Warren Quattrocchi
CSC 138- section 3

Lab 3 - TCP

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.

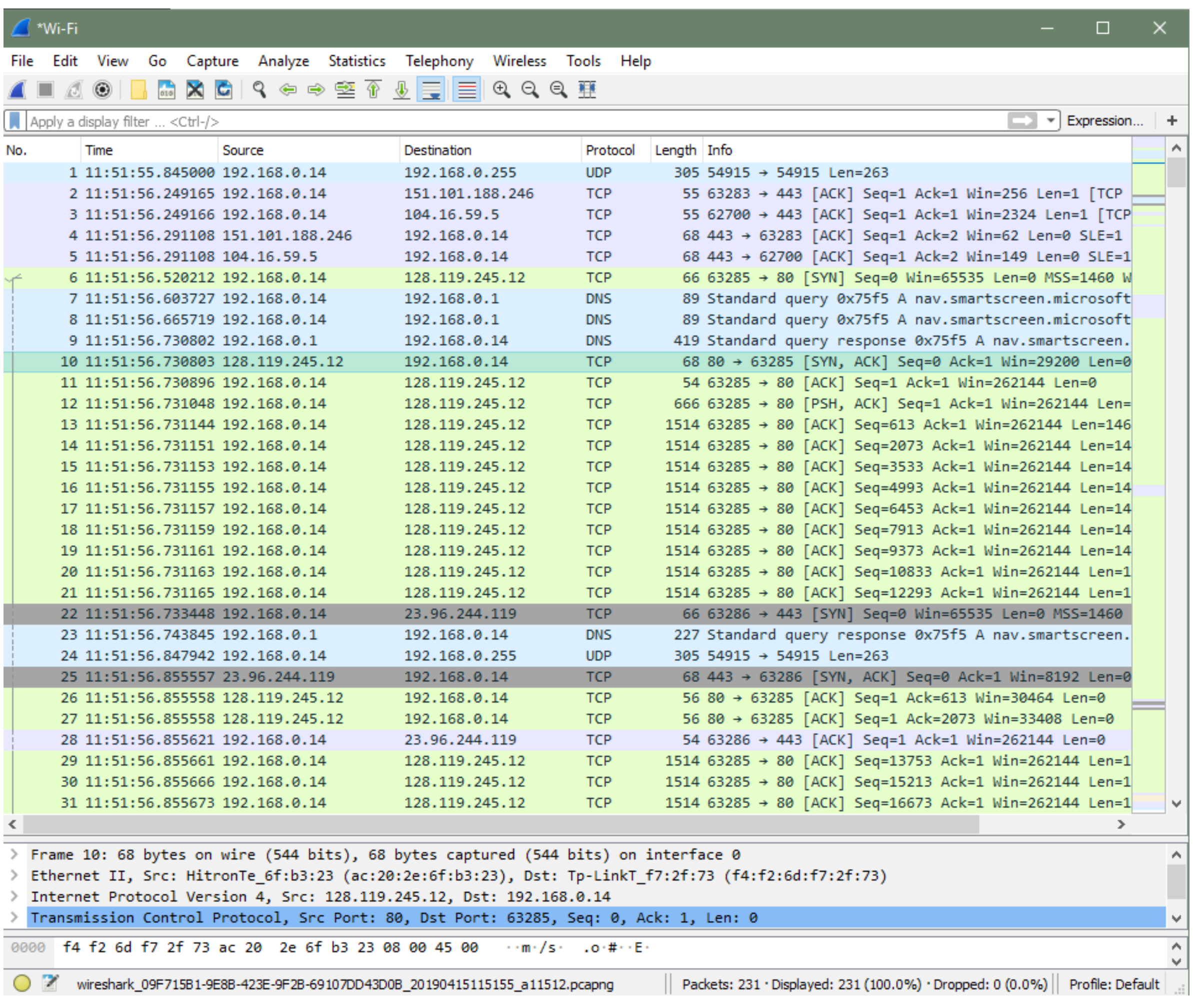
    IP address: 192.168.0.14
    Port: 63285

2. 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
    Port: 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?

        IP address: 192.168.0.14
        Port: 63285



3. **TCP Basics**

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?

   Seq = 0, it is identified by the [SYN] flag before the sequence number

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?

   Seq = 0, Ack = 1, the Ack is determined by adding 1 to the Seq # of the original SYN segment. The [SYN, ACK] flags identify 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.

   Seq = 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 239 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 239 for all subsequent segments.

   Seq = 1 sent at 11:51:56:731048 Ack received at 11:51:56:855558
   Seq = 613 sent at 11:51:56:731144 Ack received at 11:51:56:855558
   Seq = 2073 sent at 11:51:56:731151 Ack received at 11:51:56:872335
   Seq = 3533 sent at 11:51:56:731153 Ack received at 11:51:56:872336
   Seq = 4993 sent at 11:51:56:731155 Ack received at 11:51:56:872336
   Seq = 6453 sent at 11:51:56:731157 Ack received at 11:51:56:872336

   RTT
   segment 1: 124.4 ms = 0.124 seconds
   segment 2: 124.35 ms = 0.12435 seconds
   segment 3: 141.1 ms = 0.1411 seconds
   segment 4: 141.4 ms = 0.1414 seconds
   segment 5: 141.4 ms = 0.1414 seconds
   segment 6: 141.4 ms = 0.1414 seconds

   Estimated RTT
   segment 1: 0.124 seconds
   segment 2: .875 * 0.124 + 0.125 * 0.12435 = 0.12404
   segment 3: .875 * 0.12404 + 0.125 * .1411 = 0.12617
   segment 4: .875 * 0.12617 + 0.125 * .1414 = 0.12807
   segment 5: .875 * 0.12807 + 0.125 * .1414 = 0.12973
   segment 6: .875 * 0.12937 + 0.125 * .1414 = 0.13087

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

   1: 612, 2: 1460, 3: 1460, 4:1460, 5:1460, 6: 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 buffer size is 29200, the sender is not 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?

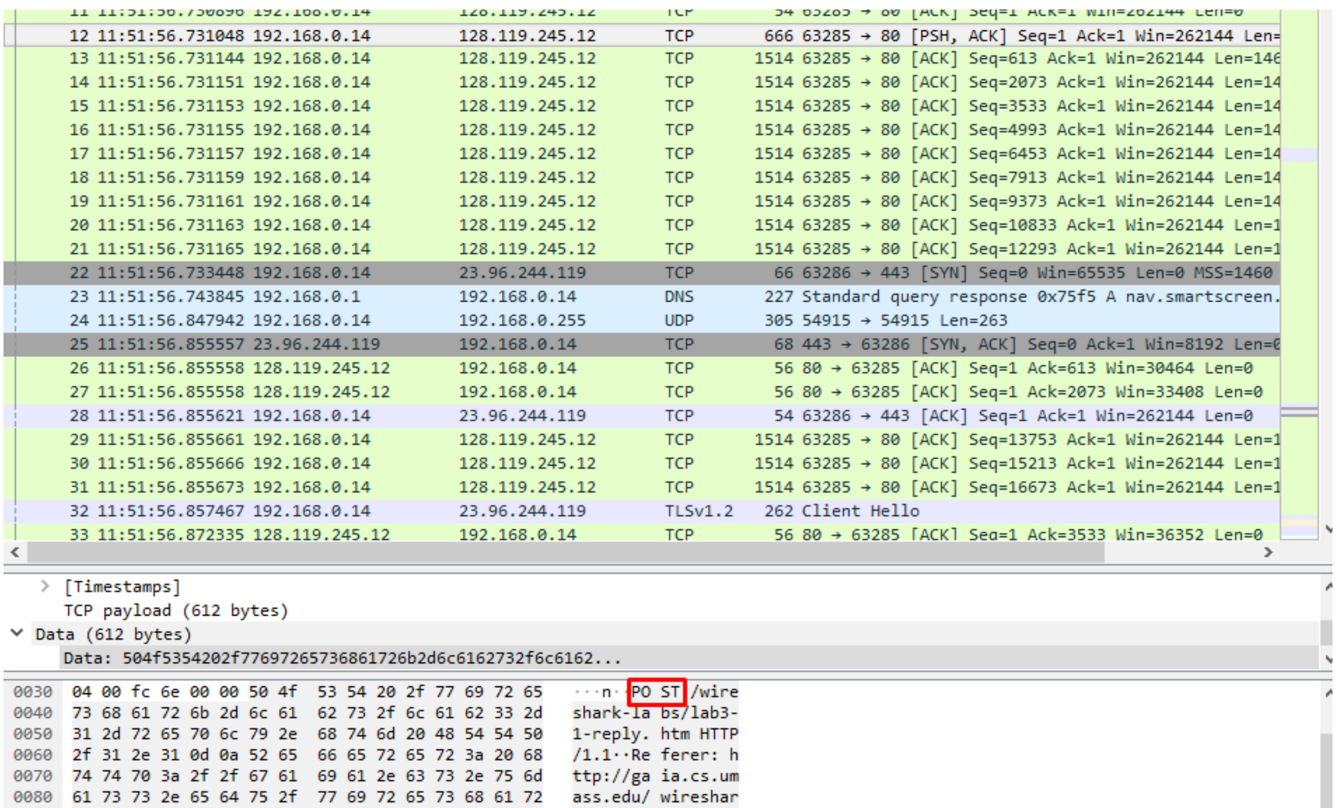
   No, they are none. I checked by looking for duplicate or lower sequence numbers

