of the SYN segment from client. Both the SYN and Acknowledgement flags are set to 1 meaning this is a SYNACK segment.

Kevin Pirabaharan
0946212

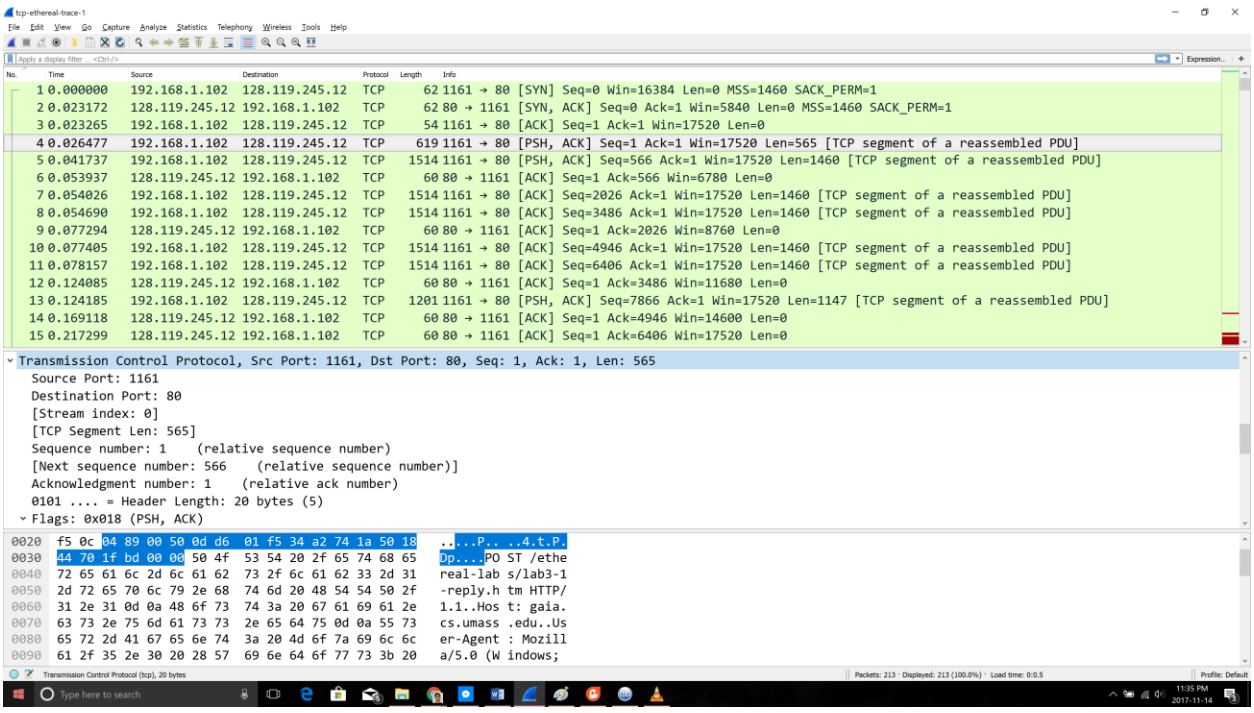


Figure 5 For Question 6

6. **What is the sequence number of the TCP segment containing the HTTP POST command?**
   The fourth segment contains the HTTP POST command, and the sequence number for this is 1.

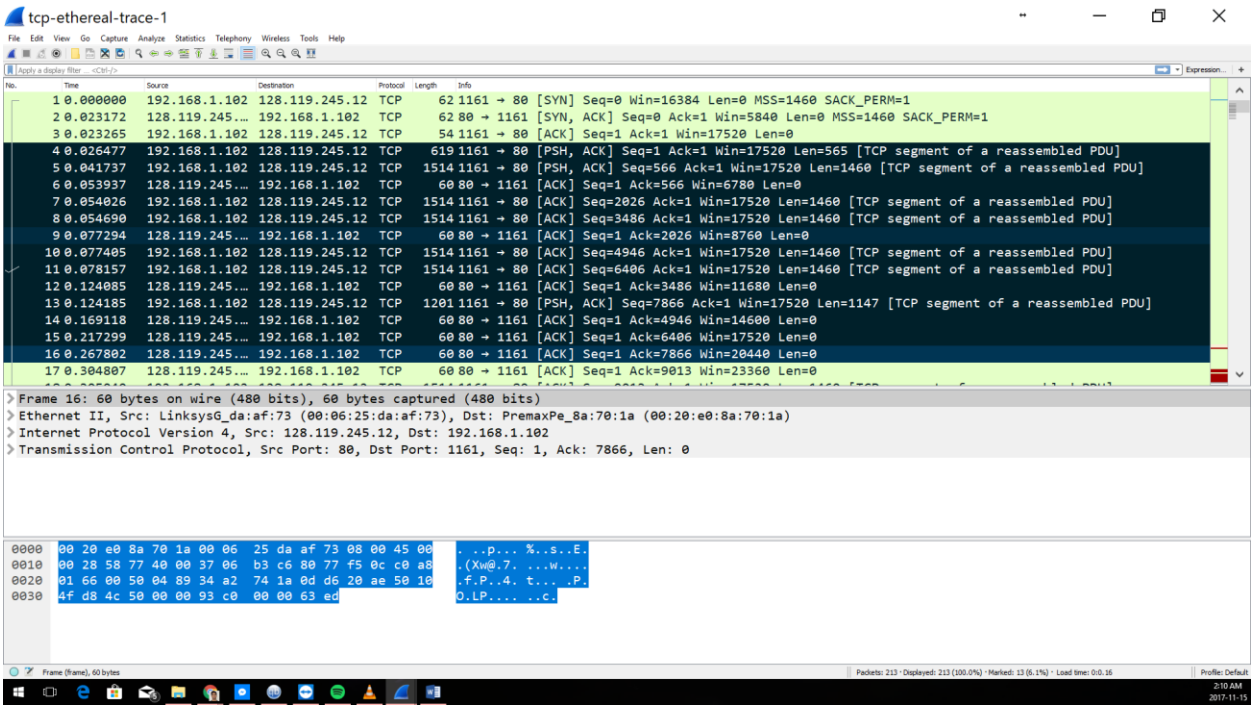


Figure 6 For Question 7

Kevin Pirabaharan
0946212

7. **What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST)? At what time was each segment sent? When was the ACK for each segment received? Given the difference between when each TCP segment was sent, and when its acknowledgement was received, what is the RTT value for each of the six segments? What is the EstimatedRTT value (see Section 3.5.3, page 242 in text) after the receipt of each ACK? Assume that the value of the EstimatedRTT is equal to the measured RTT for the first segment, and then is computed using the EstimatedRTT equation on page 242 for all subsequent segments.**

