Computer Networks CSE 4344/5344

Project 2
Transmission Control Protocol Analysis using Wireshark

Instructor: Dr. Sajib Datta

GTA (4344/5344): Jees Augustine

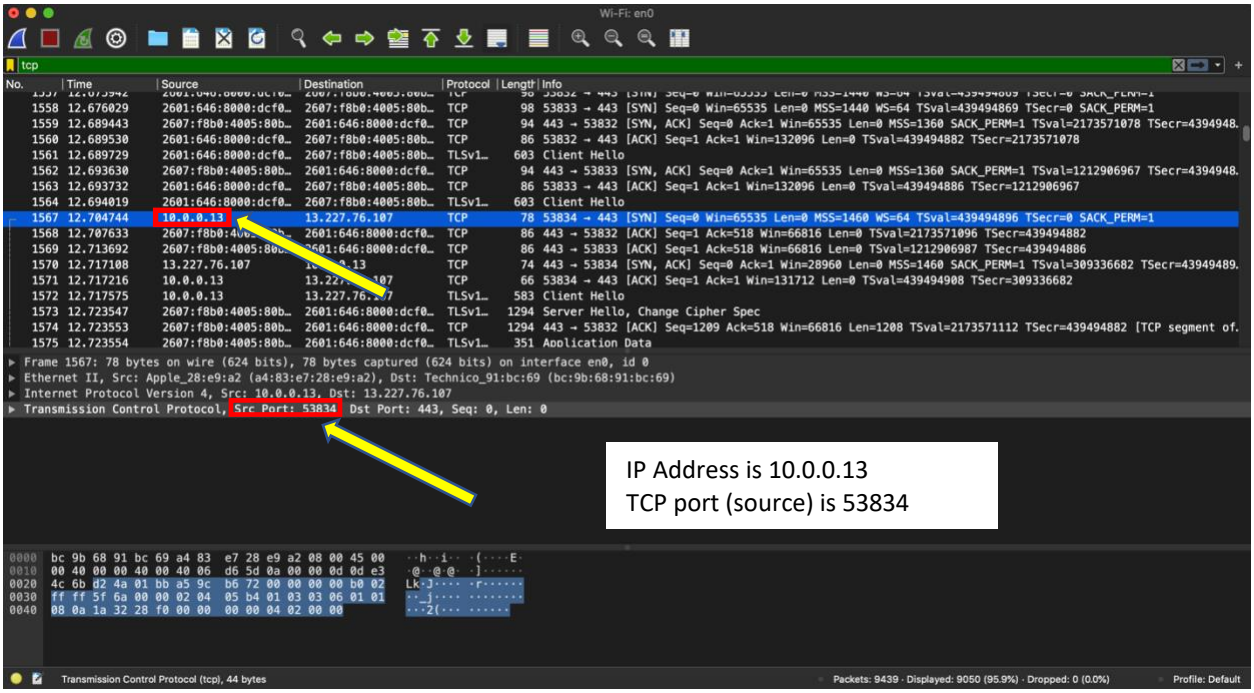
Summer 2020

Name : Jahnavi Nuthalapati

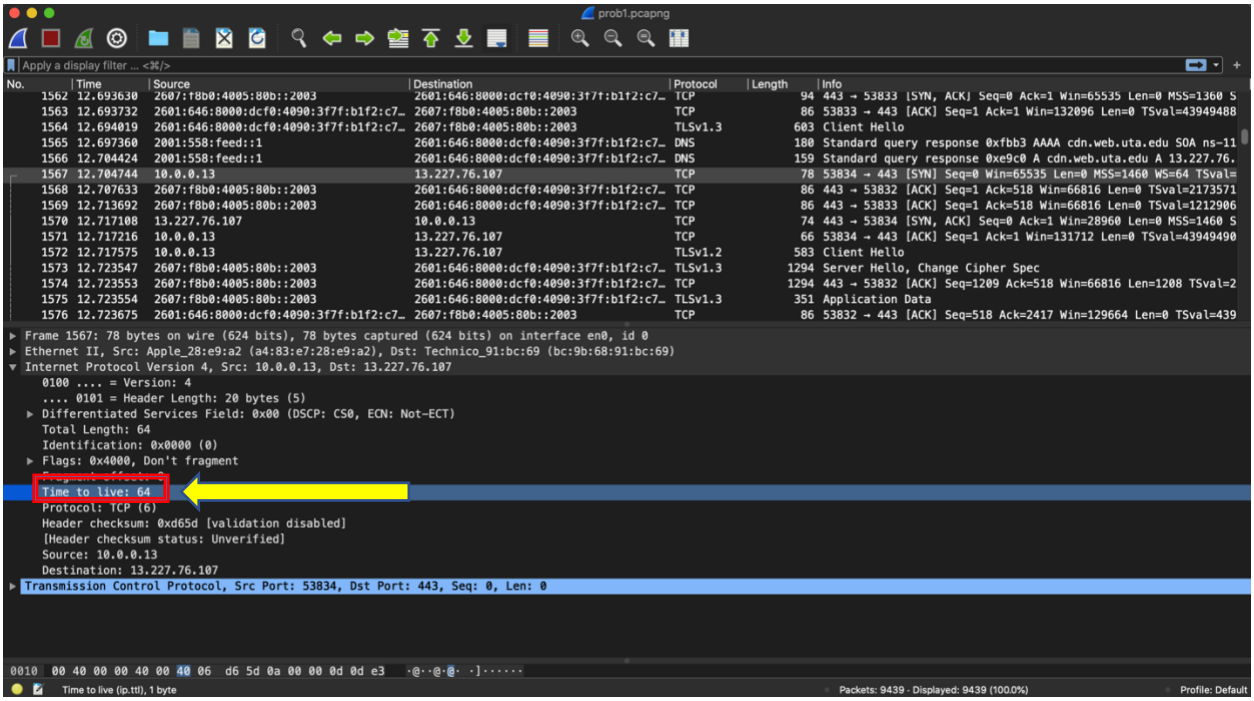
Table of Contents

Problem Set 1

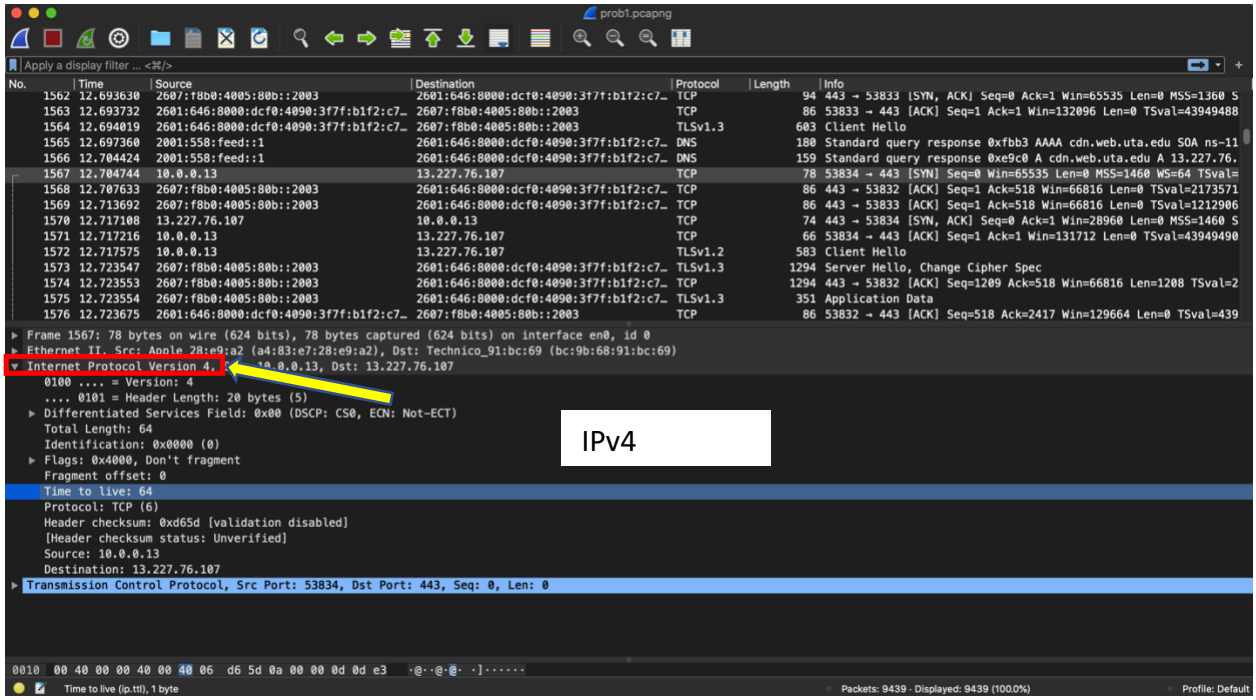
1. What is the IP address and TCP port number used by your client computer (source) to browse the page uta.edu.



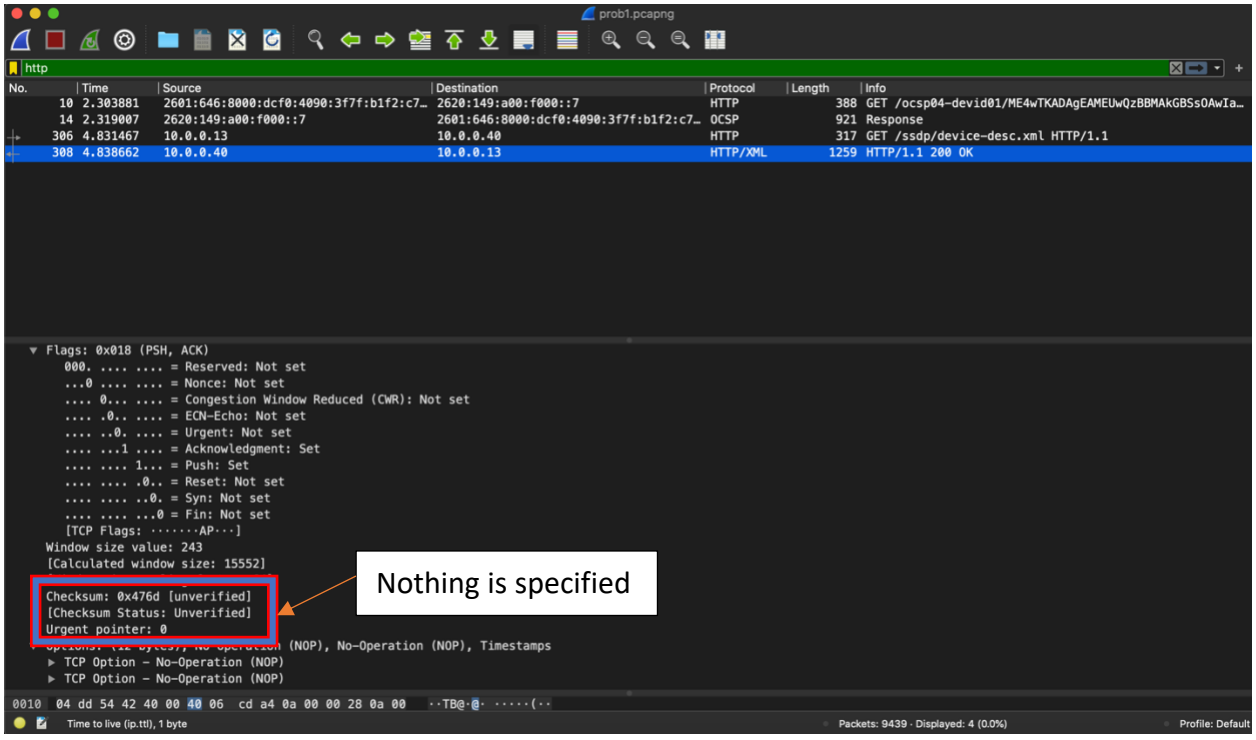
2. What is the TTL value that is used in this communication ?



