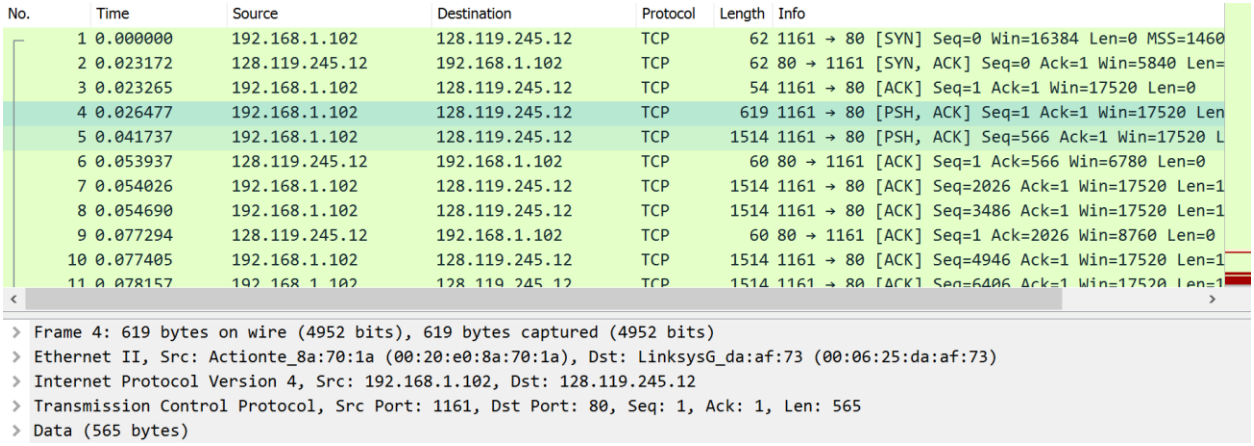
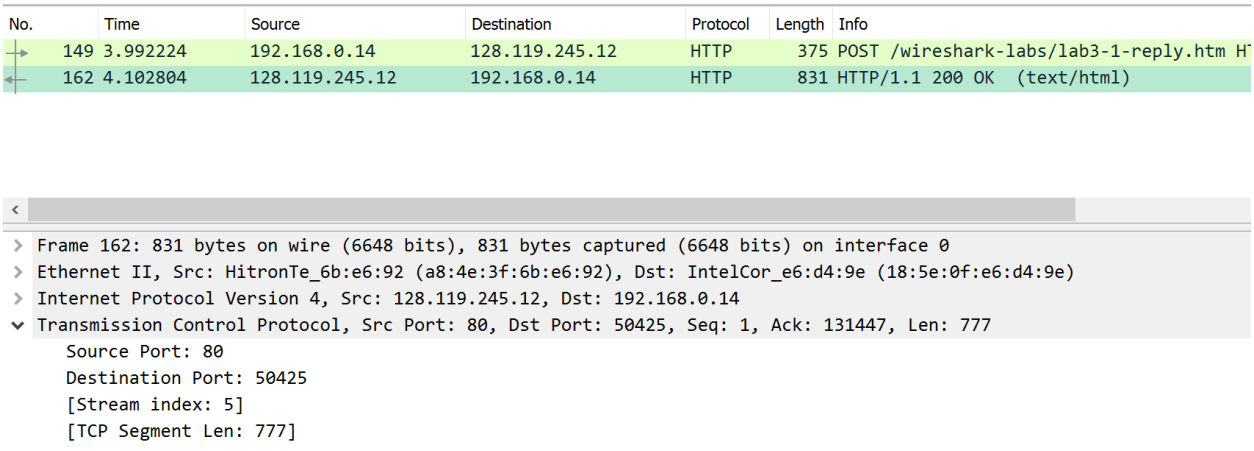
Project 2 **Alan Ou**
301320642

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.



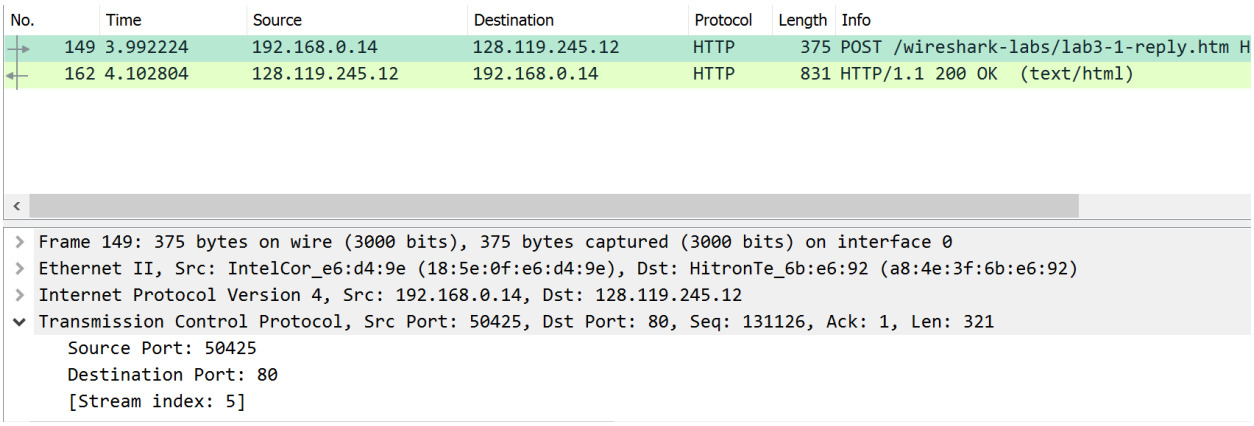
Using the *tcpethereal-trace-1* file, the IP address is **192.168.1.102** and TCP port number is **1161.**

