CS 3873: Net-Centric Computing

Assignment #: Assignment Title (if any)

Student Name: _Adrian Freeman__ Student Number: _3661616__

[Mandatory] Declaration: "I warrant that this is my own work."

Signed by _Adrian Freeman__

(You can type in your name as your signature.)

From the UNB Undergraduate Calendar, available at the website: http://www.unb.ca/academics/calendar/undergraduate/current/regulations/universitywideacademicregulations/viii-academicoffences/index.html.

"Plagiarism includes:

1. quoting verbatim or almost verbatim from any source, regardless of format, without acknowledgement;
2. adopting someone else's line of thought, argument, arrangement, or supporting evidence (such as, statistics, bibliographies, etc.) without indicating such dependence;
3. submitting someone else's work, in whatever form (essay, film, workbook, artwork, computer materials, etc.) without acknowledgement;
4. knowingly representing as one's own work any idea of another."

Penalties for plagiarism and other academic offences can be found at the above website.

1. True or False?

   a. A user requests a Web page that consists of some text and three images. For this page, the client will send one HTTP request message and receive four HTTP response messages. Assume HTTP/1.1 is used.

      False, The client will still need to make an HTTP request for each item, but with HTTP/1.1, the client only has to establish a connection with the host at the beginning, and the connection will persist.

   b. With non-persistent connections between browser and origin server, it is possible for a single TCP connection to carry two distinct HTTP request messages.

      False, Each HTTP request message requires their own TCP connection.

   c. The ***<u>Date:</u>*** header in the HTTP response message indicates when the object in the response was last modified.

      False, the ***<u>Date:</u>*** header is used to indicate the date and the time that the response was created and sent by the server

   d. HTTP response messages never have an empty entity body.

      False, HTTP response messages can have empty entity bodies. For example, the response for an HTTP HEAD request does not retrieve the body, as well as 404 NOT FOUND and a 304 NOT-MODIFIED responses.

2. (4 points) Consider the following string of ASCII characters that were captured by Wireshark when the browser sent an HTTP GET message (i.e., this is the actual content of an HTTP GET message). The characters <cr><lf> are carriage return and line-feed characters (that is, the italic character string <cr> in the text below represents the single carriage-return character that was contained at that point in the HTTP header). Answer the following questions, indicating where in the HTTP GET message below you find the answer.

```
GET /cs453/index.html HTTP/1.1<cr><lf>Host: gai
a.cs.umass.edu<cr><lf>User-Agent: Mozilla/5.0 (
Windows;U; Windows NT 5.1; en-US; rv:1.7.2) Gec
ko/20040804 Netscape/7.2 (ax) <cr><lf>Accept:ex
t/xml, application/xml, application/xhtml+xml, text
/html;q=0.9, text/plain;q=0.8,image/png,*/*;q=0.5
<cr><lf>Accept-Language: en-us,en;q=0.5<cr><lf>Accept-
Encoding: zip,deflate<cr><lf>Accept-Charset: ISO
-8859-1,utf-8;q=0.7,*;q=0.7<cr><lf>Keep-Alive: 300<cr>
<lf>Connection:keep-alive<cr><lf><cr><lf>
```

a. What is the complete URL (in the format http://.......) of the object requested by the browser?

   http://gaia.cs.umass.edu/cs453/index.html This is constructed by concatenating the directory of the object being requested to the end of the host URL.

b. What version of HTTP is the browser running?

   HTTP/1.1, found in the top line, next to the object being requested.

c. Does the browser request a non-persistent or a persistent connection?

   Persistent, which is done by requesting the Connection:keep-alive. If it were Connection:close it would be non-persistent. Found on the bottom line

d. What is the file type of the requested object?

   .html, which is indicated by the very beginning of the request message.

3. (2 points) Suppose within your Web browser you click on a link to obtain a Web page. The IP address for the associated URL has been cached in your local host, so a DNS lookup is not necessary to obtain the IP address. Suppose that the Web page associated with the link contains a small amount of HTML text, and it references to 8 very small objects on the same server. Neglect transmission times and let RTTw denote the RTT between the local host and the Web server containing the objects. How much time (in terms of RTTw) elapses

   a. if non-persistent HTTP is used with no parallel TCP connections?

      If non-persistent HTTP is used, we need to make a TCP connection for every request. We have 8 small objects and a small amount of HTML text (9 objects total) So for our total RTT, we get 2*RTT*9 = 18RTT

      Therefore **RTTw = 18RTT**

   b. if persistent HTTP is used and requests for referenced objects are sent out back-to-back without waiting for responses for previous requests?

      Persistent HTTP with Pipelined Persistent Connection: 1 RTT to establish the connection, and 1RTT to request and recieve all of the data. Total of **RTTw = 2RTT**

4. (1 point) In BitTorrent, consider a new peer Alice that joins BitTorrent without possessing any chunks. Without any chunks, she cannot become a top-four uploader for any of the other peers, since she has nothing to upload. How then will Alice get her first chunk?

   When Alice joins the torrent without any chunks, the torrent's tracker then selects a subset of peers, denoted as 'neighbouring peers' from the list of participating peers. Alice attempts to establish a TCP connection with each of her neighbouring peers, after establishing a connection with them, she can begin downloading chunks of the torrent from them.

5. (6 points) Consider distributing a file of F = 20 GB to N peers. The server has an upload rate of u_s = 1 Gbps, and each peer has a download rate of d_i = 20 Mbps and an upload rate of u_i. For N = 10, N = 100, or N = 1000, and u_i = 500 kbps, u_i = 5 Mbps, or u_i = 25 Mbps, prepare a table giving the distribution time for each of the combinations of N and u_i for both client-server distribution and P2P distribution. For simplicity, round your results for the distribution time into integers in terms of seconds.

Distribution Time for Client/Server D = max{NF/u_s , F/d_i}

u_i (Mbps)	N = 10	N = 100	N = 1000
0.5	D = max{10*20*8/1, $20*10^9*8/20*10^6$ = max{1600, 8000} = 8000s	D = max{100*20*8/1, $20*10^9*8/20*10^6$} = max{16000, 8000} = 16000s	D = max{1000*20*8/1, $20*10^9*8/20*10^6$} = max{160000, 8000} = 160000s
5	8000s	16000s	160000s
25	8000s	16000s	160000s

Distribution Time for Peer-to-Peer D = max{F/u_s, F/d_i, NF/(u_s + Nu_i)

u_i (Mbps)	N = 10	N = 100	N = 1000
0.5	D = max{20*8/1, $20*8*10^3/20$, $10*20*8*10^3/(1*10^3+10*0.5)$ = max{160, 8000, 1592} = 8000s	D = max{20*8/1, $20*8*10^3/20$, $100*20*8*10^3/(1*10^3+100*0.5)$ = max{160, 8000, 15238} =15238s	D = max{20*8/1, $20*8*10^3/20$, $1000*20*8*10^3/(1*10^3+1000*0.5)$ = max{160, 8000, 106667} =106667s

5	D = max{20*8/1, $20*8*10^3/20$, $10*20*8*10^3/(1*10^3+10*5)$ = max{160, 8000, 1524} = 8000s	D = max{20*8/1, $20*8*10^3/20$, $100*20*8*10^3/(1*10^3+100*5)$ = max{160, 8000, 10667} =10667s	D = max{20*8/1, $20*8*10^3/20$, $1000*20*8*10^3/(1*10^3+1000*0.5)$ = max{160, 8000, 26667} =26667s
25	D = max{20*8/1, $20*8*10^3/20$, $10*20*8*10^3/(1*10^3+10*25)$ = max{160, 8000, 1280} = 8000s	D = max{20*8/1, $20*8*10^3/20$, $100*20*8*10^3/(1*10^3+100*25)$ = max{160, 8000, 4571} =8000s	D = max{20*8/1, $20*8*10^3/20$, $1000*20*8*10^3/(1*10^3+1000*0.5)$ = max{160, 8000, 6154} =8000s