Segment	No.	Sequence #	Sent Time	ACK No.	ACK Rec Time	RTT (sec) ACK - Sent	ETT (seconds)
1	4	1	0.026477	6	0.053937	0.02746	0.0725
2	5	566	0.041737	9	0.077294	0.035557	0.0285
3	7	2026	0.054026	12	0.124085	0.070059	0.0337
4	8	3486	0.054690	14	0.169118	0.11443	0.0438
5	10	4906	0.077405	15	0.217299	0.13989	0.0558
6	11	6406	0.078157	16	0.267802	0.18964	0.0725

Calculations for EstimatedRTT after ACK of Segments (second):

- EstimatedRTT1 = RTT for Segment 1 = 0.02746
- EstimatedRTT2 = 0.875 * 0.02746(EstimatedRTT1) + 0.125 * 0.035557 (RTT2) = 0.0285
- EstimatedRTT3 = 0.875 * 0.0285(EstimatedRTT2) + 0.125 * 0.070059 (RTT3) = 0.0337
- EstimatedRTT4 = 0.875 * 0.0337(EstimatedRTT3) + 0.125 * 0.11443(RTT4) = 0.0438
- EstimatedRTT5 = 0.875 * 0.0438(EstimatedRTT4) + 0.125 * 0.13989(RTT5) = 0.0558
- EstimatedRTT6 = 0.875 * 0.0558(EstimatedRTT5) + 0.125 * 0.18964(RTT6) = 0.0725

Kevin Pirabaharan
0946212

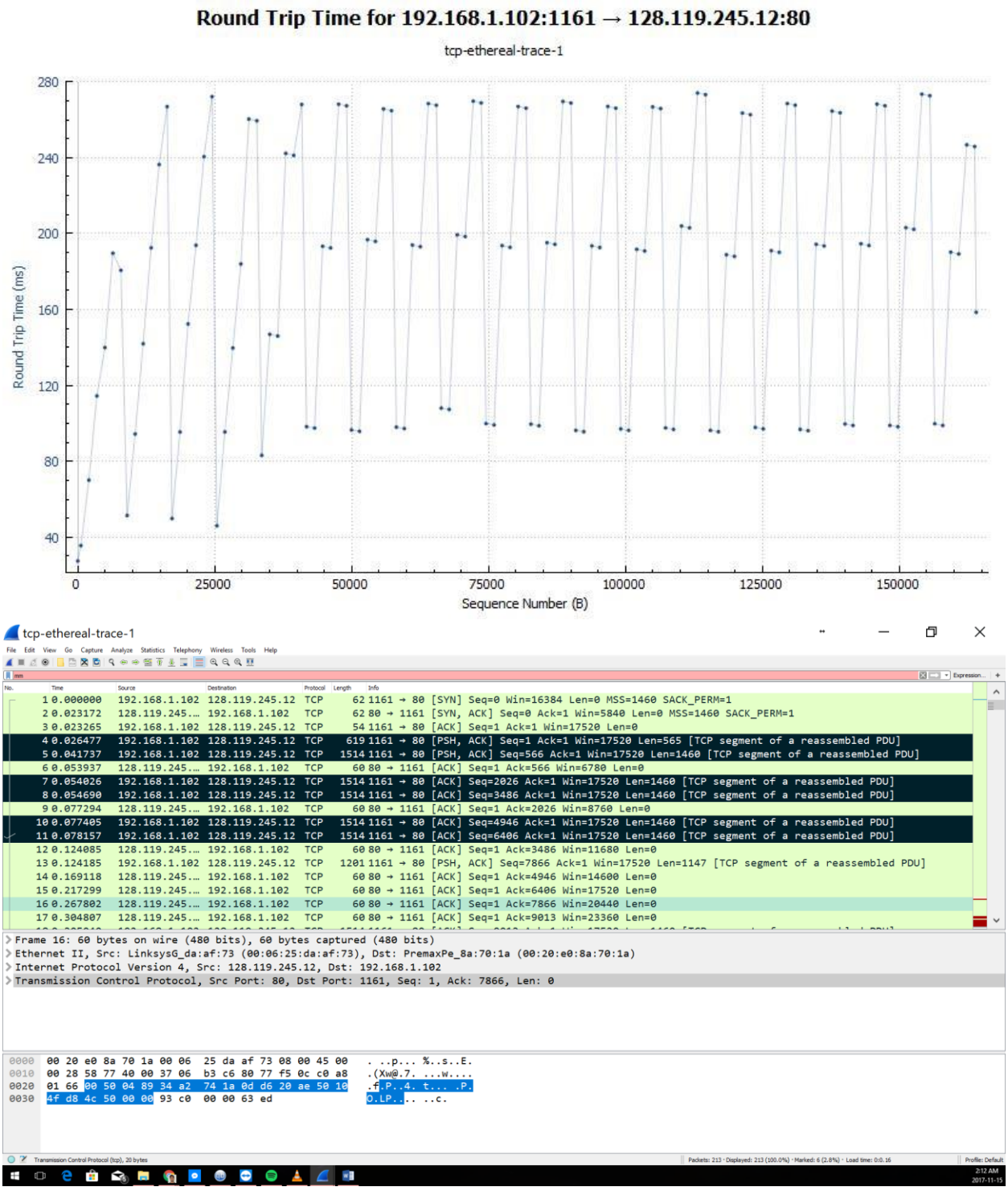


Figure 7 For Question 8

8. **What is the length of each of the first six TCP segments?**
   Length of the first TCP segment 565 bytes
   Length of each of other 5 segments: 1460 bytes

