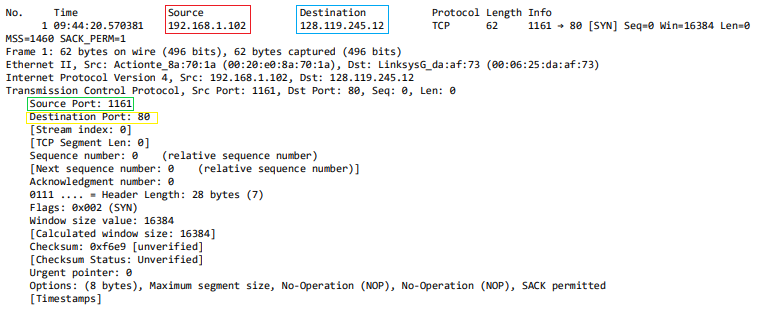
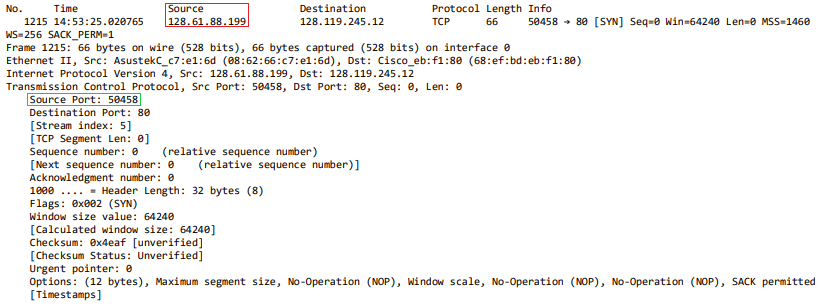
1. What is the IP address and TCP port number used by the client computer (source) that is transferring the file to gaia.cs.umass.edu? To answer this question, it’s probably easiest to select an HTTP message and explore the details of the TCP packet used to carry this HTTP message, using the “details 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.
   1. Client/Source IP address: 192.168.1.102 (Marked in Red)
   2. Client/Source Port: 1161 (Marked in Green)
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?
   1. Destination IP Address: 128.119.245.12 (Marked in Blue)
   2. Destination Port: 80 (Marked in Yellow)

Screen Shot for 1 and 2:



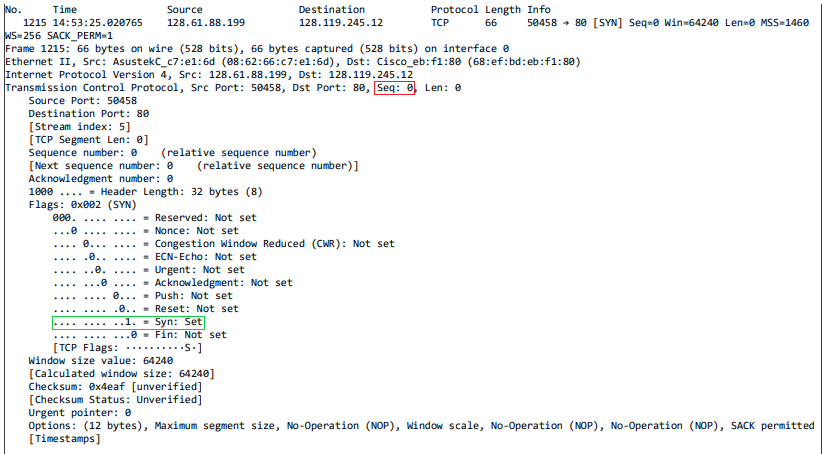
1. What is the IP address and TCP port number used by your client computer (source) to transfer the file to gaia.cs.umass.edu?
   1. My Client/Sources IP address: 128.61.88.199 (Marked in Red)
   2. My Client/Sources Port: 50458 (Marked In Green)

Screen Shot for 3:



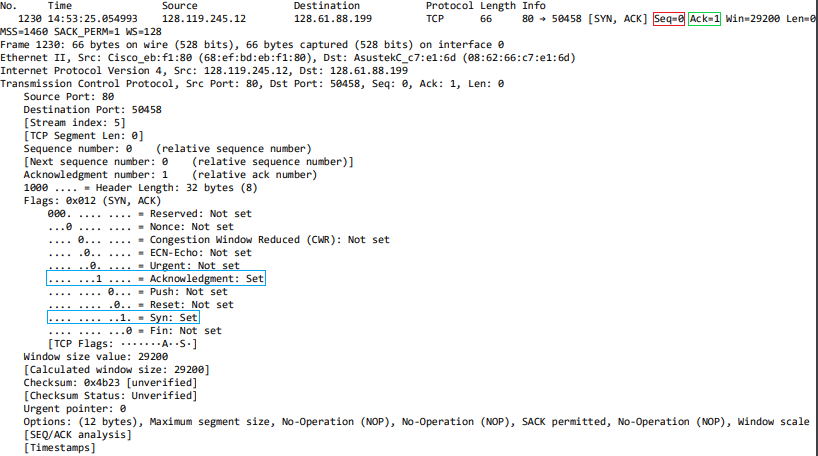
1. What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? What is it in the segment that identifies the segment as a SYN segment?
   1. The Sequence Number of the TCP SYN Segment is: 0 (Marked In Red)
   2. Under the Flags part of the packet, there the Syn flag is set to 1 which means it is a SYN segment. (Marked in Green)

Screen Shot for 4:



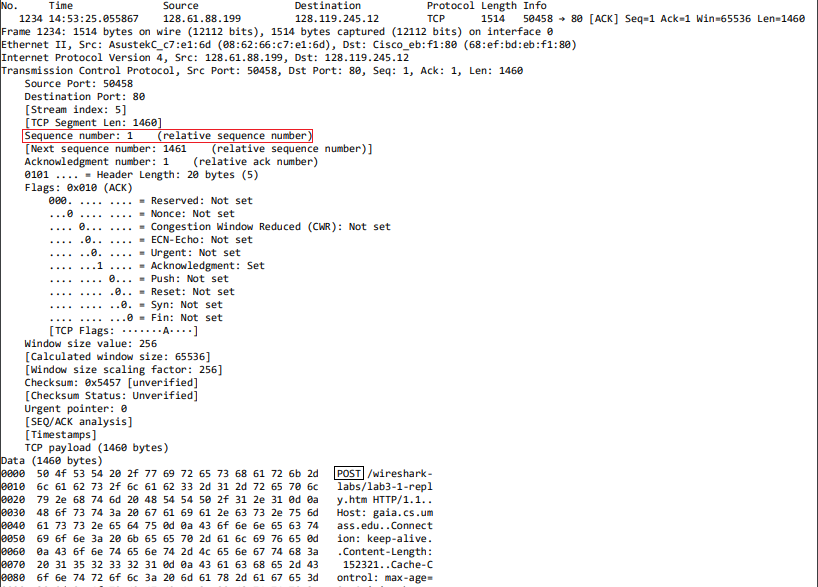
1. What is the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN? What is the value of the Acknowledgement field in the SYNACK segment? How did gaia.cs.umass.edu determine that value? What is it in the segment that identifies the segment as a SYNACK segment?
   1. Sequence Number of SNACK segment: 0 (Marked In Red)
   2. Value of the Acknowledgement field: 1 (Marked in Green)
   3. The server adds 1 to the initial sequence number of SYN segment that is received from the client.
   4. The segment is identified as a SYNACK segment if both the SYN Set and Acknowledgement Set are 1. (Marked in Blue)

Screen Shot for 5:



1. 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.
   1. The sequence number of the POST command is: 1 (Marked In Red)

