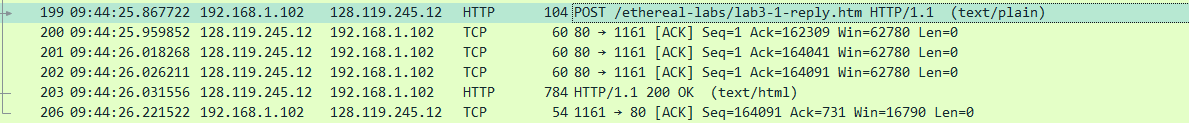
**CS 655 Computer Network**

**TCP Lab**

**Ziqi Tan**

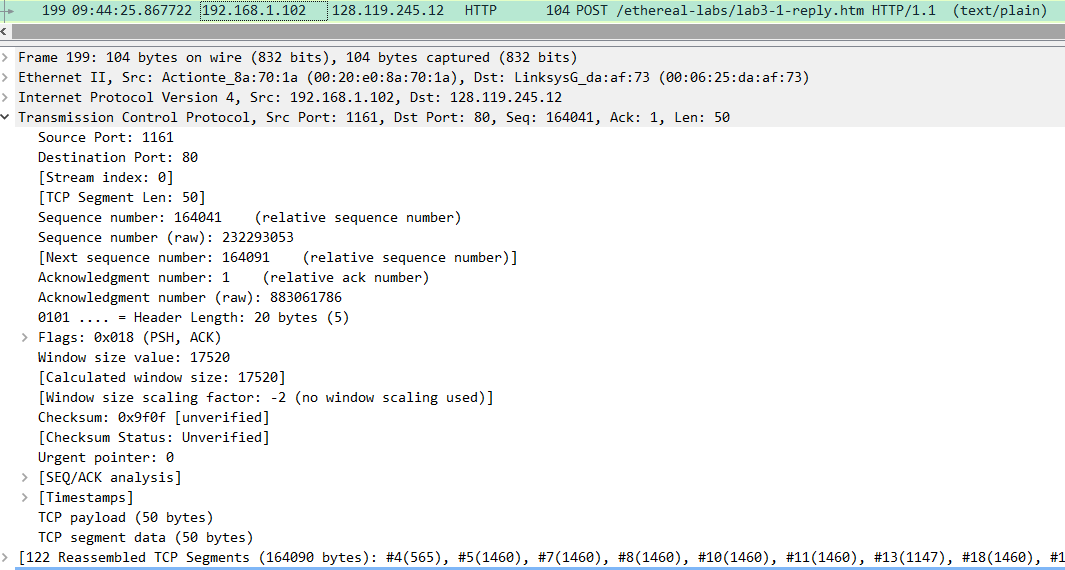
**U 88387934**

A first look at the captured trace.



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.**

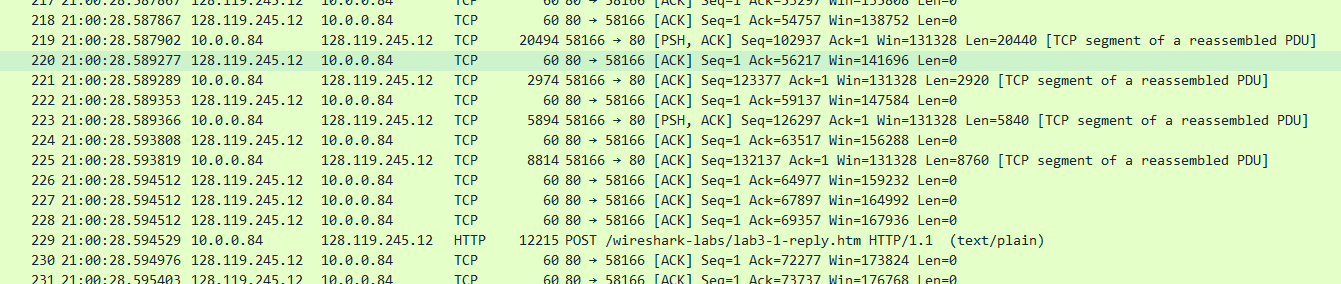
The client computer’s IP address is 192.168.1.105 and the TCP port number is 1161.