Kevin Pirabaharan
0946212

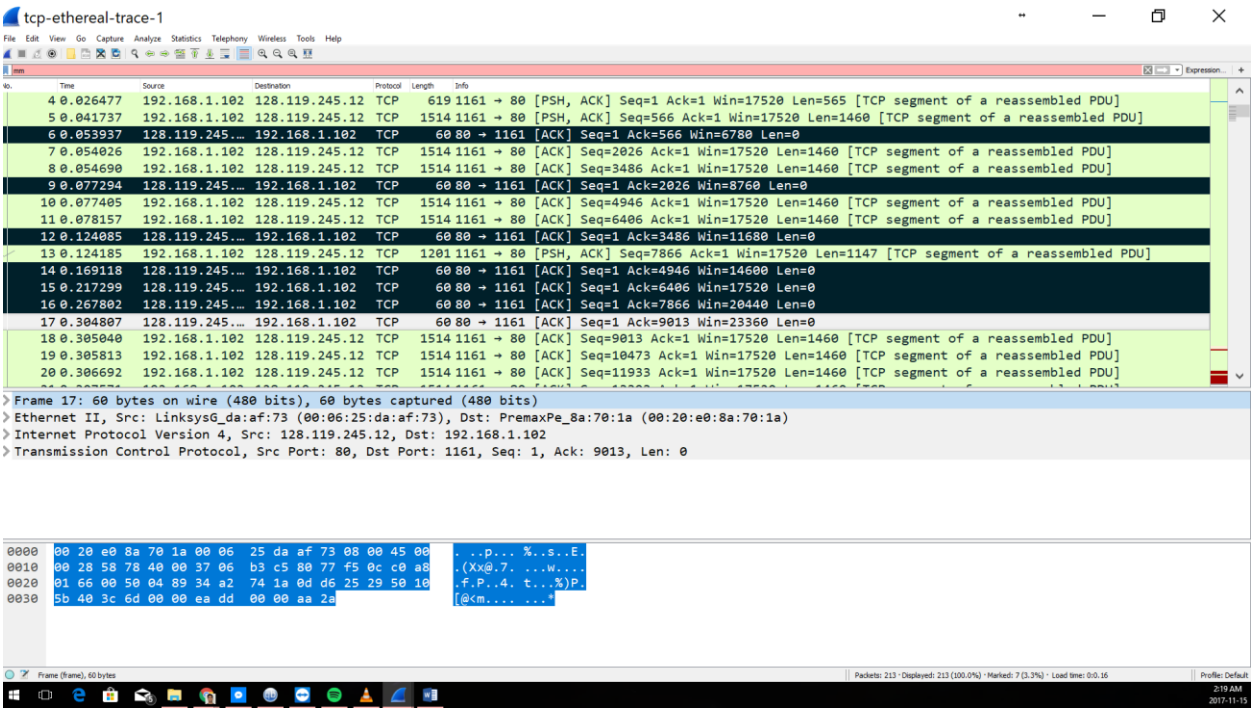
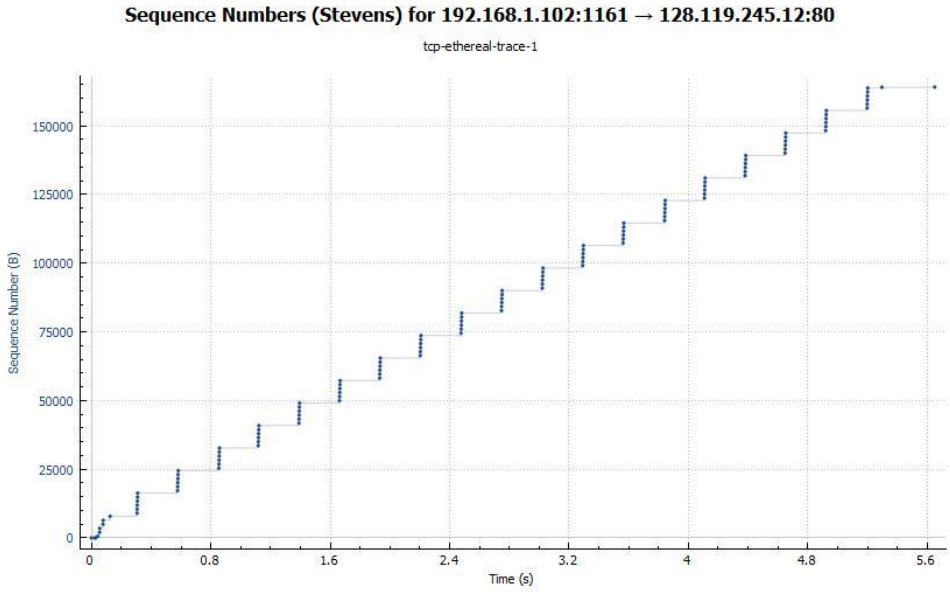


Figure 8 For Question 9

9. **What is the minimum amount of available buffer space advertised at the received for the entire trace? Does the lack of receiver buffer space ever throttle the sender?**
   The minimum amount of buffer space is 5840 bytes. The sender is never throttled due to lacking of receiver buffer space because it grows steadily to a maximum size of 62780 bytes
10. **Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?**
    No retransmission occurs in the trace file, this can be verified with the TimeSequence-Graph (Stevens). If there was a retransmission then the graph wouldn't be a straight increase in

sequence number.



11. **How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACKing every other received segment (see Table 3.2 on page 250 in the text).**
12. **What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.**