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?



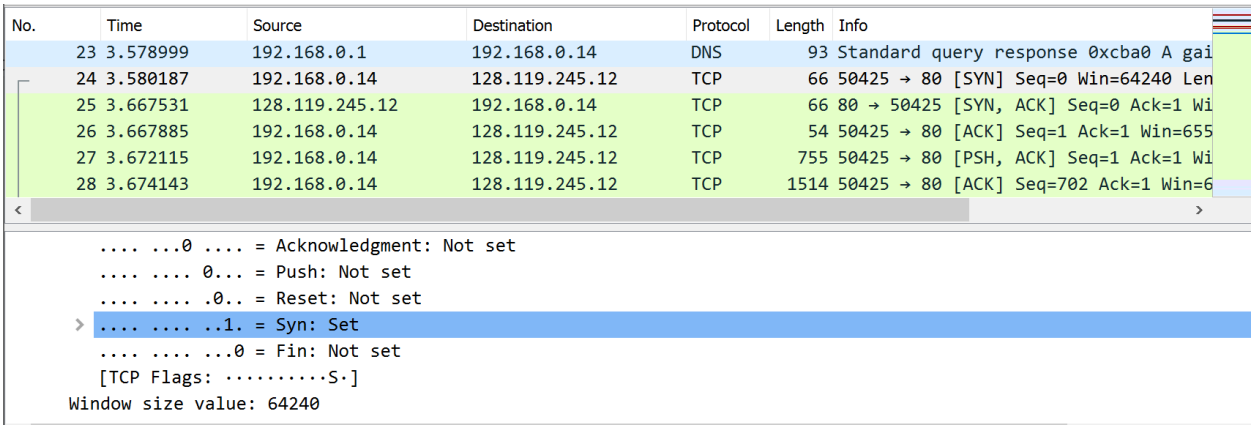
Using the trace I captured, the IP address of gaia.cs.umass.edu is **128.119.245.12**. On port number **80**.

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?



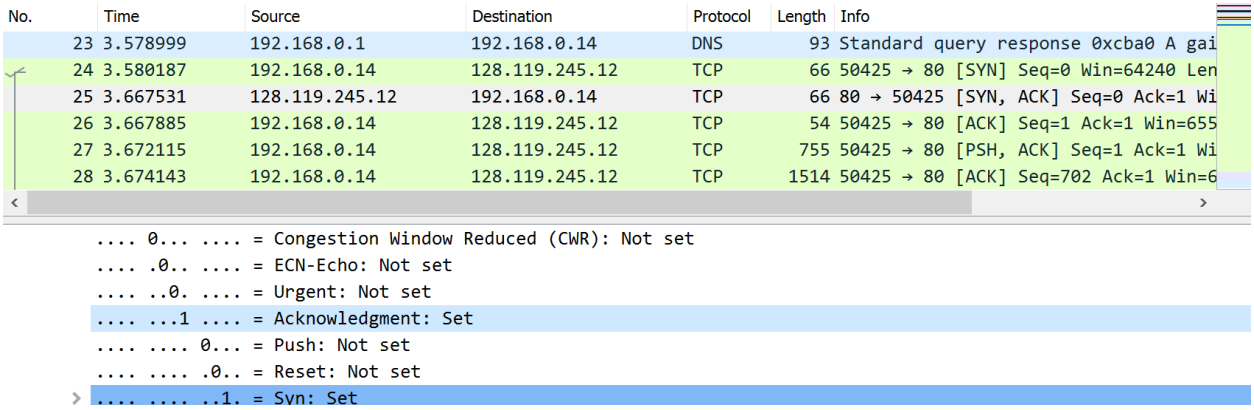
My IP address is **192.168.0.14** and TCP port number is **50425.**

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 of the TCP SYN segment is **0**. The SYN flag is set to **1**, therefore 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?

