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? To answer this question, it's probably easiest to select an HTTP message and explore the details of the TCP packet used to carry this HTTP message, using the "details of the selected packet header window" (refer to Figure 2 in the "Getting Started with Wireshark" Lab if you're uncertain about the Wireshark windows.

IP Address: 192.168.1.102

TCP Port Number(client computer): 1161

```
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 SACK_PERM=1
```

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?

IP Address: 128.119.245.12

TCP Port Number(gaia.cs.umass.edu): 80

```
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 SACK_PERM=1
```

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?

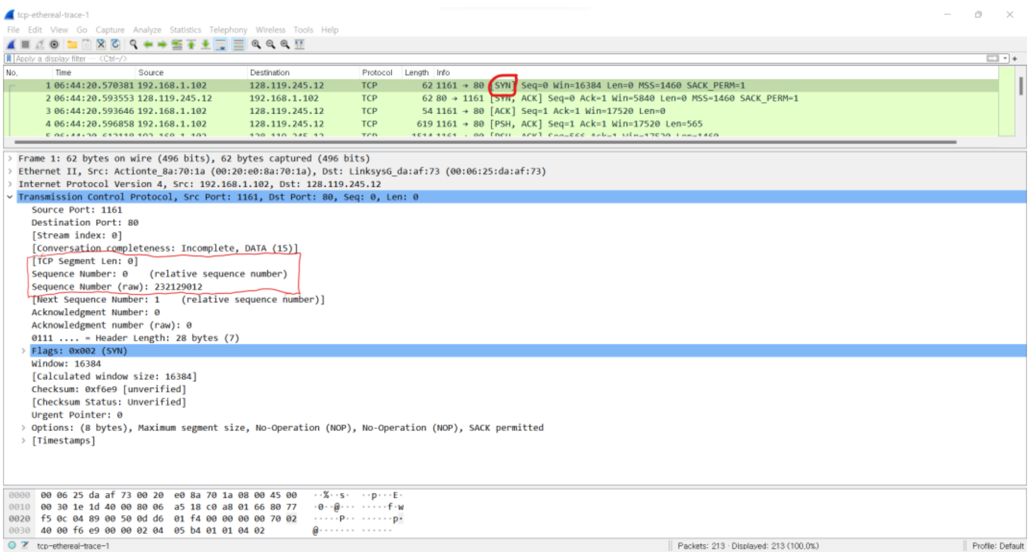
IP Address: 10.248.105.226

TCP Port Number: 55657

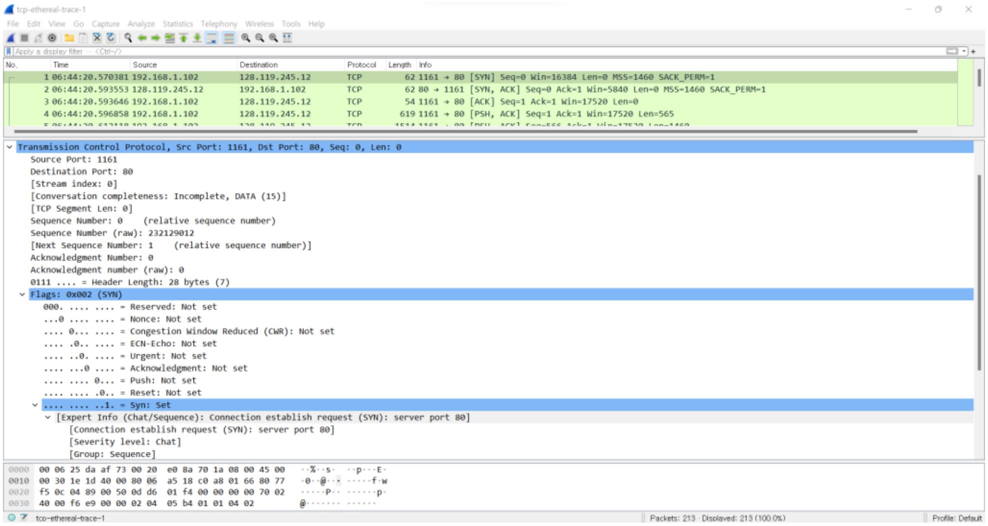
```
3 1.375225 10.248.105.226   128.119.245.12   TCP    54 55657 → 80 [FIN, ACK] Seq=1 Ack=1 Win=513 Len=0
4 1.376648 10.248.105.226   128.119.245.12   TCP    66 55659 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
5 1.387929 10.248.105.226   128.119.245.12   TCP   958 55656 → 80 [PSH, ACK] Seq=1 Ack=1 Win=514 Len=904
6 1.454292 128.119.245.12   10.248.105.226   TCP    66 80 → 55659 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1386 SACK_PERM=1 WS=128
7 1.454435 10.248.105.226   128.119.245.12   TCP    54 55659 → 80 [ACK] Seq=1 Ack=1 Win=131584 Len=0
8 1.454542 128.119.245.12   10.248.105.226   TCP    56 80 → 55657 [ACK] Seq=1 Ack=2 Win=238 Len=0
```

