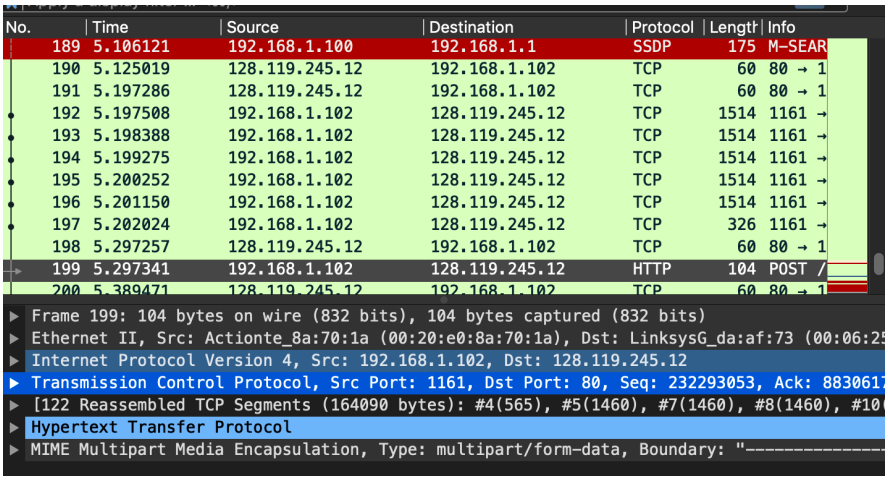
LAB 4

Exercise 1: Understanding TCP using Wireshark

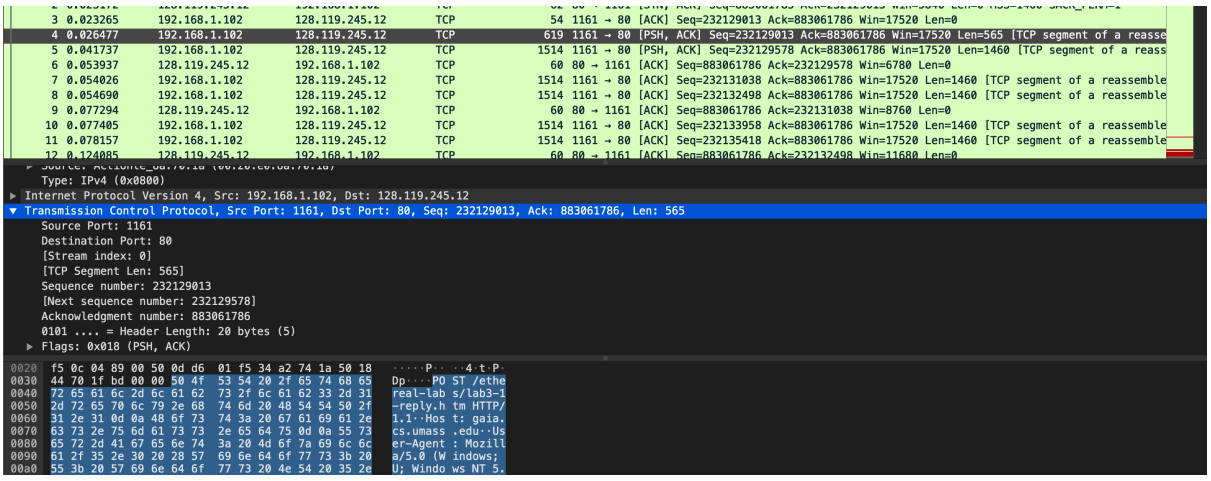
Question 1. What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection? What is the IP address and TCP port number used by the client computer (source) that is transferring the file to gaia.cs.umass.edu?



1. the IP address of gaia.cs.umass.edu: 128.119.245.12. It's sending and receiving TCP segments at port 80.

2. the IP address used by client is 192.168.1.102, port no. 1161

Question 2. What is the sequence number of the TCP segment containing the HTTP POST command? Note that in order to find the POST command, you'll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with a "POST" within its DATA field.



The 4th segment contain the HTTP POST command, the sequence number of this TCP segment is 232129013

Question 3. Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection. What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST) sent from the client to the web server (Do not consider the ACKs received from the server as part of these six segments)? 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 relevant parts of Section 3.5 or lecture slides) after the receipt of each ACK? Assume that the initial value of EstimatedRTT is equal to the measured RTT (SampleRTT) for the first segment, and then is computed using the EstimatedRTT equation for all subsequent segments. Set alpha to 0.125.

Question 4. What is the length of each of the first six TCP segments?

