Igor Oleshko
CSC 138
Professor Dai
04/12/2023

Wireshark 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?**

   - Source IP Address: 192.168.1.102
   - Source TCP Port Number: 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?**

   - Destination IP Address: 128.119.245.12
   - Destination TCP Port Number: 80

```
No.     Time           Source                Destination           Protocol Length Info
    199 5.297341       192.168.1.102         128.119.245.12        HTTP     104    POST /ethereal-labs/lab3-1-reply.htm HTTP/1.1  (text/plain)
Frame 199: 104 bytes on wire (832 bits), 104 bytes captured (832 bits)
Ethernet II, Src: Actionte_8a:70:1a (00:20:e0:8a:70:1a), Dst: LinksysG_da:af:73 (00:06:25:da:af:73)
Internet Protocol Version 4, Src: 192.168.1.102, Dst: 128.119.245.12
Transmission Control Protocol, Src Port: 1161, Dst Port: 80, Seq: 164041, Ack: 1, Len: 50
    Source Port: 1161
    Destination 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?**

   - Source IP Address: 10.117.215.138
   - Source TCP Port Number: 49766

```
No.     Time           Source                Destination           Protocol Length Info
    583 2.473466       10.117.215.138        128.119.245.12        HTTP     355    POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1  (text/plain)
Frame 583: 355 bytes on wire (2840 bits), 355 bytes captured (2840 bits) on interface \Device\NPF_{B7184193-E065-4674-90E8-E6BD1164D2FE}, id 0
Ethernet II, Src: IntelCor_0c:37:1c (3c:9c:0f:0c:37:1c), Dst: Alcatel-_b5:de:c1 (94:24:e1:b5:de:c1)
Internet Protocol Version 4, Src: 10.117.215.138, Dst: 128.119.245.12
Transmission Control Protocol, Src Port: 49766, Dst Port: 80, Seq: 152749, Ack: 1, Len: 301
    Source Port: 49766
    Destination Port: 80
```

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?**

   - SYN Sequence Number: 0
   - When the flag for the SYN is set equal to 1 (Set) then we know that this TCP segment would be identified as an SYN segment. However, if it is a 0 (Not set) that means that it is not an 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?**

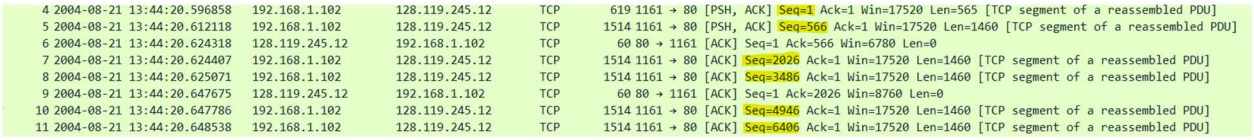
    - SYNACK Sequence Number: 0
    - Acknowledgement Field Value: 1
    - The acknowledgment value is determined by the sequence number of next byte expected from other side.
    - Like the last segment, we need the SYN flag to be equal to 1 (Set) but this time we will also need the Acknowledgment flag to be equal to 1 (Set) as well. This causes the segment to be identified as SYN ACK.

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.**

    - HTTP POST Sequence Number: 1

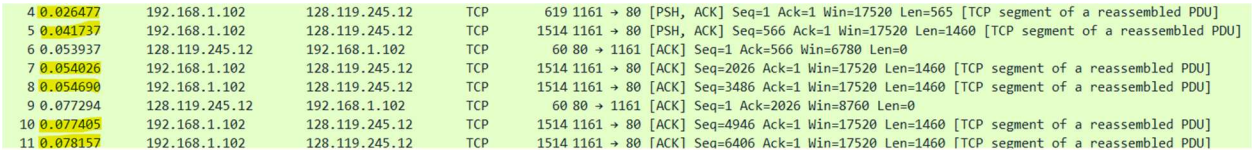
7. Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection. What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST)? At what time was each segment sent? When was the ACK for each segment received? 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.