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?

Sequence Number of TCP SYN segment: 0 (RAW: 232129012)



In the segment, there is a flag which identifies the segment as a SYN segment.



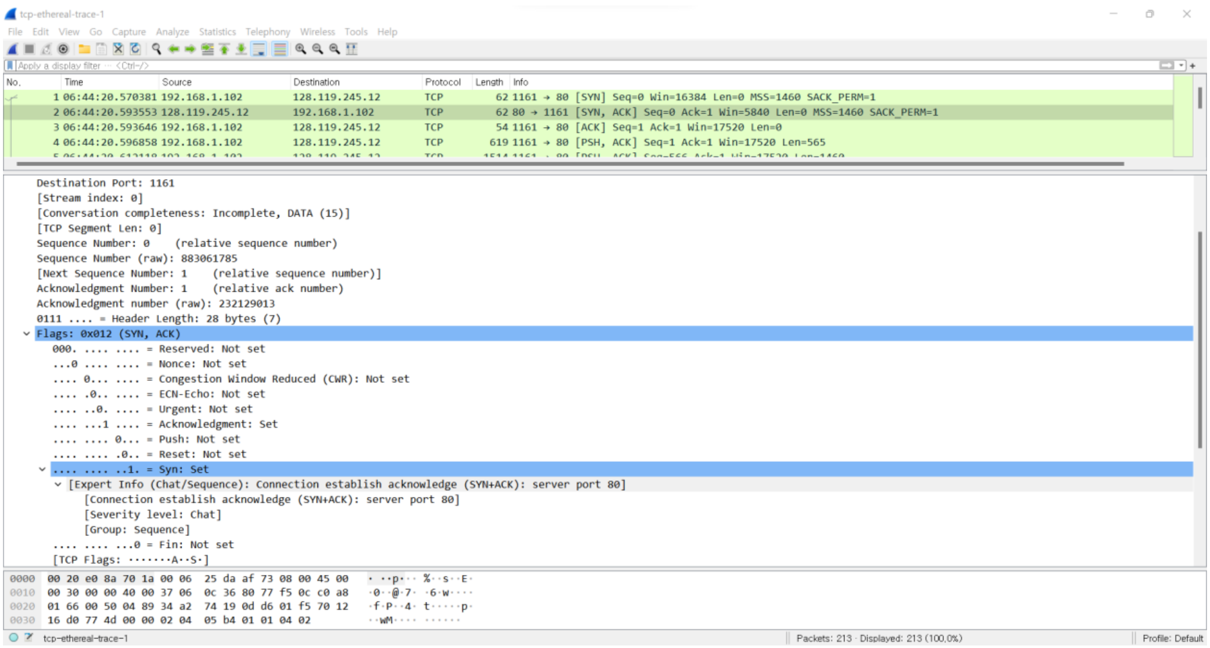
5. What is the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN? What is the value of the Acknowledgement field in the SYNACK segment? How did gaia.cs.umass.edu determine that value? What is it in the segment that identifies the segment as a SYNACK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