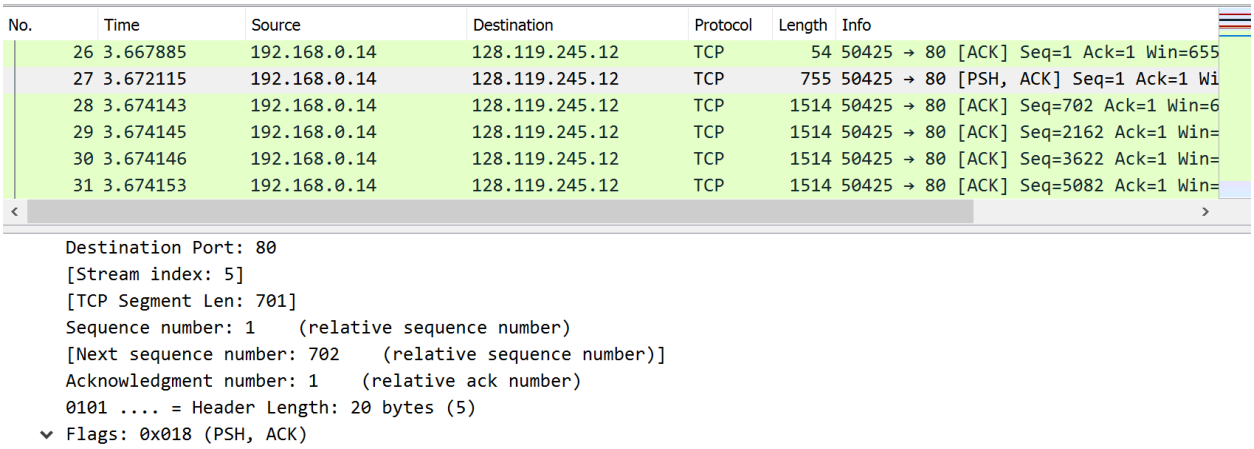


The sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN is **0**.

The value of the Acknowledgement field in the SYNACK segment is 1 and is determined by the gaia.cs.umass.edu (host) server. When the first segment is received, the host (gaia.cs.umass.edu) puts the sequence number of the next byte that the host expects to receive. As a result, since the SYN segment had an initial sequence number 0, the SYNACK segment is numbered 1.

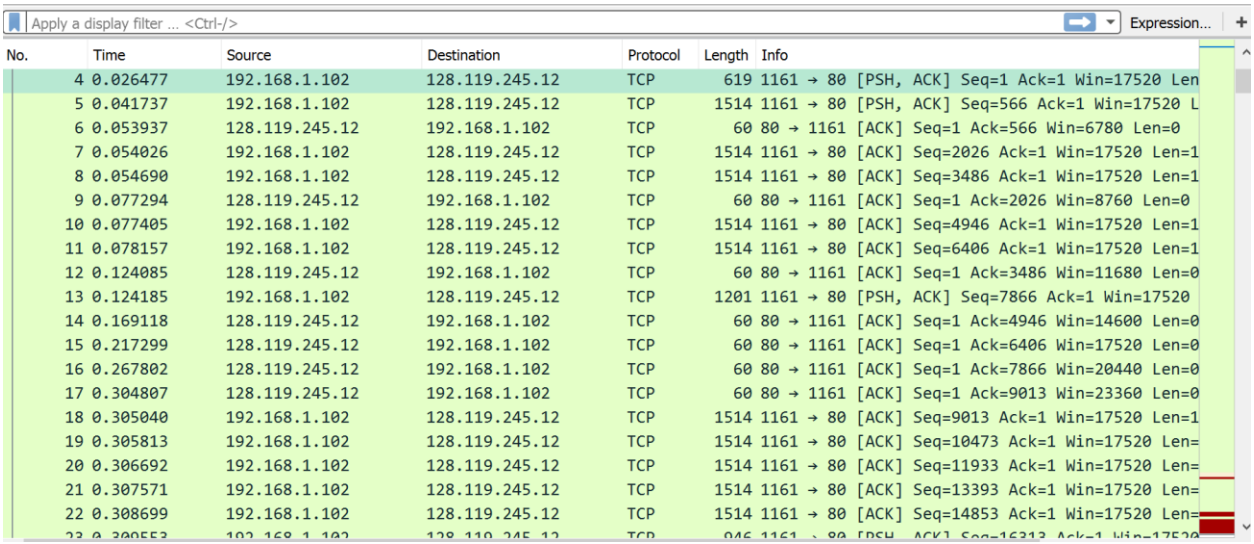
The parts of the segment that identifies the segment as a SYNACK segment are the SYN and Acknowledgement flags. In the SYNACK segment, both flags should be set to 1.

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 the HTTP POST command is **1**.

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.


Apply a display filter ... <Ctrl-/> Expression... +

No.	Time	Source	Destination	Protocol	Length	Info
4	0.026477	192.168.1.102	128.119.245.12	TCP	619	1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len
5	0.041737	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [PSH, ACK] Seq=566 Ack=1 Win=17520 L
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=1
8	0.054690	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1
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=1
11	0.078157	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1
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
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=1
19	0.305813	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=10473 Ack=1 Win=17520 Len=
20	0.306692	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=11933 Ack=1 Win=17520 Len=
21	0.307571	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=13393 Ack=1 Win=17520 Len=
22	0.308699	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=14853 Ack=1 Win=17520 Len=

Note that I used the trace from the *tcpethereal-trace-1* file to answer this question. The sequence numbers of the first six segments are **1, 566, 2026, 3486, 4946, 6406** and were sent at times **0.026477, 0.041737, 0.054026, 0.054690, 0.077405, 0.078157** respectively.

