CSE232 | Assignment 3

Report

Question 1

a. Maximum expected value of throughput is 5 Mbps. Since there are 2 links, the minimum of the two would be the maximum expected throughput as it would serve as the bottlenect between the two.

b. Bandwidth delay product can be calculated as:

BDP = Bandwidth * RTT

We know that Bandwidth = 5 Mbps and RTT = (10+15)*2 = 50ms

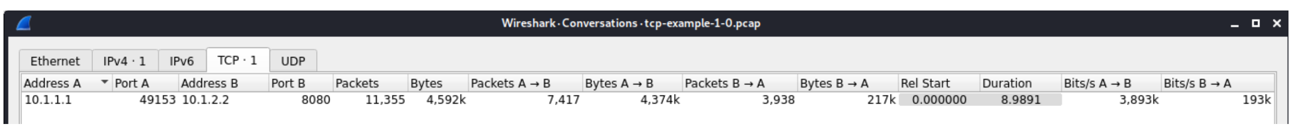
BDP = 5 * 10^6 (bits/sec) * 50 * 10^{-3} sec

BDP = 5 * 50 * 10^3 bits = 250000 bits = 250000/8 bytes = 31250 bytes

Now 1 packet = 1460 bytes

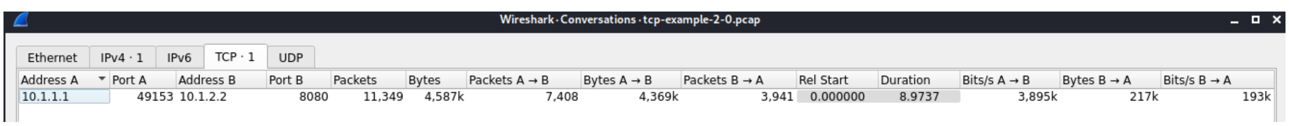
BDP = 31250/1460 packets = 21.4 packets which is approximately equal to 21 packets.

c. Average Computed throughput of the transfer: 3.894Mbps


Wireshark · Conversations · tcp-example-1-0.pcap

Ethernet | IPv4 · 1 | IPv6 | TCP · 1 | UDP

Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.1.1.1	49153	10.1.2.2	8080	11,355	4,592k	7,417	4,374k	3,938	217k	0.000000	8.9891	3,893k	193k

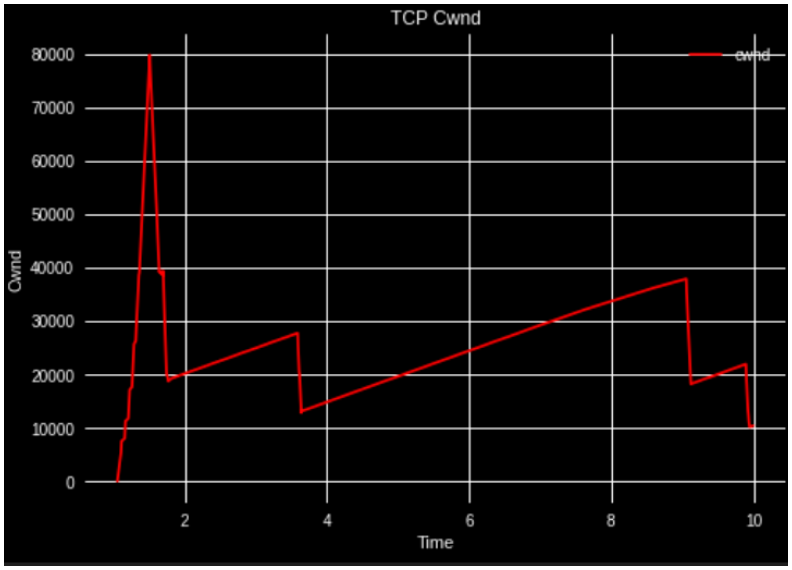

Wireshark · Conversations · tcp-example-2-0.pcap

Ethernet | IPv4 · 1 | IPv6 | TCP · 1 | UDP

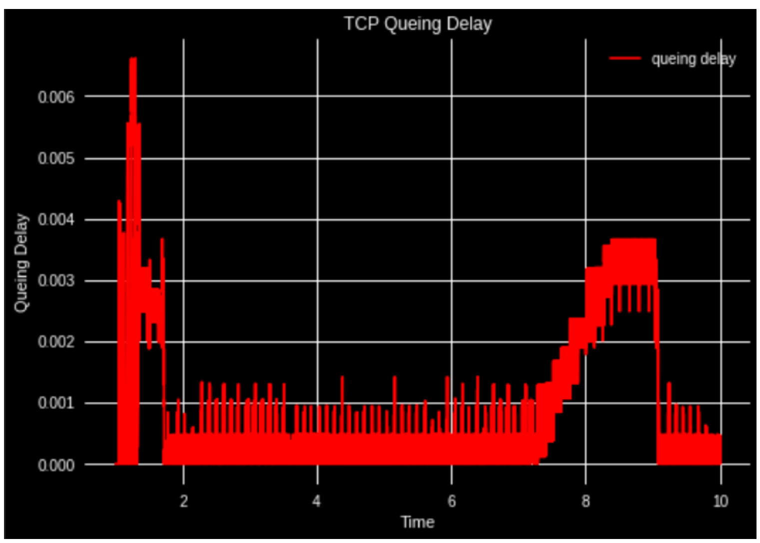
Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Rel Start	Duration	Bits/s A → B	Bytes B → A	Bits/s B → A
10.1.1.1	49153	10.1.2.2	8080	11,349	4,587k	7,408	4,369k	3,941	0.000000	8.9737	3,895k	217k	193k

d. No the achieved throughput is not approximately equal to the maximum expected value. This is because of the added delays. One of the key reasons behind this is the added delays like queuing, transmitting, propagation, network and processing among others. These restrict the traffic and thus the achieved throughput is less than the expected one. Apart from this as it is evident from the previous screen shots, the network is also suffering from packet loss which further affects the final observed throughput.

e.



f. In this and all subsequent parts asking for the queuing delay the graphs are plotted assuming ideal queuing behavior is assumed. By ideal queuing behavior, I mean that the packet being enqueued would be dequeued first.



g. The (CWND vs time) and (Queuing Delay vs time) plots are somewhat related to each other. The intuition behind the relation being that, bigger the congestion window, more the number of packets being thrown on the network. More the networks being thrown, longer the queue being formed at the receiver's end. This in turn increases the wait duration for the packets in the queue. Thus higher values of queuing delay being obtained. From the graphs it can be seen that once the cwnd crosses a particular threshold, a stark increase in the queuing delay is observed.

It can directly be seen from the graphs that when the cwnd size reaches a higher value, the queuing delay also sees a hike. In the time intervals between 1-2 seconds and 7-9 seconds the hike is pretty evident.

As time progresses and with various iterations on the cwnd size, the queuing delay would find its final value and more or less that cwnd would be optimum for the network.

Question 2

a. Average computed throughput: 4.024Mbps

Wireshark · Conversations · tcp-example-1-0.pcap

Ethernet | IPv4 · 1 | IPv6 | TCP · 1 | UDP

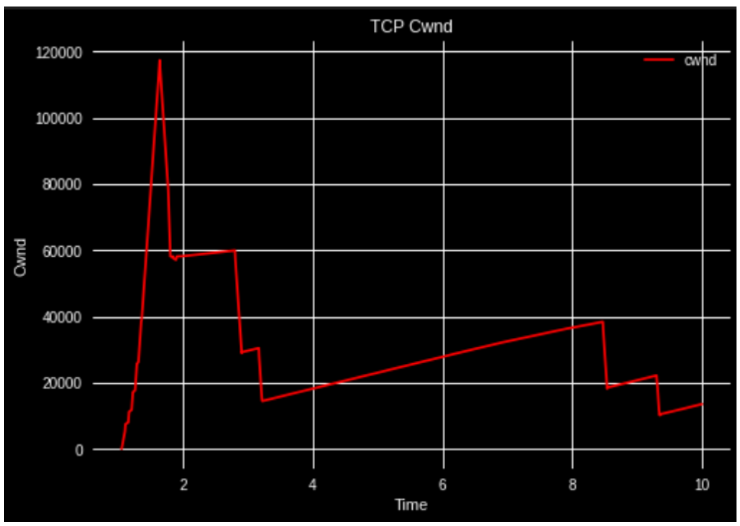
Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.1.1.1	49153	10.1.2.2	8080	11,807	4,751k	7,666	4,520k	4,141	230k	0.000000	8.9882	4,023k	205k

Wireshark · Conversations · tcp-example-2-0.pcap

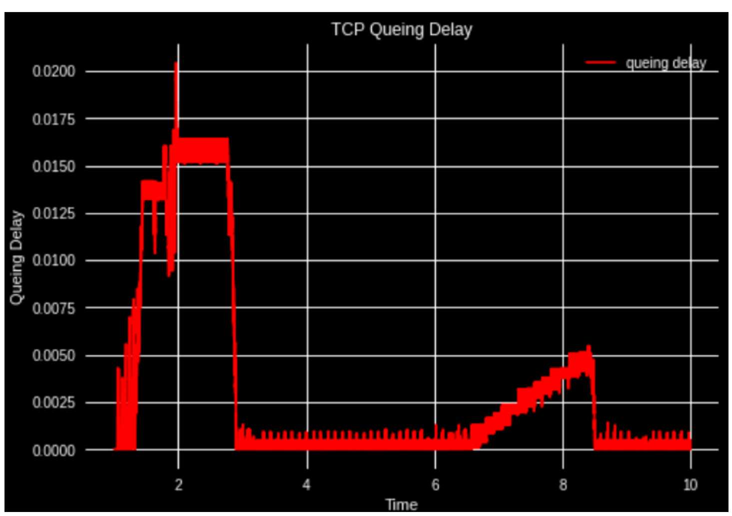
Ethernet | IPv4 · 1 | IPv6 | TCP · 1 | UDP

Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.1.1.1	49153	10.1.2.2	8080	11,800	4,745k	7,656	4,514k	4,144	230k	0.000000	8.9746	4,024k	205k

b.



c.



d. Comapring CWND vs Time plots of the 2 questions, it can be observed that when the queue size is increased the initial maximum cwnd size is increased by as high as 50% (80k to 120k). The subsequent peaks are also affected but as we progress, the final peak is approximately of the same size. This behavior can be explained by taking the increased queuing size. As the maximum queuing size is increased, the sender tries to send more packets by increasing its congestion window size. This in turn also increases the queuing delay. However courtesy to the algorithm being used to determine the most optimal congestion window size and the bandwidth, and the delays in the system, the sender learns the optimum congestion window size, and this value becomes more or less the same as the one is observed in the previous graph.

Question 3

a. Average Throughput: 4.716Mbps

Wireshark · Conversations · tcp-example-2-0.pcap

Ethernet | IPv4 · 1 | IPv6 | TCP · 1 | UDP

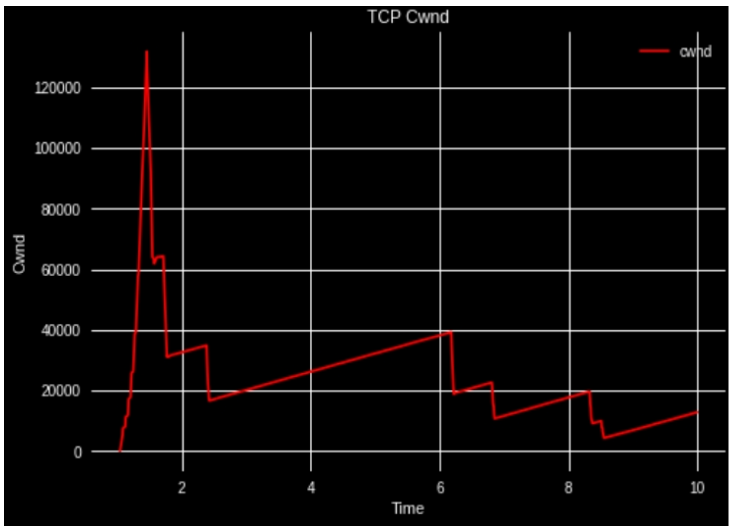
Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.1.1.1	49153	10.1.2.2	8080	13,868	5,562k	8,981	5,292k	4,887	270k	0.000000	8.9746	4,717k	240k

Wireshark · Conversations · tcp-example-1-0.pcap

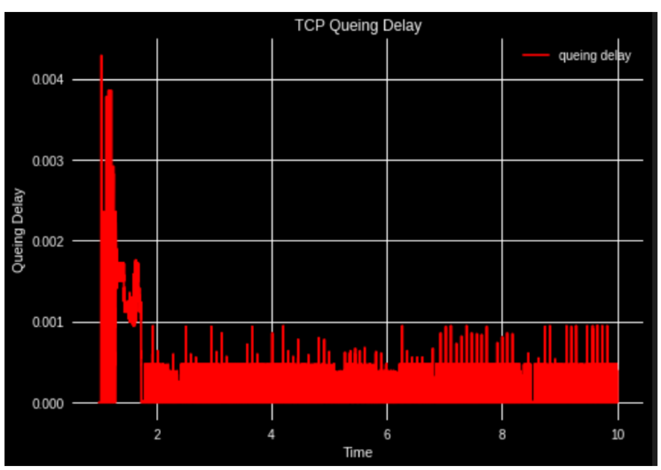
Ethernet | IPv4 · 1 | IPv6 | TCP · 1 | UDP

Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.1.1.1	49153	10.1.2.2	8080	13,875	5,567k	8,990	5,297k	4,885	270k	0.000000	8.9898	4,714k	240k

b.



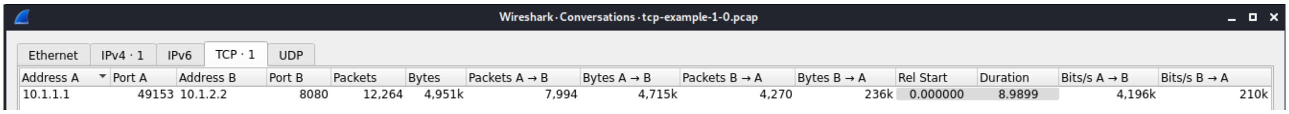
c.



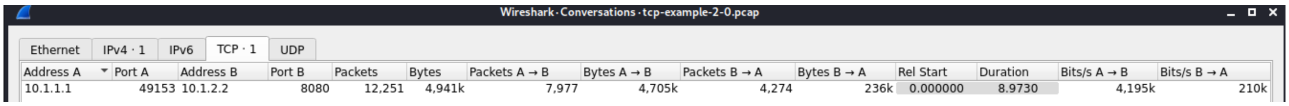
d. While comparing the two (Queuing delay vs. Time) plots, it can be directly seen that on changing the bandwidth and the delay for one of the connection, queuing delay had a significant change. In the beginning for the second graph also the queuing delay peaks. This is because when the system starts, the sender does not know as to how many packets can be sent. This leads to accumulation of the packets at the receiver end thus resulting in queuing delay. This queuing delay however is smaller than the one in ques 1 by a factor as the increased bandwidth and decreased delay value fro the second connection helps in increasing the amount of packets being processed in a given time. This hike is shortlived as the sender starts learning about the limit of receiver and slows down the sending process. Now since the throughput has increased significantly, and so has the loss decreased, the amount of time spent by the packets in the queue has reduced. Now for the given time interval, the threshold for cwnd to result in significant queuing delays has not increased, therefore the queuing delay keeps to a certain minimum, thus explaining the constant nature of the graph. After some time, there might be the scenario that the cwnd crosses that particular threshold and the queuing delay pikes once again.

