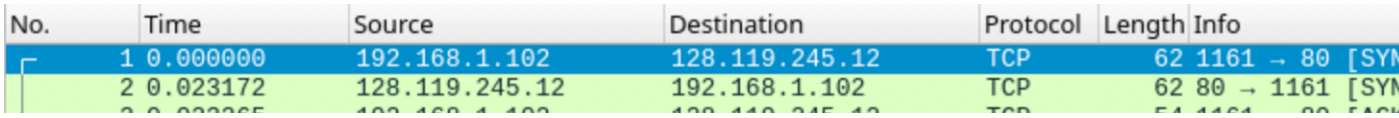
Question 1 **. What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection? What are the IP address and TCP port numbers used by the client computer (source) that is transferring the file to gaia.cs.umass.edu?**

In TCP transport layer we set up the connection first, which shows the source and the destination in No.1.

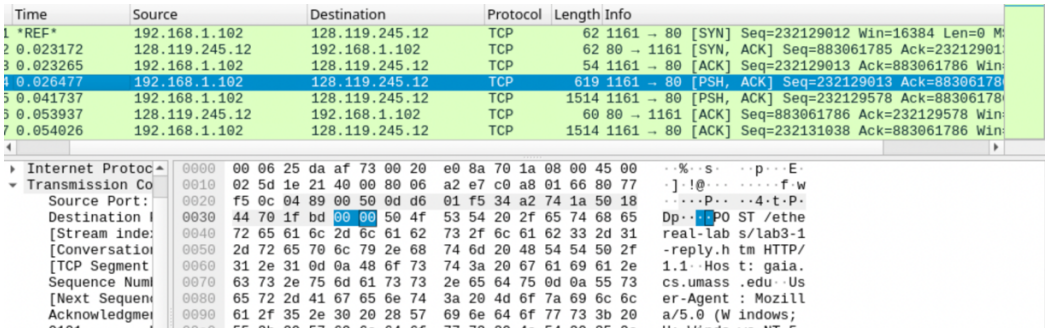
The IP address of gaia.cs.umass.edu is 128.119.245.12, and the port number is 80.
The IP address of the client computer is 192.168.1.102 and its port number is 1161.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.102	128.119.245.12	TCP	62	1161 → 80 [SYN
2	0.023172	128.119.245.12	192.168.1.102	TCP	62	80 → 1161 [SYN

Question 2. **What is the sequence number of the TCP segment containing the HTTP POST command? Note that 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.**

As first three lines are building the connection (three ways handshake), we check No.4 which request the data.
The sequence number is 232129013.



Time	Source	Destination	Protocol	Length	Info
1 *REF*	192.168.1.102	128.119.245.12	TCP	62	1161 → 80 [SYN] Seq=232129012 Win=16384 Len=0 M
2 0.023172	128.119.245.12	192.168.1.102	TCP	62	80 → 1161 [SYN, ACK] Seq=883061785 Ack=23212901
3 0.023265	192.168.1.102	128.119.245.12	TCP	54	1161 → 80 [ACK] Seq=232129013 Ack=883061786 Win
4 0.026477	192.168.1.102	128.119.245.12	TCP	619	1161 → 80 [PSH, ACK] Seq=232129013 Ack=88306178
5 0.041737	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [PSH, ACK] Seq=232129578 Ack=88306178
6 0.053937	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232129578 Win
7 0.054026	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=232131038 Ack=883061786 Win

```
▸ Internet Protoc
▾ Transmission Co
    Source Port:
    Destination
    [Stream inde
    [Conversatio
    [TCP Segment
    Sequence Numb
    [Next Sequen
    Acknowledgme
```

```
0000  00 06 25 da af 73 00 20  e0 8a 70 1a 08 00 45 00   ··%··s·  ··p···E·
0010  02 5d 1e 21 40 00 80 06  a2 e7 c0 a8 01 66 80 77   ·]·!@··· ·····f·w
0020  f5 0c 04 89 00 50 0d d6  01 f5 34 a2 74 1a 50 18   ·····P·· ··4·t·P·
0030  44 70 1f bd 00 00 50 4f  53 54 20 2f 65 74 68 65   Dp····PO ST /ethe
0040  72 65 61 6c 2d 6c 61 62  73 2f 6c 61 62 33 2d 31   real-lab s/lab3-1
0050  2d 72 65 70 6c 79 2e 68  74 6d 20 48 54 54 50 2f   -reply.h tm HTTP/
0060  31 2e 31 0d 0a 48 6f 73  74 3a 20 67 61 69 61 2e   1.1··Hos t: gaia.
0070  63 73 2e 75 6d 61 73 73  2e 65 64 75 0d 0a 55 73   cs.umass .edu··Us
0080  65 72 2d 41 67 65 6e 74  3a 20 4d 6f 7a 69 6c 6c   er-Agent : Mozill
0090  61 2f 35 2e 30 20 28 57  69 6e 64 6f 77 73 3b 20   a/5.0 (W indows;
```

Question 3. **Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection.**

a) **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 webserver (Do not consider the ACKs received from the server as part of these six segments)?**

b) **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?**

c) **What is the *EstimatedRTT* value (see relevant parts of Section 3.5 or lecture slides) after receiving 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.