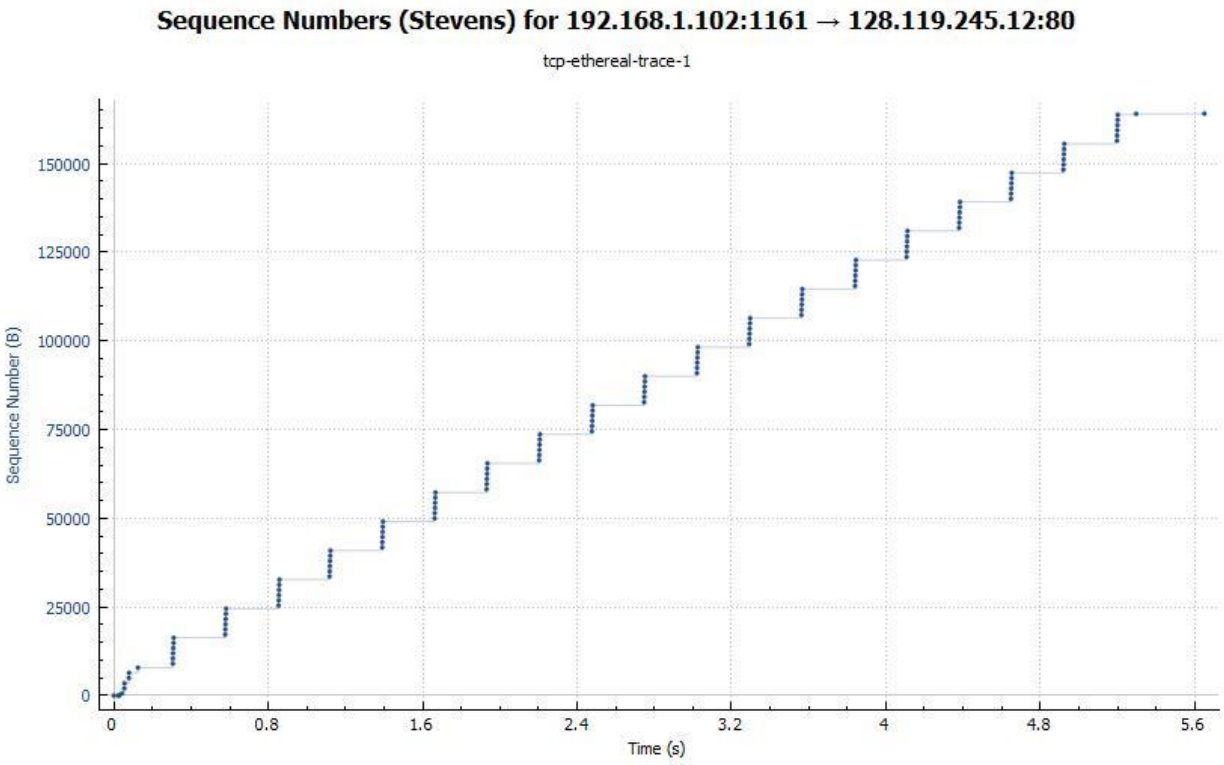
    Amount of data transmitted = 164091 (final transfer segment) - 1 (initial transfer segment)
    = 164090 bytes

    Amount of time incurred = 5.455830 (final transfer time) - 0.026477 (initial transfer segment)
    = 5.4294s

    Throughput = Amount of data transmitted / Time incurred
    = 164090 / 5.4294
    = 30.222 kbytes/sec.

13. **Use the Time-Sequence-Graph(Stevens) plotting tool to view the sequence number versus time plot of segments being sent from the client to the gaia.cs.umass.edu server. Can you identify where TCP's slowstart phase begins and ends, and where congestion avoidance takes over? Comment on ways in which the measured data differs from the idealized behavior of TCP that we've studied in the text.**

Kevin Pirabaharan
0946212



The slow-start phase seems to go for the first 0.2 – 1 second, after that the session goes into congestion avoidance state. The graph does not grow linearly, instead the sender sends packets in groups of 6. The reason behind this is because the HTTP server has enforced a rate-limit.

14. Answer each of two questions above for the trace that you have gathered when you transferred a file from your computer to gaia.cs.umass.edu

- Throughput

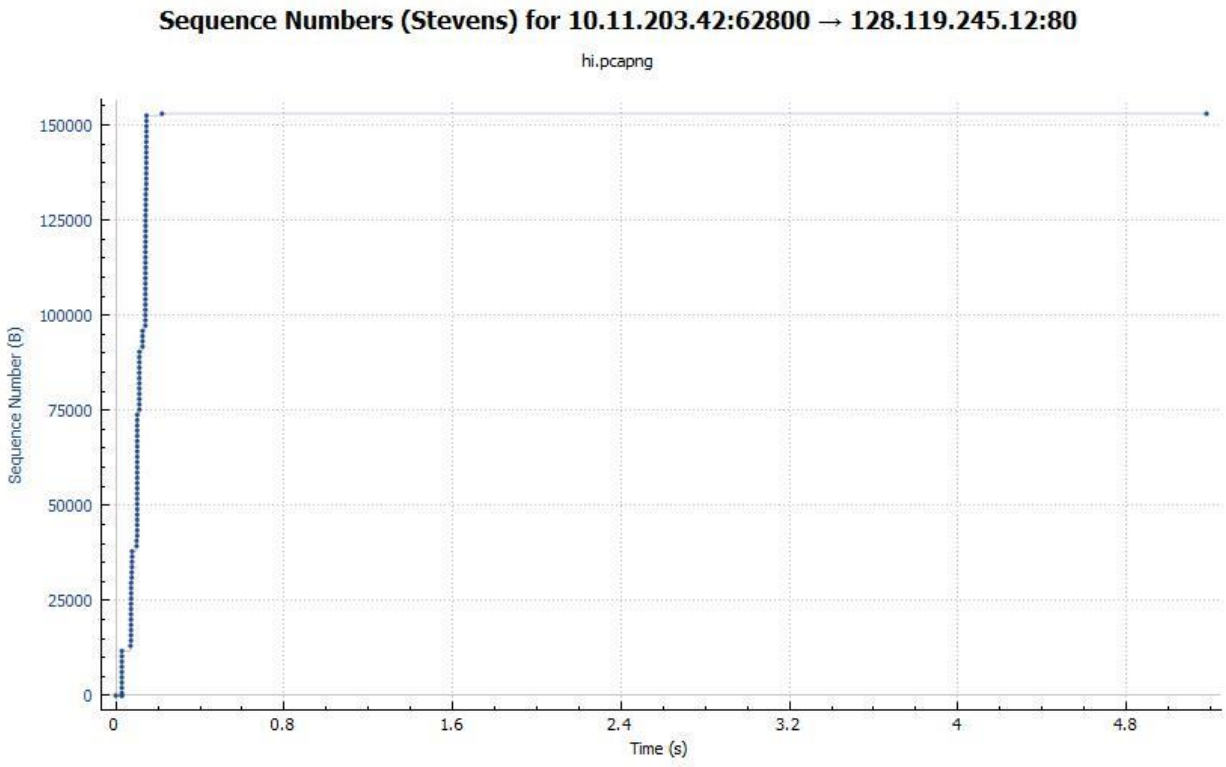
  Amount of data transmitted = final transfer segment - initial transfer segment
  = 152983 – 1
  = 152982

  Amount of time incurred = final transfer time - initial transfer segment
  = 2.394179 – 2.180408
  = 0.213771 bytes

  Throughput = Amount of data transmitted / Time incurred
  = 152982 / 0.213771
  = 715634.955 kbytes/sec.

- Graph sequences

Kevin Pirabaharan
0946212



The slow-start phase seems to go for the first 0.1 second, after that the session goes into congestion avoidance state. The graph grows linearly, there is no buffer limit and so the sender sends multiple packets at once.