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?**

The IP address of gaia.cs.umass.edu is 128.119.245.12 and the port number is 80.

If you have been able to create your own trace, answer the following question:



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?**

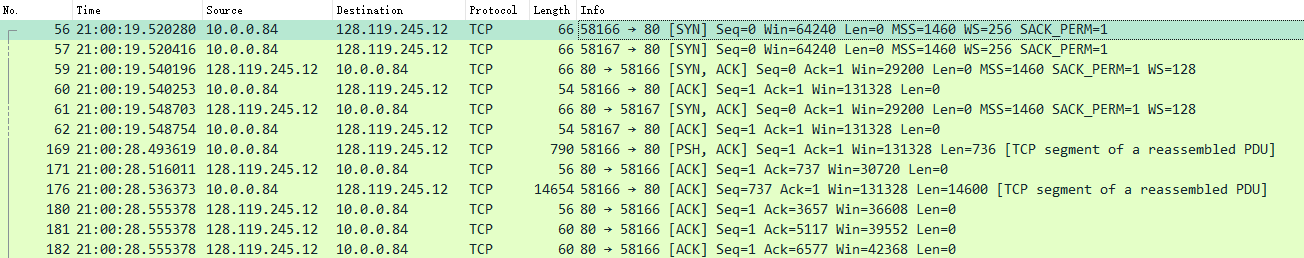
The IP address of my computer is 10.0.0.84 and the port number is 51456.

The IP address of gaia.cs.umass.edu is 128.119.245.12 and the port number is 80.

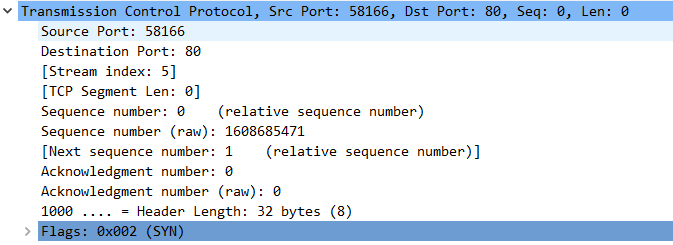
**I will use my own trace to answer the following questions.**

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?**

The sequence number is 0.



The flag 0x002 (SYN) identifies the segment as a SYN segment.



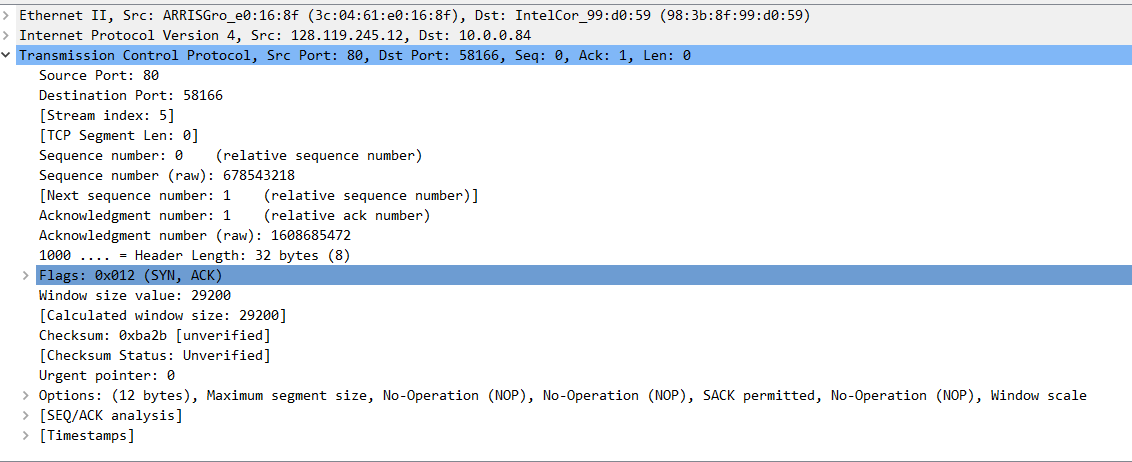
1. **What is the sequence number of the SYNACK segment sent by** **gaia.cs.umass.edu to the client computer in reply to the SYN? What is the value of the Acknowledgement field in the SYNACK segment? How did gaia.cs.umass.edu determine that value? What is it in the segment that identifies the segment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