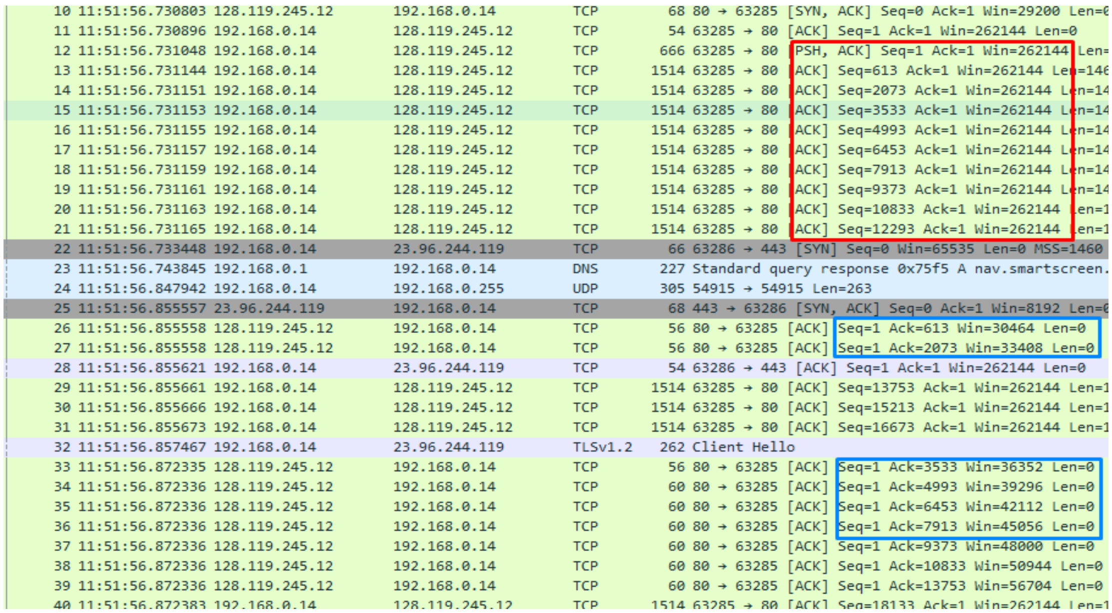
11. How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACKing every other received segment (see Table 3.2 on page 247 in the text).