Part 2: Wireshark Lab: UDP v7.0

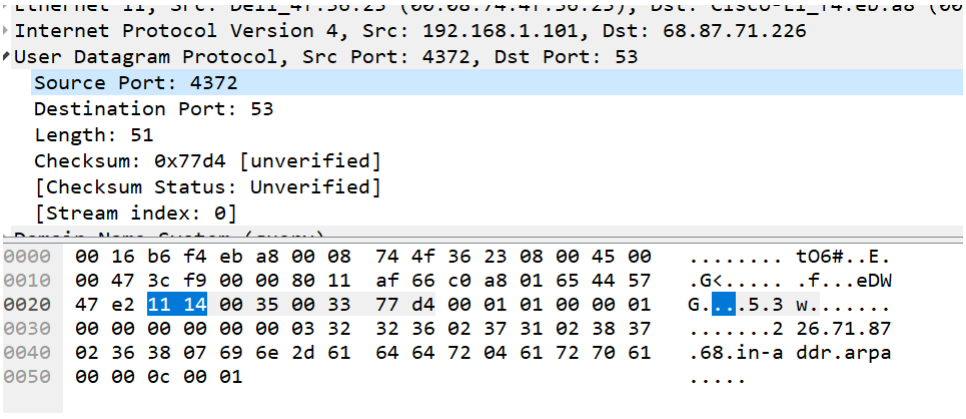


Figure 9 For Questions 1 -3

1. **Select one UDP packet from your trace. From this packet, determine how many fields there are in the UDP header. Name these fields.**
   There are four fields: source port, destination port, length, and checksum.
2. **By consulting the displayed information in Wireshark's packet content field for this packet, determine the length (in bytes) of each of the UDP header fields.**
   Each of the UDP header fields is 2 bytes long (as highlighted in the screen shot above).

Kevin Pirabaharan
0946212

3. **The value in the Length field is the length of what? (You can consult the text for this answer). Verify your claim with your captured UDP packet.**
   Value in the length field is sum of the 8 header bytes + 45 bytes of data/payload = 53 bytes
4. **What is the maximum number of bytes that can be included in a UDP payload? (Hint: the answer to this question can be determined by your answer to 2. above)**
   Max possible bytes $2^{16} - 1 = 65535$. This gives 65535 – 8 (minus the header) = 65527 bytes.
5. **What is the largest possible source port number? (Hint: see the hint in 4.)**
   $2^{16} - 1 = 65535$
6. **What is the protocol number for UDP? Give your answer in both hexadecimal and decimal notation. To answer this question, you'll need to look into the Protocol field of the IP datagram containing this UDP segment (see Figure 4.13 in the text, and the discussion of IP header fields).**

```
Fragment offset: 0
Time to live: 128
Protocol: UDP (17)
Header checksum: 0xaf66 [validation disabled]
[Header checksum status: Unverified]
Source: 192.168.1.101
Destination: 68.87.71.226
```

```
0000  00 16 b6 f4 eb a8 00 08  74 4f 36 23 08 00 45 00
0010  00 47 3c f9 00 00 80 11  af 66 c0 a8 01 65 44 57
0020  47 e2 11 14 00 35 00 33  77 d4 00 01 01 00 00 01
0030  00 00 00 00 00 00 03 32  32 36 02 37 31 02 38 37
0040  02 36 38 07 69 6e 2d 61  64 64 72 04 61 72 70 61
0050  00 00 0c 00 01
```

   Protocol number in decimal: 17
   Hexadecimal: 0x11
7. **Examine a pair of UDP packets in which your host sends the first UDP packet and the second UDP packet is a reply to this first UDP packet. (Hint: for a second packet to be sent in response to a first packet, the sender of the first packet should be the destination of the second packet). Describe the relationship between the port numbers in the two packets.**
   Sent

7	0.060268	192.168.1.101	68.87.71.226	DNS	71	Standard query 0x0004 A www.mit.edu
8	0.074984	68.87.71.226	192.168.1.101	DNS	87	Standard query response 0x0004 A www.mit.e
9	2.874248	128.119.245.…	192.168.1.101	TLS…	123	Application Data
10	2.886418	192.168.1.101	128.119.245.93	TLS…	91	Application Data
11	2.901973	128.119.245.…	192.168.1.101	TLS…	107	Application Data
12	2.914501	192.168.1.101	128.119.245.93	TLS…	91	Application Data
13	2.929459	128.119.245.…	192.168.1.101	TLS…	107	Application Data
14	2.930094	192.168.1.101	128.119.245.93	TLS…	123	Application Data
15	2.945019	128.119.245.…	192.168.1.101	TLS…	155	Application Data
16	3.122854	192.168.1.101	128.119.245.93	TCP	54	4294 → 993 [ACK] Seq=144 Ack=277 Win=64946
17	3.202518	192.168.1.101	128.119.245.93	TLS…	267	Application Data
18	3.222514	128.119.245.…	192.168.1.101	TLS…	667	Application Data

```
> Frame 7: 71 bytes on wire (568 bits), 71 bytes captured (568 bits)
> Ethernet II, Src: Dell_4f:36:23 (00:08:74:4f:36:23), Dst: Cisco-Li_f4:eb:a8 (00:16:b6:f4:eb:a8)
> Internet Protocol Version 4, Src: 192.168.1.101, Dst: 68.87.71.226
> User Datagram Protocol, Src Port: 4375, Dst Port: 53
> Domain Name System (query)
```

   Response

Kevin Pirabaharan
0946212

8	0.074984	68.87.71.226	192.168.1.101	DNS	87	Standard query response 0x0004 A www.mit.edu
9	2.874248	128.119.245.…	192.168.1.101	TLS…	123	Application Data
10	2.886418	192.168.1.101	128.119.245.93	TLS…	91	Application Data
11	2.901973	128.119.245.…	192.168.1.101	TLS…	107	Application Data
12	2.914501	192.168.1.101	128.119.245.93	TLS…	91	Application Data
13	2.929459	128.119.245.…	192.168.1.101	TLS…	107	Application Data
14	2.930094	192.168.1.101	128.119.245.93	TLS…	123	Application Data
15	2.945019	128.119.245.…	192.168.1.101	TLS…	155	Application Data
16	3.122854	192.168.1.101	128.119.245.93	TCP	54	4294 → 993 [ACK] Seq=144 Ack=277 Win=64946 L…
17	3.202518	192.168.1.101	128.119.245.93	TLS…	267	Application Data
18	3.222514	128.119.245.…	192.168.1.101	TLS…	667	Application Data

```
> Frame 8: 87 bytes on wire (696 bits), 87 bytes captured (696 bits)
> Ethernet II, Src: Cisco-Li_f4:eb:a8 (00:16:b6:f4:eb:a8), Dst: Dell_4f:36:23 (00:08:74:4f:36:23)
> Internet Protocol Version 4, Src: 68.87.71.226, Dst: 192.168.1.101
> User Datagram Protocol, Src Port: 53, Dst Port: 4375
> Domain Name System (response)
```

The source (red, for example a client) send a request packet to the destination (blue, for example a server), but on the response the sender's port becomes the destination and vice versa (same thing goes for IP).