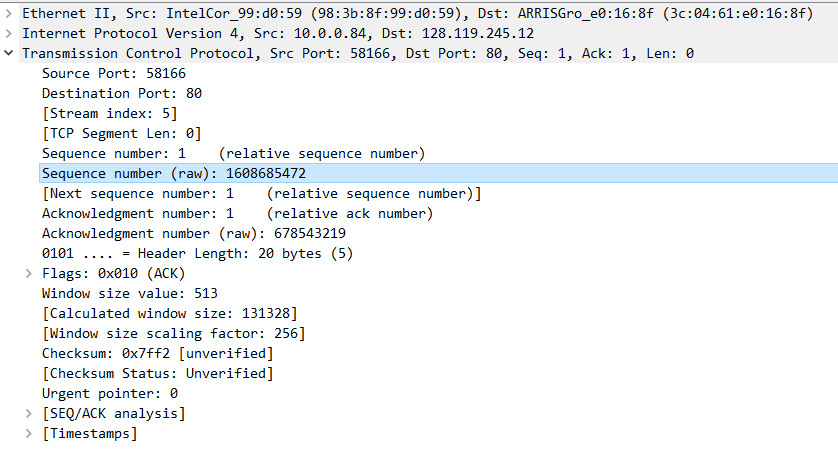
The sequence number is 0.

Acknowledgement number: 1.

The acknowledgment number that “gaia.cs.umass.edu” puts in its segment is the sequence number of the next byte “gaia.cs.umass.edu” is expecting from the client computer. Let’s see the next TCP segment sent from the client computer whose sequence number is exactly 1.

The Flags: 0x012 (SYN, ACK) identifies the segment as a SYNACK segment.