   1460 bytes

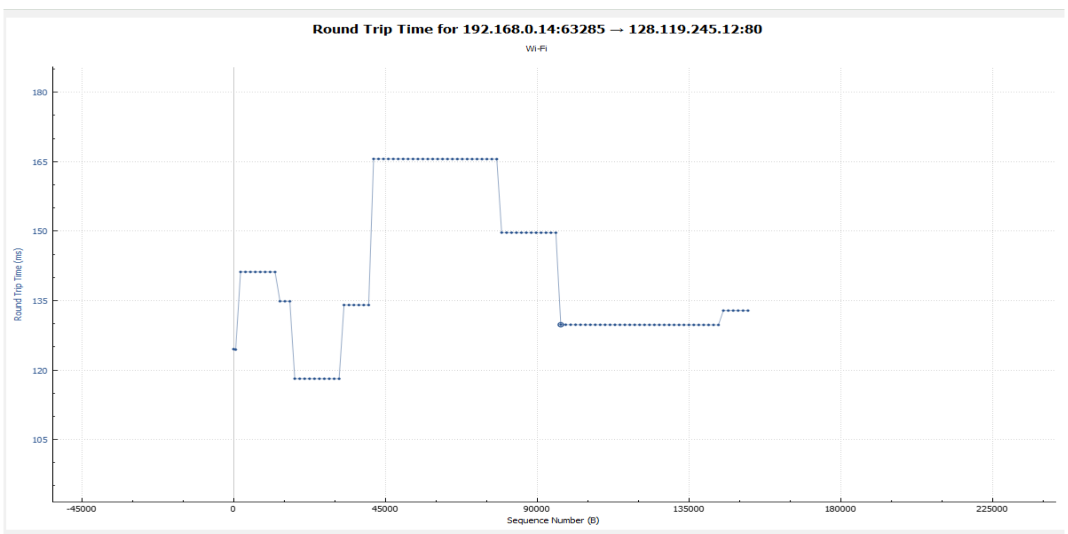
12. What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.

   Start time: 56.520212
   end time: 57:290348
   770.137 ms total in connection
   152,938 bytes sent – last ack received
   198.585 bytes/ms