The ACKs for the first six segments were received at times **0.053937, 0.077294, 0.124085, 0.169118, 0.217299, 0.267802** respectively.

Segment 1 RTT = 0.053937 - 0.026477 = **0.02746**

Segment 2 RTT = 0.077294 - 0.041737 = **0.035557**

Segment 3 RTT = 0.124085 - 0.054026 = **0.070059**

Segment 4 RTT = 0.169118 - 0.054690 = **0.11443**

Segment 5 RTT = 0.217299 - 0.077405 = **0.13989**

Segment 6 RTT = 0.267802 - 0.078157 = **0.18964**

After receipt of ACK 1

$EstimatedRTT_1$ **= 0.02746**

After receipt of ACK 2

$EstimatedRTT_2$ = 0.875 * 0.02746 + 0.125 * 0.035557 = **0.028472125**

After receipt of ACK 3

$EstimatedRTT_3$ = 0.875 * 0.028472125 + 0.125 * 0.070059 = **0.033760484**

After receipt of ACK 4

$EstimatedRTT_4$ = 0.875 * 0.033760484 + 0.125 * 0.11443 = **0.043765424**

After receipt of ACK 5

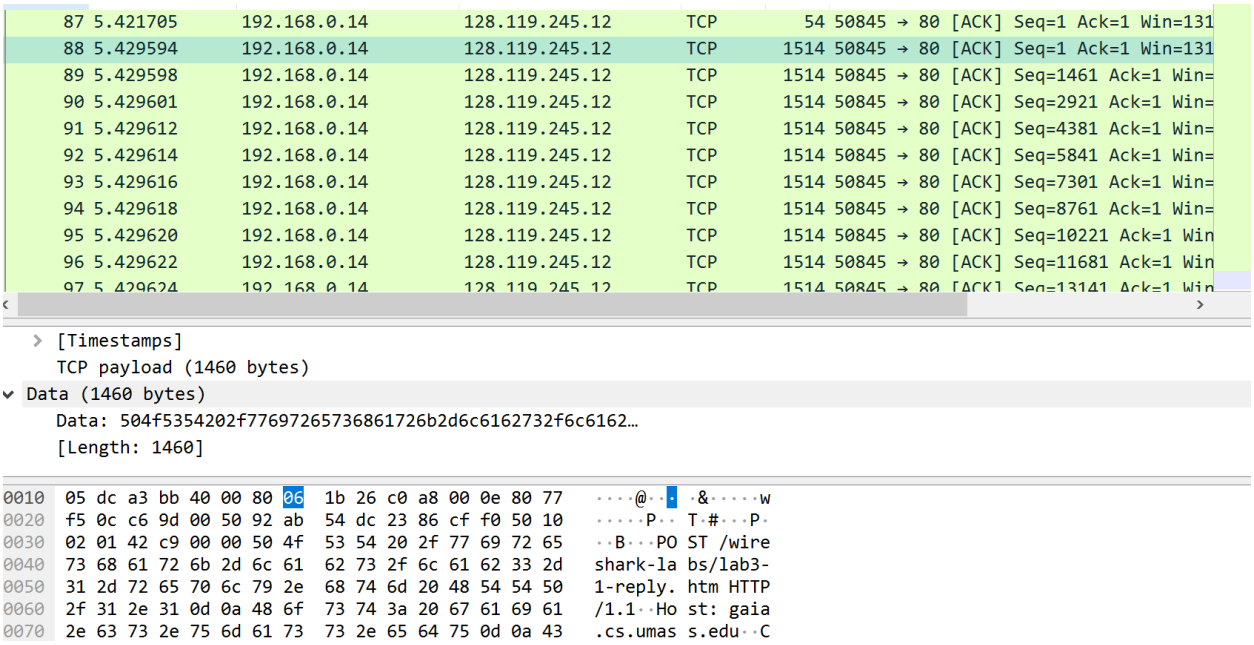
$EstimatedRTT_5$ = 0.875 * 0.043765424 + 0.125 * 0.13989 = **0.055780996**