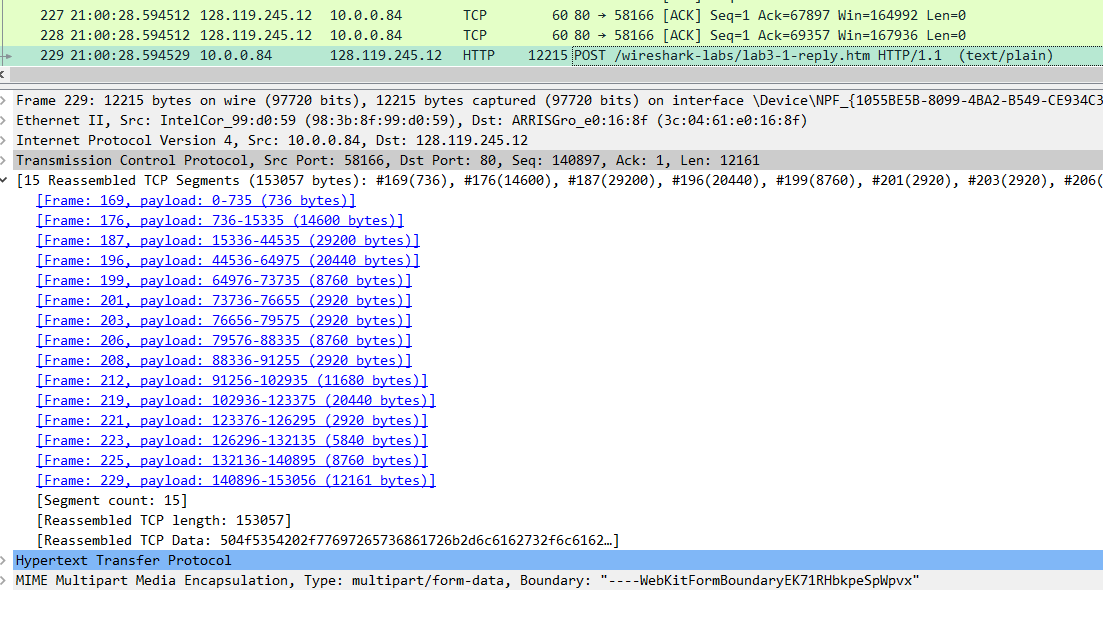


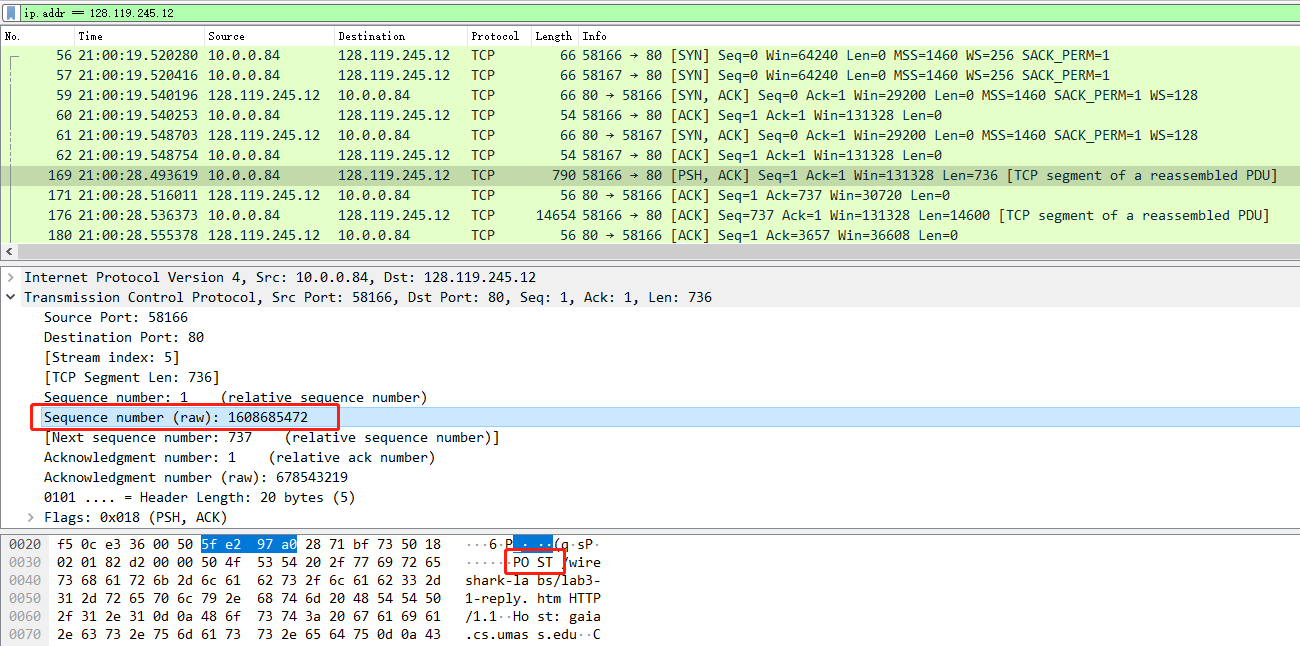


1. **What is the sequence number of the TCP segment containing the HTTP POST command? Note that in order to find the POST command, you’ll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with a “POST” within its DATA field.**

The sequence number is 1.

From the HTTP POST request, find the first frame number 169 which is the first post request TCP segment. Then, turn to frame 169 and we find the sequence number is 1. In the raw data, we can see the “POST”.





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 Estimated RTT value (see Section 3.5.3, page 242 in text) after the receipt of each ACK?** **Assume that the value of the Estimated RTT is equal to the measured RTT for the first segment, and then is computed using the Estimated RTT 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.