    - Sequence numbers of the first six segments
        - **First:** 1
        - **Second:** 566
        - **Third:** 2026
        - **Fourth:** 3486
        - **Fifth:** 4946
        - **Sixth:** 6406



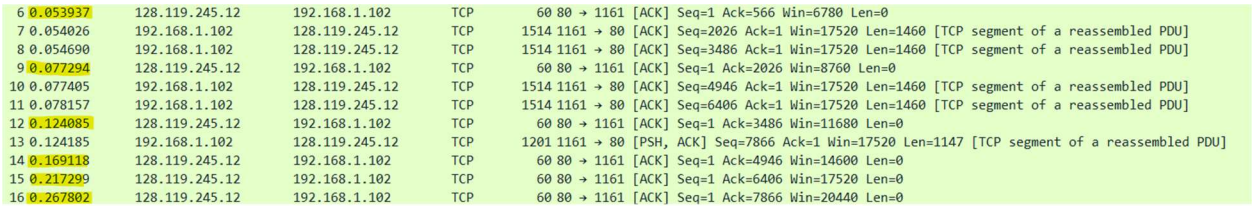
4	2004-08-21 13:44:20.596858	192.168.1.102	128.119.245.12	TCP	619	1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len=565 [TCP segment of a reassembled PDU]
5	2004-08-21 13:44:20.612118	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [PSH, ACK] Seq=566 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
6	2004-08-21 13:44:20.624318	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=566 Win=6780 Len=0
7	2004-08-21 13:44:20.624407	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=2026 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
8	2004-08-21 13:44:20.625071	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
9	2004-08-21 13:44:20.647675	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=2026 Win=8760 Len=0
10	2004-08-21 13:44:20.647786	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=4946 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
11	2004-08-21 13:44:20.648538	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]

- Time at which each segment got sent
  - **First:** 0.026477 seconds
  - **Second:** 0.041737 seconds
  - **Third:** 0.054026 seconds
  - **Fourth:** 0.054690 seconds
  - **Fifth:** 0.077405 seconds
  - **Sixth:** 0.078157 seconds



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 [TCP segment of a reassembled PDU]
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 [TCP segment of a reassembled PDU]
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 [TCP segment of a reassembled PDU]
8	0.054690	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
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 [TCP segment of a reassembled PDU]
11	0.078157	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]

- Time at which the ACK for each segment was received
  - **First:** 0.053937 seconds
  - **Second:** 0.077294 seconds
  - **Third:** 0.124085 seconds
  - **Fourth:** 0.169118 seconds
  - **Fifth:** 0.217299 seconds
  - **Sixth:** 0.267802 seconds



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 [TCP segment of a reassembled PDU]
8	0.054690	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
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 [TCP segment of a reassembled PDU]
11	0.078157	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
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 [TCP segment of a reassembled PDU]
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

- Calculated RTT based on difference
  - **First:** 0.053937 - 0.026477 = 0.02746 seconds
  - **Second:** 0.077294 - 0.041737 = 0.035557 seconds
  - **Third:** 0.124085 - 0.054026 = 0.070059 seconds
  - **Fourth:** 0.169118 - 0.054690 = 0.114428 seconds
