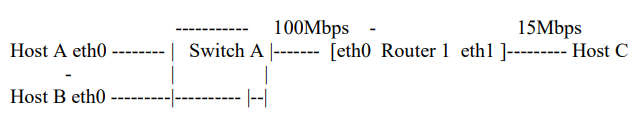
Anthony Jones

CPSC 3600 HW # 2

Spring 2021

3/12/2021



Q1a. For this example, we will use the same address assignment provided. In this case, the network connected to Switch A has the network address 192.168.0.0/24 and the network of the right (the other side) has the network address 192.168.1.0/24. This leaves us with 8 bits to determine hosts.

The first host will be the internal interface for the router which will be the gateway between the two networks. This is technical split between two ethernet interfaces, eth0 and eth1. These addresses will be given as 192.168.0.1 and 192.168.1.1 respectively.

The next host will be Host A. Because this is on the left side, it will be on the 192.168.0.0/24 network, and because the letter representing it is A, we will use its decimal equivalent 10 to make the host address 192.168.0.10. Host B is similar, belong to the same network address, but we will use its decimal equivalent 11 to make its host address be 192.168.0.11. The final host, Host C, will be on the other network. For consistency we will use the decimal equivalent 12 to make its address 192.168.1.12.

Q1b. (1000 original bytes + 28 bytes of overhead + 20 byte header + 8 byte UDP header) / (100 Mbps \* 125000 byte per Mb) + (1000 original bytes + 28 bytes of overhead + 20 byte header + 8 byte UDP header) / (15 Mbps \* 125000 byte per Mb) = 0.00064768 seconds.

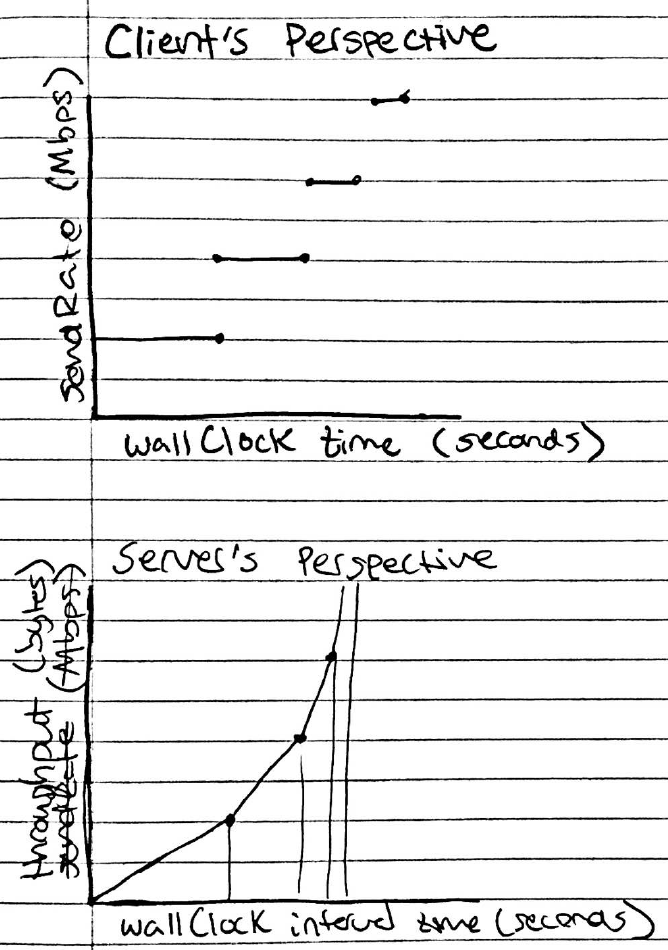
Q1c. This problem is easy as it is sending a 1000 byte packet from Host A to Host C and back. We already know the time it takes to travel between the two, and since it does it twice, the total Round trip time would be 0.00064768 \* 2 = 0.00129536 seconds. Also, the host has a 10000 millisecond delay built into each response through the invocation, so the expected real time performance is 10.00129536 seconds between iterations.

Q2a. Basically, as far as my interpretation goes, this question is asking for how long the code is forcing the client to wait before sending its packet. In this case we are limiting a 100 Mbps connection to a 5 Mbps traffic rate. Since the packet size is (1000 + 28 + 20 + 8) = 1056 bytes, the time that the client is delayed every iteration is 1056 / (95 Mbps \* 125000 byte per Mb) = 0.00008892631 seconds. In other words, the client waits those many seconds every time it sends packets of 1056 bytes in order to limit the traffic to the 5 Mbps threshold.