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

The sequence numbers of the first six segments in the TCP connection are 1, 737, 15337, 45537, 64977, 73737. **The sent time and the length** are shown below.

169 **21:00:28.493619** 10.0.0.84 128.119.245.12 TCP 790 58166 → 80 [PSH, ACK]

**Seq=1** Ack=1 Win=131328 **Len=736** [TCP segment of a reassembled PDU]

176 **21:00:28.536373** 10.0.0.84 128.119.245.12 TCP 14654 58166 → 80 [ACK]

**Seq=737** Ack=1 Win=131328 **Len=14600** [TCP segment of a reassembled PDU]

187 **21:00:28.555414** 10.0.0.84 128.119.245.12 TCP 29254 58166 → 80 [PSH, ACK]

**Seq=15337** Ack=1 Win=131328 **Len=29200** [TCP segment of a reassembled PDU]

196 **21:00:28.572104** 10.0.0.84 128.119.245.12 TCP 20494 58166 → 80 [PSH, ACK]

**Seq=44537** Ack=1 Win=131328 **Len=20440** [TCP segment of a reassembled PDU]

199 **21:00:28.574148** 10.0.0.84 128.119.245.12 TCP 8814 58166 → 80 [PSH, ACK] **Seq=64977** Ack=1 Win=131328 **Len=8760** [TCP segment of a reassembled PDU]

201 **21:00:28.574593** 10.0.0.84 128.119.245.12 TCP 2974 58166 → 80 [ACK]

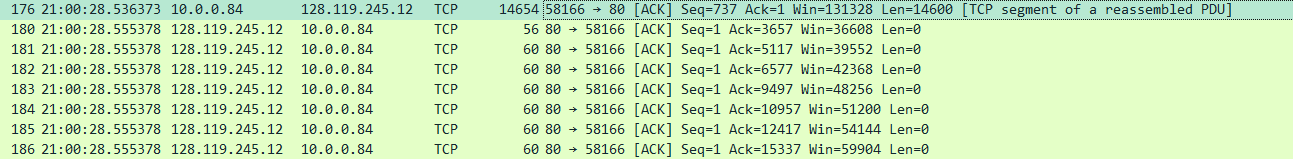
**Seq=73737** Ack=1 Win=131328 **Len=2920** [TCP segment of a reassembled PDU]

The ACK for sequence number 1:

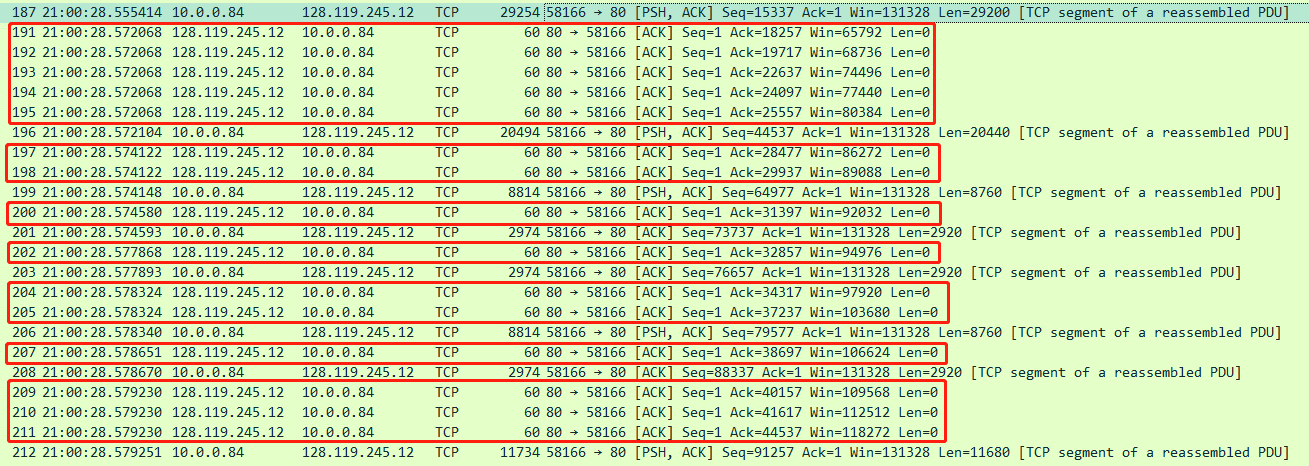
171 21:00:28.516011 128.119.245.12 10.0.0.84 TCP 56 80 → 58166 [ACK]