The sequence number of the SYNACK segment sent by gaia.cs.umass.edu is 0(raw: 883061785)

The value of the Acknowledgement field in the SYNACK segment is 1(raw: 232129013)

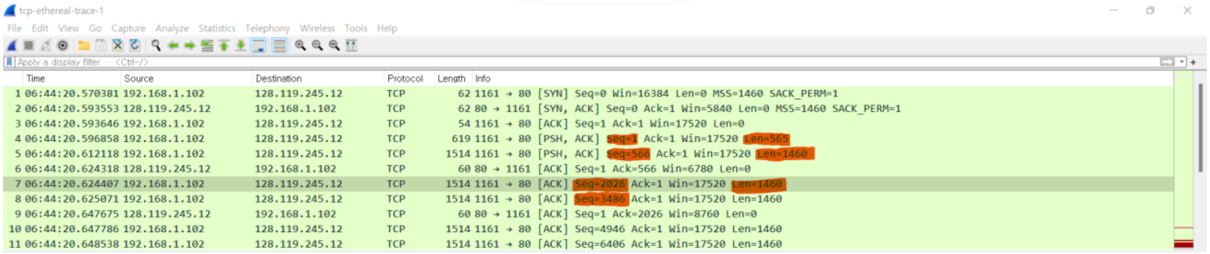
'gaia.cs.umass.edu' determined that value by using flags represented in hexadecimals.

In SYNACK segment, there are flags which identifies the segment as a SYNACK segment.



6. 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 sequence number of the TCP segment containing HTTP POST is the cumulative sum of the length of previous segments.



No.	Time	Source	Destination	Protocol	Length	Info
1	06:44:20.570381	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	06:44:20.593553	128.119.245.12	192.168.1.102	TCP	62	80 → 1161 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 SACK_PERM=1
3	06:44:20.593646	192.168.1.102	128.119.245.12	TCP	54	1161 → 80 [ACK] Seq=1 Ack=1 Win=17520 Len=0
4	06:44:20.596858	192.168.1.102	128.119.245.12	TCP	619	1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len=565
5	06:44:20.612118	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [PSH, ACK] Seq=566 Ack=1 Win=17520 Len=1460
6	06:44:20.624318	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=566 Win=6780 Len=0
7	06:44:20.624407	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=2026 Ack=1 Win=17520 Len=1460
8	06:44:20.625071	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460
9	06:44:20.647675	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=2026 Win=8760 Len=0
10	06:44:20.647786	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=4946 Ack=1 Win=17520 Len=1460
11	06:44:20.648538	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460

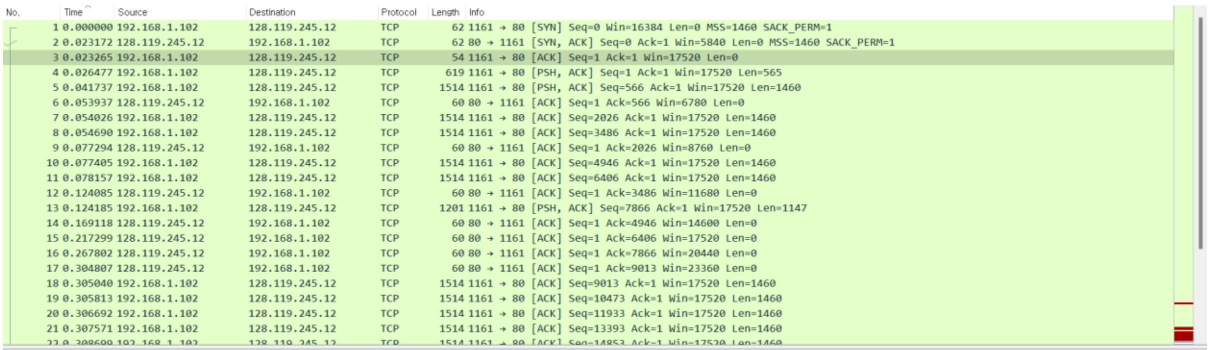
7. Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection. What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST)?