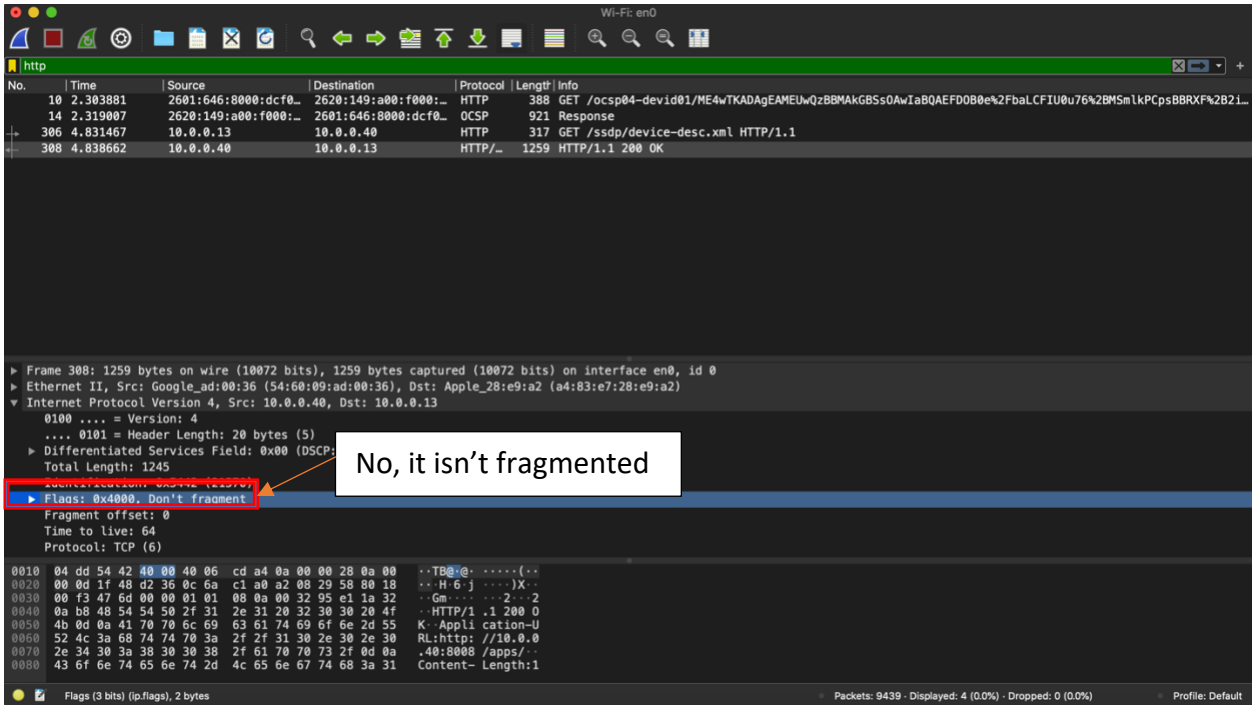
3. Did you Use IPv4 or IPv6 for communication ?



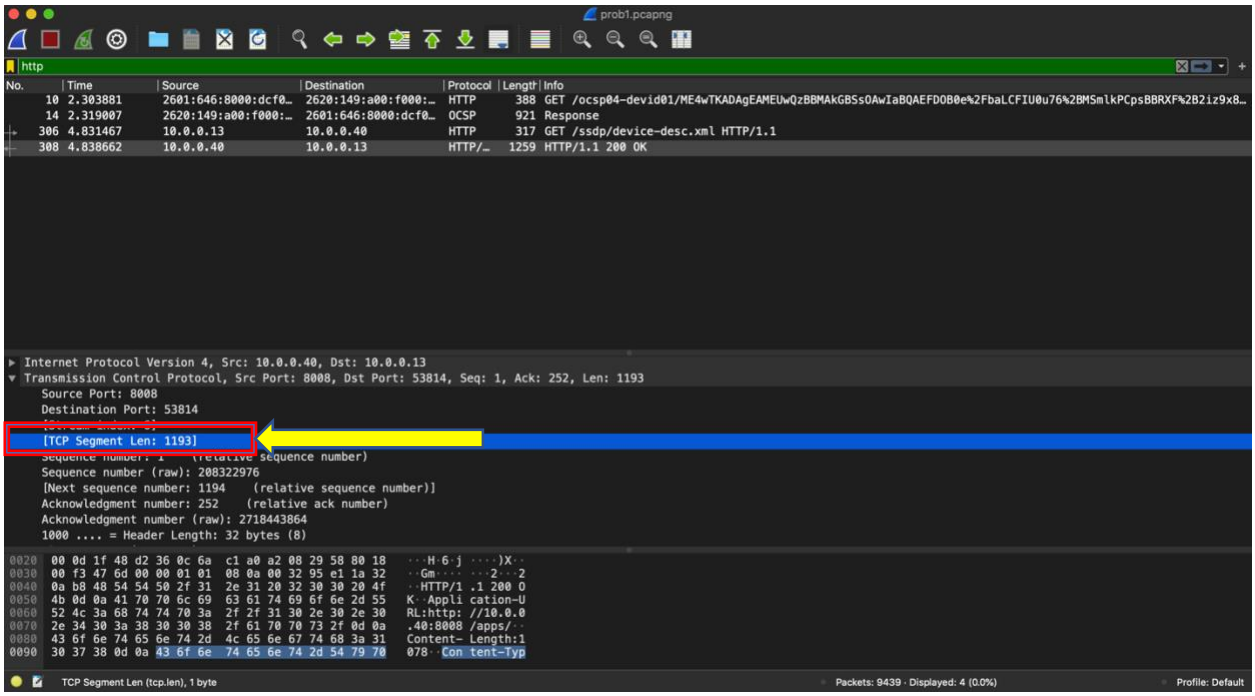
4. Does your optional field has some particular information or not.



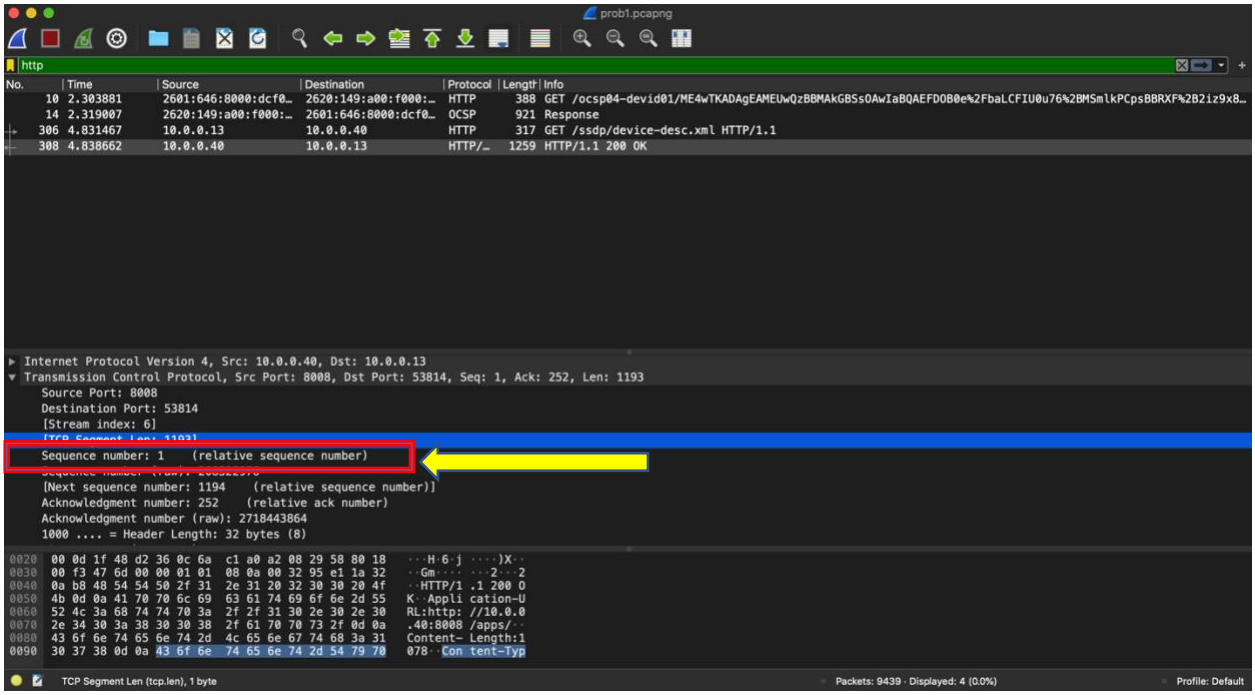
5. Is the Packet Fragmented



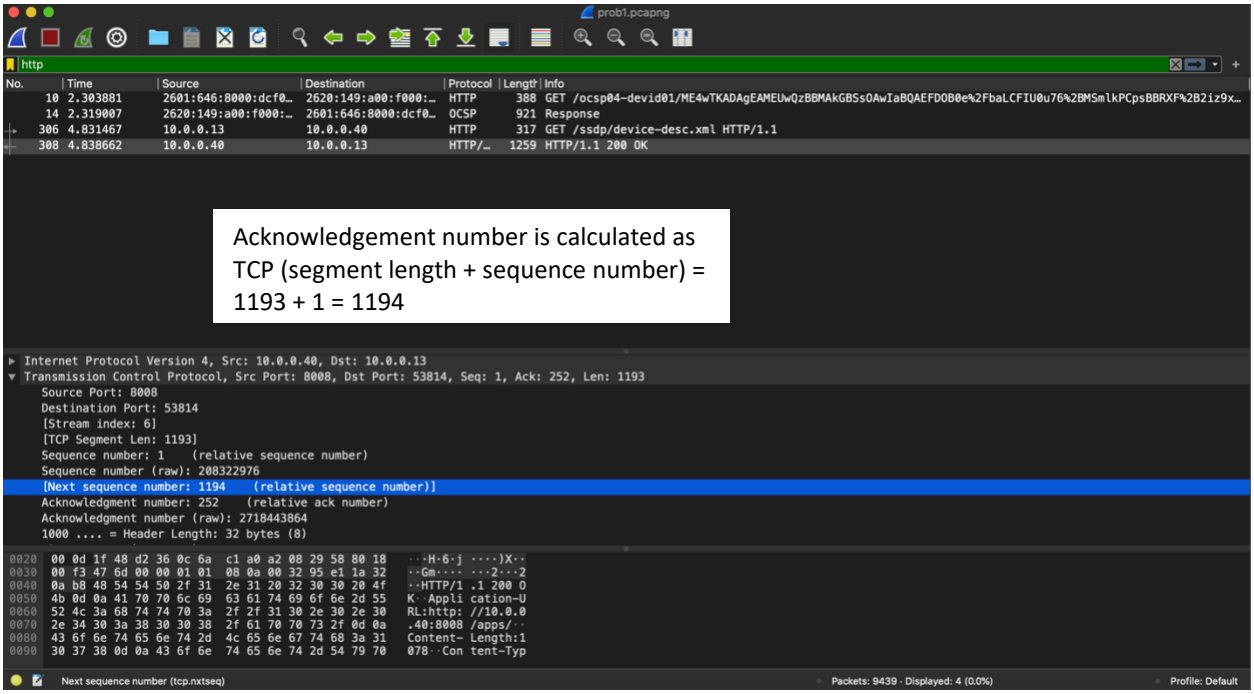
6. What is the TCP segment length ?



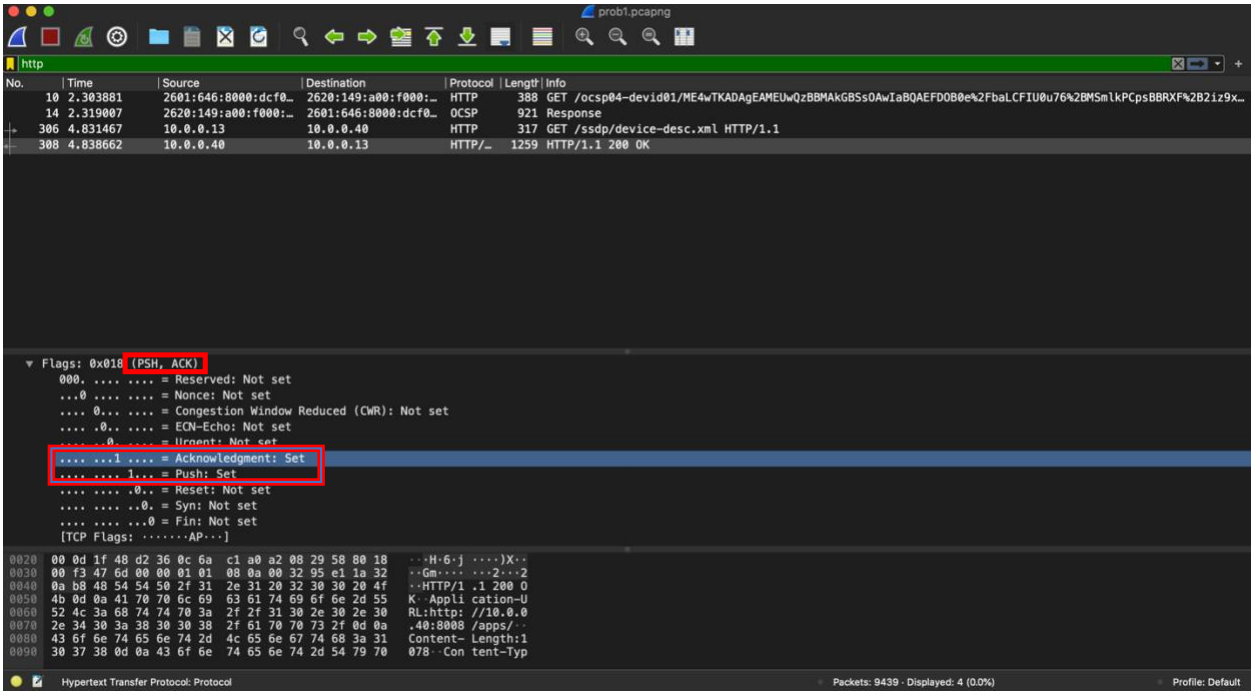
7. What is the Sequence Number of TCP segment (you can use the relative sequence number).



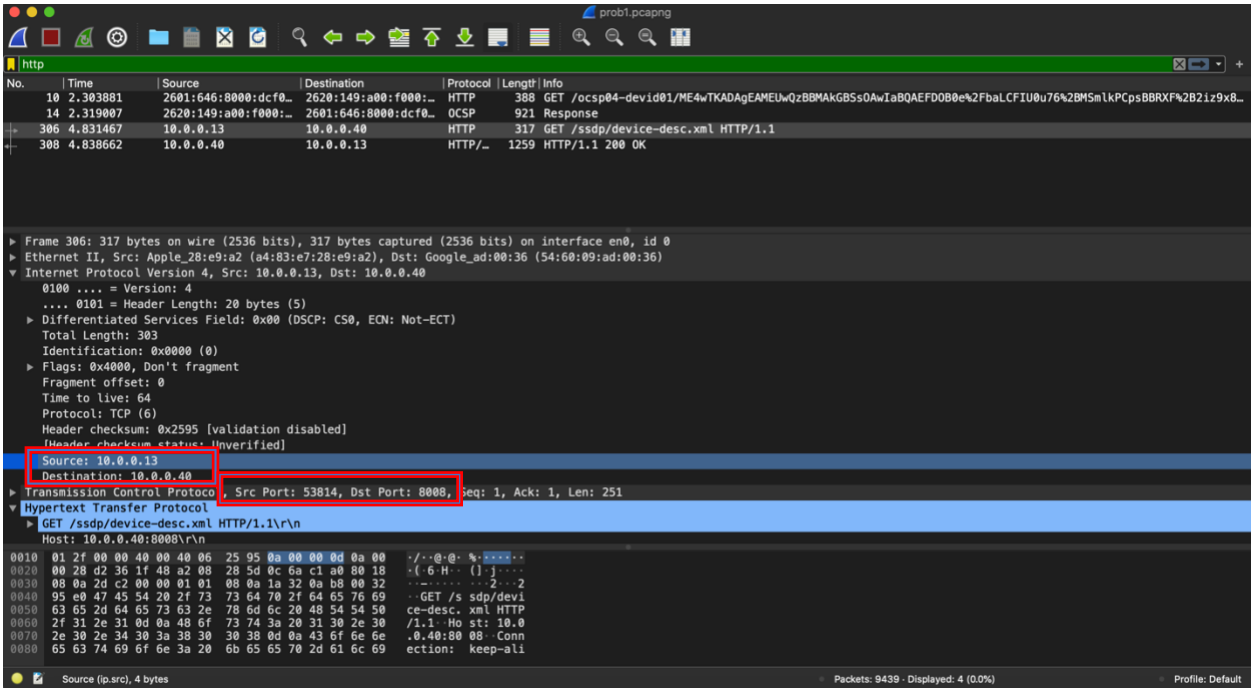
8. Calculate the acknowledgement number based on the the two questions above. Verify your solution with the Wireshark values.