Sequence numbers the first six segments:

Segment No.	ACK No.	Sequence No.	Sent time	ACK received time	RTT(seconds)	length
4	6	232129013	0.026477	0.053937	0.02746	565
5	9	232129578	0.041737	0.077294	0.035557	1460
7	12	232131038	0.054026	0.124085	0.070059	1460
8	14	232132498	0.054690	0.169118	0.11443	1460
10	15	232133958	0.077405	0.217299	0.13989	1460
11	16	232135418	0.078157	0.267802	0.18964	1460

EstimatedRTT = (1 - a)* RTT + a * SampleRTT

segment 4:
EstimatedRTT = RTT for Segment 4 = 0.02746 s

segment 5:
EstimatedRTT = 0.875 * 0.02746 + 0.125 * 0.035557 = 0.0285 s

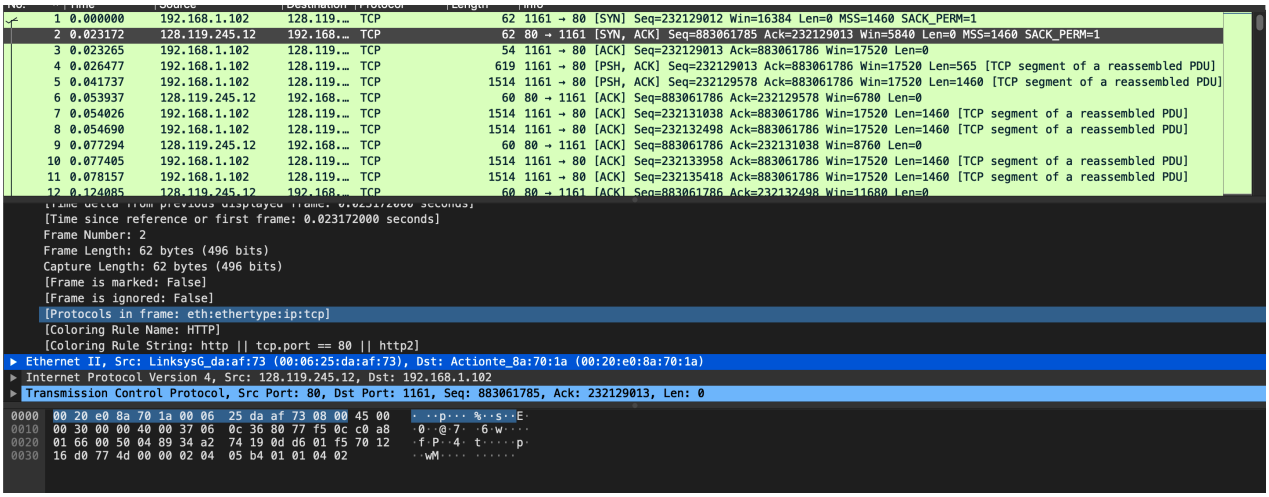
segment 7:
EstimatedRTT = 0.875 * 0.0285 + 0.125 * 0.070059 = 0.0337 s