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.

Note: Wireshark has a nice feature that allows you to plot the RTT for each of the TCP segments sent. Select a TCP segment in the "listing of captured packets" window that is being sent from the client to the gaia.cs.umass.edu server. Then select: Statistics->TCP Stream Graph- >Round Trip Time Graph.

The sequence numbers of first six segments in the TCP connection are 0, 1, 1, 566, 2026, 3486.



No.	Time	Source	Destination	Protocol	Length	Info
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 SACK_PERM=1
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
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
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
8	0.054690	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460
9	0.077294	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=2026 Win=8760 Len=0
10	0.077405	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=4946 Ack=1 Win=17520 Len=1460
11	0.078157	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460
12	0.124085	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=3486 Win=11680 Len=0
13	0.124185	192.168.1.102	128.119.245.12	TCP	1201	1161 → 80 [PSH, ACK] Seq=7866 Ack=1 Win=17520 Len=1147
14	0.169118	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=4946 Win=14600 Len=0
15	0.217299	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=6406 Win=17520 Len=0
16	0.267802	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=7866 Win=20440 Len=0
17	0.304807	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=9013 Win=23360 Len=0
18	0.305040	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=9013 Ack=1 Win=17520 Len=1460
19	0.305813	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=10473 Ack=1 Win=17520 Len=1460
20	0.306692	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=11933 Ack=1 Win=17520 Len=1460
21	0.307571	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=13393 Ack=1 Win=17520 Len=1460

(Note: Time Display is Seconds since Beginning of Capture)