After receipt of ACK 6

$EstimatedRTT_6$ = 0.875 * 0.055780996 + 0.125 * 0.18964 = **0.072513371**

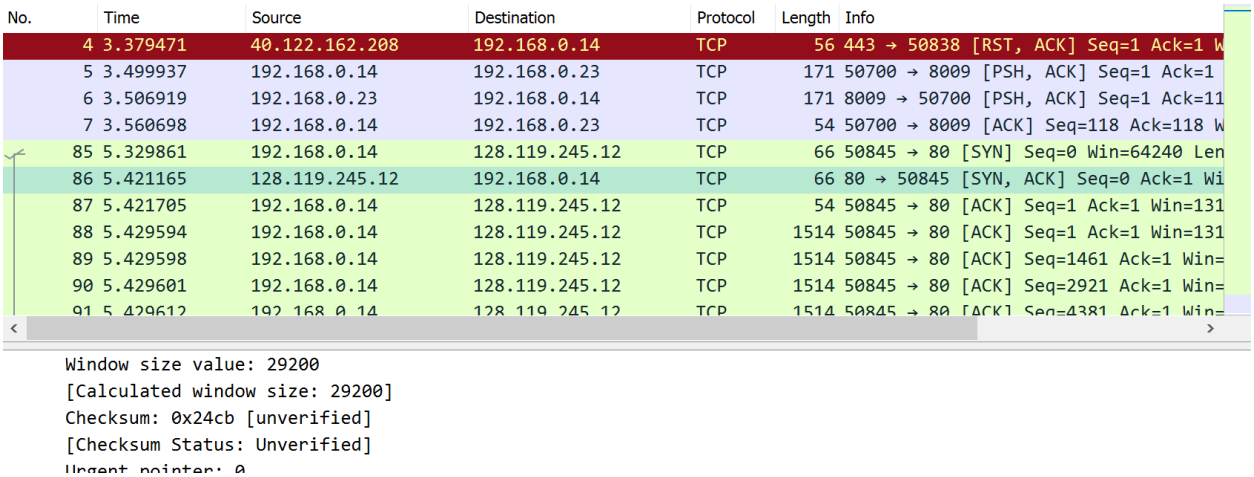
8. What is the length of each of the first six TCP segments?

Note that I restarted the capture for this question.



The first segment is 1460 bytes, and so were the rest. So, all of the first six TCP segments were 1460 bytes.

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?



No.	Time	Source	Destination	Protocol	Length	Info
4	3.379471	40.122.162.208	192.168.0.14	TCP	56	443 → 50838 [RST, ACK] Seq=1 Ack=1 W
5	3.499937	192.168.0.14	192.168.0.23	TCP	171	50700 → 8009 [PSH, ACK] Seq=1 Ack=1
6	3.506919	192.168.0.23	192.168.0.14	TCP	171	8009 → 50700 [PSH, ACK] Seq=1 Ack=11
7	3.560698	192.168.0.14	192.168.0.23	TCP	54	50700 → 8009 [ACK] Seq=118 Ack=118 W
85	5.329861	192.168.0.14	128.119.245.12	TCP	66	50845 → 80 [SYN] Seq=0 Win=64240 Len
86	5.421165	128.119.245.12	192.168.0.14	TCP	66	80 → 50845 [SYN, ACK] Seq=0 Ack=1 Wi
87	5.421705	192.168.0.14	128.119.245.12	TCP	54	50845 → 80 [ACK] Seq=1 Ack=1 Win=131
88	5.429594	192.168.0.14	128.119.245.12	TCP	1514	50845 → 80 [ACK] Seq=1 Ack=1 Win=131
89	5.429598	192.168.0.14	128.119.245.12	TCP	1514	50845 → 80 [ACK] Seq=1461 Ack=1 Win=
90	5.429601	192.168.0.14	128.119.245.12	TCP	1514	50845 → 80 [ACK] Seq=2921 Ack=1 Win=
91	5.429612	192.168.0.14	128.119.245.12	TCP	1514	50845 → 80 [ACK] Seq=4381 Ack=1 Win=

```
Window size value: 29200
[Calculated window size: 29200]
Checksum: 0x24cb [unverified]
[Checksum Status: Unverified]
Urgent pointer: 0
```

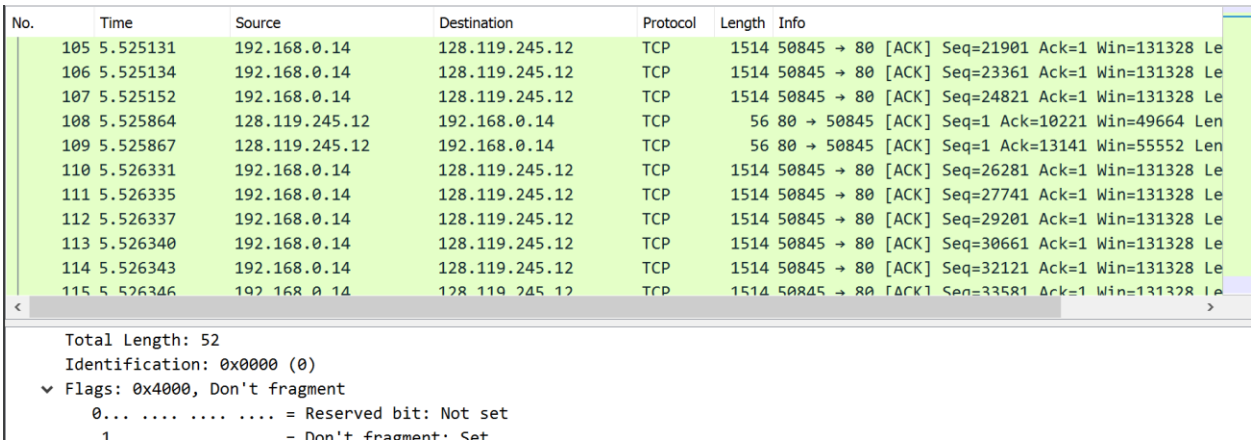
The minimum of available buffer space advertised at the received for the entire trace is **29200**.