10	11:51:56.730803	128.119.245.12	192.168.0.14	TCP	68	80 → 63285 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0
11	11:51:56.730896	192.168.0.14	128.119.245.12	TCP	54	63285 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=0
12	11:51:56.731048	192.168.0.14	128.119.245.12	TCP	666	63285 → 80 [PSH, ACK] Seq=1 Ack=1 Win=262144 Len=
13	11:51:56.731144	192.168.0.14	128.119.245.12	TCP	1514	63285 → 80 [ACK] Seq=613 Ack=1 Win=262144 Len=146
14	11:51:56.731151	192.168.0.14	128.119.245.12	TCP	1514	63285 → 80 [ACK] Seq=2073 Ack=1 Win=262144 Len=14
15	11:51:56.731153	192.168.0.14	128.119.245.12	TCP	1514	63285 → 80 [ACK] Seq=3533 Ack=1 Win=262144 Len=14
16	11:51:56.731155	192.168.0.14	128.119.245.12	TCP	1514	63285 → 80 [ACK] Seq=4993 Ack=1 Win=262144 Len=14
17	11:51:56.731157	192.168.0.14	128.119.245.12	TCP	1514	63285 → 80 [ACK] Seq=6453 Ack=1 Win=262144 Len=14
18	11:51:56.731159	192.168.0.14	128.119.245.12	TCP	1514	63285 → 80 [ACK] Seq=7913 Ack=1 Win=262144 Len=14
19	11:51:56.731161	192.168.0.14	128.119.245.12	TCP	1514	63285 → 80 [ACK] Seq=9373 Ack=1 Win=262144 Len=14
20	11:51:56.731163	192.168.0.14	128.119.245.12	TCP	1514	63285 → 80 [ACK] Seq=10833 Ack=1 Win=262144 Len=1
21	11:51:56.731165	192.168.0.14	128.119.245.12	TCP	1514	63285 → 80 [ACK] Seq=12293 Ack=1 Win=262144 Len=1
22	11:51:56.733448	192.168.0.14	23.96.244.119	TCP	66	63286 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460
23	11:51:56.743845	192.168.0.1	192.168.0.14	DNS	227	Standard query response 0x75f5 A nav.smartscreen.
24	11:51:56.847942	192.168.0.14	192.168.0.255	UDP	305	54915 → 54915 Len=263
25	11:51:56.855557	23.96.244.119	192.168.0.14	TCP	68	443 → 63286 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0
26	11:51:56.855558	128.119.245.12	192.168.0.14	TCP	56	80 → 63285 [ACK] Seq=1 Ack=613 Win=30464 Len=0
27	11:51:56.855558	128.119.245.12	192.168.0.14	TCP	56	80 → 63285 [ACK] Seq=1 Ack=2073 Win=33408 Len=0
28	11:51:56.855621	192.168.0.14	23.96.244.119	TCP	54	63286 → 443 [ACK] Seq=1 Ack=1 Win=262144 Len=0
29	11:51:56.855661	192.168.0.14	128.119.245.12	TCP	1514	63285 → 80 [ACK] Seq=13753 Ack=1 Win=262144 Len=1
30	11:51:56.855666	192.168.0.14	128.119.245.12	TCP	1514	63285 → 80 [ACK] Seq=15213 Ack=1 Win=262144 Len=1
31	11:51:56.855673	192.168.0.14	128.119.245.12	TCP	1514	63285 → 80 [ACK] Seq=16673 Ack=1 Win=262144 Len=1
32	11:51:56.857467	192.168.0.14	23.96.244.119	TLSv1.2	262	Client Hello
33	11:51:56.872335	128.119.245.12	192.168.0.14	TCP	56	80 → 63285 [ACK] Seq=1 Ack=3533 Win=36352 Len=0
34	11:51:56.872336	128.119.245.12	192.168.0.14	TCP	60	80 → 63285 [ACK] Seq=1 Ack=4993 Win=39296 Len=0
35	11:51:56.872336	128.119.245.12	192.168.0.14	TCP	60	80 → 63285 [ACK] Seq=1 Ack=6453 Win=42112 Len=0
36	11:51:56.872336	128.119.245.12	192.168.0.14	TCP	60	80 → 63285 [ACK] Seq=1 Ack=7913 Win=45056 Len=0
37	11:51:56.872336	128.119.245.12	192.168.0.14	TCP	60	80 → 63285 [ACK] Seq=1 Ack=9373 Win=48000 Len=0
38	11:51:56.872336	128.119.245.12	192.168.0.14	TCP	60	80 → 63285 [ACK] Seq=1 Ack=10833 Win=50944 Len=0
39	11:51:56.872336	128.119.245.12	192.168.0.14	TCP	60	80 → 63285 [ACK] Seq=1 Ack=13753 Win=56704 Len=0
40	11:51:56.872383	192.168.0.14	128.119.245.12	TCP	1514	63285 → 80 [ACK] Seq=18133 Ack=1 Win=262144 Len=1

