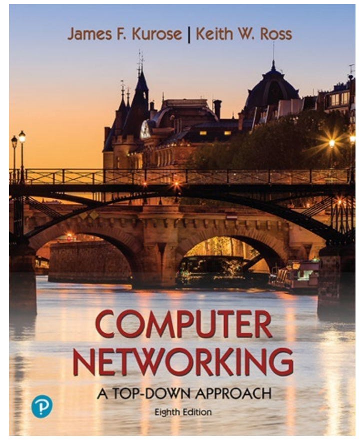
Wireshark Lab: TCP v8.0
SOLUTIONS

Supplement to *Computer Networking: A Top-Down Approach, 8th ed.,* J.F. Kurose and K.W. Ross

"Tell me and I forget. Show me and I remember. Involve me and I understand." Chinese proverb



The answers below are based on the trace file *tcp-wireshark-trace1-1* in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces-8.1.zip

TCP Basics

Answer the following questions for the TCP segments:

- *What is the IP address and TCP port number used by the client computer (source) that is transferring the alice.txt file to gaia.cs.umass.edu?*

Solution: Client computer (source)
IP address: `192.168.86.68`
TCP port number: `55639`

- *What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?*

Solution: Destination computer: gaia.cs.umass.edu
IP address: `128.119.245.12`
Receiving on TCP port number: `80`
Sending on TCP port number: `55639`

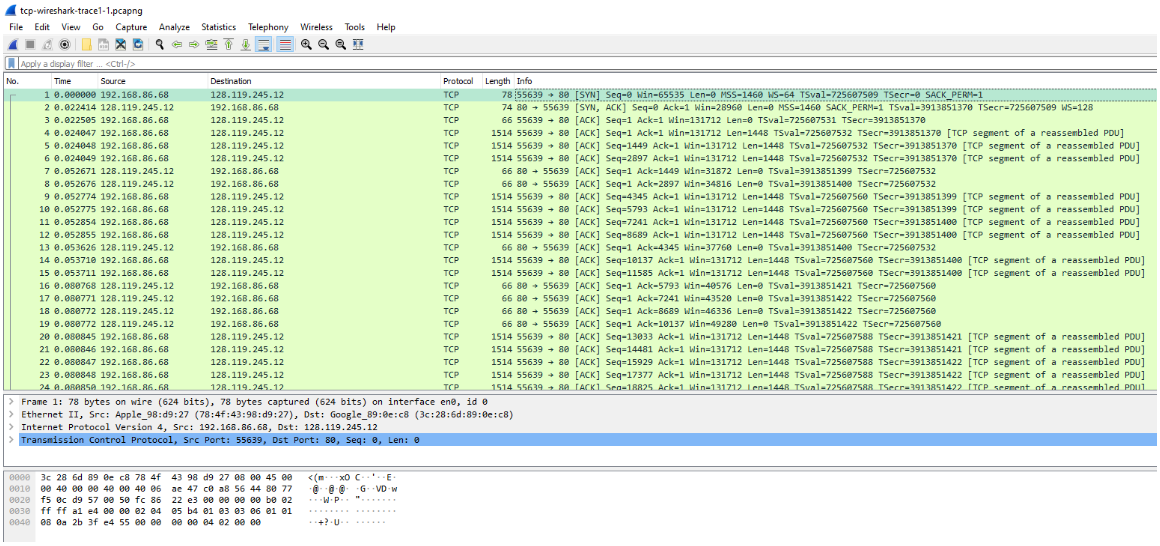


Figure 1: IP addresses and TCP port numbers of the client computer (source) and gaia.cs.umass.edu

- *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?*

Solution: Sequence number of the TCP SYN segment is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu. The raw value is 4236649187 in this trace.

The SYN flag is set to 1 and it indicates that this segment is a SYN segment.

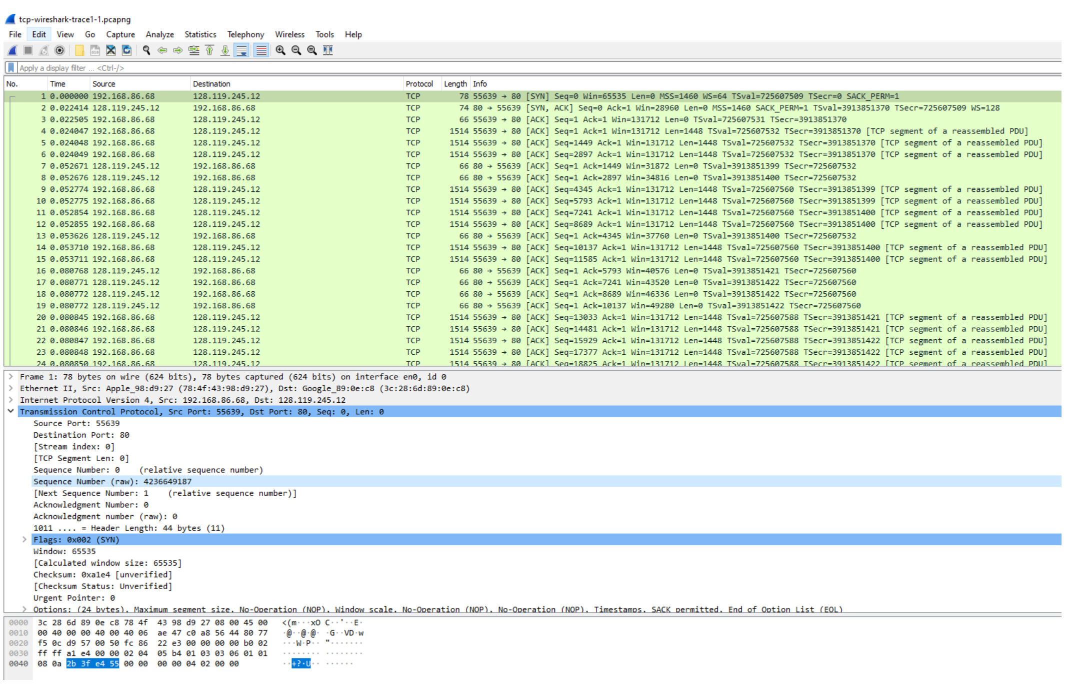


Figure 2: Sequence number of the TCP SYN segment

- *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?*

Solution: Sequence number of the SYNACK segment from gaia.cs.umass.edu to the client computer in reply to the SYN has the value of `1068969752` in this trace.

The value of the ACKnowledgement field in the SYNACK segment is 4236649188. The value of the ACKnowledgement field in the SYNACK segment is determined by gaia.cs.umass.edu by adding 1 to the initial sequence number of SYN segment from the client computer (i.e., the sequence number of the SYN segment initiated by the client computer is 4236649187.).

The SYN flag and Acknowledgement flag in the segment are set to 1 and they indicate that this segment is a SYNACK segment.

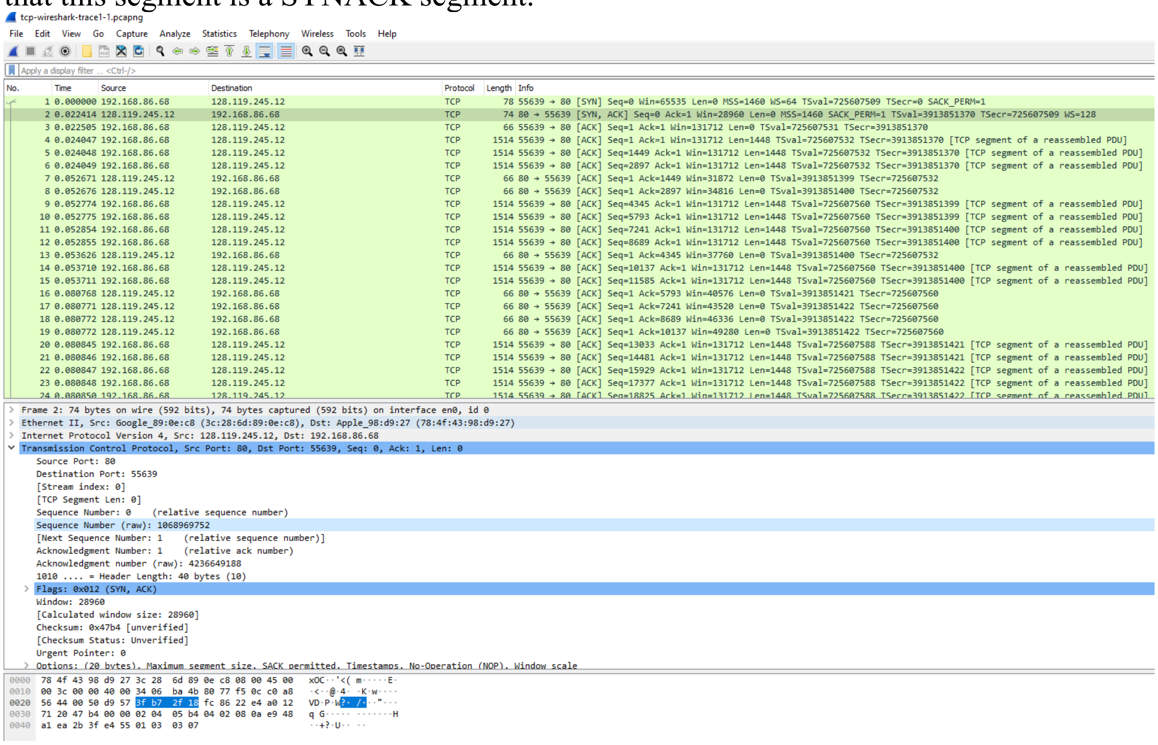


Figure 3: Sequence number and Acknowledgement number of the SYNACK segment

- *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.*

Solution: No. 4 segment is the TCP segment containing the HTTP POST command. The sequence number of this segment has the relative value of 1/raw value of 4236649188.