9. What are the fields in the TCP Flags. No need to give any values but give the field names given in Wireshark

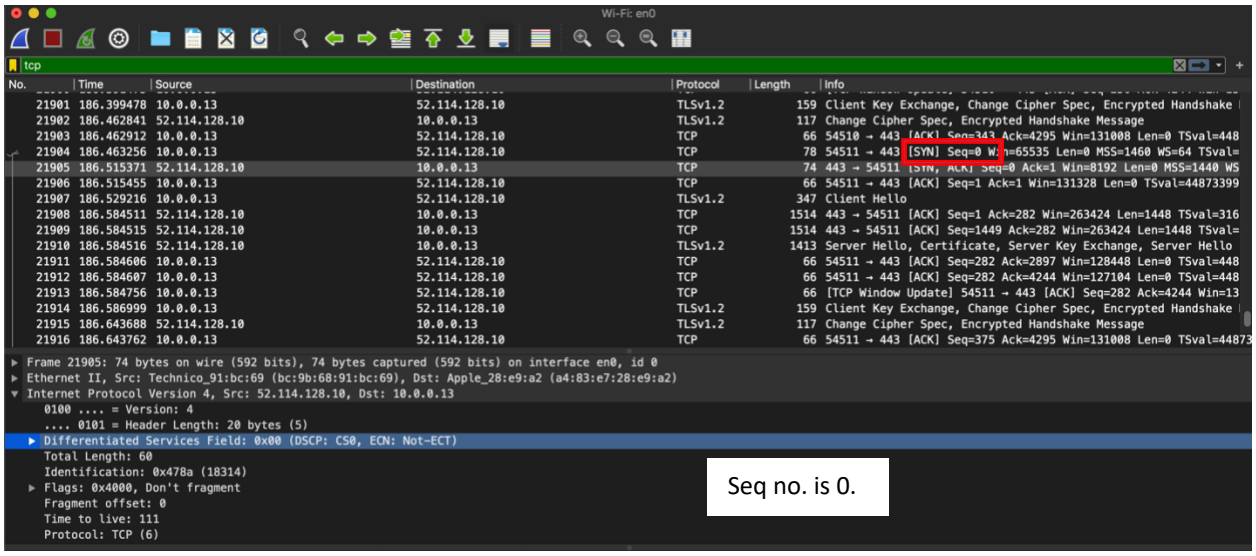


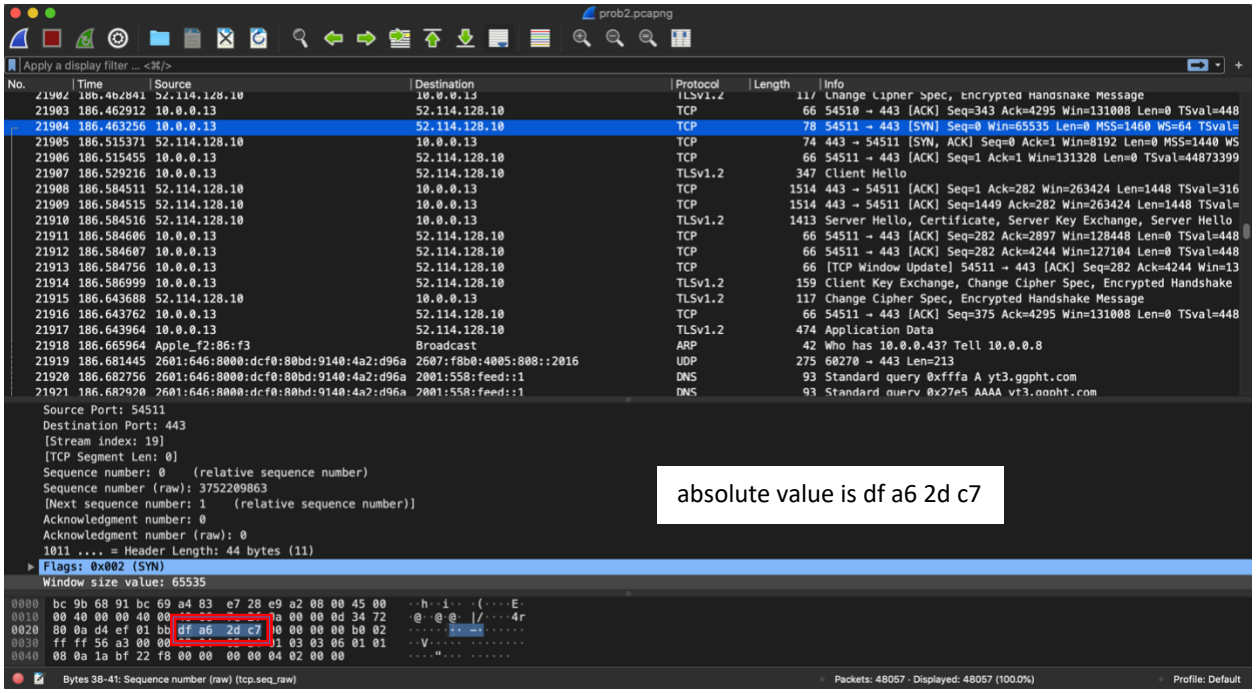
10. What is the IP address of uta.edu ? On what port number is it sending and receiving TCP segments for this connection?



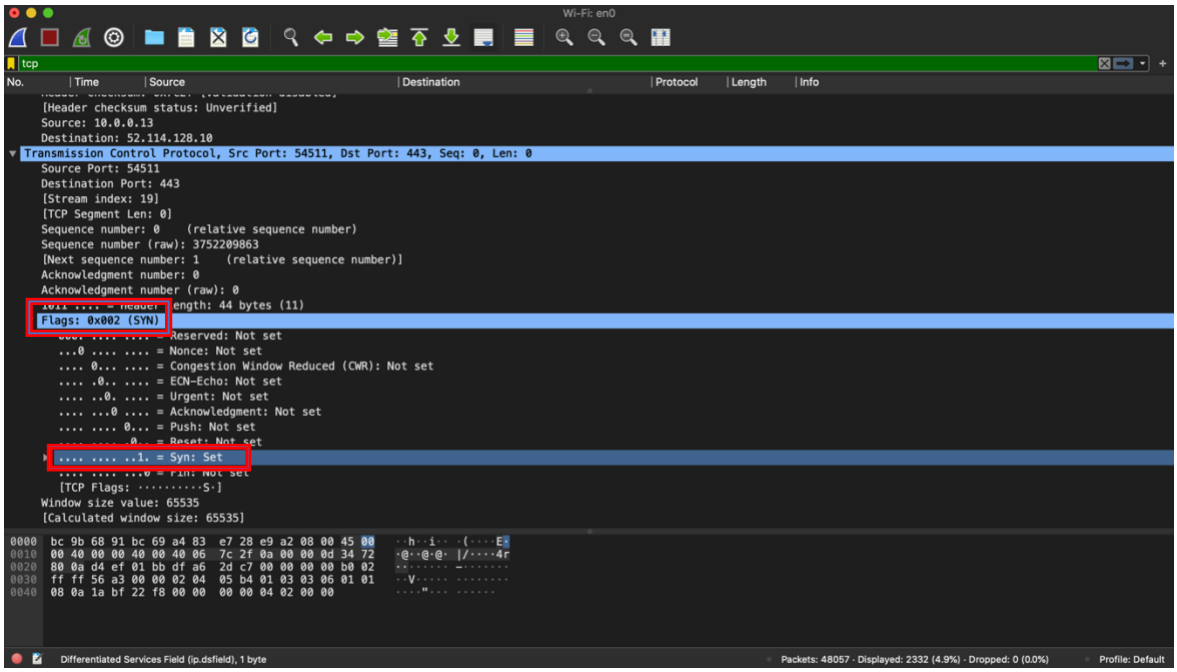
Problem Set 2

1. What is the sequence number (absolute) of the TCP SYN segment that is used to initiate the TCP connection between the client computer and youtube.com?

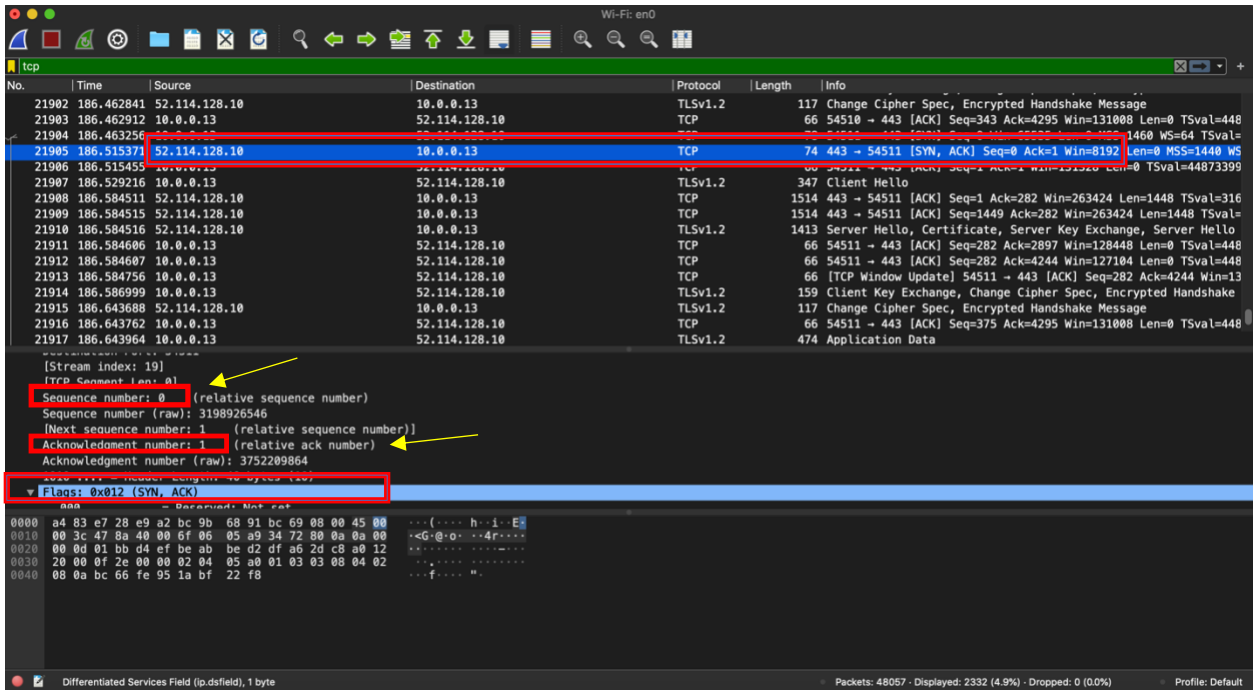




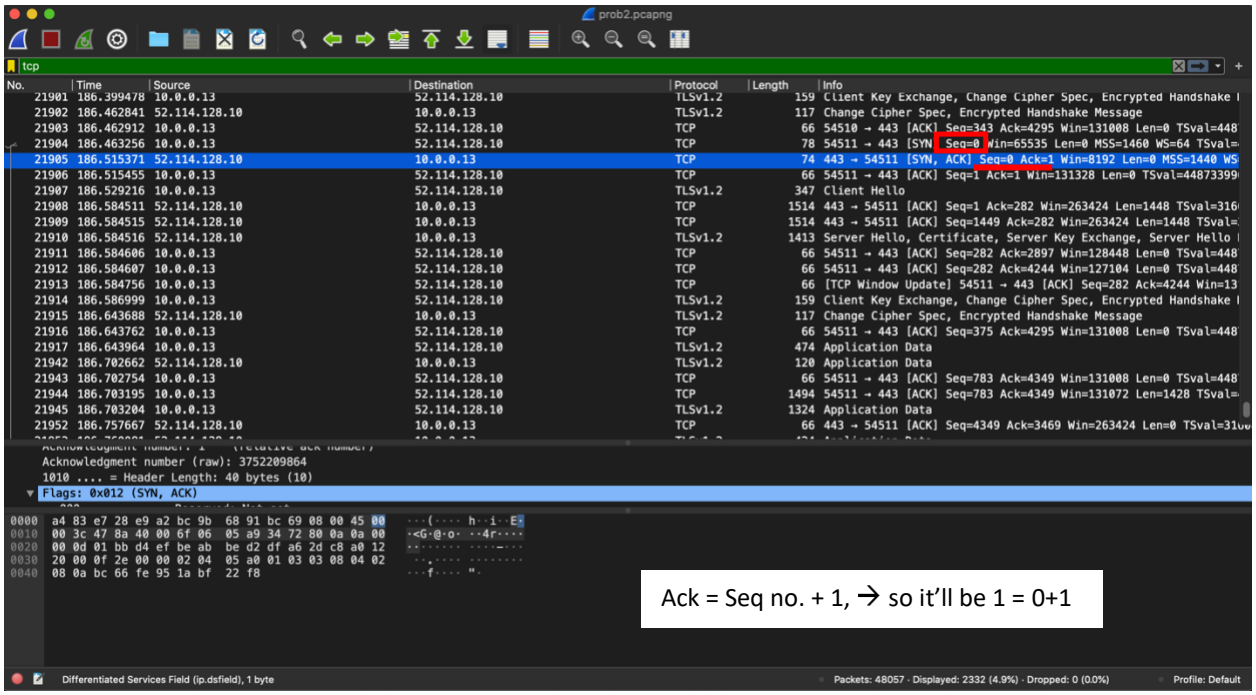
2. What is it in the segment that identifies the segment as a SYN segment? (The highlighted boxes)



3. What is the sequence number of the SYNACK segment sent by youtube.com to the client computer in reply to the SYN? What is the value of the Acknowledgement field in the SYNACK segment?