Q2b. (1000 original bytes + 28 bytes of overhead + 20 byte header + 8 byte UDP header) / (5 Mbps \* 125000 byte per Mb) + (1000 original bytes + 28 bytes of overhead + 20 byte header + 8 byte UDP header) / (15 Mbps \* 125000 byte per Mb) = 0.0022528 seconds. Note that the only difference in the calculation is that the client sends data at 5 Mbps instead of 100 Mbps. The server can still send its data back using its normal 15 Mbps traffic rate.

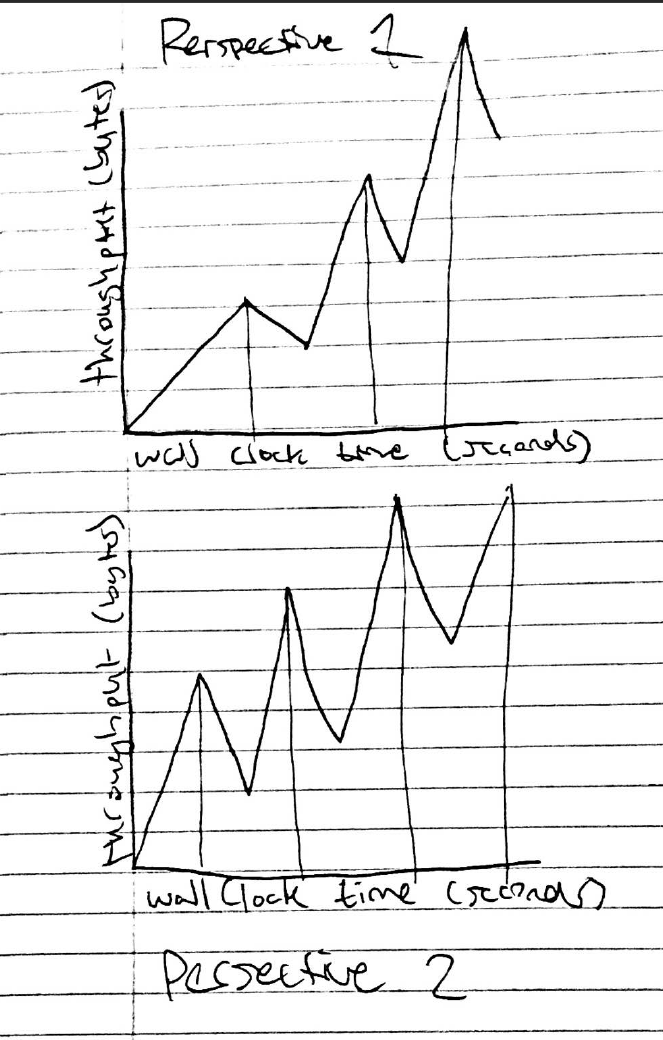
Q2c. ( (1 Gigabyte) \* (1000000000 bytes / 1 Gigabyte) + 28 bytes of overhead + 20 byte header + 8 byte UDP header) / (5 Mbps \* 125000 byte per Mb) + ((1 Gigabyte) \* (1000000000 bytes / 1 Gigabyte) + 28 bytes of overhead + 20 byte header + 8 byte UDP header) / (15 Mbps \* 125000 byte per Mb) = 2133.3334528 seconds. We swapped out the original packet size in the calculation for both components.

Q3a.



As you can see, from the client’s perspective the send Rate doubles after smaller and smaller time periods. This is because the RTT for each iteration changes as the send Rate gets faster. In fact, this would actually cap pretty fast because the traffic is only 100 Mbps, but the graph is a step function because the send Rate can only update after every delay. From the server’s perspective, the throughput does constantly increase so it is continuous. Instead the rate at which throughput comes in doubles after shorter and shorter interval times. Technically if we counted the time through iterations, this would be constant as every iteration passes the same packet size and therefore is the same throughput.

Q3b.



It was hard for me to find a great interpretation for this problem. I believe one possibility is that the throughput between the different perspectives might oscillate, especially if the clients started sending their packets at different times. For example, after 3 seconds Host A might double its send rate, but Host B would still be sending at a low rate of 10 Kbps. Then after another second, Host B could double, but Host A remains the same. This would effectively change the throughputs of each server perspective.