EstimatedRTT = 0.875*EstimatedRTT+0.125*sampleRTT

Question 4. **What is the length of each of the first six TCP segments? (same six segments as Question 3)**

Segment (determine by source and destination IP address)	Length (find in header)	Seq. number	Sent time	Received time (find in ACK -> length+seq)	Sample RTT (Received time – sent time)	Estimated RTT
1 (no.4)	565	232129013	0.026477	0.05394	0.02746	0.02746 (same as sample RTT as it's the first column)
2 (no.5)	1460	232129578	0.041737	0.077294 (ACK= no.9)	0.03556	0.0284721 3
3 (no.7)	1460	232131038	0.054690	0.0124085 (ACK= no.12)	0.070059	0.0327848 75
4 (no.8)	1480	232132498	0.054690	0.169118 (ACK=no.14)	0.114428	0.038331
5 (no.10)	1480	232133958	0.077405	0.217299 (ACK=no.15)	0.139894	0.04151425

6 (no.11)	1480	232135418	0.078157	0.267802 (ACK=no.16)	0.189645	0.047733125

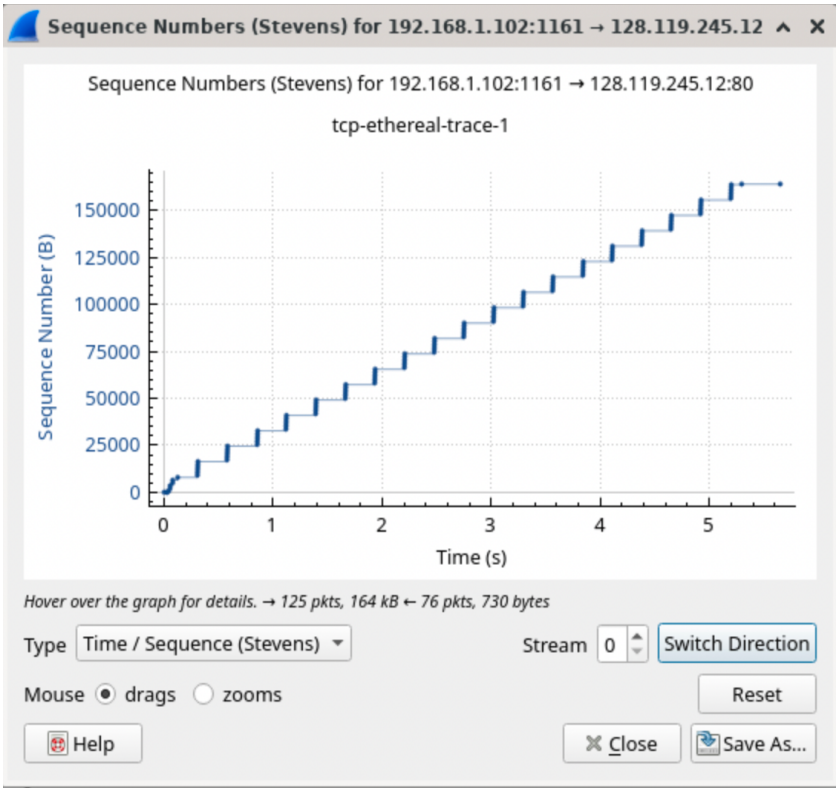
Question 5. **What is the minimum amount of available buffer space advertised at the receiver for the entire trace? Does the lack of receiver buffer space ever throttle the sender?**

Check SYN ACK packet (no. 2nd) -> window (maximum value 5840). As the size is increasing, the lack of receiver buffer space does jot throttle the sender.

Question 6. Are there any retransmitted segments in the trace file? To answer this question, what did you check for (in the trace)?

(1. Check is there same sequence number + same content OR wireshark->statistics->TCP stream graph-> time sequence sevens)

The sequence numbers increase step by step in TCP steam graph which means we did not retransmit the graph.



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)?

We observe the packet no.53- no.62 and there are 6 packets with 4 ACK.

No.53–No.58: 6 packets (we can't fine the ACK for no54 & no.55) –> which no.54 — no.56 combine their ACK in no.60

No59–No. 62: 4 ACK

53	1.117333	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=232162601 Ack=883061786 Win=
54	1.118133	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=232164061 Ack=883061786 Win=
55	1.119029	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=232165521 Ack=883061786 Win=
56	1.119858	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=232166981 Ack=883061786 Win=
57	1.120902	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=232168441 Ack=883061786 Win=
58	1.121891	192.168.1.102	128.119.245.12	TCP	946 1161 → 80 [PSH, ACK] Seq=232169901 Ack=883061786
59	1.200421	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=883061786 Ack=232164061 Win=
60	1.265026	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=883061786 Ack=232166981 Win=
61	1.362074	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=883061786 Ack=232169901 Win=
62	1.389886	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=883061786 Ack=232170793 Win=

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

Throughput = (Total amount of data)/ (total transmission time) = (#202_seq - #4_seq – 1)/(#202_ack_time - #4_seq_time)

$$(232293103 - 232129013 - 1) \div (5.45583 - 0.026477) = 30222.5697979$$

Exercise 2

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 is 2818463618.

***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?**

As it is sent by the server to the client (check the ip), which the sequence number is 1247095790.

The ACK is 2818463619.

***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 of the ACK segment CK is 2818463619. The ACK segment is 1247095791. This segment does not contain any data because the sequence number is same as segment #298.

***Question 4* . Who has done the active close? Is it the 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?**

Both client and server done the active close. As both of them sent [FIN, ACK] flags to each other without receiving ACK flag. In addition, the sequence number in segment #304 is equal to the value of the acknowledgement field in segment #305. The type of closure is 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?**

client to server = 2818463653 - 2818463618 - 2(SYN & FIN) = 33 bytes

server to client = 1247095831 - 1247095790 - 2(SYN & FIN) = 40 bytes

As SYN and FIN flags increase 1 at ACK without containing any data.
This relationship can be used to calculate the data transfer.