Digital Communications in the Modern World

Exercise 2

Hand in by Thursday 2/12-2004 at 23:59

Note: **bps** denotes *bits* per second (**Bps** denotes *bytes* per second).

1. *(10 points)* A group of innovative students want to establish a startup company called DoggieNet for peer2peer networking on the campus. The idea is based on training a dog to carry a box with three magnetic tapes between any two computers. Each tape contains 7gigabytes. The dog can travel at 18 km/hour. The transmission lines in the campus have a data transmission rate of 150 Mbps. Would you invest in the stocks of DoggieNet? Or in other words: is there a range of distances in which DoggieNet has a higher transmission rate than the LAN in the campus? (ignore any overhead, only compare the actual data transmission).

2. *(10 points)* Suppose two hosts, A and B, are separated by 10,000 km and are connected by a direct link of R = 1Mbps. Suppose the propagation speed over the link is $2.5 \cdot 10^8$ meters/sec.
   a. Calculate the "bandwidth-delay product," $R \cdot t_{prop}$.
   b. Consider sending a file of 400,000 bits from host A to host B. Suppose the file is sent continuously as one big message. What is the maximum number of bits that will be in the link at any given time?
   c. Provide an interpretation of the bandwidth-delay product.
   d. What is the width (in meters) of a bit in the link?
   e. Derive a general expression for the width of a bit in terms of the propagation speed s, the bandwidth R, and the length of the link m.

3. *(10 points)* Suppose that on average ***10*** packets arrive simultaneously every $\boldsymbol{10 \cdot (L/R)}$ seconds at a router. The size of each packet is $\boldsymbol{L}$ bits. The transmission rate of the router is $\boldsymbol{R}$ bps. What is the average queuing delay (d_{queue})?

4. *(10 points)* Answer the following questions by searching the HTTP/1.1 RFC (RFC 2616) at http://www.w3.org/Protocols/rfc2616/rfc2616.html.
   a. In a persistent connection, what causes a web server to close a connection to the web client? (closing the socket that does a TCP close)
   b. What should a web client do in case the web server closes the connection?

5. *(5 points)* Both UDP and TCP use port numbers to identify the destination entity when delivering a message. Give two reasons for why these protocols invented a new abstract ID (port numbers), instead of using process IDs, which existed

when these protocols were designed.

6. *(10 points)* Real Time Player is used to transmit streaming CD-quality audio (such as listening to a radio station through the computer). It makes a pair of 16-bit samples 44,100 times/sec; one sample for each of the stereo channels. Assume the TCP implementation works with 1024-byte payload TCP packets. How many packets must RTP transmit for listening to one second of audio?

7. *(5 points)* What can go wrong if there are duplicate "live" packets with the same sequence number in a TCP connection? Give an innovative example.

8. *(10 points)* One way to avoid the existence of duplicate packets with the same sequence number is to have a have a "maximum packet lifetime" in order to minimize the chance of a "live" packet being around when the sequence numbers wrap around and reach that number again. Packets whose "maximum packet lifetime" is greater than the local time on the hosts and routers are deleted. This requires a relation between the maximum packet life time, the size of the sequence number space and the rate at which sequence number can progress.

   A 64-bit sequence number seems sufficient for most real life scenarios. However, theoretically, an optical fiber can run at 75 Tbps (1 Tera is 1024 Giga). Assume that each byte has its own sequence number, as TCP does.

   a. What maximum packet life time is required to make sure that future 75 Tbps networks do not have wraparound problems even with 64-bit sequence numbers?
   b. What maximum packet life time is required in the same network but with the 32-bit sequence numbers that are used by TCP today?

9. *(15 points)* In order to use the maximum packet lifetime the hosts and routers need to have roughly synchronized clocks. Otherwise the maximum packet lifetime stamped on the source host will be meaningless on the routers and target hosts. Due to the physics of the clock crystals, clocks drift apart over time. As it is unfeasible to equip every host (routers and clients) with a GPS thus they all need to periodically synchronize their clocks from each other or from a server.

   Let us define clock synchronization as follows:
   $\exists\ \gamma, t_0 \in R^+$, s.t. $\forall$real time $t \geq t_0$ and $\forall$ two hosts h_i , h_j:
   |clock of host h_i at t - clock of host h_j at t | $\leq\ \gamma$.

   The goal is to attain a γ which is as small as possible.

   a. Describe a simple application layer client-server clock synchronization algorithm. Assume that you have no control over or knowledge about the network layers beneath. Try to get a γ which is as small as possible. You can assume that the time server is always functioning correctly and gets its time through a GPS.

b. What is the γ of your algorithm?
c. Now assume that your application has some knowledge and control over the transport layer in the sense that it can choose between UDP and TCP. How that either choice affect your algorithm?
d. Would you choose using UDP or TCP for your application?

10. *(15 points)* All the P2P networks we learnt in class (Napster, Gnutella, KaZaA, BitTorrent, etc.) are called **unstructured** P2P networks in which there is no real structure between the clients. They all go through centralized servers. The latest trend in P2P networking is **structured** networks in which there is no central server.

Suppose there is a general hash function that gives every object (document, audio file, etc.) a unique real number in the range [0..1]. Suppose there are *n* nodes in the network and every node is responsible for a distinct range in [0..1]. Thus, if object *YellowSubmarine.mp3* gets the number 0.1111 and node with id *x* is responsible for the range [0.1110…0.1112] then node *x* actually holds this object.

For simplicity assume that every object is held by only one node. A simple solution would be that every node holds the mapping of
<node id ⇔ node's object range>, for all of the other *n* nodes. The hash function would thus return the unique name of the object and every node would immediately know the exact id of the node where the object is located. Holding such a big *O(n)*-sized table is not scalable and not practical for a large *n*.

Describe an algorithm in which every node holds a *O(logn)*-sized mapping table and that needs at most *O(logn)* hops to find the id of the node holding the requested object. Specifically you need to show:

1. What mapping table every node holds.
2. How to route in *O(logn)* hops to a requested object using the mapping table.
3. Prove or explain your solution.

You can ignore "join" and "leave" issues. Assume that the objects are already in the "right" places according to the hash function. You do not need to handle any fault tolerance, i.e. all the nodes are up and function correctly, no node is malicious (they do not try to sabotage). You can assume that the node ids are numbers from *1* to *n*. Every node knows its own id. You cannot use broadcast. Every node can only send messages to the nodes in its mapping table.

Hint: Every node should hold the mapping of only *logn* out of the *n* nodes in the network. Think of a mapping rule for the nodes that are present in the mapping table. Think of how to use the mapping table in order to go from one node to the other until you reach the requested node in O*(logn)* hops.