Part 3: Wireshark Lab: IP v7.0

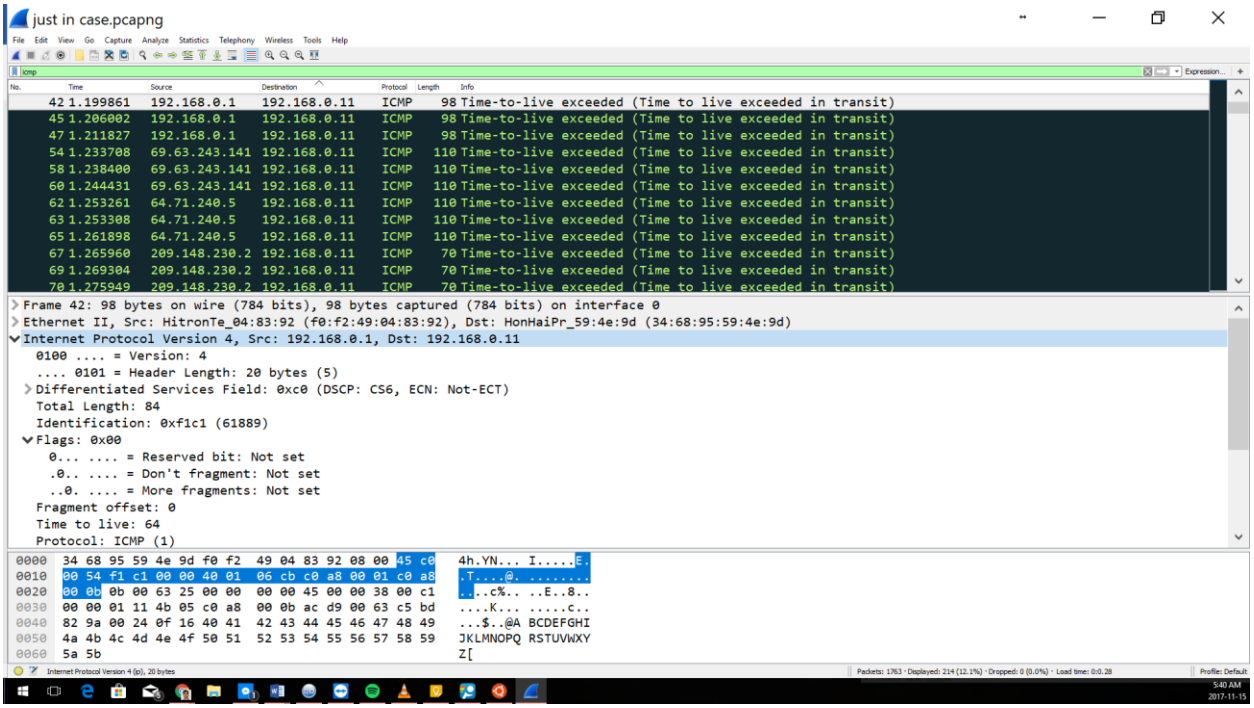


Figure 10 For Questions 1 - 4

1. **Select the first ICMP Echo Request message sent by your computer, and expand the Internet Protocol part of the packet in the packet details window. What is the IP address of your computer?**
   192.168.0.11
2. **Within the IP packet header, what is the value in the upper layer protocol field?**
   Protocol: ICMP (1)

Kevin Pirabaharan
0946212

3. **How many bytes are in the IP header? How many bytes are in the payload of the IP datagram? Explain how you determined the number of payload bytes.**
   Header = 20 bytes
   Payload = Total Length – Header Length
   = 56 – 20 = 36 bytes
4. **Has this IP datagram been fragmented? Explain how you determined whether or not the datagram has been fragmented.**
   This datagram hasn't been fragmented because none of the fragment bits have been set.
5. **Which fields in the IP datagram always change from one datagram to the next within this series of ICMP messages sent by your computer?**
   The check sum changes and so does the sequence number in the times that it is shown
6. **Which fields stay constant? Which of the fields must stay constant? Which fields must change? Why?**
   The header length and time to live are pre-set so they don't change, while the fragment number, sequence number, flags, total length and checksum aren't so they will change (variance in each segment).
7. **Describe the pattern you see in the values in the Identification field of the IP datagram.**
   Identification field is increases by 1 on every new outgoing message.