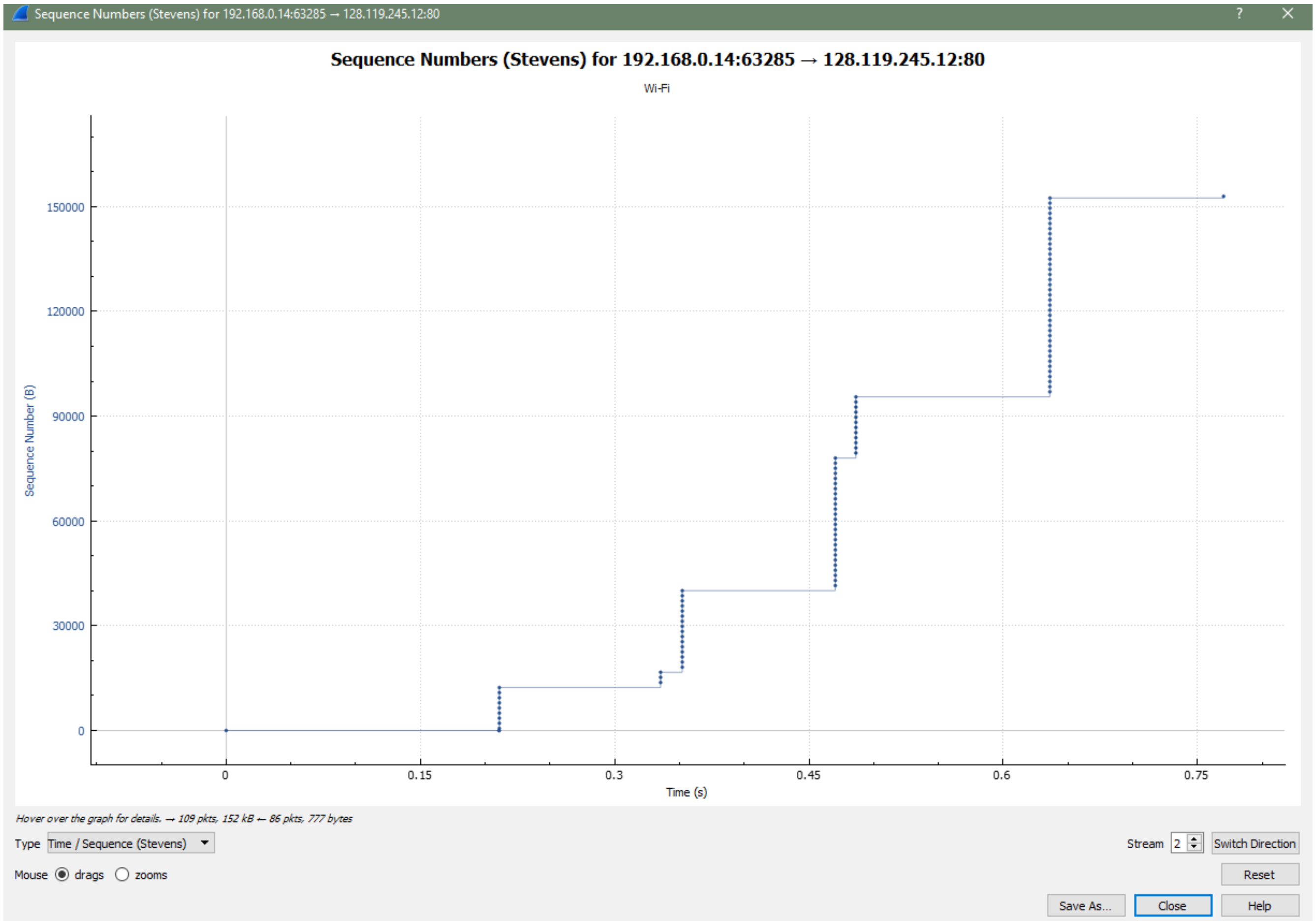


4. TCP congestion control in action

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.

    Yes, there are clear increases in speed and plateaus in the graph where it appears congestion control is kicking in to avoid data loss.

    The growth in the chart appears very linear, as opposed to the exponential growth discussed during lecture.


Sequence Numbers (Stevens) for 192.168.0.14:63285 → 128.119.245.12:80
Sequence Numbers (Stevens) for 192.168.0.14:63285 → 128.119.245.12:80
Wi-Fi
Sequence Number (B)
150000
120000
90000
60000
30000
0
0
0.15
0.3
0.45
0.6
0.75
Time (s)
Hover over the graph for details. → 109 pkts, 152 kB ← 86 pkts, 777 bytes
Type
Time / Sequence (Stevens)
Mouse
drags
zooms
Stream
2
Switch Direction
Reset
Save As...
Close
Help