No, the sender is never throttled.

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, there were no retransmitted segments. If there were retransmitted segments, then it would be explicitly stated in the info section with the words “TCP RETRANSMISSION”.

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).



No.	Time	Source	Destination	Protocol	Length	Info
105	5.525131	192.168.0.14	128.119.245.12	TCP	1514	50845 → 80 [ACK] Seq=21901 Ack=1 Win=131328 Le
106	5.525134	192.168.0.14	128.119.245.12	TCP	1514	50845 → 80 [ACK] Seq=23361 Ack=1 Win=131328 Le
107	5.525152	192.168.0.14	128.119.245.12	TCP	1514	50845 → 80 [ACK] Seq=24821 Ack=1 Win=131328 Le
108	5.525864	128.119.245.12	192.168.0.14	TCP	56	80 → 50845 [ACK] Seq=1 Ack=10221 Win=49664 Len
109	5.525867	128.119.245.12	192.168.0.14	TCP	56	80 → 50845 [ACK] Seq=1 Ack=13141 Win=55552 Len
110	5.526331	192.168.0.14	128.119.245.12	TCP	1514	50845 → 80 [ACK] Seq=26281 Ack=1 Win=131328 Le
111	5.526335	192.168.0.14	128.119.245.12	TCP	1514	50845 → 80 [ACK] Seq=27741 Ack=1 Win=131328 Le
112	5.526337	192.168.0.14	128.119.245.12	TCP	1514	50845 → 80 [ACK] Seq=29201 Ack=1 Win=131328 Le
113	5.526340	192.168.0.14	128.119.245.12	TCP	1514	50845 → 80 [ACK] Seq=30661 Ack=1 Win=131328 Le
114	5.526343	192.168.0.14	128.119.245.12	TCP	1514	50845 → 80 [ACK] Seq=32121 Ack=1 Win=131328 Le
115	5.526346	192.168.0.14	128.119.245.12	TCP	1514	50845 → 80 [ACK] Seq=33581 Ack=1 Win=131328 Le

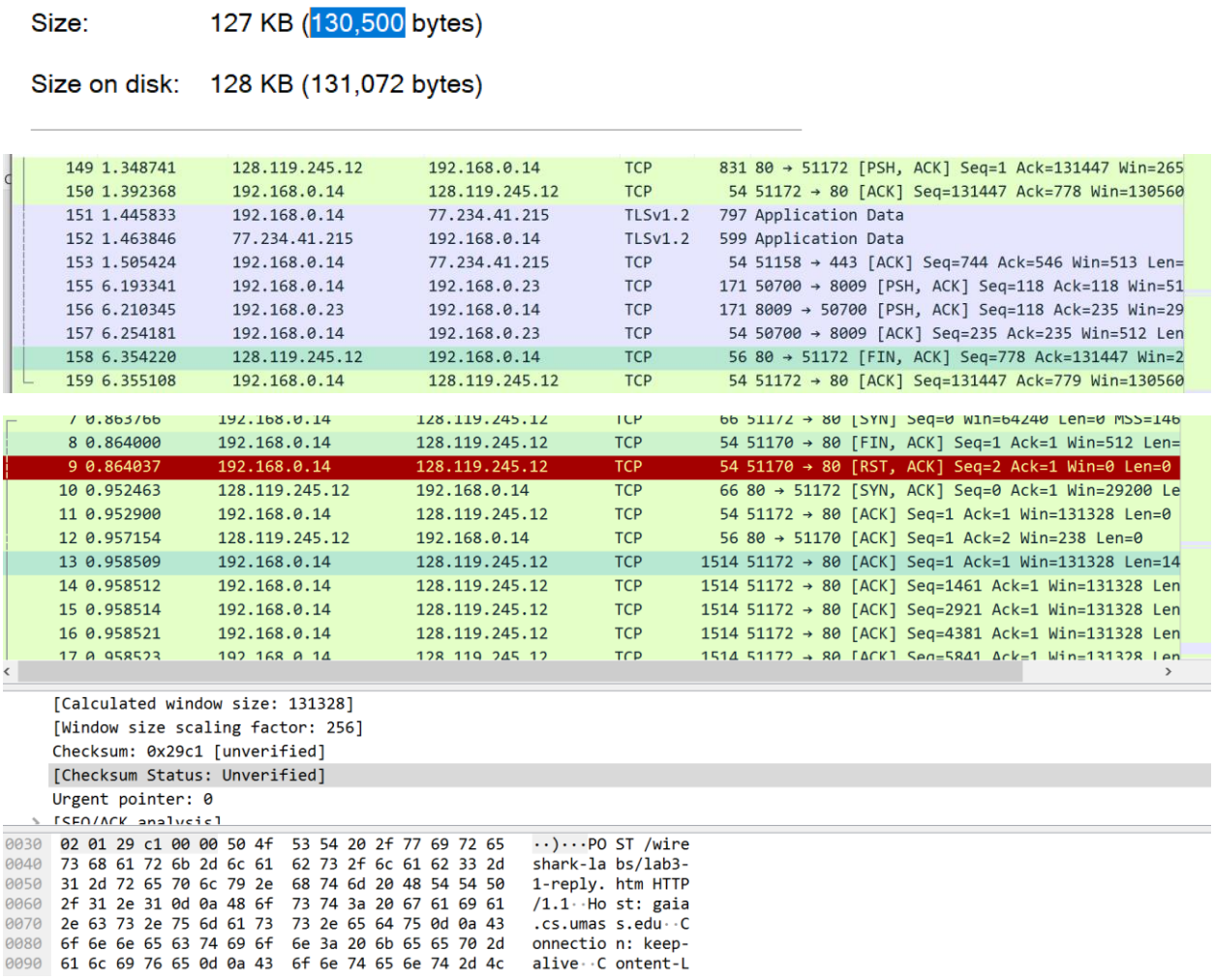
```
Total Length: 52
Identification: 0x0000 (0)
Flags: 0x4000, Don't fragment
    0... .... .... .... = Reserved bit: Not set
    .1.. .... .... .... = Don't fragment: Set
```