<Sequence Number: Time-segment sent>

0: 0s

1: 0.023172

1: 0.026477

566: 0.041737

2026: 0.054026

3486: 0.054690

<Sequence Number: Time-ACK Received>

0: 0.023172s

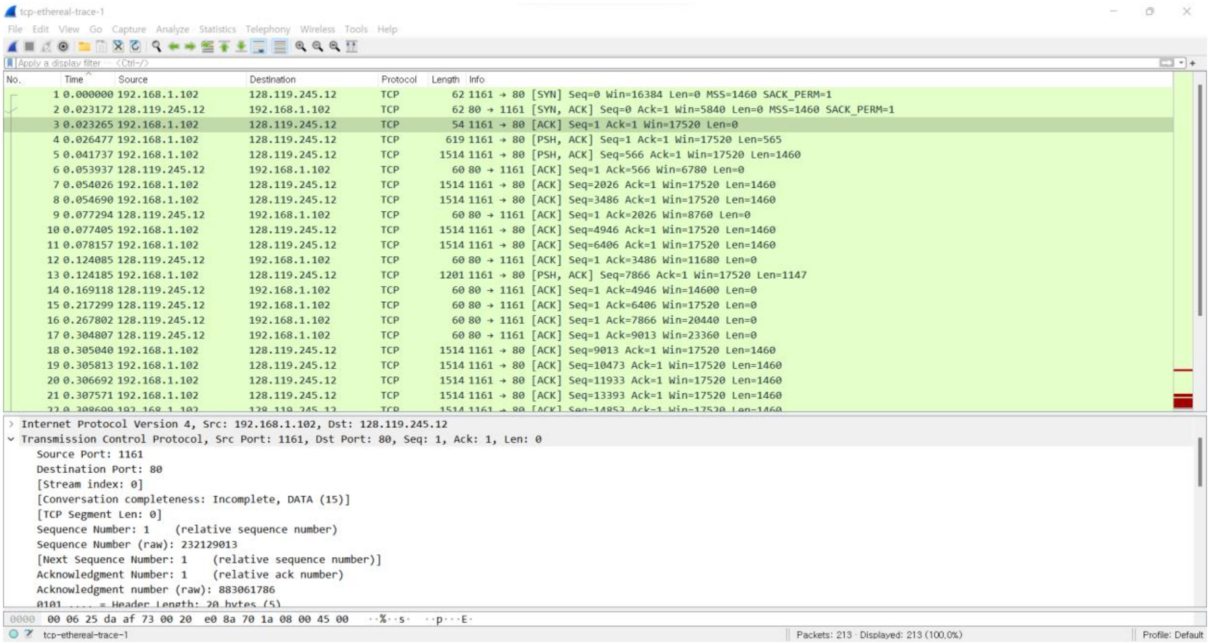
1: -

1: -

566: 0.053937s

2026: 0.077294s

3486: 0.124085s



<Sequence Number : RTT>

0: 0.0023172s

1: -

1: -

566: 0.0122s

2026: 0.023268s

3486: 0.069395s

<Sequence Number: Estimated RTT> (Note: $\alpha = 0.125$)

0: 0.0012172s

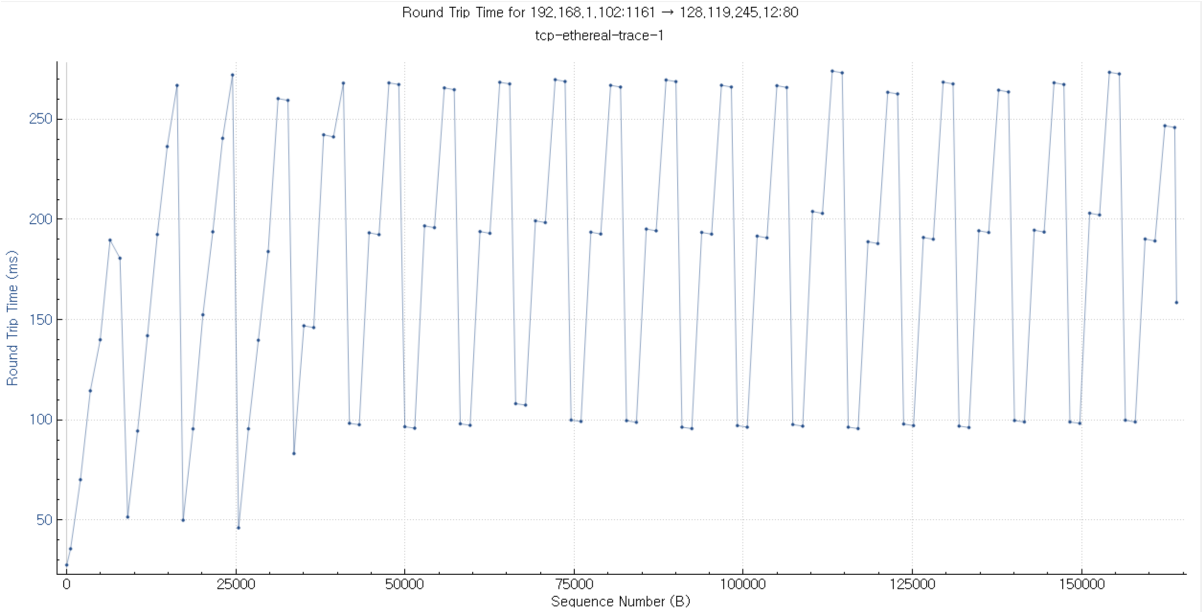
1: -

1: -

566: 0.0076744188s

2026: 0.163763665s

3486: 0.298418207s



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

<Sequence Number: Length (bit)>

0: 0

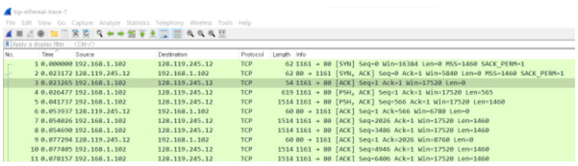
1: 0

1: 565

566: 1460

2026: 1460

3486: 1460



tcp-ethereal-trace-1

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter … <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
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 SACK_PERM=1
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
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
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
8	0.054690	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460
9	0.077294	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=2026 Win=8760 Len=0
10	0.077405	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=4946 Ack=1 Win=17520 Len=1460
11	0.078157	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460

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 available buffer space is 5840 bytes.