segment 8:
EstimatedRTT = 0.875 * 0.0337+ 0.125 * 0.11443 = 0.0438 s

segment 10:
EstimatedRTT = 0.875 * 0.0438 + 0.125 * 0.13989 = 0.0558 s

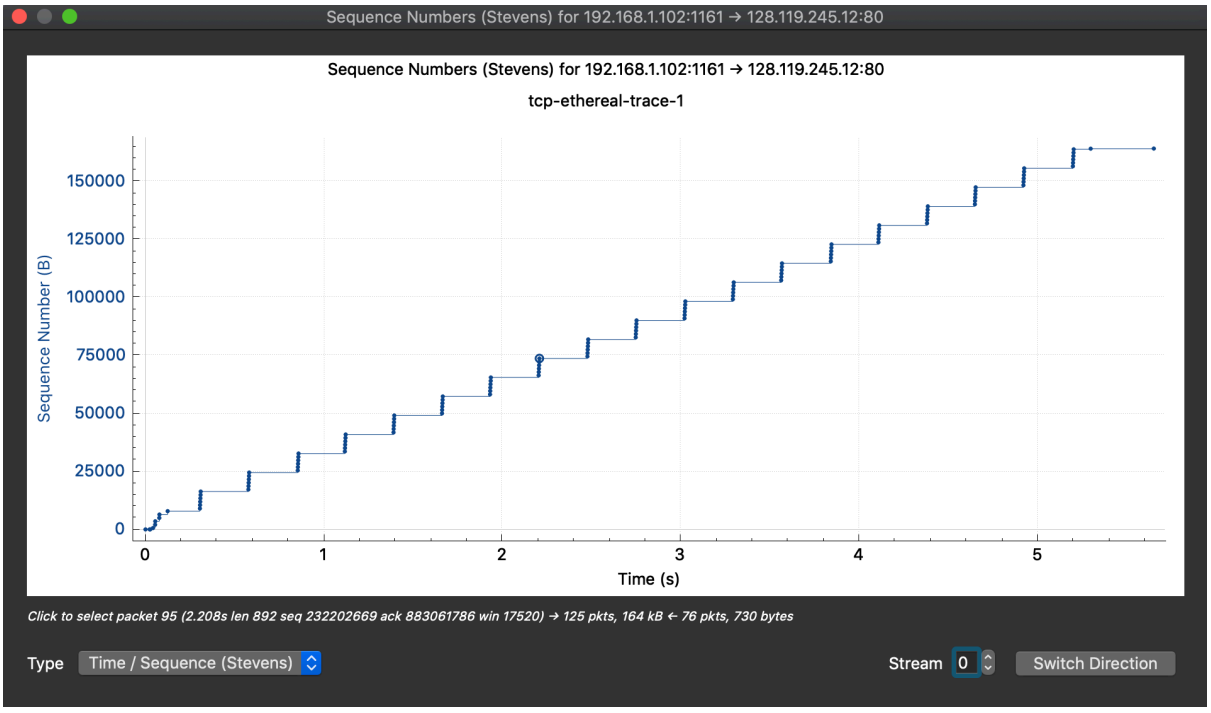
segment 11:
EstimatedRTT = 0.875 * 0.0558 + 0.125 * 0 .18964 = 0.0725 s

Question 5. What is the minimum amount of available buffer space advertised at the receiver for the entire trace? Does the lack of receiver buffer space ever throttle the sender?



The minimum amount of available buffer space is 5840 bytes. That is shown in the first acknowledge in segment 2. The sender has never been throttled.

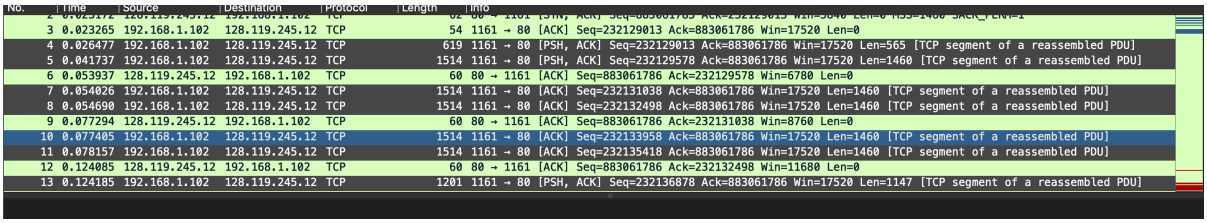
Question 6. Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?



There is no retransmitted segments in the trace file. It can be known from the Time-Sequence-Graph (Stevens) of this trace. The sequence numbers increase monotonically with respect to time, which means that no retransmission happened. Otherwise, there will be some segments with lower sequence number than its neighbouring segment.

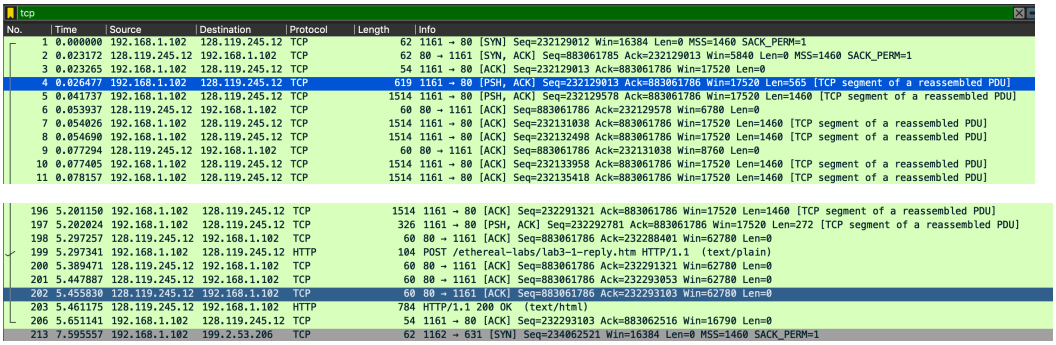
Question 7. How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACKing every other received segment (recall the discussion about delayed acks from the lecture notes or Section 3.5 of the text).

It can be known from the difference between two sequence numbers of consecutives ACKs. Also, the difference is indicated by the Len header in ACK segment. By inspecting the amount of f acknowledged data by each ACK, I notice that the segment of NO.60 acknowledged data with 2920 bytes. That is 2920 = 1460 * 2 bytes.
It means that the receiver is acking other received segment.