Seq=1 **Ack=737** Win=30720 Len=0

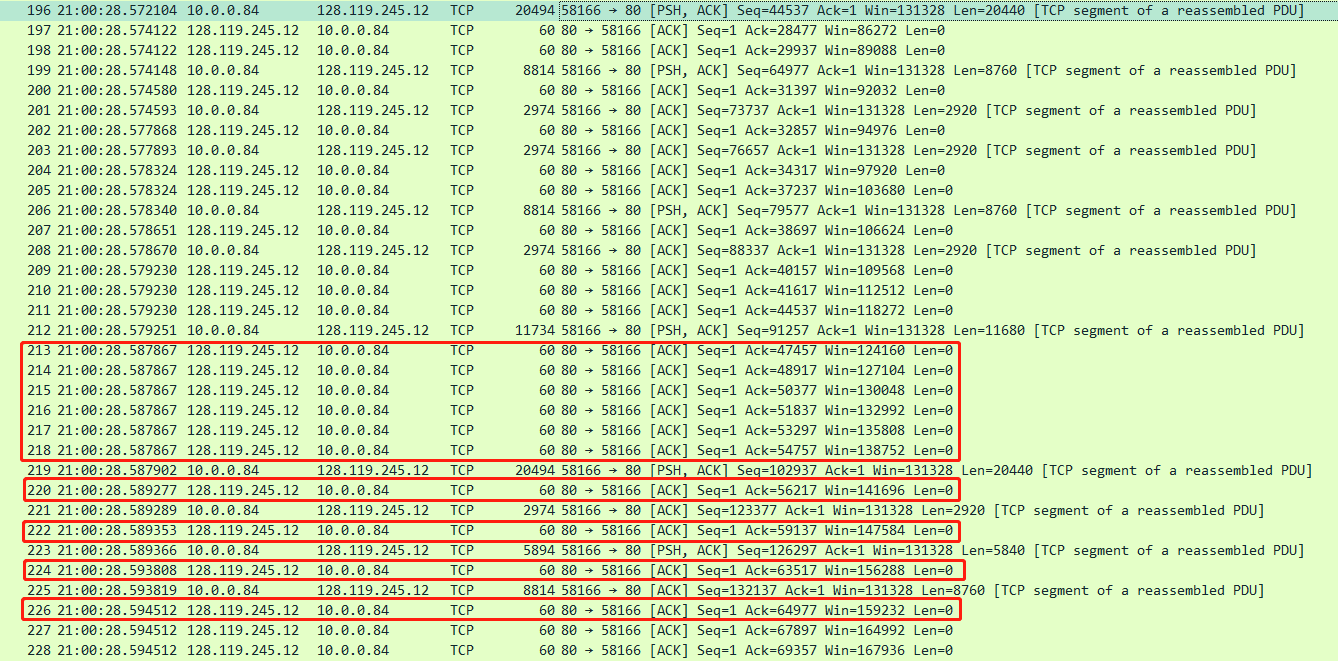
The ACKs for sequence number 737:



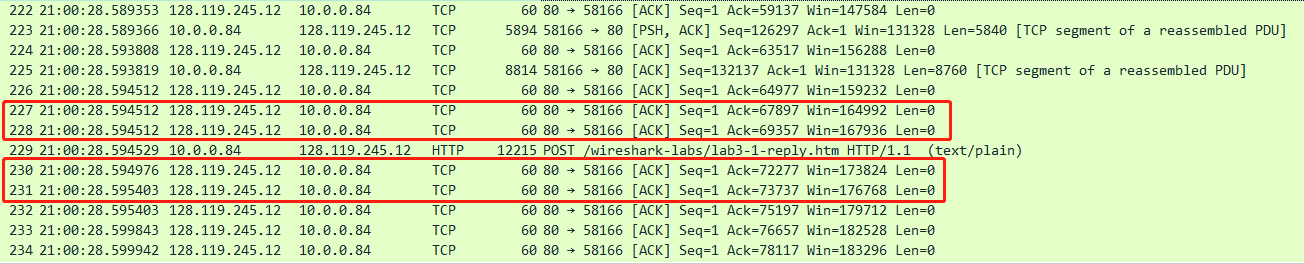
The ACKs for sequence number 15337 are circled in red rectangles:



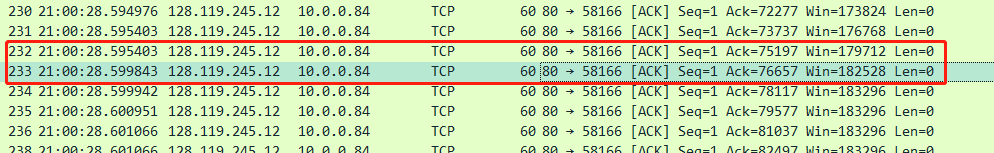
The ACKs for sequence number 45537 are circled in red tangles:



The ACKs for sequence number 64977 are circled in red tangles:



The ACKs for sequence number 73737 are circled in red tangles:



In SEQ/ACK analysis of the wireshark, we have the measured RTT:

ACK to frame 169: 0.022392000 seconds

ACK to frame 176: 0.019005000 seconds

ACK to frame 187: 0.023816000 seconds

ACK to frame 196: 0.022408000 seconds

ACK to frame 199: 0.021255000 seconds

ACK to frame 201: 0.025250000 seconds

We use the equation in section 3.5.2:

Where is recommended as 0.125.

Assume that the value of the Estimated RTT is equal to the measured RTT for the first segment.

Estimated RTT:

ACK to frame 176: (1-0.125) \* 0.022392000 + 0.125 \* 0.019005000 = 0.022 seconds

ACK to frame 187: (1-0.125) \* 0.022 + 0.125 \* 0.023816000 = 0.0222 seconds

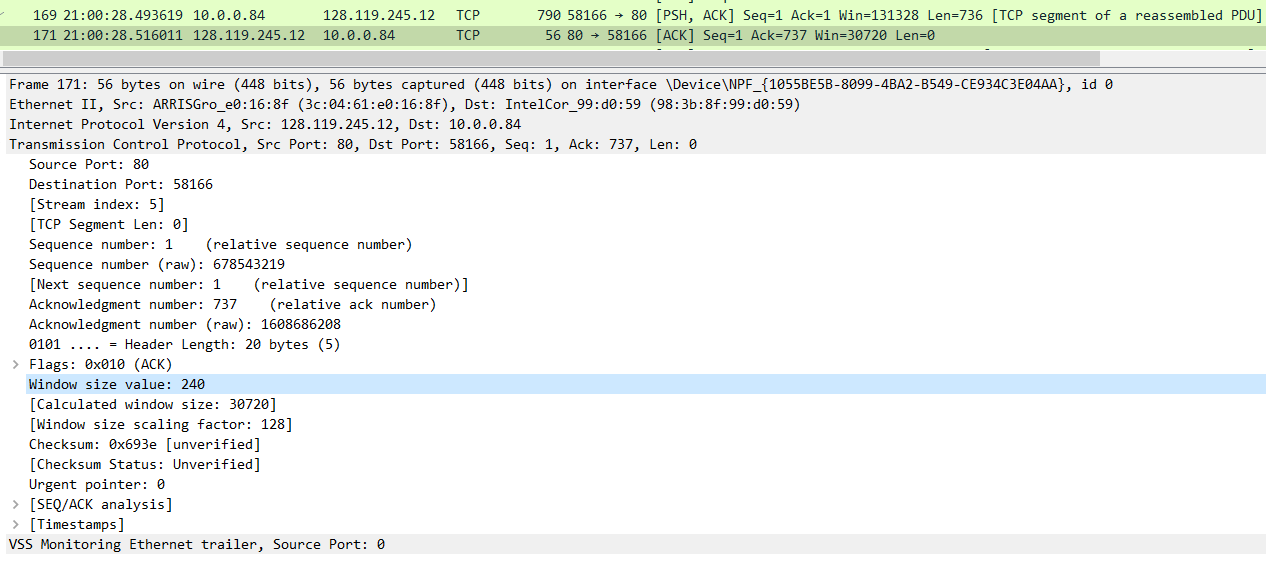
ACK to frame 196: (1-0.125) \* 0.0222 + 0.125 \* 0.022408000 = 0.0222 seconds