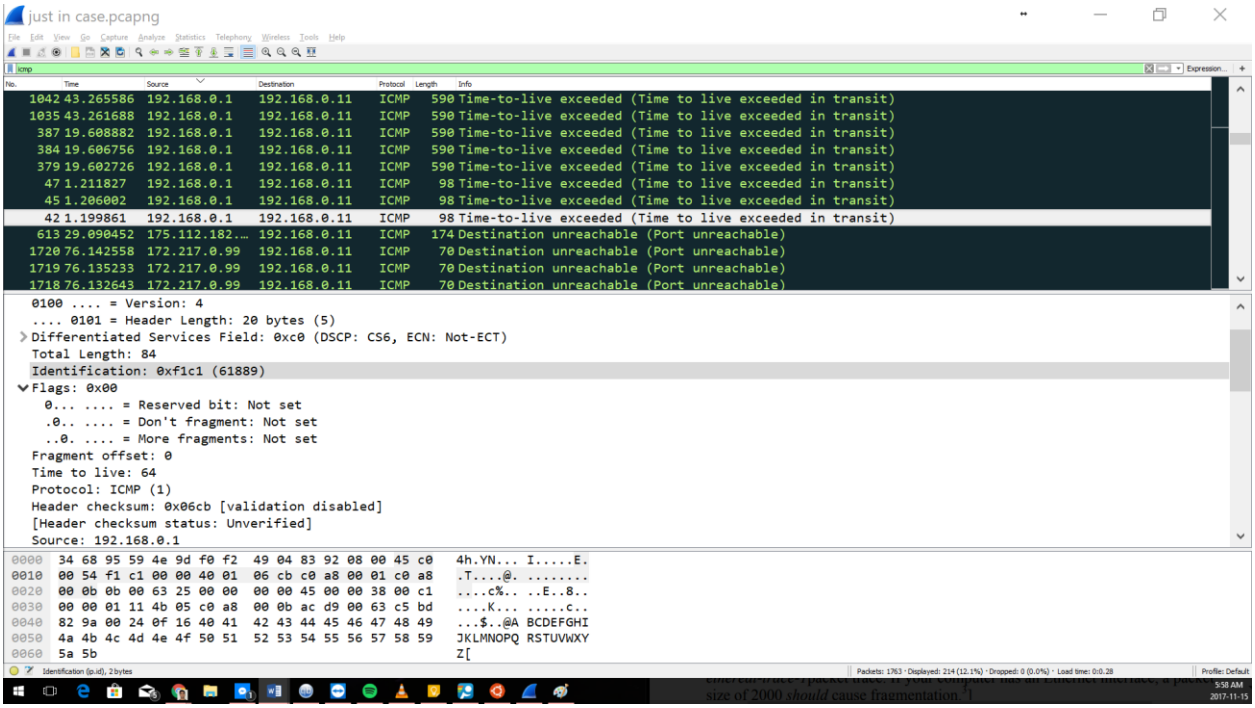


Figure 11 For Questions 8-9

8. **What is the value in the Identification field and the TTL field?**
   Identification Field: 0xf1c1 (61889)
   Time to Live: 64
9. **Do these values remain unchanged for all of the ICMP TTL-exceeded replies sent to your computer by the nearest (first hop) router? Why?**

Kevin Pirabaharan
0946212

The Identification field changes so that it can be unique every time, while time to live remains the same because it is a pre-set therefore it is constant.

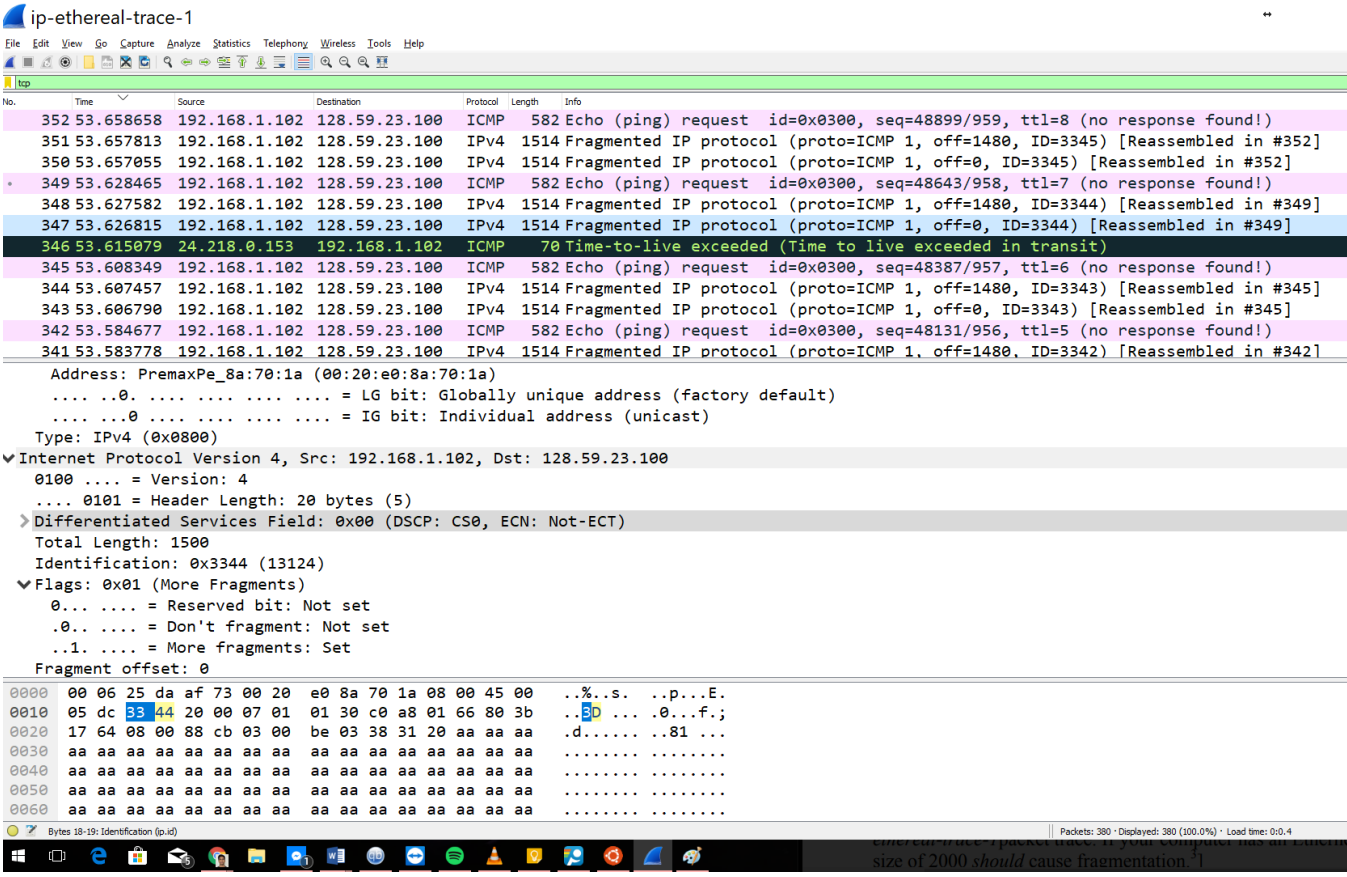
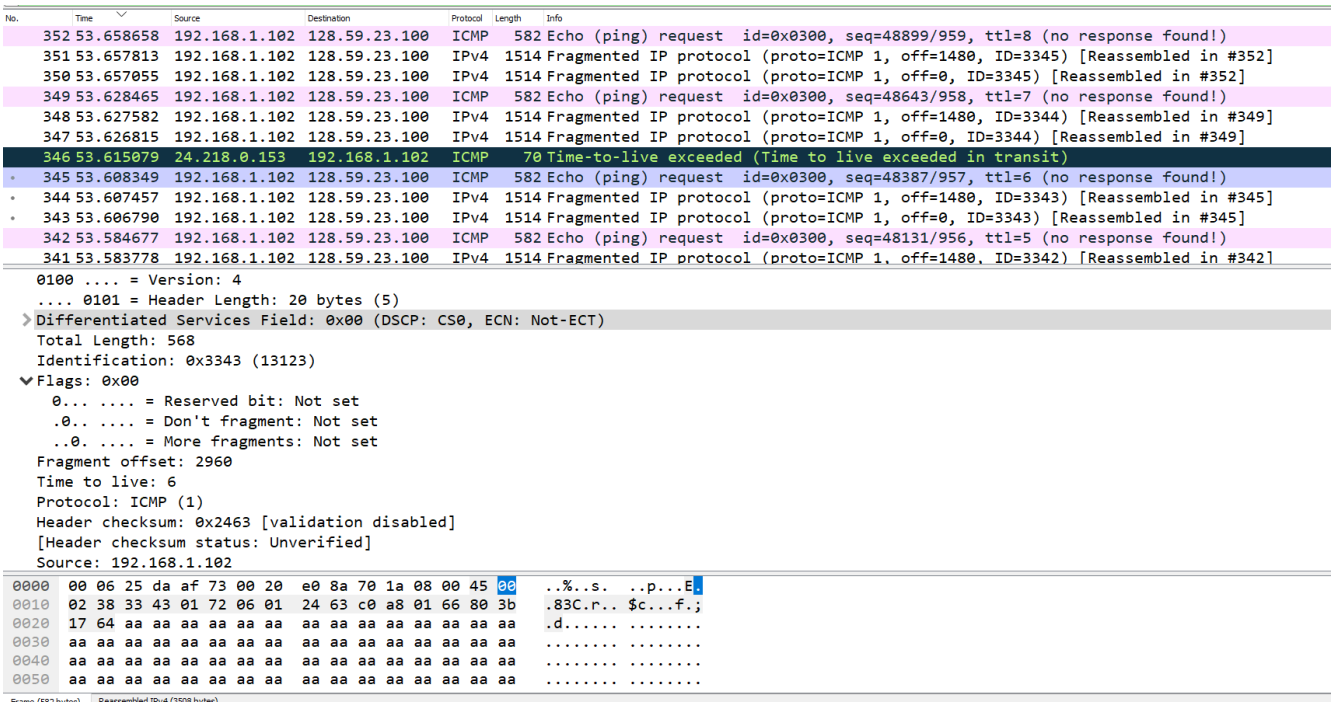