ACK No.	Len	Sequence NO.
4	565	232129013
5	1460	232129578
7	1460	232131038
8	1460	232132498
10	1460	232133958
11	1460	232135418
13	1161	232136878
…		

Question 8. What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.



To compute the throughput for the TCP connection, we need to extract a certain period of time and acquire the total amount of bytes transmitted and the difference of timestamps

between start and end. Since no retransmission occur in the trace file, simply trace the sequence number of first ACK and last ACK.

Amount of bytes:
232293103 – 232129013 = 164090 bytes
total transmission time:
5.455830 – 0.026477 = 5.4294 s
throughput:
164090 / 5.4294 = 30222.5 byte/s

Exercise 2: TCP Connection Management

Consider the following TCP transaction between a client (10.9.16.201) and a server (10.99.6.175).

No	Source IP	Destination IP	Protocol	Info
295	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [SYN] Seq=2818463618 win=8192 MSS=1460
296	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [SYN, ACK] Seq=1247095790 Ack=2818463619 win=262144 MSS=1460
297	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [ACK] Seq=2818463619 Ack=1247095791 win=65535
298	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [PSH, ACK] Seq=2818463619 Ack=1247095791 win=65535
301	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [ACK] Seq=1247095791 Ack=2818463652 win=262096
302	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [PSH, ACK] Seq=1247095791 Ack=2818463652 win=262144
303	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [ACK] Seq=2818463652 Ack=1247095831 win=65535
304	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [FIN, ACK] Seq=2818463652 Ack=1247095831 win=65535
305	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [FIN, ACK] Seq=1247095831 Ack=2818463652 win=262144
306	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [ACK] Seq=2818463652 Ack=1247095832 win=65535
308	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [ACK] Seq=1247095831 Ack=2818463653 win=262144

Question 1 . What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and server?

the sequence number: 281846318

Question 2. What is the sequence number of the SYNACK segment sent by the server to the client computer in reply to the SYN? What is the value of the Acknowledgement field in the SYNACK segment? How did the server determine that value?

the sequence number: 1247095790. The value of Acknowledgement field: 281846319. That is the value of sequence number of TCP SYN segment from client plus one.

Question 3 . What is the sequence number of the ACK segment sent by the client computer in response to the SYNACK? What is the value of the Acknowledgment field in this ACK segment? Does this segment contain any data?

the sequence number: 2818463619
the value of the Acknowledgement field: 1247095791
No data contained in this segment. It is just the last segment of the three way handshake

Question 4 . Who has done the active close? client or the server? how you have determined this? What type of closure has been performed? 3 Segment (FIN/FINACK/ACK), 4 Segment (FIN/ACK/FIN/ACK) or Simultaneous close?

this is actually a simultaneous close. Both the client and the server initialize the Fin without receiving FIN from other side. By inspecting, I notice that the sequence number and ACK number is somewhat unusual. In the 305th and 306th records, the ACK has not increase by 1 in the FIN that has been acknowledged. So the server has not received the closure segment when it send the 306th segment. So, this is actually a simultaneous close.

Question 5. How many data bytes have been transferred from the client to the server and from the server to the client during the whole duration of the connection? What relationship does this have with the Initial Sequence Number and the final ACK received from the other side?

Consider the following TCP transaction between a client (10.9.16.201) and a server (10.99.6.175).

No	Source IP	Destination IP	Protocol	Info
295	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [SYN] Seq=2818463618 win=8192 MSS=1460
296	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [SYN, ACK] Seq=1247095790 Ack=2818463619 win=262144 MSS=1460
297	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [ACK] Seq=2818463619 Ack=1247095791 win=65535
298	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [PSH, ACK] Seq=2818463619 Ack=1247095791 win=65535
301	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [ACK] Seq=1247095791 Ack=2818463652 win=262096
302	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [PSH, ACK] Seq=1247095791 Ack=2818463652 win=262144
303	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [ACK] Seq=2818463652 Ack=1247095831 win=65535
304	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [FIN, ACK] Seq=2818463652 Ack=1247095831 win=65535
305	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [FIN, ACK] Seq=1247095831 Ack=2818463652 win=262144
306	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [ACK] Seq=2818463652 Ack=1247095832 win=65535
308	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [ACK] Seq=1247095831 Ack=2818463653 win=262144

computing by tracing the sequence number
from client to server:
2818463653 – 2818463618 – 2 = 33 bytes
from server to client:
1247095832 – 1247095790 – 2 = 40 bytes

relation: the difference between initial sequence number and final ACK from the other end indicate the total amount of bytes transmitted during the connection