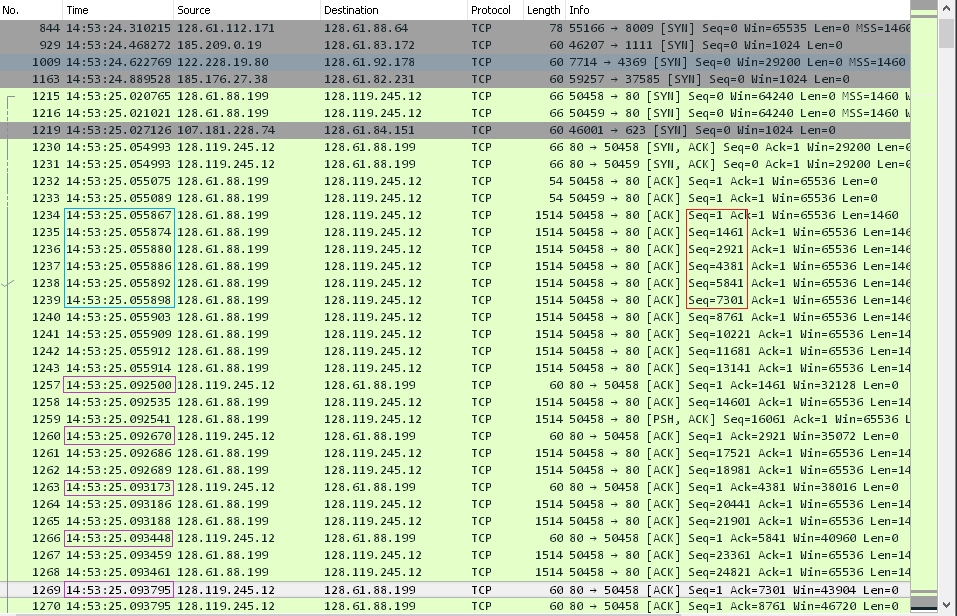
Screen Shot for 6:



1. 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.
   1. The first 6 sequence numbers are: 1, 1461, 2921, 4381, 5841, 7301 (Marked in Red)
   2. Each Segment was sent at: 14:53:25.055867, 14:53:25.055874, 14:53:25.055880, 14:53:25.055886, 14:53:25.055892, 14:53:25.055898 (Marked in Blue)
   3. Each Segment was received at:(Could not find the receive time for Sequence No. 1) 14:53:25.092500, 14:53:25.092670, 14:53:25.093173, 14:53:25.093448, 14:53:25.093795 (Marked in Purple)

|  |  |  |  |
| --- | --- | --- | --- |
| Sequence Number: | Time Sent: | Time Received: | RTT: |
| 1 | 14:53:25.055867 | N/A | 0.0404  (Found RTT based on Graph) |
| 1461 | 14:53:25.055874 | 14:53:25.092500 | 0.03992825 |
| 2921 | 14:53:25.055880 | 14:53:25.092670 | 0.03959875 |
| 4381 | 14:53:25.055886 | 14:53:25.093173 | 0.040010875 |
| 5841 | 14:53:25.055892 | 14:53:25.093448 | 0.03969450 |
| 7301 | 14:53:25.055898 | 14:53:25.093795 | 0.040087125 |

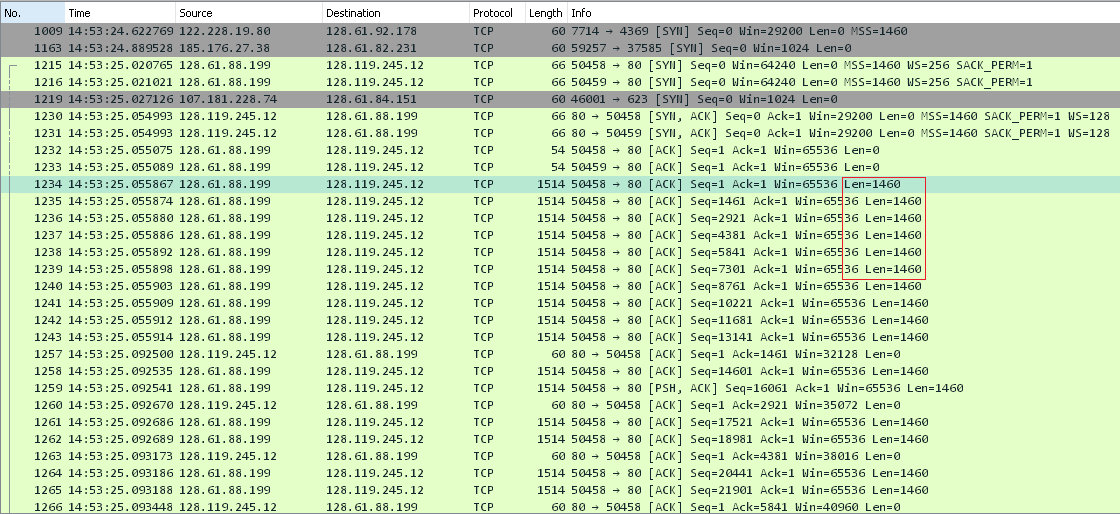
Screen Shot for 7:



1. What is the length of each of the first six TCP segments?
   1. Table:

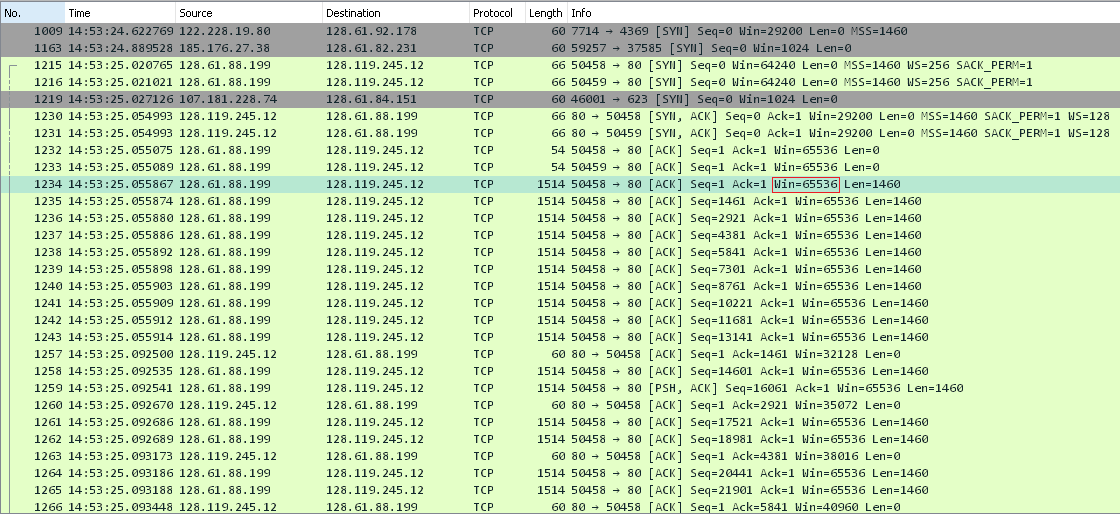
|  |  |  |
| --- | --- | --- |
| Packet No. | Sequence No. | TCP Segment Len:  (Marked in Red) |
| 1234 | 1 | 1460 |
| 1235 | 1464 | 1460 |
| 1236 | 2921 | 1460 |
| 1237 | 4381 | 1460 |
| 1238 | 5841 | 1460 |
| 1239 | 7201 | 1460 |

Screen Shot for 8:

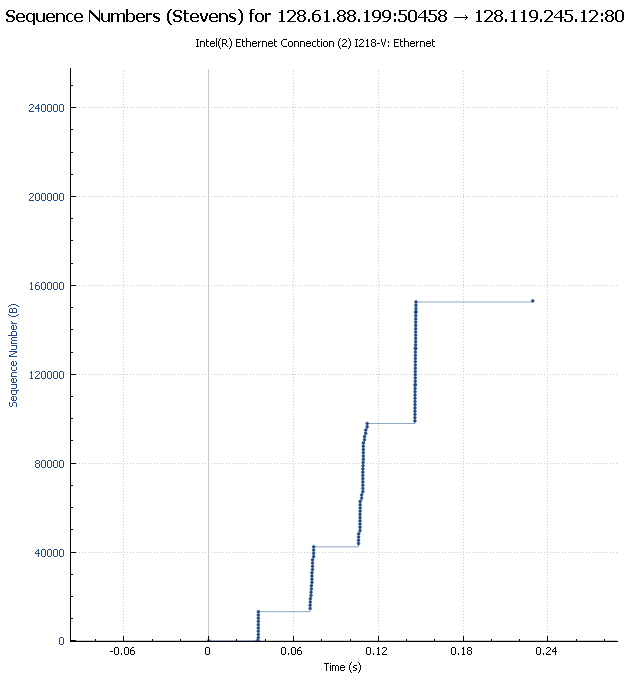


1. What is the minimum amount of available buffer space advertised at the received for the entire trace? Does the lack of receiver buffer space ever throttle the sender?
   1. The minimum amount of available buffer space advertised is: 65536 (Marked in Red)
   2. There is plenty of receiver buffer space so the sender is not throttled

Screen Shot of 9:

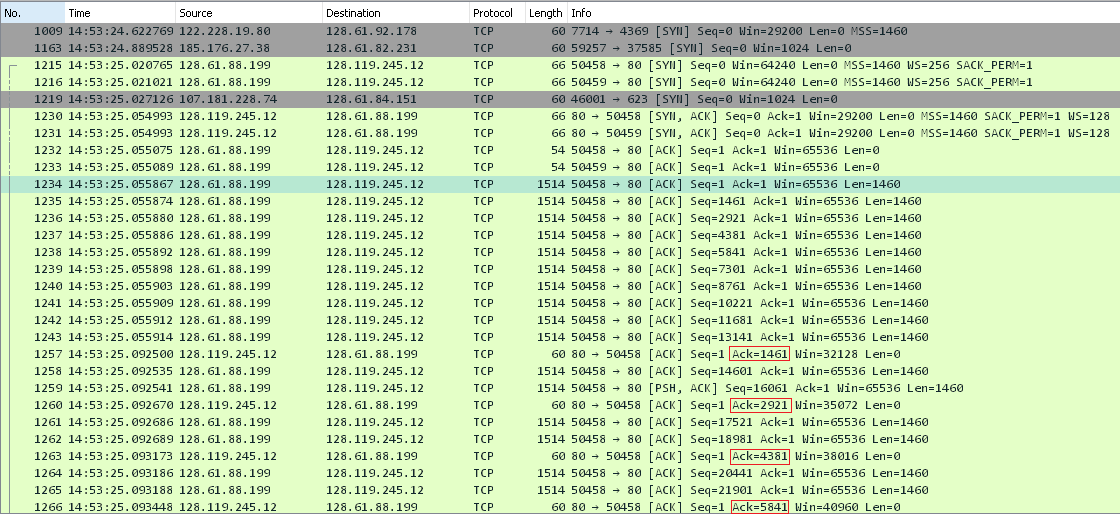


1. Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?
   1. No there were no retransmitted segments, you can determine this by looking at the figure below, where the same sequence no doesn’t appear at different times:



1. 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).
   1. Based on the screen shot the receiver Acknowledges 1460 bytes at a time (Marked in Red)

Screen Shot of 11:



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

Screen Shot of 12:





1. 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.
   1. The slow start phase looks to go from 0 sec to about 0.16 sec (Marked in Red)
   2. The congestion avoidance takes over at about about 0.32sec (Marked in Green)
   3. The graph is shows that the TCP Transmit window is not increase linearly instead it looks like we are transmitting a consistent 6 packets, this may be cause by the rate-limit of HTTP
2. My Graph’s results:
   1. The slowstart phase begins at 0.035 secs and ends at about 0.11375 sec (Marked in Red)
   2. Congestion avoidance takes over at what looks to be about 0.14875 sec (Marked in Green)

Screen Shot of 14:

