1. Transcript of interaction

```
MacBook-Pro ~ % telnet pollett.org 70 Trying
173.13.143.73...
Connected to pollett.org.
Escape character is '^]'.
/
i=====================================     fake  null  0
i= Welcome to the Pollett.org Server =     fake  null  0
i=   Maintained by Chris Pollett     =     fake  null  0
i=====================================     fake  null  0
i     fake  null  0
ICheck out my Picture /me1.jpg   pollett.org     70
i     fake  null  0
0A Brief Bio    /bio.txt   pollett.org     70
i     fake  null  0
0My Phlog  /phlog.txt pollett.org     70
i     fake  null  0
iOn This Machine     fake  null  0
i===============     fake  null  0
i     fake  null  0
1Experience the natural numbers one-by-one /addone/0  pollett.org
70
i     fake  null  0
7Ask The Magic Eight Ball a Question  /magic-eight-ball
pollett.org     70
i     fake  null  0
iPlaces to Visit:     fake  null  0
i================     fake  null  0
i     fake  null  0
hAtto Server GitHub Repository  URL:https://github.com/cpollett/atto
pollett.org     70
i     fake  null  0
1Floodgap - Gopher Central /     gopher.floodgap.com   70
i     fake  null  0
1Technical Info About Gopher    /gopher/tech    gopher.floodgap.com
70
.
Connection closed by foreign host.
MacBook-Pro ~ %
```

Note: Starting from here, answers are often found in **bold.**

2.

a. Assume that there is no time spent in the queue.
   Units are confirmed to be in megabyte, kilobyte, etc., and NOT mibibyte, kibibytes, etc.

   Time to transfer the file is T. All items are in seconds (s).

   T = *propagation* + *transmit* + *handshake*

   Handshake is given to be 2 x RTT. RTT = 200 ms = 0.2 s
   handshake = 0.2 s * 2 = 0.4 s

   propagation delay is "the time it takes to propagate a signal through the system." As it takes 200 ms for the signal to round trip (RTT), assuming the signal has the same speed throughout, we can assume that half of RTT is the time spent to go through the wire (one way). Furthermore, since data is being transferred **continuously**, it only needed to be applied once:

   propagation = RTT/2 = 0.2 s / 2 = 0.1 s

   Given a bandwidth of 2.5 megabits per seconds, and a packet size of 4kB (confirmed to be 1000 bytes, not 1024):

   4kB = 4000 bytes = 32000 bits/package
   2.5 Mbps = 2,500,000 bits/s

   32000 bits/packet / 2,500,000 bits/s
   = 0.0128 s/packet

   1500 kB / 4 kB = 375 packets
   transmit = 375 packets * 0.0128 s/packet = 4.8 s

   With the above values, we can then calculate T, the time it takes the transfer the file:

   T = 0.4 s + 4.8 s + 0.1 s = **5.3 seconds**

---

b. New limitation: Must wait 1 RTT before sending the next packet. In between 375 packets, there are 374 RTTs to wait for. To find the time, simply add it to the result from part A, which is the time without any delay.

   T = 5.3 s + (374 packets * RTT) = **80.1 seconds**

---

c. Infinite bandwidth means transmit time is now 0.
   New info: Up to 10 packets can be sent per RTT.

   T = *handshake* + *propagation* (due to *transmit* = 0)

   Transmit may now be 0, but now the time taken is moved to propagation (time to travel along wire) due to having to wait 1 RTT after sending each burst of packets. (half the time spent to actually send, the other half to confirm before sending the next batch)

   You can approximate the number of round trips that needed to be done with the following: 375 packets / 10 packets per trips = 37.5 trips, which would then be rounded up to 38 trips.

   However, it should be noted that for the last batch of packets, we only need to consider one-way time (half of RTT), as there will not be any more packets to send afterwards. This leaves us with 37.5 RTT to go through all the packets, plus the handshake to start.

   propagation = 37.5 RTT

   T = 37.5 RTT + *handshake* = 39.5 RTT * 0.2 s/RTT = **7.9 seconds**

---

d. Transmit time is still 0. However, the amount of packets that is sent doubles every RTT or packet burst. Since we have 375 packets to go through, we will have to look for which iteration sends that amount, or beyond. Total packets sent can be calculated with 2^i - 1.

Iteration	Packets Sent	Packets Sent TOTAL
1	1	1
2	2	3
3	4	7
4	8	15
5	16	31
6	32	63
7	64	127
8	128	255
9	256	511

As seen here, at the 9th wave of packets, we will exceed 375 total packets. Note that we are still following the rule of 1 RTT before sending each packet burst, with the exception of the last batch, which we only need to consider the one way time for.
As such, *propagation = 8.5 RTT*

T = *handshake* + 8.5 RTT = 10.5 RTT = **2.1 seconds**

3.

a. 256-byte packets
Speed provided to be 2 x 10^8 m/s

Propagation delay = distance / speed
= (75 km * 1000 m/km) / (2 x 10^8 m/s) = 3/8000 s

Transmit delay = size/bandwidth

Solving: Transmit delay = Propagation delay , to get bandwidth

→ (256 bytes * 8 bits/byte) / Bandwidth = 3 / 8000 s
→ Bandwidth = 2048 bits / (3/8000 s) = 5461333.333 bits/s, or bps
= **5.461333333 Mbps**

b. 1024-byte packets

   Propagation delay = 3/8000 s

   Transmit delay = Propagation delay

   → (1024 bytes * 8 bits/byte) / Bandwidth = 3 / 8000 s
   → Bandwidth = 8192 bits / (3/8000 s) = 21845333.33 bits/, or bps
   = **21.84533333 Mbps**

4.

a. Here, we only have one store-forward operation, meaning that there is one switch between the 2 hosts, meaning we have 2 links before arriving at the destination. For each link, there is a propagation delay of 5 microseconds. However, we also need to take into account the transmit delay of both as well.

   T = (2 * *propagation*) + (2 * *transmit*)

   transmit = size/bandwidth = 4000 bits / 100,000,000 bps
   = 4 * 10^-5 s = 40 μs

   T = (2 * 5 μs) + (2 * 40 μs) = **90 μs, or 90 microseconds**

b. With 3 switches, there are now 4 links, rather than 2. However, everything else is the same.

   T = (4 * *propagation*) + (4 * *transmit*)

   T = (4 * 5 μs) + (4 * 40 μs) = **180 μs or 180 microseconds**

c. Back to single switch, but with the cut-through switching. Only the first 150 bits of the packet has to be received to begin the forwarding process, meaning we won't have 2 full transmit delays. Rather, it would be one full transmit to first get the entire file in the wire + time it takes for the first 150 bits to enter the switch. This method should be faster (expect a smaller time). Propagation delay is still present for each link however.

   T = (2 * *propagation*) + (*transmit* + *cut-through time*) *cut-through time* = 150 bits / 100,000,000 bps = 1.5 * 10^-6 s = 1.5 μs

T = (2 * 5 μs) + (40 μs + 1.5 μs) = **51.5 μs**

5. This differs from STDM (synchronous time-division multiplexing) in that it takes into consideration whether the host(s) have something to send or not by having those without anything to forfeit the medium. STDM on the other hand, have neatly divided time segments for each host, regardless of whether they would actually use it or not. However, the time allotted for the hosts and the order in which the flow occurs are less predictable with the round-robin scheme.

   Assume that after a selected host transmits their packet, another host is picked almost immediately for the next transmission. The network utilization of this scheme compared to STDM should be higher, as the medium would not be forced to waste time on hosts not sending anything since they are excluded.