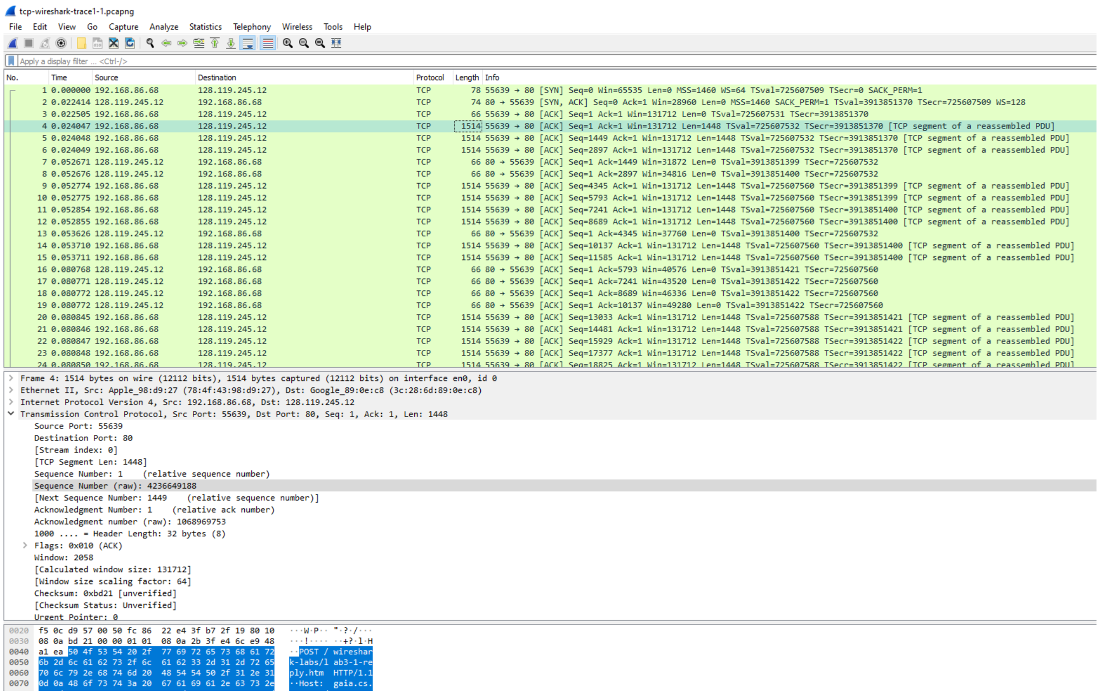


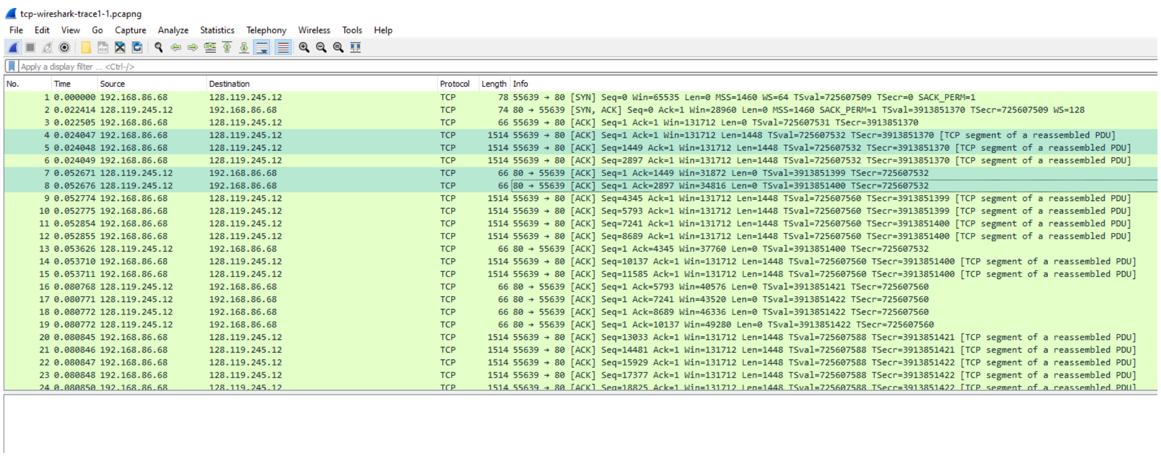
Figure 4: Sequence number of the TCP segment containing the HTTP POST command

- *Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection.*
  - *At what time was the first segment (the one containing the HTTP POST) in the data-transfer part of the TCP connection sent?*
  - *At what time was the ACK for this first data-containing segment received?*
  - *What is the RTT for this first data-containing segment?*
  - *What is the RTT value between the second data-carrying TCP segment and its ACK?*
  - *What is the* `EstimatedRTT` *value (see Section 3.5.3, in the text) after the ACK for the second data-carrying segment is received? Assume that in making this calculation after the received of the ACK for the second segment, that the initial value of* `EstimatedRTT` *is equal to the measured RTT for the first segment, and then is computed using the* `EstimatedRTT` *equation on page 242, and a value of* $\alpha = 0.125$.

Solution: The HTTP POST segment is considered as the first segment, and it was sent at time 0.024047 s.
Segment 7 is the first ACK; it was received at 0.052671 s.
The RTT for the first data containing segment is 0.052671 – 0.024047 = 0.028624 s.

Segment 5 is the second data segment sent at 0.024048 s and segment 8 is the corresponding ACK 1 received at 0.052676 s. The RTT value is 0.052676 – 0.024048 = 0.028628 s.
The EstimatedRTT after the second data-carrying segment is:

```
EstimatedRTT = 0.875 * EstimatedRTT + 0.125 * SampleRTT =
0.875 * 0.028624 + 0.125 * 0.028628 = 0.0286245 s
```

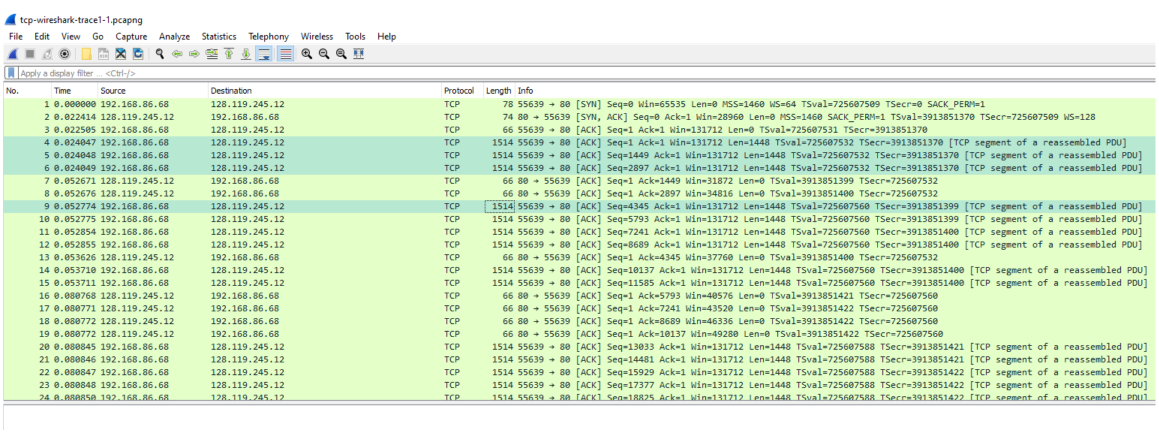

tcp-wireshark-trace1-1.pcapng

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.86.68	128.119.245.12	TCP	78	55639 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=725607509 TSecr=0 SACK_PERM=1
2	0.022414	128.119.245.12	192.168.86.68	TCP	74	80 → 55639 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=3913851370 TSecr=725607509 WS=128
3	0.022505	192.168.86.68	128.119.245.12	TCP	66	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=725607531 TSecr=3913851370
4	0.024047	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP segment of a reassembled PDU]
5	0.024048	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1449 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP segment of a reassembled PDU]
6	0.024049	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=2897 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP segment of a reassembled PDU]
7	0.052671	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=1449 Win=31872 Len=0 TSval=3913851399 TSecr=725607532
8	0.052676	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=2897 Win=34816 Len=0 TSval=3913851400 TSecr=725607532
9	0.052774	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=4345 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [TCP segment of a reassembled PDU]
10	0.052775	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=5793 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [TCP segment of a reassembled PDU]
11	0.052854	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=7241 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP segment of a reassembled PDU]
12	0.052855	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=8689 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP segment of a reassembled PDU]
13	0.053626	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=4345 Win=37760 Len=0 TSval=3913851400 TSecr=725607532
14	0.053710	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=10137 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP segment of a reassembled PDU]
15	0.053711	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=11585 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP segment of a reassembled PDU]
16	0.080768	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=5793 Win=40576 Len=0 TSval=3913851421 TSecr=725607560
17	0.080771	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=7241 Win=43520 Len=0 TSval=3913851422 TSecr=725607560
18	0.080772	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=8689 Win=46336 Len=0 TSval=3913851422 TSecr=725607560
19	0.080772	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=10137 Win=49280 Len=0 TSval=3913851422 TSecr=725607560
20	0.080845	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=13033 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851421 [TCP segment of a reassembled PDU]
21	0.080846	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=14481 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851421 [TCP segment of a reassembled PDU]
22	0.080847	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=15929 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851422 [TCP segment of a reassembled PDU]
23	0.080848	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=17377 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851422 [TCP segment of a reassembled PDU]
24	0.080850	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=18825 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851422 [TCP segment of a reassembled PDU]

Figure 5: First two segments and corresponding ACKs

- *What is the length of each of the first four TCP segments?*

Solution: Length of each of the first 4 TCP segments: 1448 bytes + 32 bytes of header.

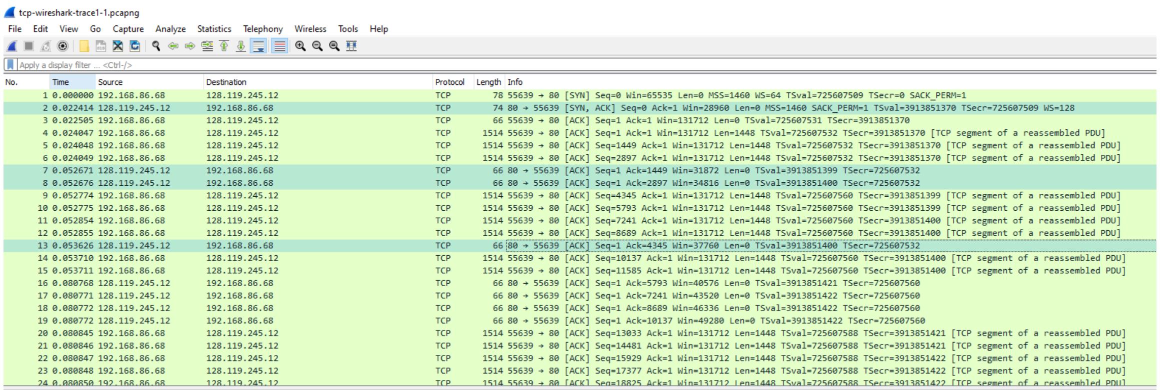

tcp-wireshark-trace1-1.pcapng

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.86.68	128.119.245.12	TCP	78	55639 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=725607509 TSecr=0 SACK_PERM=1
2	0.022414	128.119.245.12	192.168.86.68	TCP	74	80 → 55639 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=3913851370 TSecr=725607509 WS=128
3	0.022505	192.168.86.68	128.119.245.12	TCP	66	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=725607531 TSecr=3913851370
4	0.024047	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP segment of a reassembled PDU]
5	0.024048	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1449 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP segment of a reassembled PDU]
6	0.024049	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=2897 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP segment of a reassembled PDU]
7	0.052671	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=1449 Win=31872 Len=0 TSval=3913851399 TSecr=725607532
8	0.052676	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=2897 Win=34816 Len=0 TSval=3913851400 TSecr=725607532
9	0.052774	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=4345 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [TCP segment of a reassembled PDU]
10	0.052775	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=5793 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [TCP segment of a reassembled PDU]
11	0.052854	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=7241 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP segment of a reassembled PDU]
12	0.052855	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=8689 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP segment of a reassembled PDU]
13	0.053626	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=4345 Win=37760 Len=0 TSval=3913851400 TSecr=725607532
14	0.053710	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=10137 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP segment of a reassembled PDU]
15	0.053711	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=11585 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP segment of a reassembled PDU]
16	0.080768	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=5793 Win=40576 Len=0 TSval=3913851421 TSecr=725607560
17	0.080771	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=7241 Win=43520 Len=0 TSval=3913851422 TSecr=725607560
18	0.080772	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=8689 Win=46336 Len=0 TSval=3913851422 TSecr=725607560
19	0.080772	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=10137 Win=49280 Len=0 TSval=3913851422 TSecr=725607560
20	0.080845	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=13033 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851421 [TCP segment of a reassembled PDU]
21	0.080846	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=14481 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851421 [TCP segment of a reassembled PDU]
22	0.080847	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=15929 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851422 [TCP segment of a reassembled PDU]
23	0.080848	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=17377 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851422 [TCP segment of a reassembled PDU]
24	0.080850	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=18825 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851422 [TCP segment of a reassembled PDU]

Figure 6: Lengths of segments 1-4

- *What is the minimum amount of available buffer space advertised to the client by gaia.cs.umass.edu among these first four data-carrying TCP segments? Does the lack of receiver buffer space ever throttle the sender for these first four data-carrying segments?*

Solution: The minimum amount of buffer space (receiver window) advertised at gaia.cs.umass.edu for the entire trace is 28960 bytes, which shows in the first acknowledgement from the server. This receiver window grows steadily after that. The sender is never throttled due to lack of receiver buffer space by inspecting this trace.



tcp-wireshark-trace1-1.pcapng

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.86.68	128.119.245.12	TCP	78	55639 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=725607509 TSecr=0 SACK_PERM=1
2	0.022414	128.119.245.12	192.168.86.68	TCP	74	80 → 55639 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=3913851370 TSecr=725607509 WS=128
3	0.022505	192.168.86.68	128.119.245.12	TCP	66	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=725607531 TSecr=3913851370
4	0.024047	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP segment of a reassembled PDU]
5	0.024048	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1449 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP segment of a reassembled PDU]
6	0.024049	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=2897 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP segment of a reassembled PDU]
7	0.052671	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=1449 Win=31872 Len=0 TSval=3913851399 TSecr=725607532
8	0.052676	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=2897 Win=34816 Len=0 TSval=3913851400 TSecr=725607532
9	0.052774	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=4345 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [TCP segment of a reassembled PDU]
10	0.052775	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=5793 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [TCP segment of a reassembled PDU]
11	0.052854	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=7241 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP segment of a reassembled PDU]
12	0.052855	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=8689 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP segment of a reassembled PDU]
13	0.053626	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=4345 Win=37760 Len=0 TSval=3913851400 TSecr=725607532
14	0.053710	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=10137 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP segment of a reassembled PDU]
15	0.053711	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=11585 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP segment of a reassembled PDU]
16	0.080768	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=5793 Win=40576 Len=0 TSval=3913851421 TSecr=725607560
17	0.080771	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=7241 Win=43520 Len=0 TSval=3913851422 TSecr=725607560
18	0.080772	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=8689 Win=46336 Len=0 TSval=3913851422 TSecr=725607560
19	0.080772	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=10137 Win=49280 Len=0 TSval=3913851422 TSecr=725607560
20	0.080845	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=13033 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851421 [TCP segment of a reassembled PDU]
21	0.080846	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=14481 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851421 [TCP segment of a reassembled PDU]
22	0.080847	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=15929 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851422 [TCP segment of a reassembled PDU]
23	0.080848	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=17377 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851422 [TCP segment of a reassembled PDU]
24	0.080850	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=18825 Ack=1 Win=131712 Len=1448 TSval=725607588 TSecr=3913851422 [TCP segment of a reassembled PDU]

Figure 7: Minimum receive window advertised at gaia.cs.umass.edu for the first four data-carrying segments

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

Solution: There are no retransmitted segments in the trace file. We can verify this by checking the sequence numbers of the TCP segments in the trace file. In the *Time-Sequence-Graph (Stevens)* of this trace, all sequence numbers from the source (192.168.86.68) to the destination (128.119.245.12) are increasing monotonically with respect to time. If there is a retransmitted segment, the sequence number of this retransmitted segment should be smaller than those of its neighboring segments.

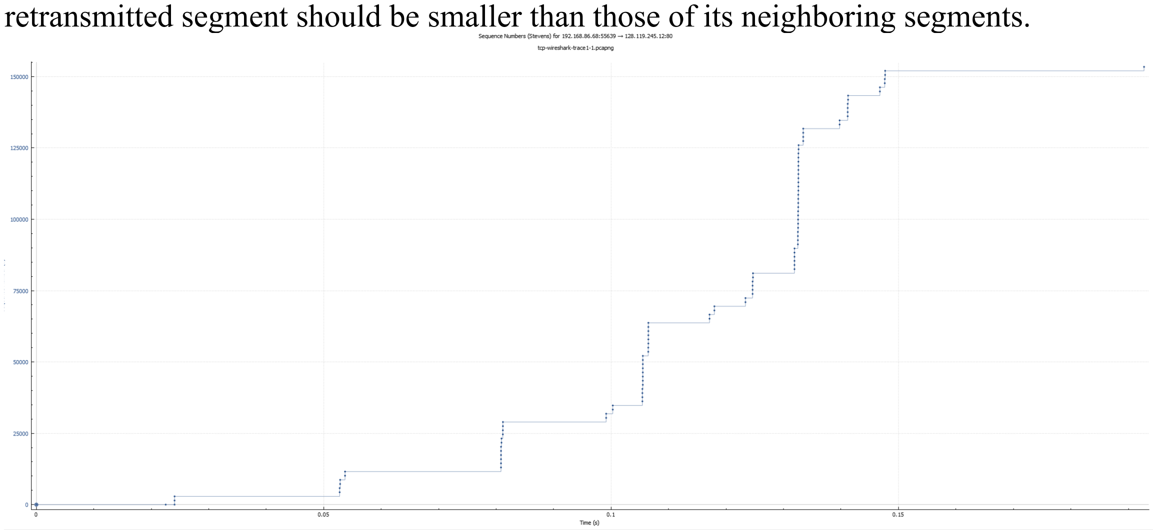


Figure 8: Sequence numbers of the segments from the source (`192.168.86.68`) to the destination (`128.119.245.12`)

- *How much data does the receiver typically acknowledge in an ACK among the first ten data-carrying segments sent from the client to gaia.cs.umass.edu? Can you identify cases where the receiver is ACKing every other received segment (see Table 3.2 in the text) among these first ten data-carrying segments?*

Solution: The acknowledged sequence numbers of the ACKs are listed as follows.

	acknowledged sequence number	acknowledged data
ACK 1	1449	1448
ACK 2	2897	1448
ACK 3	4345	1448
ACK 4	5793	1448
ACK 5	7241	1448
ACK 6	8689	1448
ACK 7	10137	1448
ACK 8	11585	1448
ACK 9	13033	1448
ACK 10	14481	1448

The difference between the acknowledged sequence numbers of two consecutive ACKs indicates the data received by the server between these two ACKs. By inspecting the amount of acknowledged data by each ACK, there are no cases where the receiver is ACKing every other segment for the first 10 segments.

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

Solution: The computation of TCP throughput largely depends on the selection of averaging time period. As a common throughput computation, in this question, we select the average time period as the whole connection time. Then, the average throughput for this TCP connection is computed as the ratio between the total amount data and the total transmission time. The total amount data transmitted can be computed by the difference between the sequence number of the first TCP segment (i.e., 1 byte for No. 4 segment) and the acknowledged sequence number of the last ACK (153426 bytes for No. 178 segment). Therefore, the total data are 153426 - 1 = 153425 bytes. The whole transmission time is the difference of the time instant of the first TCP segment (i.e., 0.024047 second for No.4 segment) and the time instant of the last ACK (i.e., 0.191496 second for No. 178 segment). Therefore, the total transmission time is 0.191496 - 0.024047 = 0.167449 seconds. Hence, the throughput for the TCP connection is computed as 153425/0.167449 = 916.249 KByte/sec.

- *Consider the "fleets" of packets sent around t = 0.025, t = 0.053, t = 0.082 and t = 0.1 in Figure 5. Comment on whether this looks as if TCP is in its slow start phase, congestion avoidance phase or some other phase. Figure 6 shows a slightly different view of this data.*

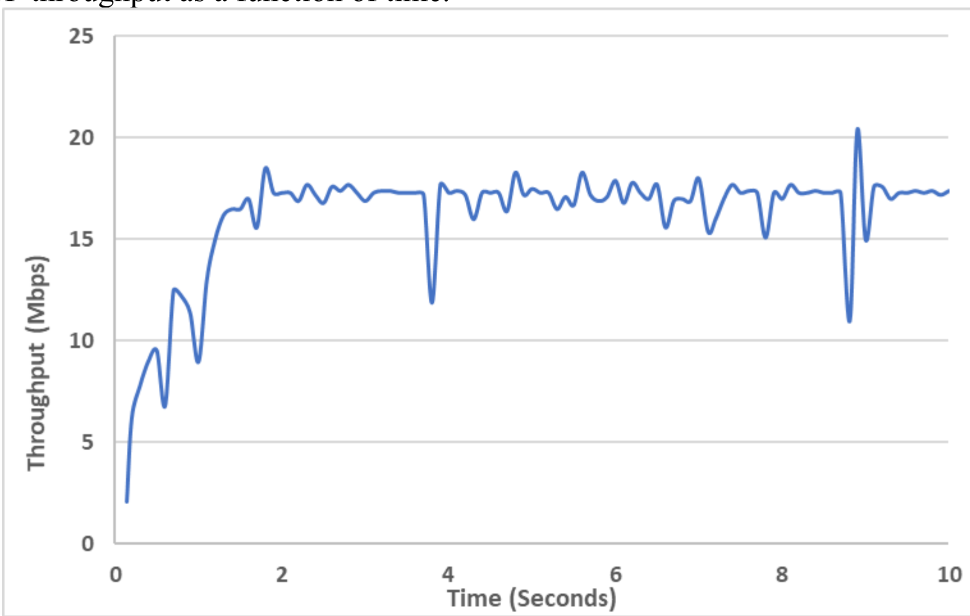
Solution: TCP sends 3 segments at $t = 0.025$, 6 segments at $t = 0.053$, 12 segments at $t = 0.082$, and 24 segments at $t = 0.1$. Since the window doubles every time, it looks like slow start.

- *Answer again question 12 for your own trace.*

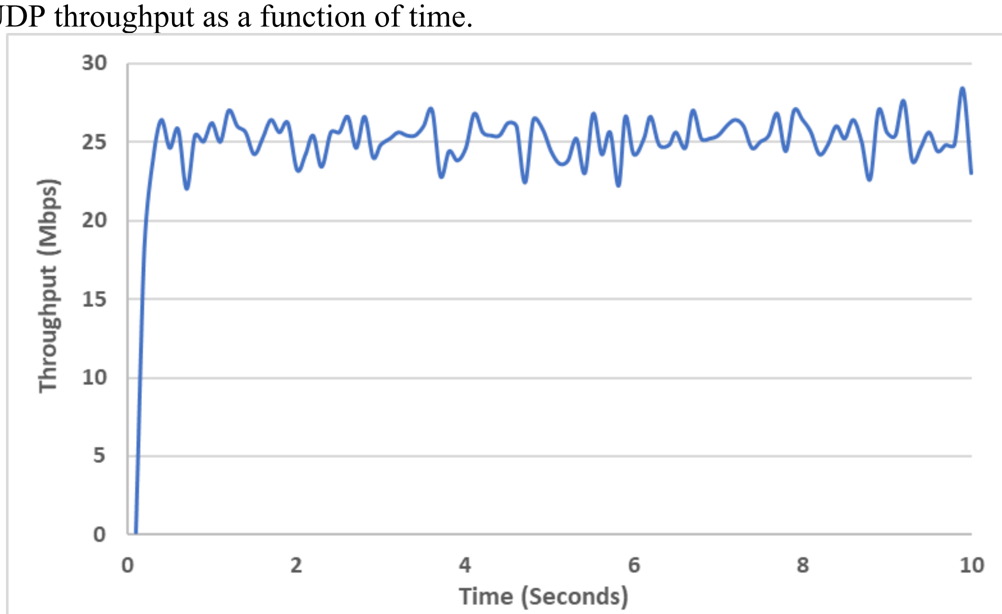
Solution: It depends on your trace.

Measuring bandwidth with iperf

a) TCP throughput as a function of time.