Difference between two consecutive acknowledgements = 13141 – 10221 = 2920

The receiver typically acknowledges **2920** bytes per ACK.

For all cases, the receiver is ACKing every other received segment.

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



Throughput is the “bytes transferred per unit time” as defined the question. As a result, I first took the file size, which on my computer is 130,500 bytes then I calculated the transmission time.

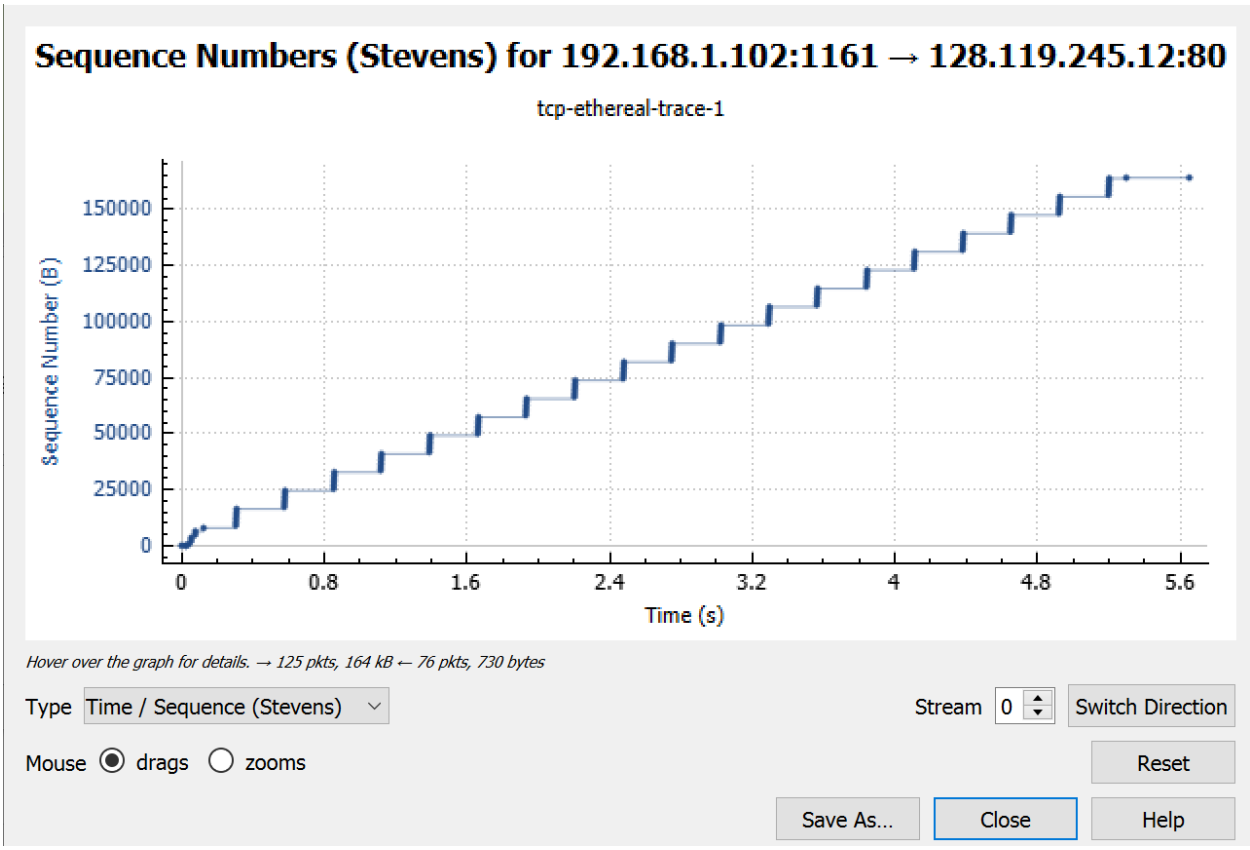
I took the time when the first TCP was sent, and I subtracted it from the Fin ACK.