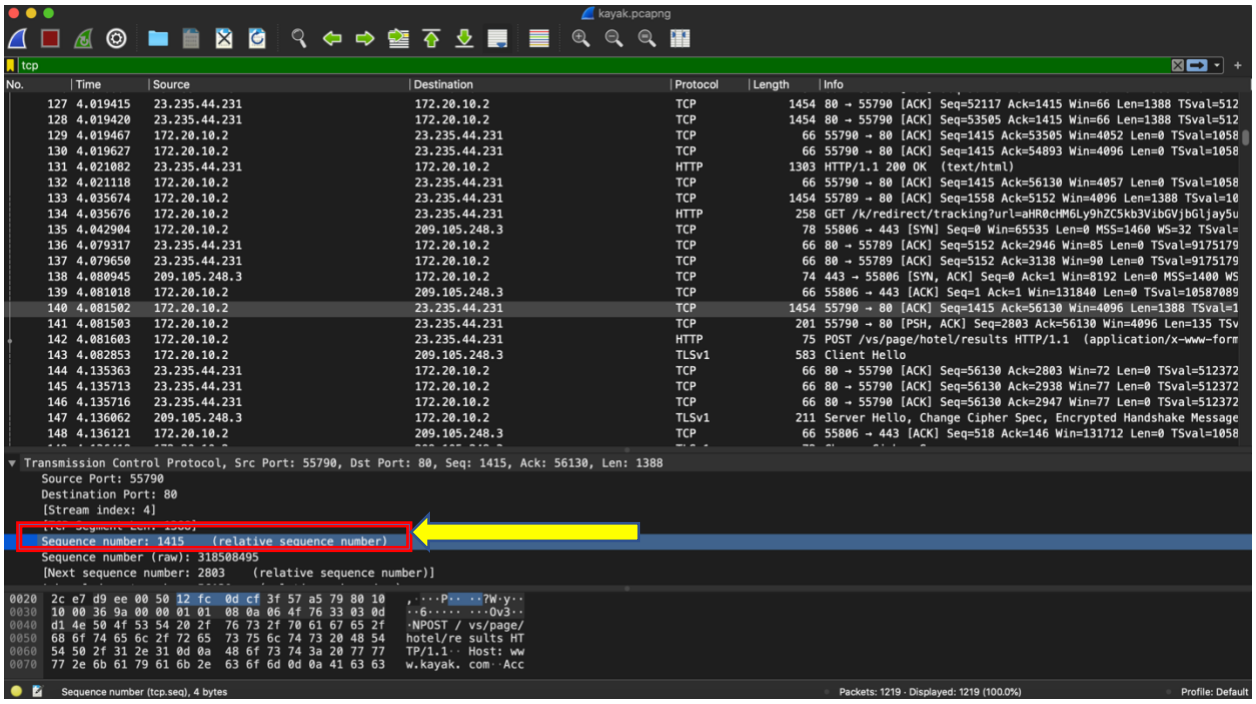


4. How did youtube.com determine that value? What is it in the segment that identifies the segment as a SYNACK segment?



Problem Set 3

1. What is the sequence number of the TCP segment containing the first HTTP POST command?



2. Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection.

   (The screenshots related to this question can be found below the calculations 2q)

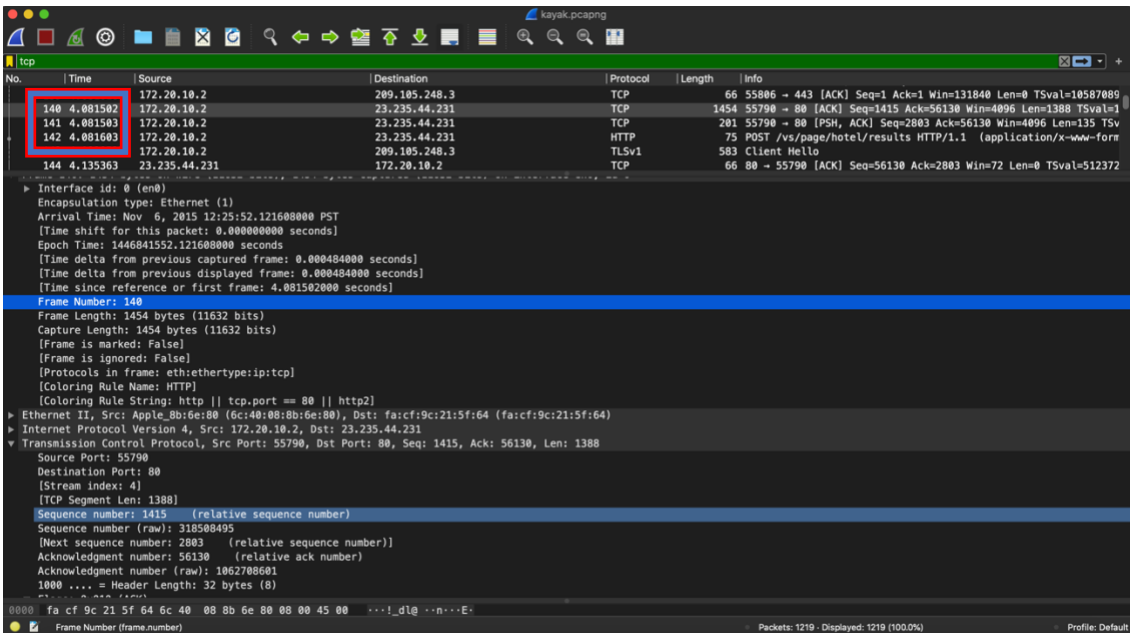
(i) What are the sequence numbers of the first four segments in the TCP connection (including the segment containing the HTTP POST)?

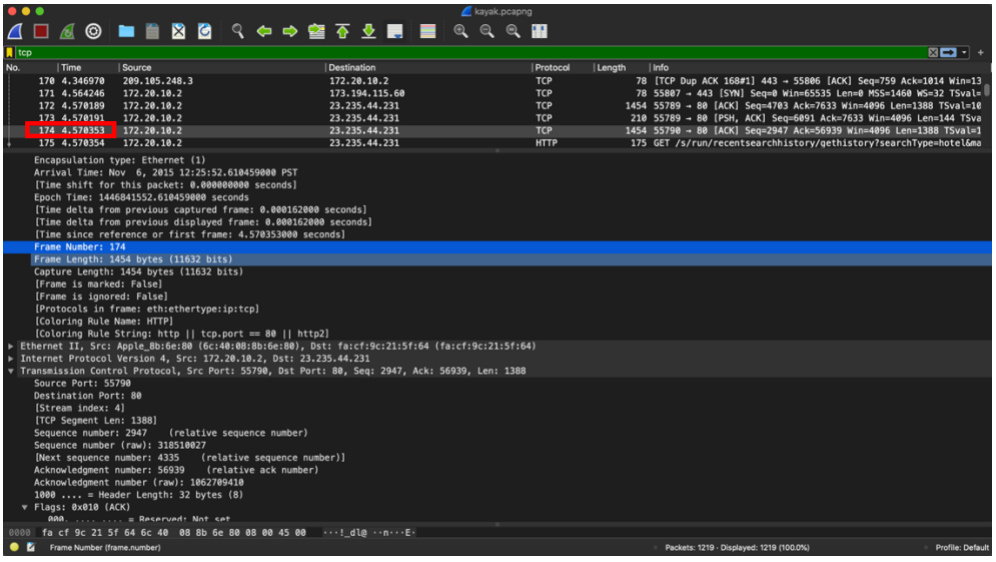
Segment Number	Sequence Number
140	1415
141	2803
142	2938
174	2947

(ii) At what time was each segment sent?

(1st screenshot depicts the time for segment numbers 140, 141, 142 and 2nd for 174)

Segment Number	Time at which the segment was sent
140	4.081502
141	4.081503
142	4.081603
174	4.570353





(iii) When was the ACK for each segment received?

(See the highlighted boxes in the screenshots below)

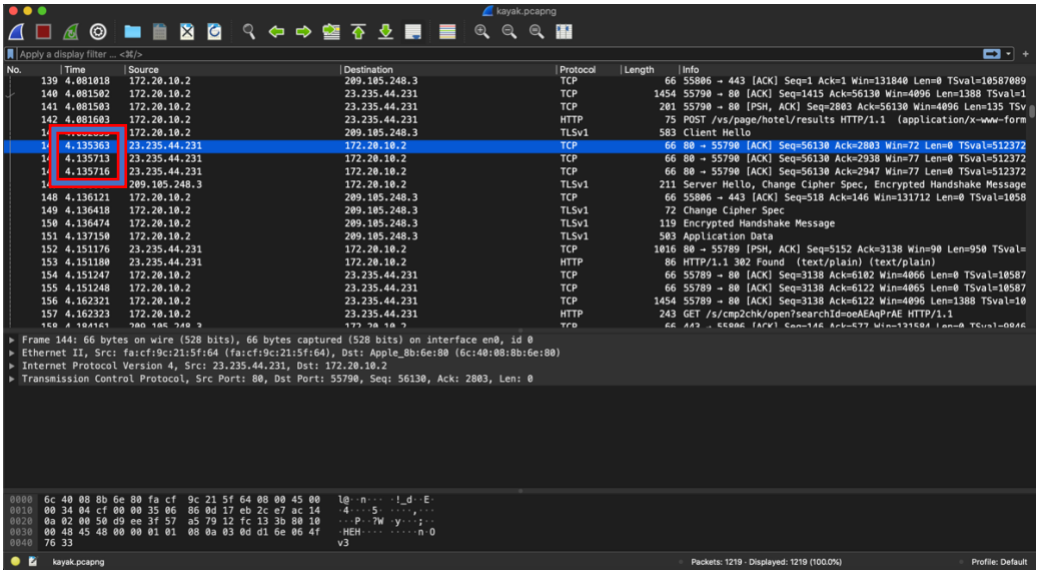
Received time for ACK

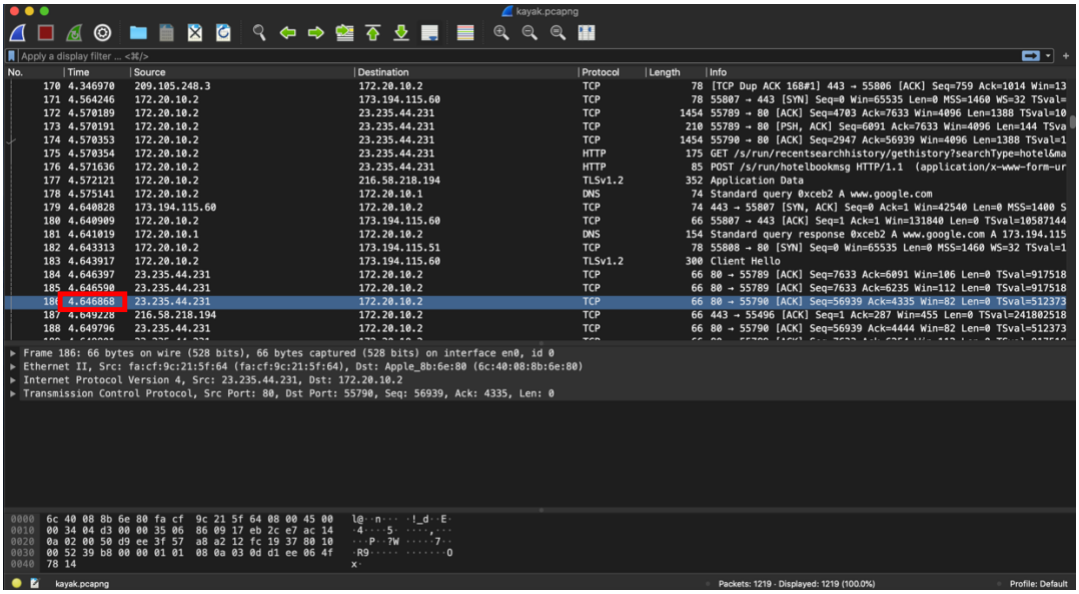
4.135363

4.135713

4.135716

4.6468





(iv) Given the difference between when each TCP segment was sent, and when its acknowledgement was received,

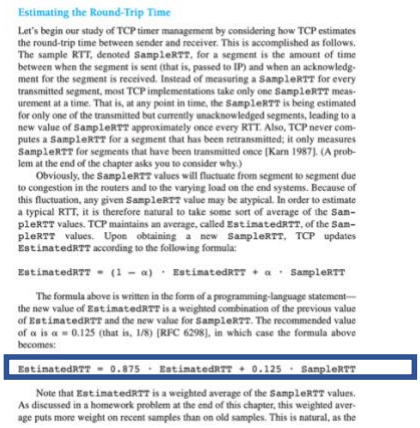
(v) what is the RTT value for each of the four segments?

(vi) What is the EstimatedRTT value (see Section 3.5.3, page 239 in text) after the receipt of each ACK?

(vii) 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 239 for all subsequent segments

Ans for iv, v, vi and vii is – (tabular form)

The Estimated RTT is to be calculated by considering the previous values also, so for the segments S1, S2, S3 and S4 the values will be (According to the TextBook,)



Estimating the Round-Trip Time

Let's begin our study of TCP timer management by considering how TCP estimates the round-trip time between sender and receiver. This is accomplished as follows. The sample RTT, denoted `SampleRTT`, for a segment is the amount of time between when the segment is sent (that is, passed to IP) and when an acknowledgment for the segment is received. Instead of measuring a `SampleRTT` for every transmitted segment, most TCP implementations take only one `SampleRTT` measurement at a time. That is, at any point in time, the `SampleRTT` is being estimated for only one of the transmitted but currently unacknowledged segments, leading to a new value of `SampleRTT` approximately once every RTT. Also, TCP never computes a `SampleRTT` for a segment that has been retransmitted; it only measures `SampleRTT` for segments that have been transmitted once [Karn 1987]. (A problem at the end of the chapter asks you to consider why.)

Obviously, the `SampleRTT` values will fluctuate from segment to segment due to congestion in the routers and to the varying load on the end systems. Because of this fluctuation, any given `SampleRTT` value may be atypical. In order to estimate a typical RTT, it is therefore natural to take some sort of average of the `SampleRTT` values. TCP maintains an average, called `EstimatedRTT`, of the `SampleRTT` values. Upon obtaining a new `SampleRTT`, TCP updates `EstimatedRTT` according to the following formula:

```
EstimatedRTT = (1 − α) · EstimatedRTT + α · SampleRTT
```

The formula above is written in the form of a programming-language statement—the new value of `EstimatedRTT` is a weighted combination of the previous value of `EstimatedRTT` and the new value for `SampleRTT`. The recommended value of α is α = 0.125 (that is, 1/8) [RFC 6298], in which case the formula above becomes:

```
EstimatedRTT = 0.875 · EstimatedRTT + 0.125 · SampleRTT
```

Note that `EstimatedRTT` is a weighted average of the `SampleRTT` values. As discussed in a homework problem at the end of this chapter, this weighted average puts more weight on recent samples than on old samples. This is natural, as the

NOW, RTT is calculated as (Received time for ACK – Time at which the segment was sent)

So, 4.135363 – 4.081502 = 0.053861

4.135713 - 4.081503 = 0.05421

4.135716 - 4.081603 = 0.054113

4.646868 - 4.570353 = 0.076515 will be the estimated RTT values.

(Assuming the sample rtt to be the measured rtt, as given in the question) THE ESTIMATED RTT WILL BE –

segment1 (s1)→ at first it will be the RTT value of the S1 itself = 0.05386 seconds

s2 → 0.875 * 0.053861 + 0.125 * 0.05421 = 0.05390 seconds

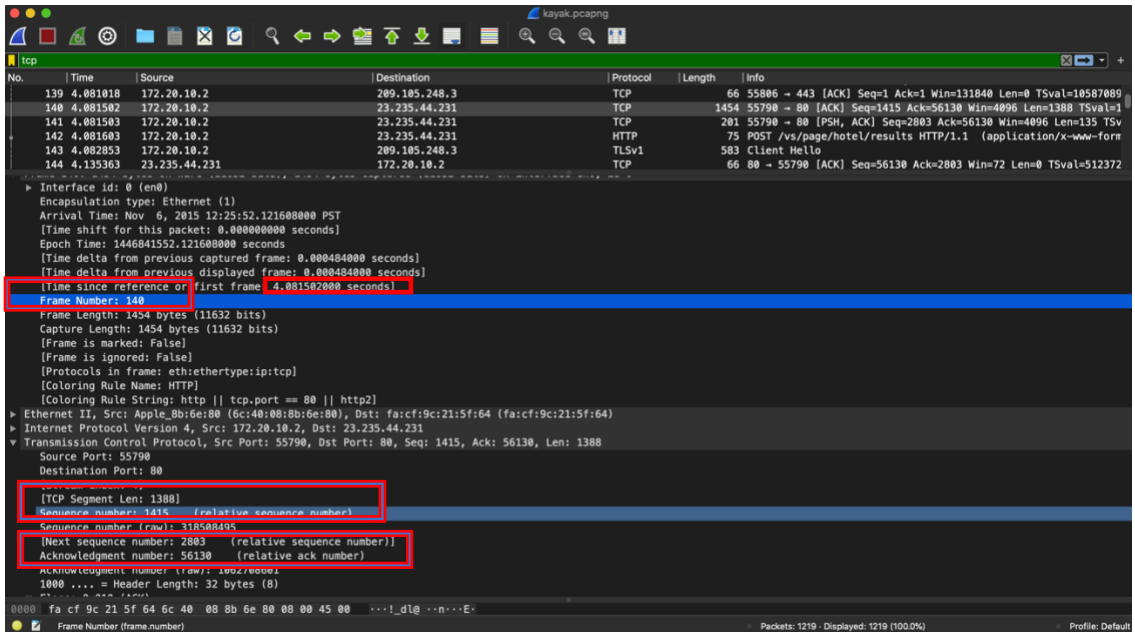
s3 → 0.875 * 0.0539 + 0.125 * 0.054113 = 0.05392 seconds

s4 → 0.875 * 0.05392 + 0.125 * 0.0765 = 0.0567 seconds

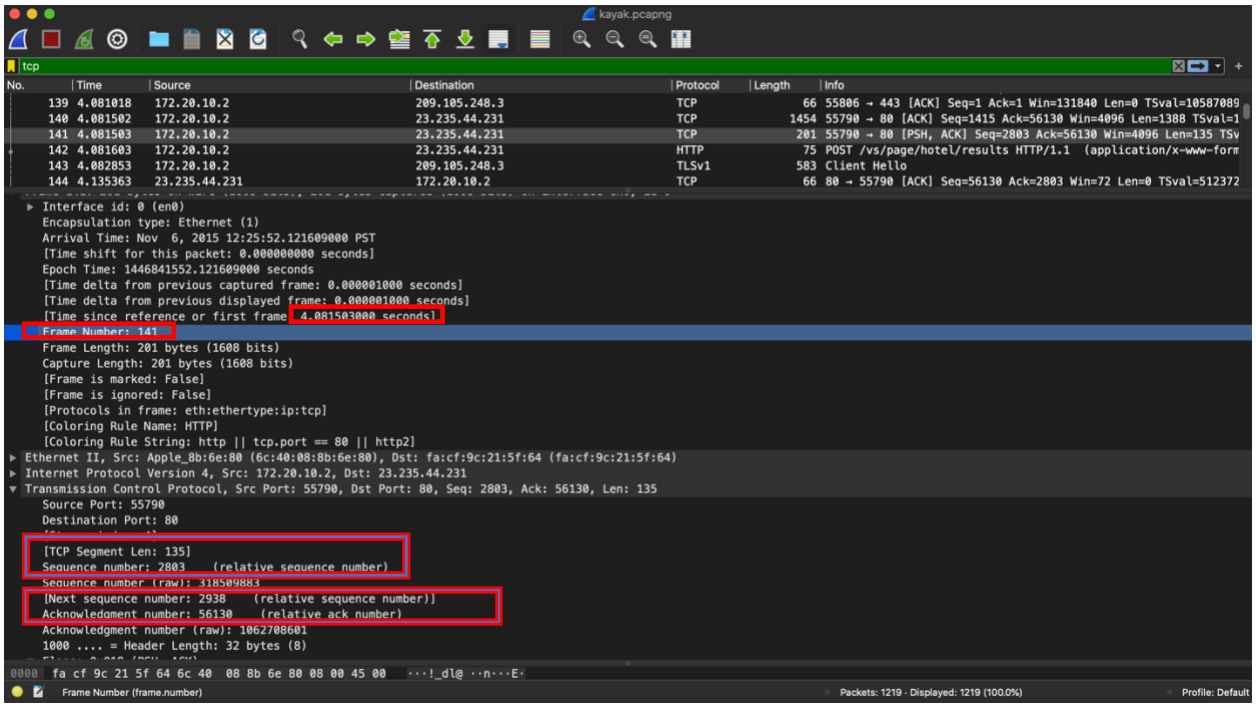
Received time for ACK	RTT (seconds) Value	Estimated RTT Value
4.135363	0.053861	0.05386
4.135713	0.05421	0.05390
4.135716	0.054113	0.05392
4.646868	0.076515	0.0567