Question 4

a. Average Throughput: 4.195Mbps

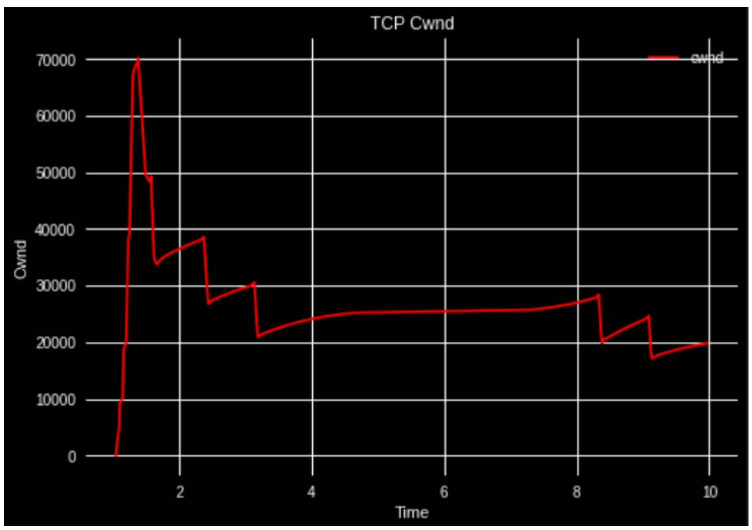


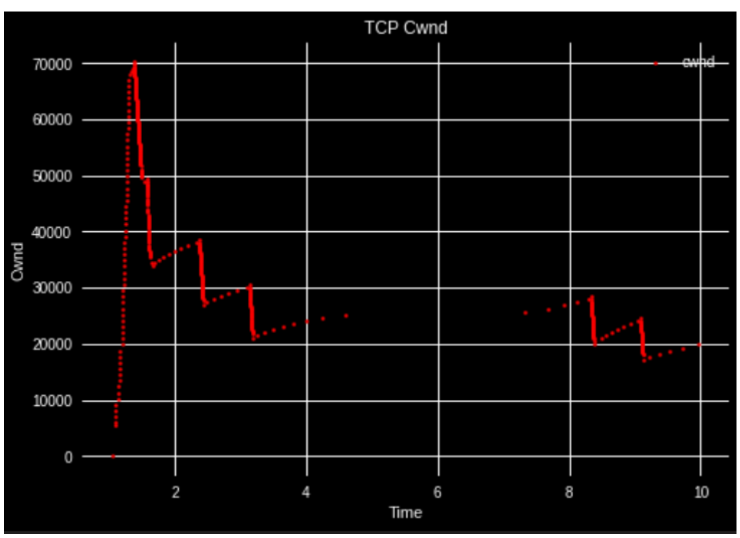
Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.1.1.1	49153	10.1.2.2	8080	12,264	4,951k	7,994	4,715k	4,270	236k	0.000000	8.9899	4,196k	210k



Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.1.1.1	49153	10.1.2.2	8080	12,251	4,941k	7,977	4,705k	4,274	236k	0.000000	8.9730	4,195k	210k

b.





c. On comparing the (cwnd vs time) graphs in this and the earlier question, it is quite evident that the graphs have significant changes. These changes arise as the tcp variant being used has changed from New Reno to Cubic. The various changes can be categorised in three parts. Firstly during slow start, it is quite evident that the starting slopes are similar. Only difference between the two is their peak or the threshold value defined for both or when they start detecting packet loss. For new reno it is about 80k while for tcp cubic it is around 70k. When this threshold is reached, both of them enter in the congestion avoidance phase. During congestion avoidance, the cwnd value and the ssthresh values are changed so as to accommodate the changes in cwnd which are desired. For new reno, the values in the congestion avoidance phase reduce drastically as compared to the values being reduced in the cubic function where a multiplicative decrease by a factor takes place. As it is evident from the graphs, the increase and decrease both are somewhat gradual in the cubic variant but the same are more steep and immediate in the new reno variant. The reno also has the capability of fast recovery where fast retransmit is sent and half of the current cwnd is stored as ssthresh and as the new cwnd, thus directly skipping the slow start process again and directly going onto the congestion avoidance phase. This is a little different in the cubic variant as the values there increase and decrease gradually, courtesy to the cubic function being used there. This usually helps the system to find the optimal value quickly (due to which there is some sort of gap in values in between also). This is because after entering congestion avoidance from fast recovery, it starts increasing the window size using concave function. This is done till the window size suffers a loss and then it is increased convexly after staying there for a short time. This staggered decline is evident in the seconds following the sharp incline. In the graph, from 3-8 seconds, the window size increases cubically and just as it starts increasing convexly, another packet loss is suffered which further shows staggered decline.