6.354220 – 0.958509 = 5.395711

Then I divided the file size by the time.

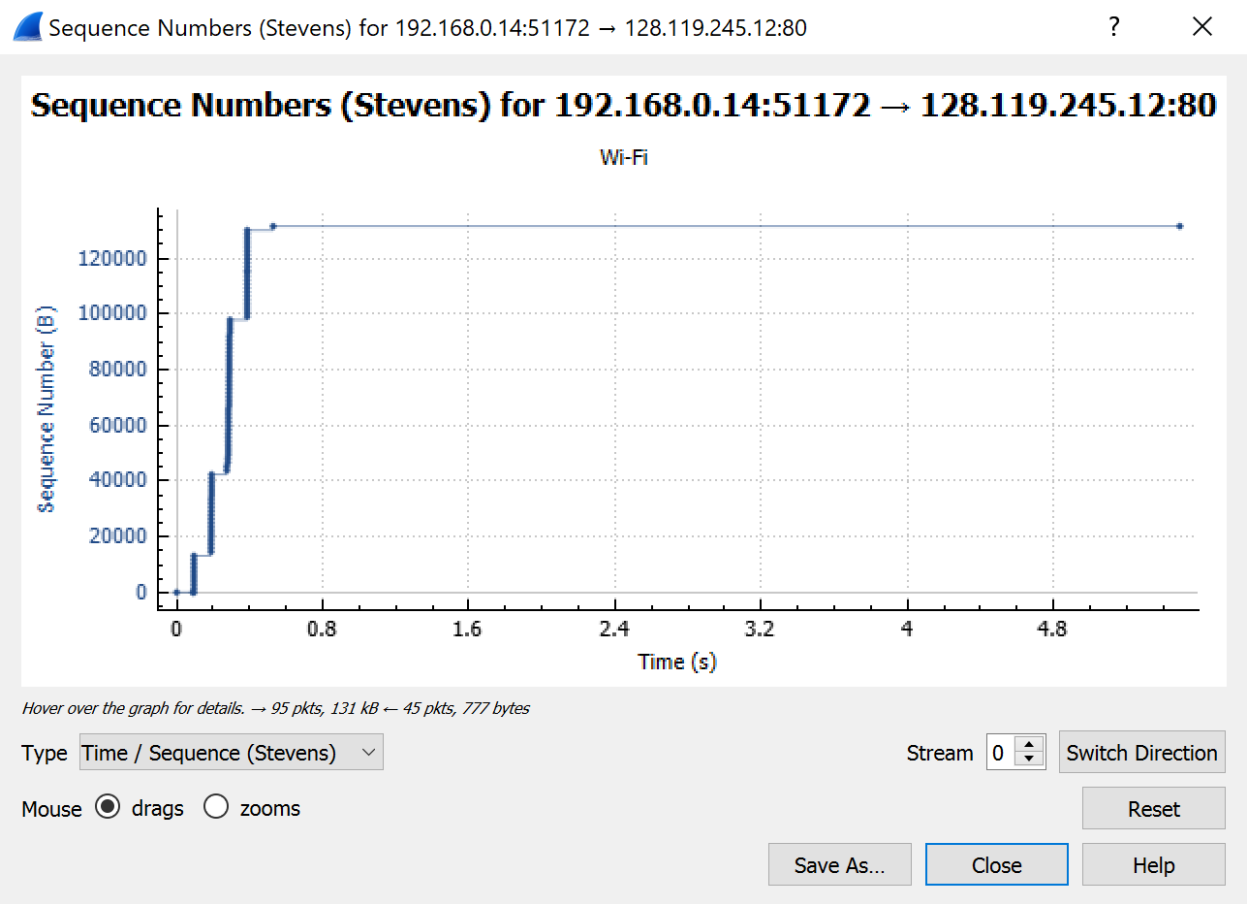
130,500 / 5.395711 = **24,185.88 bytes/second**

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.



Slowstart appears to begin at 0.02648 and ends at 0.07741. Congestion avoidance begins at 0.1242.

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



Slowstart appears to begin at 0.1 and ends around 0.6. Congestion avoidance begins at 0.7.