The lack of receiver buffer space throttled the sender. Whenever, the buffer space is determined too small for sender, the sender sent packet which length was zero.

No.	Time	Source	Destination	Protocol	Length	Info
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 SACK_PERM=1
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
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
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
8	0.054690	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460
9	0.077294	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=2026 Win=8760 Len=0
10	0.077405	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=4946 Ack=1 Win=17520 Len=1460
11	0.078157	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460
12	0.124085	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=3486 Win=11680 Len=0
13	0.124185	192.168.1.102	128.119.245.12	TCP	1201	1161 → 80 [PSH, ACK] Seq=7866 Ack=1 Win=17520 Len=1147
14	0.169118	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=4946 Win=14600 Len=0

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

There were no retransmitted segments in the trace file. Although, there was a duplicate sequence number of 1, the first 1 had length of 0 so that there is no change on the next sequence number. Except this case, there was no duplicated sequence number from the sender side. Through these reasons, there was no retransmitted segment in the trace file.

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

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).

The receiver typically acknowledges 20 bytes in an ACK.

```
[Next Sequence Number: 104965    (relative sequence number)]
Acknowledgment Number: 1    (relative ack number)
Acknowledgment number (raw): 883061786
0101 .... = Header Length: 20 bytes (5)
```

The receiver can acknowledge every other received segment by using the header.

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

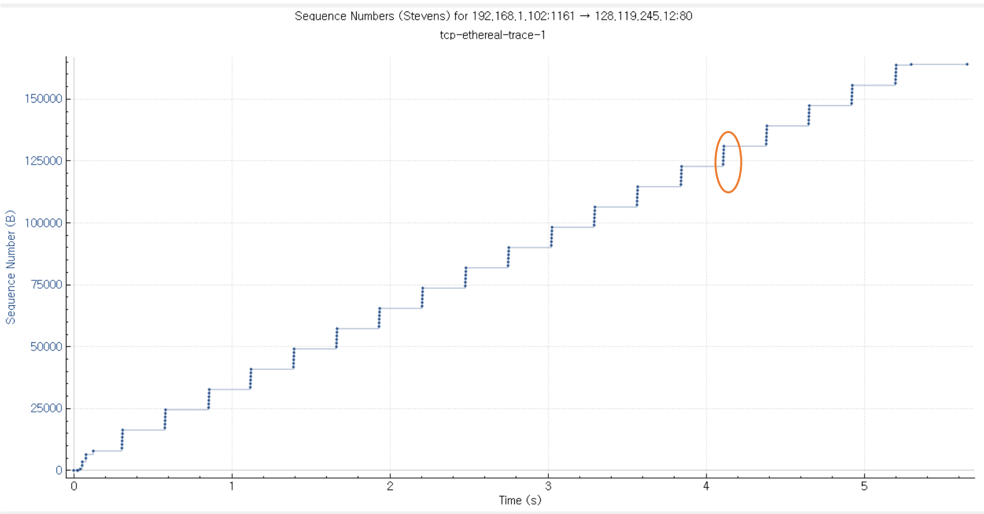
The throughput for the TCP connection is 16812.0 bytes per second. The reason is that the smallest bandwidth for the TCP connection was 54 bytes, and it took 0.003212 seconds so that the throughput for the TCP connection is around 17Kbyte/s.