Below are the screenshots – (2q)

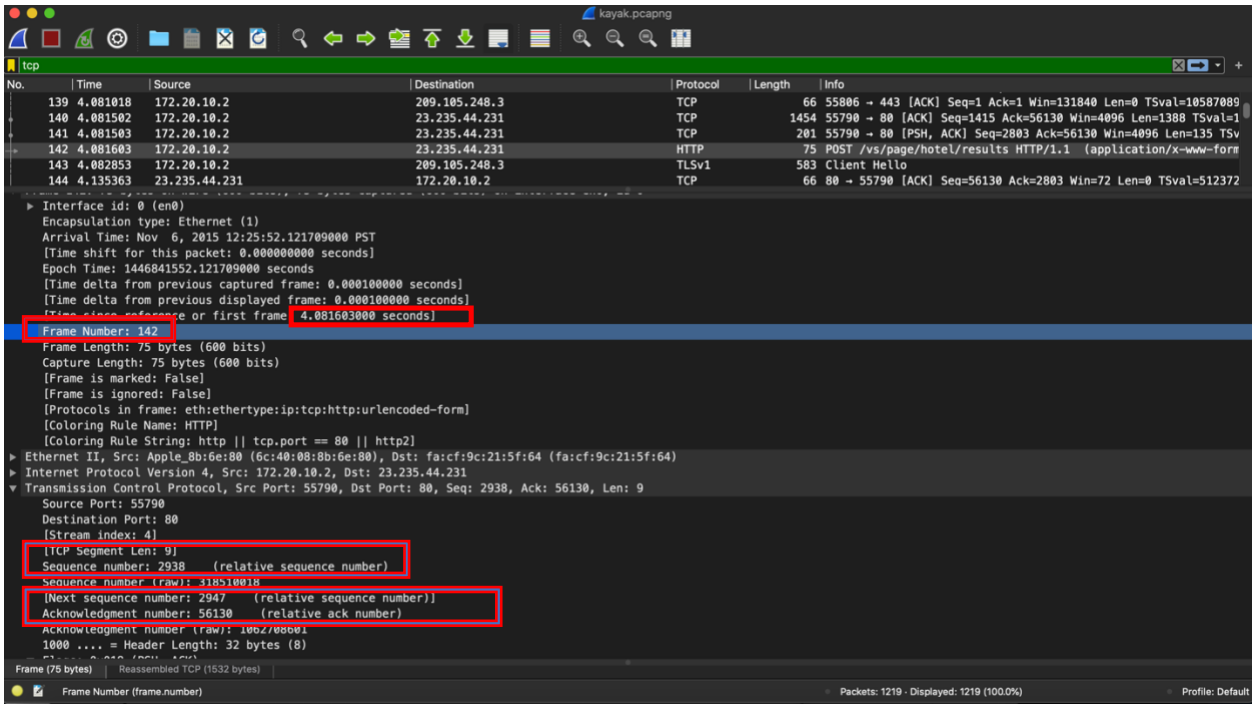
The Segment 140



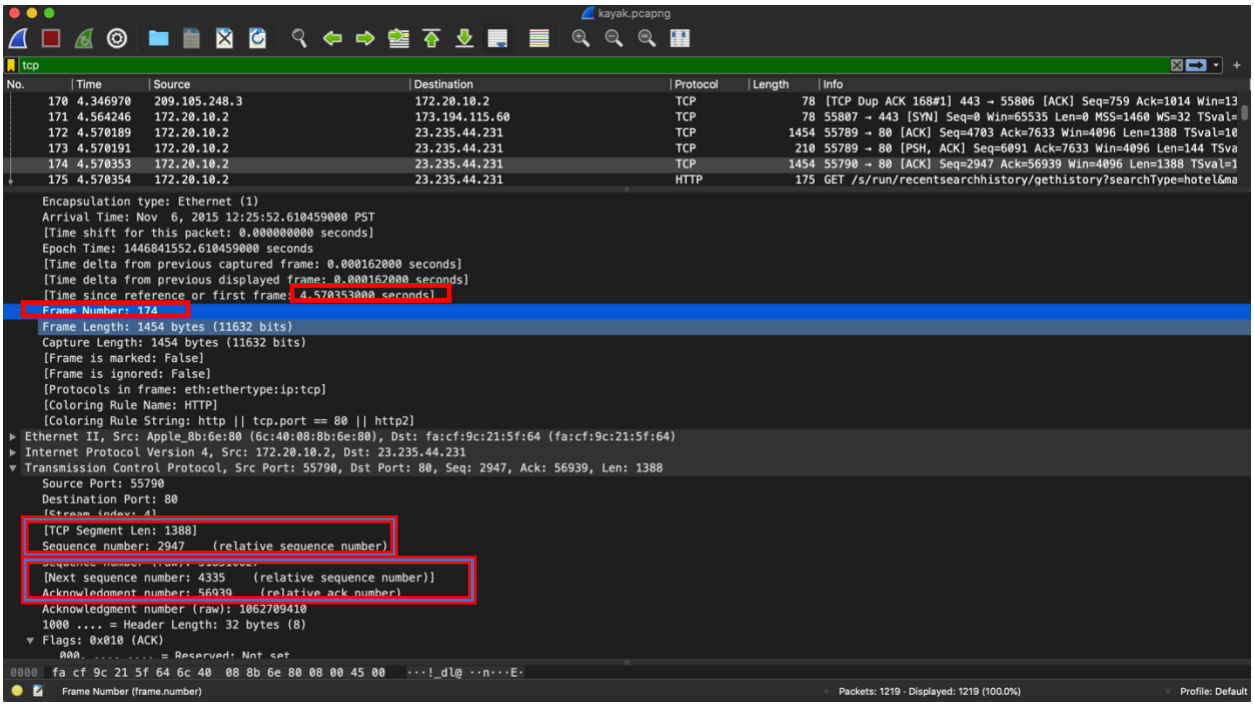
The Segment 141



The Segment 142



The Segment 174



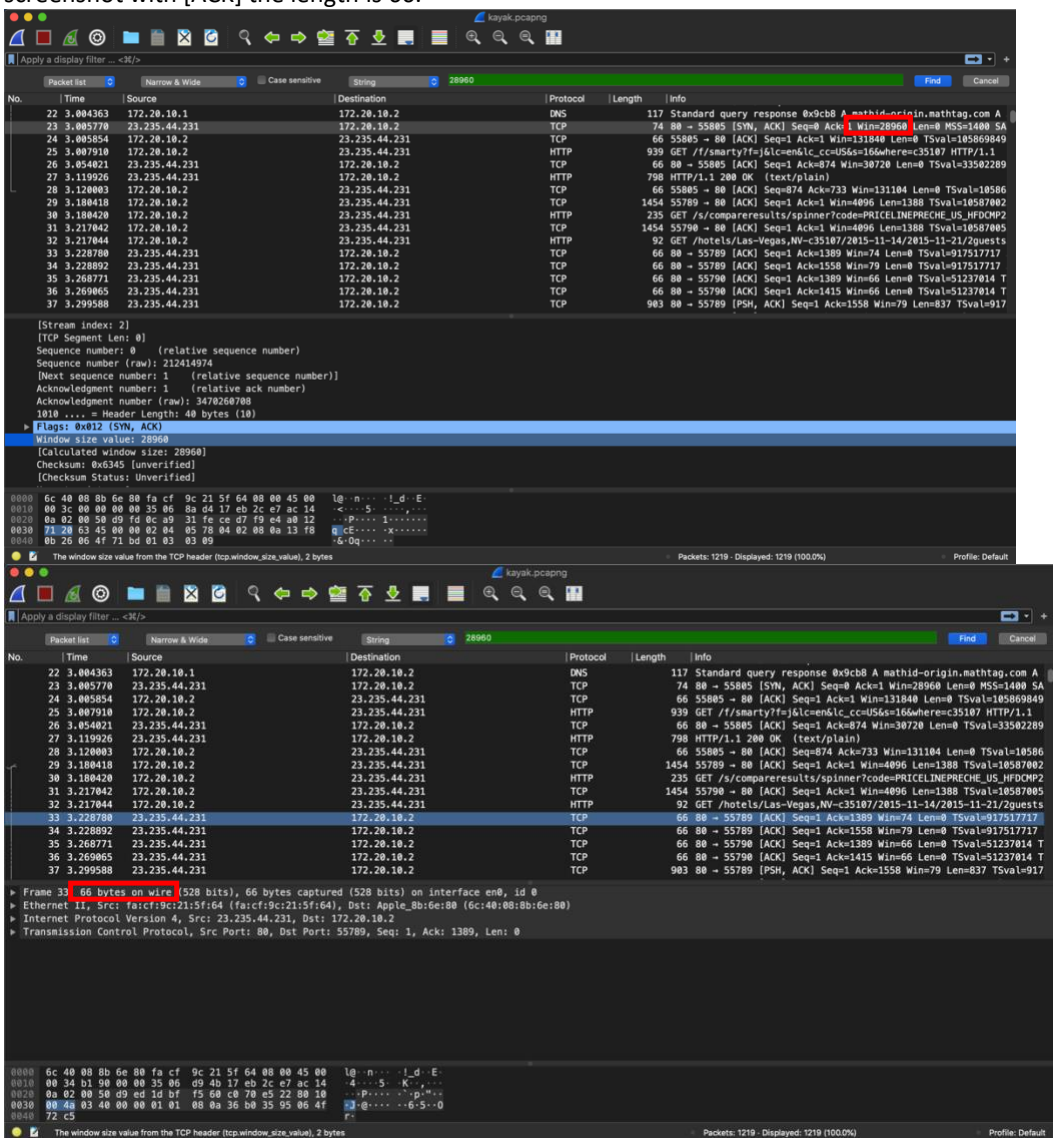
3) What is the length of each of the first four TCP Segments?

(The Segment lengths and numbers have also been highlighted in the above screenshots)

Segment Number	Segment Length
140	1388
141	135
142	9
174	1388

4) What is the minimum amount of available buffer space advertised at the receiver for the entire trace?

Ans) Here as highlighted, in the frame 23 it's the trace packet in the list with [SYN, ACK] respective source and destination address with window size 28960 in total. And with another frame as mentioned in the next screenshot with [ACK] the length is 66.



