Homework Solution 3

1. (25%) True or false?
   a. (5%) A user requests a Web page that consists of some text and three images. For this page, the client will send one request message and receive four response messages.
   False.
   There are four connections since each connection transports exactly one request message and one response message. So, each object will have its one request message instead of there being only one request message.

   b. (5%) Two distinct Web pages (for example, `www.mit.edu/research.html` and `www.mit.edu/students.html`) can be sent over the same persistent connection.
   True.
   It is because both of these web pages are on the same physical server (www.mit.edu).

   c. (5%) With nonpersistent connections between browser and origin server, it is possible for a single TCP segment to carry two distinct HTTP request messages.
   False.
   In a nonpersistent connection, the connection closes after each connection. In this case, the connection will close once the first message is received, and there will be a new connection opened to send the second message.

   d. (5%) The `Date:` header in the HTTP response message indicates when the object in the response was last modified.
   False.
   The “Date:” is the time at which the request was created and not when the object was last modified.

   e. (5%) HTTP response messages never have an empty message body.
   False.
   Some HTTP response messages have an empty message body. For example, HTTP Status-Code of 204 and 304 MUST NOT include a message body. (RFC 2616)

2. (20%)
   a. (6%) How does SMTP mark the end of a message body?
      SMTP uses a line containing only a period to mark the end of a message body.

   b. (6%) How about HTTP?
      HTTP uses “Content-Length header field” to indicate the length of a message body.

   c. (8%) Can HTTP use the same method as SMTP to mark the end of a message body? Explain.
      No, HTTP cannot use the method used by SMTP, because HTTP message could be binary data, whereas in SMTP, the message body must be in 7-bit ASCII format.

3. (20%) Suppose within your Web browser you click on a link to obtain a Web page. The IP address for the associated URL is not cached in your local host, so a DNS lookup is necessary to obtain the IP address. Suppose that *n* DNS servers are visited before your host receives the IP address from DNS; the successive visits incur an RTT of RTT_1, …, RTT_n. Further suppose that the Web page associated with the link contains exactly one object, consisting of a small amount of HTML text. Let RTT_0 denote the RTT between the local host and the server containing the object. Assuming zero transmission time of the object, how much time elapses from when the client clicks on the link until the client receives the object?
   The total amount of time to get the IP address is
   $$RTT_1 + RTT_2 + \cdots + RTT_n.$$
   Once the IP address is known, RTT_0 elapses to set up the TCP connection and another RTT_0 elapses to request and receive the small object. The total response time is
   $$2RTT_0 + RTT_1 + RTT_2 + \cdots + RTT_n.$$

4. (20%) Suppose Bob joins a BitTorrent torrent, but he does not want to upload any data to any other peers (so called free-riding).
   a. (10%) Bob claims that he can receive a complete copy of the file that is shared by the swarm. Is Bob’s claim possible? Why or why not?
      Yes. His first claim is possible, as long as there are enough peers staying in the swarm for a long enough time. Bob can always receive data through optimistic unchoking by other peers.

b. (10%) Bob further claims that he can further make his “free-riding” more efficient by using a collection of multiple computers (with distinct IP addresses) in the computer lab in his department. How can he do that?
His second claim is also true. He can run a client on each machine, and let each client do “free-riding”, and combine those collected chunks from different machines into a single file. He can even write a small scheduling program to let different machines only asking for different chunks of the file. This is actually a kind of Sybil attack in P2P networks.

5. (15%) Install and compile the Java programs TCPClient and UDPClient on one host and TCPServer and UDPServer on another host.
   a. (5%) Suppose you run TCPClient before you run TCPServer. What happens? Why?
   If you run TCPClient first, then the client will attempt to make a TCP connection with a non-existent server process. A TCP connection will not be made.

   b. (5%) Suppose you run UDPClient before you run UDPServer. What happens? Why?
   UDPClient doesn't establish a TCP connection with the server. Thus, everything should work fine if you first run UDPClient, then run UDPServer, and then type some input into the keyboard.

   c. (5%) What happens if you use different port numbers for the client and server sides?
   If you use different port numbers, then the client will attempt to establish a TCP connection with the wrong process or a non-existent process. Errors will occur.