Computer Systems Tutorial

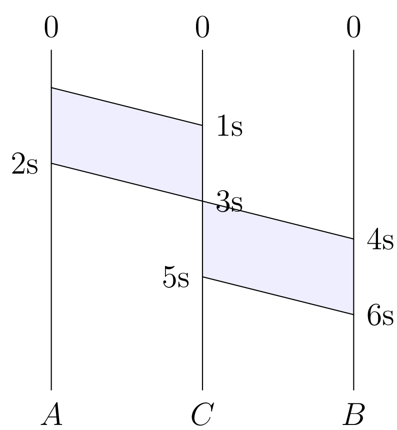
December 1^{st} 2023

1 Preliminaries

Let's recall how store-and-forward packet switching works. Suppose we have the following configuration and want to send a packet of size 40 KBytes from A to B and C is the intermediate router.

$$A \longrightarrow C \longrightarrow B$$

If the switching is store-and-forward, then C should receive the whole packet before it can start sending it to B. Suppose the data rate of channels $A \rightarrow C$ and $C \rightarrow B$ are the same and equal to 20 KBytes per second (160 Kbits/sec) and the propagation delay of both channels is 1 second. Then the following sequence diagram describes the time when the packet finally arrives at B.



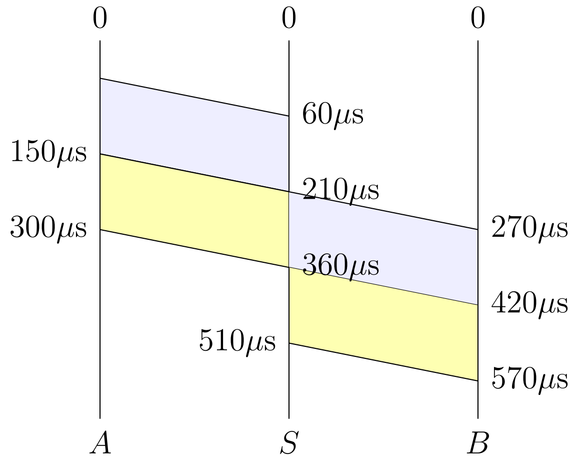
Because the bandwidth of the channel is 20 KBytes/sec, A needs 2 seconds to put all the data onto the wire and because the propagation delay is 1 second, the first bit arrives at C at 1s and the last bit at 3s. Similarly for the trip between C and B.

Recall that **throughput** is the actual amount of data that is transferred per time unit (the bandwidth is the maximal amount of data that can be transferred). Because the amount of data we sent above is more than the channel can get per second, we get that the throughput is equal to the bandwidth.

2 Exercises

Exercise 1 Suppose the path from A to B has a single switch S in between: $A \longrightarrow S \longrightarrow B$. Each link has a propagation delay of 60 µsec and a bandwidth of 2 bytes/µsec. How long would it take to send two back-to-back 300-byte packets from A to B?

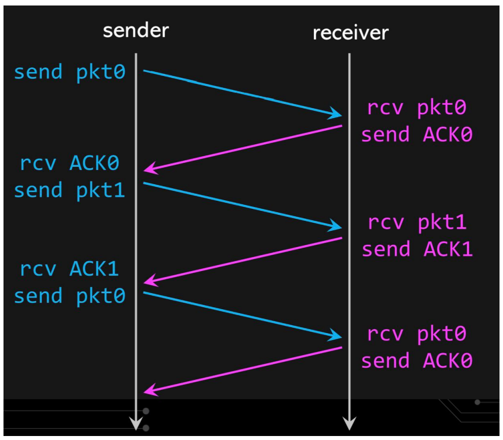
Solution Consider the diagram below.



The first bit of the first packet takes 60μs to arrive at S and 150μs are needed to put the whole packet onto the wire (300 bytes divided by 2 bytes/μsec) and hence the last bit of the first packet arrives at S at 210μs. Then the second packet can start being transmitted and the first packet can be forwarded by S (because we send back-to-back and store-and-forward). Repeating the same reasoning for the second packet and for the trip from S to B.