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

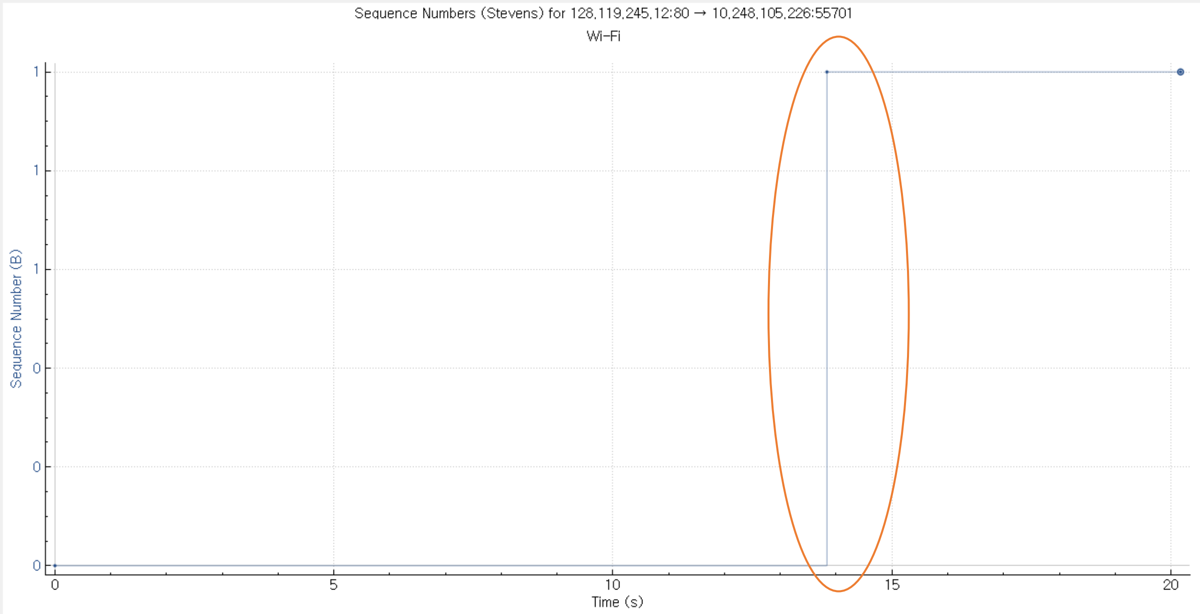
13. Use the Time-Sequence-Graph(Stevens) plotting tool to view the sequence number versus time plot of segments being sent from the client to the gaia.cs.umass.edu server. Can you identify where TCP's slowstart phase begins and ends, and where congestion avoidance takes over? Comment on ways in which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

The TCP's slow-start phase begins where the vertical edge of the graph starts, and ends where the vertical edge of the graph ends. (The orange circled area in the figure below.)

The data differs the idealized behavior of TCP which we handled in the class. The Stevens plotting showing the behavior like a stair, and the ideal behavior handled in class was oscillating between the maximum Window size and its half size. The reason of both plotting showing a different behavior is that Stevens plotting is setting the entire sequence number of segments and time as a variable, and the ideal behavior of TCP is setting the window size and time as a variable. Thus, these are the reason why each of the plotting shows different behavior.



14. Answer Question 13 for the trace that you captured when you transferred a file from your own computer to gaia.cs.umass.edu



The TCP's slow-start phase begins where the vertical edge of the graph starts, and ends where the vertical edge of the graph ends. (The orange circled area in the figure above.)

The data differs the idealized behavior of TCP which we handled in the class. The Stevens plotting showing the behavior like a stair, and the ideal behavior handled in class was oscillating between the maximum Window size and its half size. The reason of both plotting showing a different behavior is that Stevens plotting is setting the entire sequence number of segments and time as a variable, and the ideal behavior of TCP is setting the window size and time as a variable. Thus, these are the reason why each of the plotting shows different behavior. Furthermore, my actual trace is different because there was less communication between sender and receiver than the pre-record file. This can be expected that the size of segment and throughput of TCP connection is much more larger than the pre-record trace.