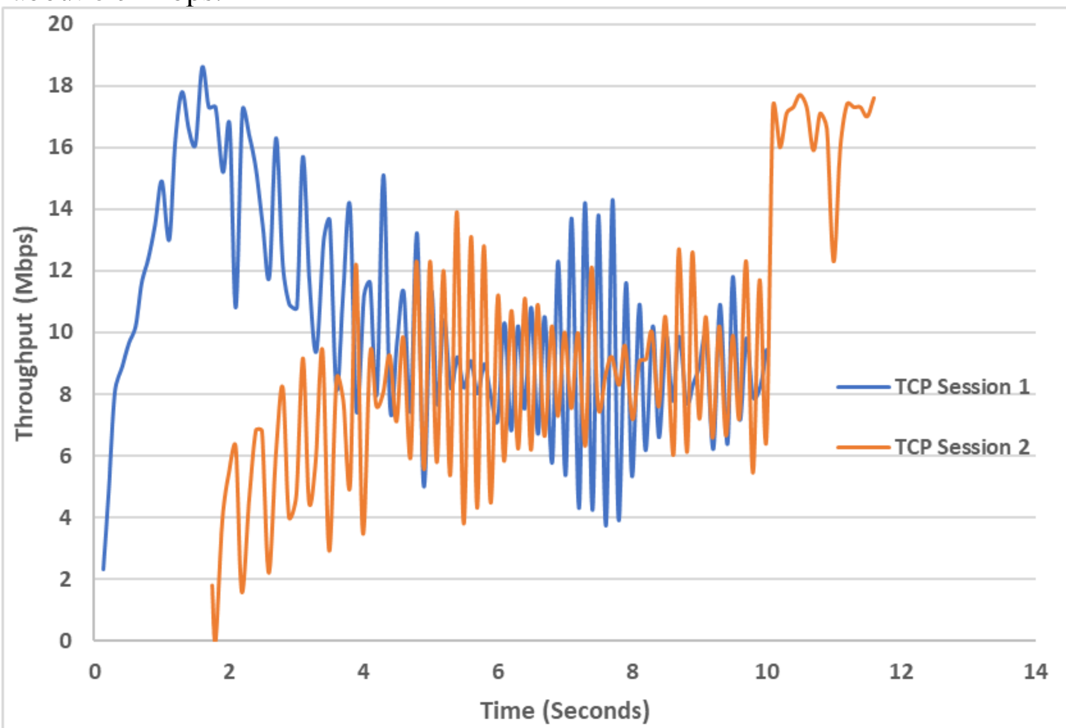


b) UDP throughput as a function of time.

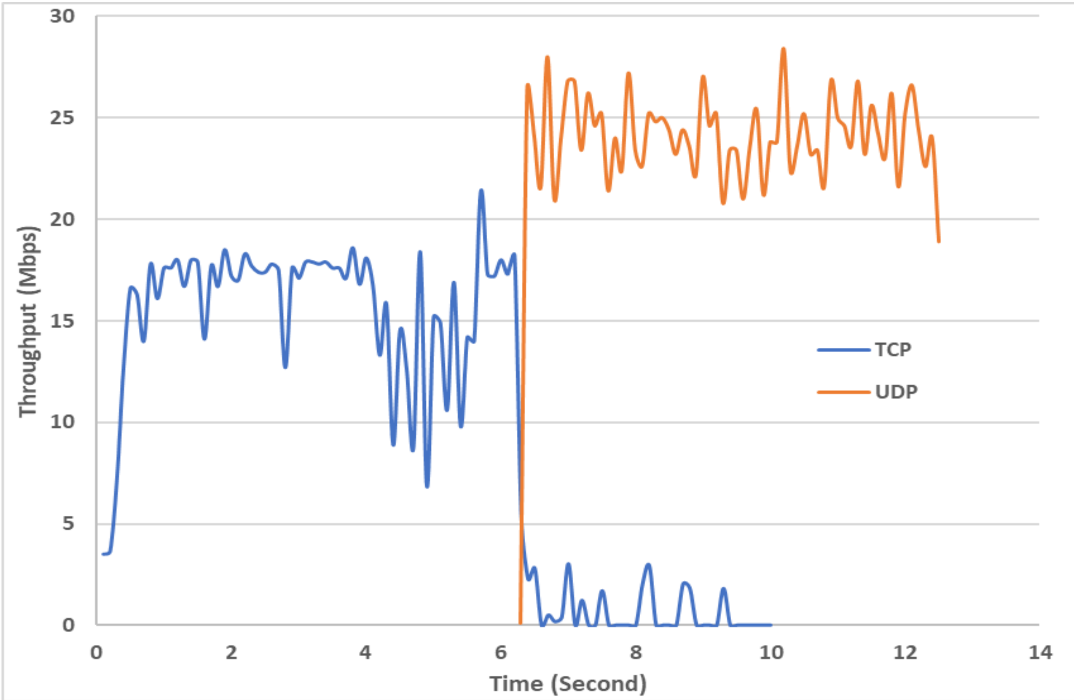


c) In Figure (a), we observe the TCP slow start phase in the first 2 seconds. After 2 seconds, TCP is in congestion avoidance (steady state). In Figure (b), there is no slow start phase for UDP as expected. Also, UDP achieves 5 Mbps higher throughput than TCP (possibly due to lower header overhead and lack of congestion control).

d) Throughput of 2 TCP connections as a function of time. After the 4th second, the 2 connections share the bandwidth fairly, each achieving an average throughput of about 8-9 Mbps.



e) Throughput of one TCP and one UDP connection as a function of time. The UDP connection starts at t=6.3 sec. Due to the lack of congestion control in UDP, the UDP connection utilizes the full bandwidth (maintaining a throughput of 25 Mbps) while the TCP connection almost starves, achieving a throughput of only 3 Mbps or less.



Throughput (Mbps)
30
25
20
15
10
5
0
0
2
4
6
8
10
12
14
Time (Second)
TCP
UDP