Figure 12 For Questions 10 and 11

10. **Find the first ICMP Echo Request message that was sent by your computer after you changed the Packet Size in pingplotter to be 2000. Has that message been fragmented across more than one IP datagram?**
    Yes it has been fragmented
11. **Print out the first fragment of the fragmented IP datagram. What information in the IP header indicates that the datagram been fragmented? How long is this IP datagram?**
    The more fragments bit being set to 1 is a sign of fragmentation. The fragment offset being 0 means that this is the first fragment. The length is 1500.
12. **Print out the second fragment of the fragmented IP datagram. What information in the IP header indicates that this is not the first datagram fragment? Are the more fragments? How can you tell?**

Kevin Pirabaharan
0946212



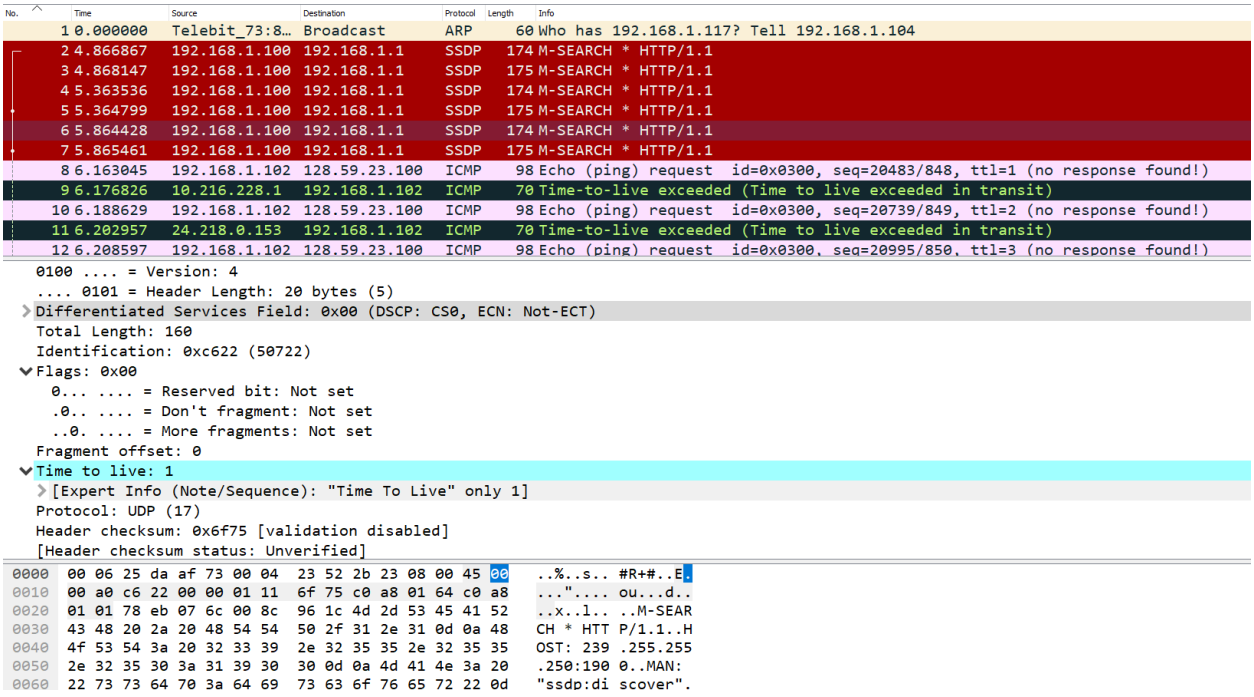
Since the offset is 2960, it means that this isn't the first fragment. However since the more fragments bit is 0, it means that there are no more fragments

13. **What fields change in the IP header between the first and second fragment?**

    The fields that change are the flags, header checksum, total length and fragment offset.

14. **How many fragments were created from the original datagram?**

    3 Fragments were created from original datagram



15. **What fields change in the IP header among the fragments?**

    Fragment offset, total length, more fragments bit, TTL and the checksum