ACK to frame 199: (1-0.125) \* 0.0222 + 0.125 \* 0.021255000 = 0.0221 seconds

ACK to frame 201: (1-0.125) \* 0.0221 + 0.125 \* 0.02525000 = 0.0226 seconds

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?**

The minimum amount of available buffer space is 240.

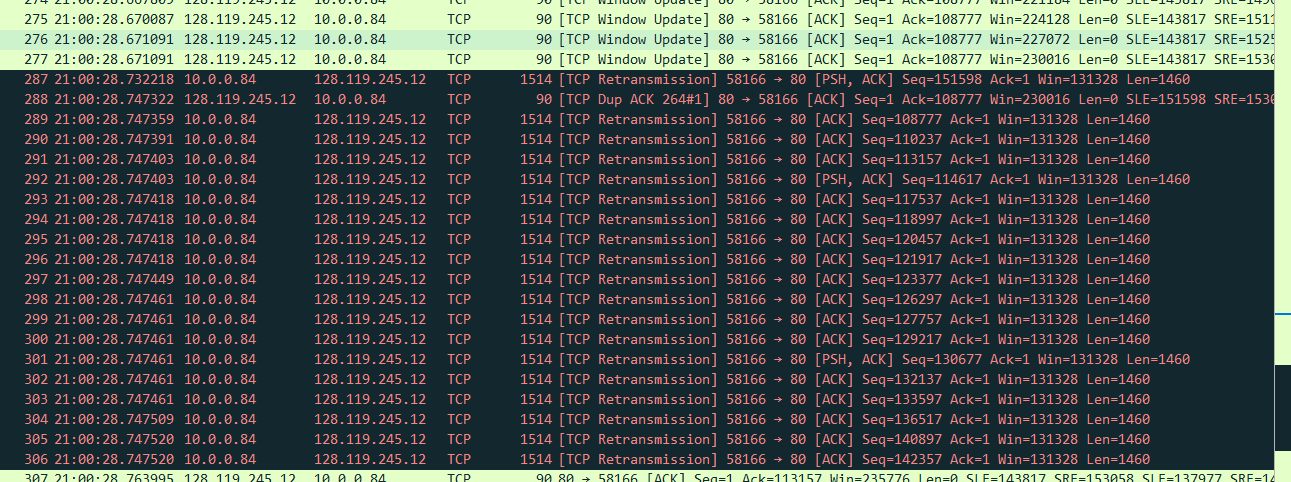


The lack of receiver buffer space never throttled the sender.

240, 286, 309, 331, 377, 400, 423, 468, 514, 537, 582, 605, 628, 274, 696, 719, 742, 765, 810, 833, 856, 879, 924, …, 1221, …, 1500, …, 2207.

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

Yes! Check the retransmission information.



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).**

The receiver typically acknowledges 1460 bytes data.

For the sequence number 1 from the client computer, the receiver sent an immediate cumulative ACK. For the other cases, the receiver always sent a partial ACK. Thus, the receiver sent multiple ACKs to acknowledge one TCP segment.

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

The throughput is almost 1.5 Mbps.

Wireshark -> Statistics -> TCP stream graph -> Throughput