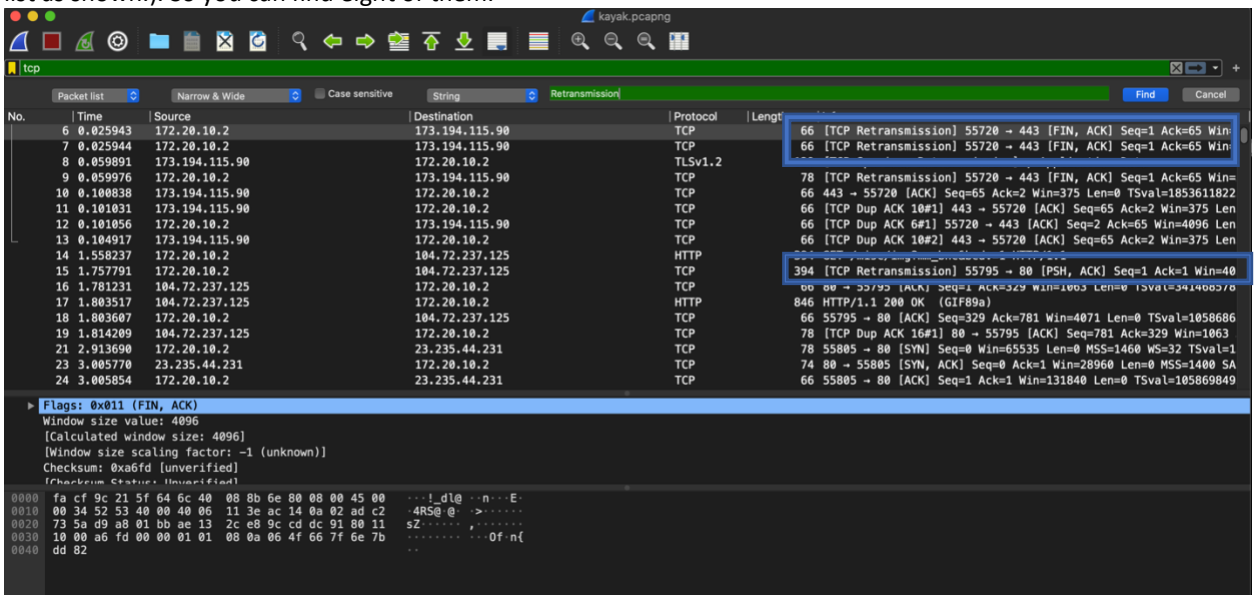
5) Does the lack of receiver buffer space ever throttle the sender?

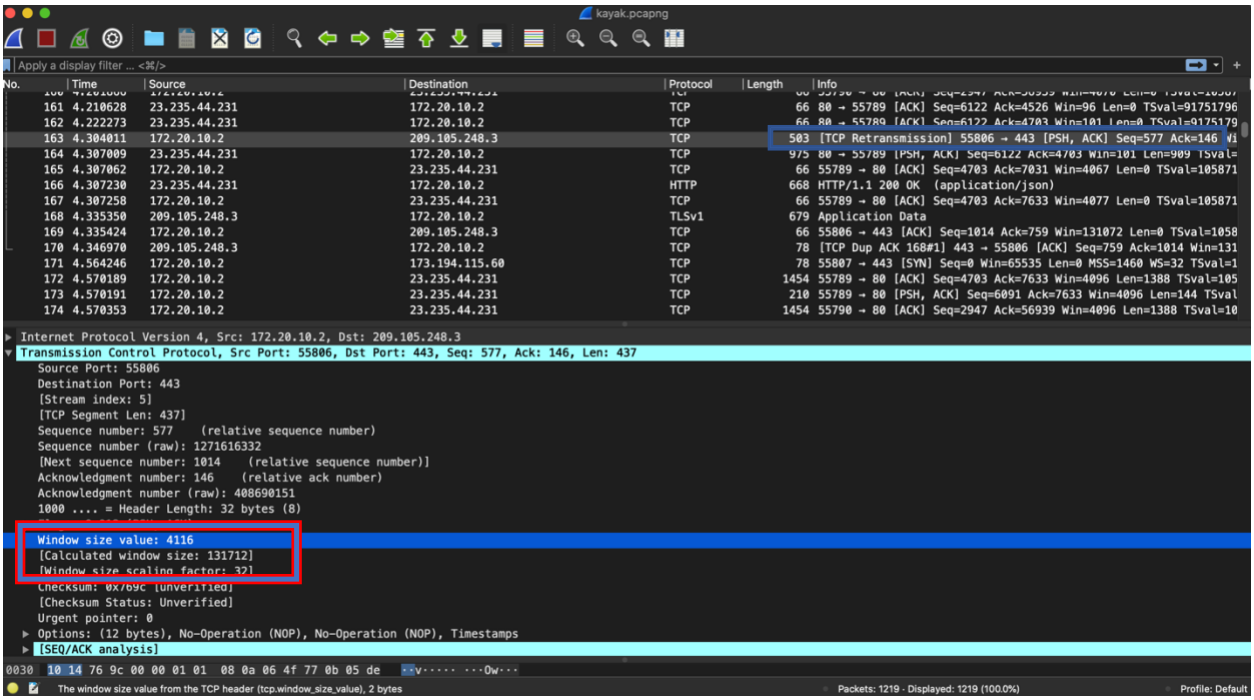
Ans) No, lack of receiver buffer space **will not** ever throttle the sender since it isn't possible to approach to close for the buffer size.

6) Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?

Ans) By checking and counting the retransmitted segments filter in the kayak trace. By which it can be said that YES!, retransmitted segments in the trace file exists, which are eight of them and can be found out by –

(press cmd + f or ctrl + f then type “Retransmission”(case sensitive R capital letter) then search in the packet list as shown.). So you can find eight of them.





7) How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACK-ing every other received segment (see Table 3.2 on page 247 in the text).

The ACK data is given by the difference between the current ACK no. and previous ACK no.

Ans) Segment Number	Acknowledgement number	Acknowledged data in bytes
140	2803	0 (intial)
141	2938	135 (2938 - 2803)
142	2947	9 (2947 - 2938)
174	4335	1388 (4335 – 2947)

8) What is the throughput (bytes transferred per unit time) for the TCP connection (Just consider a single connection)? Think on how to calculate the throughput!

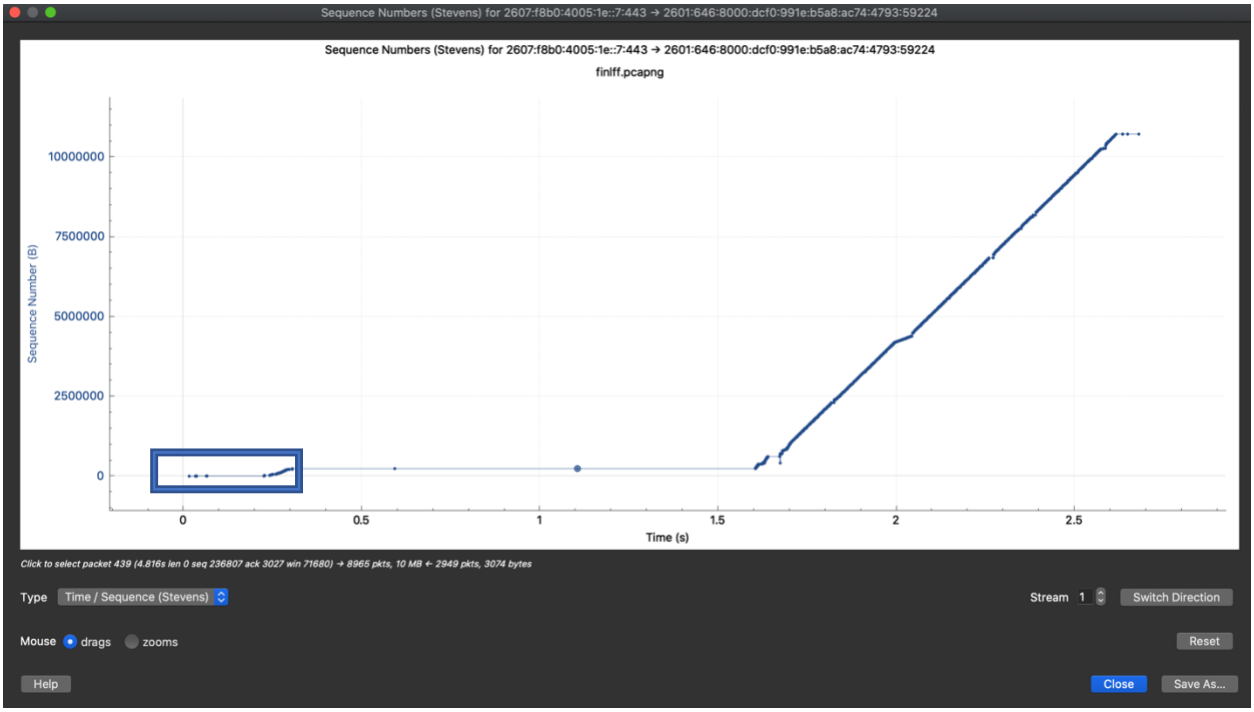
Ans) The ratio of quantity of transferred data to the whole transmission time is termed as the average throughput for a TCP connection. So to find the data that has been transferred it is given by the difference between sequence number of first TCP and the ACK no. of the last ACK. Therefore, it is given by, Throughput = (quantity of transferred data)/(total time) ➔ 2920/0.56536 = 5164.79 kilobytes per second

9) Explain how you calculated this value.

Ans) Initially as mentioned, the data that has been transferred it is given by the difference between sequence number of first TCP, i.e 140 and the ACK no. of the last ACK, i.e 186. So, it will be the difference of 4335 and 1415, which is 2920. Now for the time, it's the subtraction of first byte time sent and last ACK time, i.e difference of 4.64686 and 4.08150, which is 0.56536

Problem Set 4

1) Can you identify where TCP's slow-start phase begins and ends
Ans) The TCP slow start phase begins at 0.016 and ends at 0.225



2) Where congestion avoidance takes over?
Ans) It takes over after when slow start phase finishes, so it'll be 0.225.

3) Comment on ways in which the measured data differs from the idealized behavior of TCP that we've studied in the text.
Ans) The main issue arises when there's a lot of traffic in the network, i.e congestion, so in order to solve this the TCP protocol will have to work on it's properties to reduce the size of the window as stated. But there are also numerous issues like sending the packets again, loss of the packets and delay in time. As TCP is capable of working on it's properties like slow start and controlling the congestion, it's one of the plus point to it. The cause behind the objects setback or slowing down is because of the slow start stage and also the objects size is said to be small. And the communication will be terminated just before the slow start stage end.