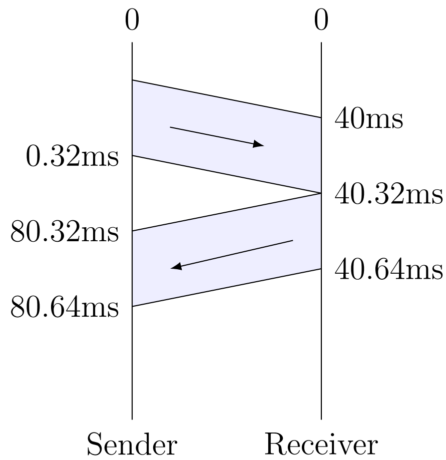
Exercise 2 We know that RDT3.0 is correct, but it suffers greatly in terms of performance due to being a Stop-and-Wait protocol. Let's assume that we have a 100 Mbps link between two devices and 40ms propagation delay, what will be the **throughput** on this link for a packet size of 4 KBytes when using RDT 3.0 protocol? Before we can send the next packet we need to wait for a packet with an acknowledgement and assume that this packet's size is also 4 KBytes and also assume that no packet is corrupted or lost.

Solution Recall how RDT works



That is, the receiver has to send an acknowledgement packet to the sender to confirm that it has received the packet. Only after the acknowledgement can the sender send another packet. In this question, we will assume that the receiver sends 4 KBytes acknowledgement response (although in practice the response contains only a header with no body).

Now consider the diagram below



The first bit takes 40 ms to arrive at the Receiver, but to put 4 Kbytes (32 KBits) of data onto the wire takes only 0.32 ms and therefore the last bit arrives at 40.32 ms. Then the sender should send these 4 Kbytes back following the same procedure and the last bit of acknowledgement gets to Sender at 80.64 ms. Ultimately, we've managed to transfer only 4 Kbytes in 80.64 ms, or $\frac{32000 \text{ bits}}{80.64 \text{ ms}} \equiv 397$ bits/ms, or, 397 Kbits/sec and hence this is our throughput (the actual amount of data transferred per unit of time). Recalling that the bandwidth was 100 Mbits/sec we get that the actual throughput is 250 times less than what we can achieve.

Exercise 3 For each IP network prefix given (with length), identify which of the subsequent IP addresses are part of the same subnet.

a **10.0.130.0/23:** 10.0.130.23, 10.0.129.1, 10.0.131.12, 10.0.132.7

b **10.0.132.0/22:** 10.0.130.23, 10.0.135.1, 10.0.134.12, 10.0.136.7

c **10.0.64.0/18:** 10.0.65.13, 10.0.32.4, 10.0.127.3, 10.0.128.4

Solution We will see the solution for (a) and the rest are similar. Recall, that the notation $x.y.z.w/m$ means that m first bits correspond to the subnet and the rest bits are available for hosts (and for the reserved subnet and broadcast IP addresses within the subnet). To see the range of available addresses we need to first convert the denary IP address into binary:

$$10.0.130.0/23 = \underline{00001010.00000000.1000001}0.00000000$$

where the subnet's part is underlined. The rest of the bits are available addresses within the subnet. Because the rest of the bits are all zeroes this is also the smallest address in the subnet (or the subnet's address). The largest available address is when all the bits are ones (this is also a broadcast address):

$$\underline{00001010.00000000.1000001}1.11111111$$

or, in denary,

$$10.0.131.255$$

All that is left is to check whether a given IP address falls in this range.

1. 10.0.130.0 < 10.0.130.23 < 10.0.131.255 and hence it is inside the range
2. 10.0.129 < 10.0.130.0 and hence it is not inside the range
3. 10.0.130.0 < 10.0.131.12 < 10.0.131.255 and hence it is inside the range
4. 10.0.132.7 > 10.0.131.255 and hence it is outside the range.

Also note that only the hosts within the same subnet (i.e. host 1 and 3) can communicate directly. If host 1 wants to communicate with host 2 it has to delegate this job to the router which connects two networks.

Exercise 4 By using Wireshark trace the TCP packets that arise when communicating with a web server over HTTP. In particular, have a look at the initial handshake and sequence and ACK numbers.

Solution You can check out this video: `https://youtu.be/3Zb_EebU22o?si=F9MnfE1mGEJv53nz`. This is pretty much the same thing I showed at the tutorial.