  - **Fifth:** 0.217299 - 0.077405 = 0.139894 seconds
  - **Sixth:** 0.267802 - 0.078156 = 0.189646 seconds

- Estimated RTT
  - **First:** 0.02746 seconds
  - **Second:** 0.875 * 0.02746 + 0.125 * 0.035557 = 0.028472 seconds
  - **Third:** 0.875 * 0.028472 + 0.125 * 0.070059 = 0.033670 seconds
  - **Fourth:** 0.875 * 0.033670 + 0.125 * 0.114428 = 0.043765 seconds
  - **Fifth:** 0.875 * 0.043765 + 0.125 * 0.139894 = 0.055781 seconds
  - **Sixth:** 0.875 * 0.055781 + 0.125 * 0.189646 = 0.072514 seconds

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

- **First:** 565 bytes
- **The Rest:** 1460 bytes

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 [TCP segment of a reassembled PDU]
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 [TCP segment of a reassembled PDU]
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 [TCP segment of a reassembled PDU]
8	0.054690	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
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 [TCP segment of a reassembled PDU]
11	0.078157	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]

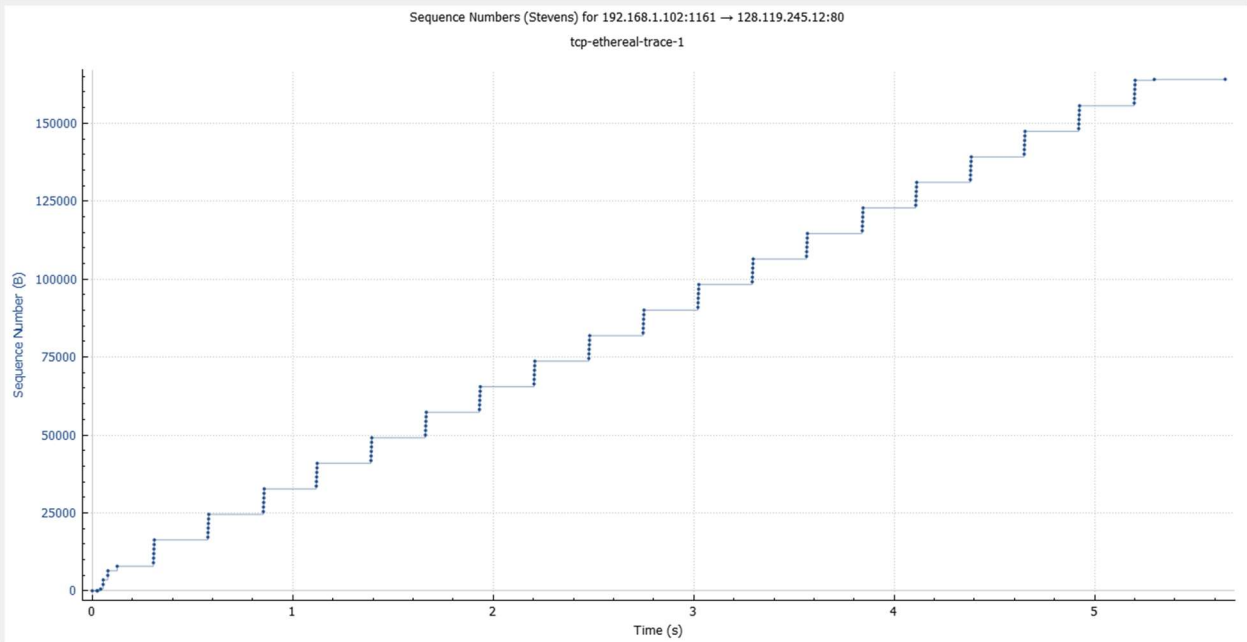
9. What is the minimum amount of available buffer space advertised at the received for the entire trace? Does the lack of receiver buffer space ever throttle the sender?

- The minimum amount of available buffer space is 5840 bytes. You can see that the buffer space grows as we continue sending data across the TCP pipeline. However, the available buffer space maxes out at 62780 bytes.
- The lack of receiver buffer space does not seem to influence throttling of the sender.

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
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 [TCP segment of a reassembled PDU]
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 [TCP segment of a reassembled PDU]
6	0.053937	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=566 Win=6780 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?

- No, there were no retransmitted segments in the trace file and we were able to check this by creating a Time Sequence (Stevens) graph which is showcased below. Looking at the graph we can see that all of the packets are being transmitted in an up direction which means that the sequence number goes up as time goes on. However, if we had a retransmitted segment that means that the sequence number for that segment would have a lower value than the neighboring segments since we are going back down in length.



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).

- Typically, the receiver would acknowledge 1460 bytes in an ACK.
- There are many difference cases where the receiver is ACKing every other received segment. We can see that here in segment 59 we went from 35049 to 37969 which is a 2920 bytes and if we divide that by two then we would get 1460.

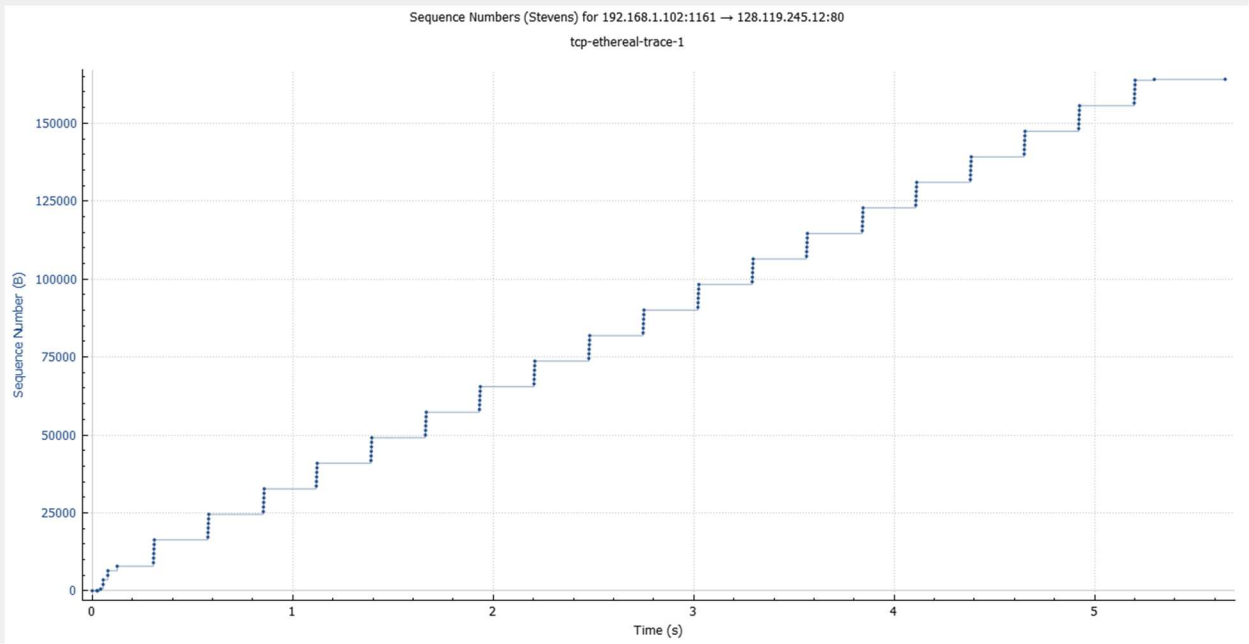
54	1.118133	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=35049 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
55	1.119029	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=36509 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
56	1.119858	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=37969 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
57	1.120902	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=39429 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
58	1.121891	192.168.1.102	128.119.245.12	TCP	946	1161 → 80 [PSH, ACK] Seq=40889 Ack=1 Win=17520 Len=892 [TCP segment of a reassembled PDU]
59	1.200421	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=35049 Win=62780 Len=0
60	1.265026	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=37969 Win=62780 Len=0

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

- To solve for the throughput, we will have to determine the difference in time between the initial TCP segment and the final TCP segment before the HTTP OK. Along with the difference in Bytes between the initial TCP segment (1) and the final TCP segment (164091). Doing the difference in bytes over difference in time gives us throughput.
  - **Time:** Final Time – Initial Time = 5.455830 - 0.026477 = 5.429353 seconds
  - **Bytes:** Final Bytes – Initial Bytes = 164091 - 1 = 164090 bytes
  - **Throughput:** Bytes/Times = 164090 / 5.429353 = 30,222.8 bytes per 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.

- We know that TCP's slow start phase occurs when the connection begins and the system continues increasing the rate until the first loss event. Therefore, I believe that the initial slow start phase begins in the very beginning and ends with packet 13 at 0.124185 seconds. Packet #13 is the last packet before the segments start going in a vertical-looking manner.
- I believe that congestion avoidance takes over at segment 18 which is the first segment that is seen in a vertical fashion. Here congestion avoidance starts to take over. The way that the measured data differs is that it is always on an positive trend while the text shows that sometimes it goes up and down.



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

- It doesn't seem like this TCP has a slow start as it doesn't continue sending packets until its first loss event.
- However, it does seem like the congestion avoidance starts at segment 20 which is the segment right when the vertical segment pattern begins and we can see that the pattern continues through multiple occasions.