Programming Assignment #2

CSCE 5580 - Computer Networks

Spring 2018

100 Points

Due: 03/19/2018, 11:59 PM

Instructions:

i). Compile the C programs and make sure it's working.
ii). Comment your code.
iii). Create a readme file text file that describes the working and usage of the code.
iv). Please create a zip archive of your assignment folder (readme, code, and header files) and upload the zip file.
v). Not following the above instructions could result up to 20% deduction from your assignment score.

Objective:

i). Create a web proxy server that can be connected by a single client and would only allow http requests.
ii). The proxy server should be able to cache up to five recent websites.

Requirements:

1. Create a C-based client-server architecture using sockets.

2. The proxy server should be able to accept and service single client's http requests.

3. The proxy server should run on cse01.cse.unt.edu machine and the client should run on cse02.cse.unt.edu machine

4. The proxy server should be able to cache five recent requested webpages.

5. When a page is requested, the proxy sever should be able to check if the requested page is current. If the page is not current, update the cache with the current page and return the current page. If the page is current, return the page from the cache.

Procedure:

1. Create a C-based server that can accept single client's request using sockets.

2. The created proxy server should also be able to connect to the client requested website through port 80.

3. Make sure the proxy server runs on cse01.cse.unt.edu and the format to start the proxy server as follows

pserver <port_number>

where pserver is the proxy server executable and port_number is the port number on which the proxy server listens.

4. Create a C-based client that can connect to the proxy server using sockets.

5. Make sure the client runs on cse02.cse.unt.edu and connects the proxy server.

The user can request the desired web page using the below format

client <port_number>

url: <url>

where client is the client executable, port_number is the port number on which the client connects the server and url is the requested url

6. Once the proxy server gets a request from the client, it then forwards the request to the web server. Figure 1 shows the overall architecture.

7. The proxy server checks for the response from the web server.

8. If the HTTP response is 200, the returned web page from the web server is cached in the proxy server. The proxy server stores the webpage in a file and assigns a filename based on the time of visit. The filename format is YYYYMMDDhhmmss.

Where YYYY is the year, MM is the month, DD is the day, hh is the hour in 24-hour format, mm is the minutes, and ss is the seconds when the website was visited

9. A list file (list.txt) is created which stores the URL of the webpage and the associated cached web page filename.

10. The list file stores five recent URLs. The cached websites that are not listed in the list file should be deleted.

11. Once the returned web page is cached, the web page is forwarded to the client. Verify to see if the returned page is same as the browser returned page.

12. If the HTTP response is not 200, do not cache the web page instead forward the HTTP response to the client.

13. When the client requests a webpage that is in the list.txt file, the proxy server checks if the page is current by contacting the server. If the page is modified, the proxy server updates the cache with new page and returns the page otherwise returns the page from the cache.

14. Test web caching by accessing multiple websites

15. A sample list.txt file is available on Canvas for reference.

Deliverables:

1. Commented server and client C code

2. A readme file text file that describes how to compile, execute, and test the C codes.

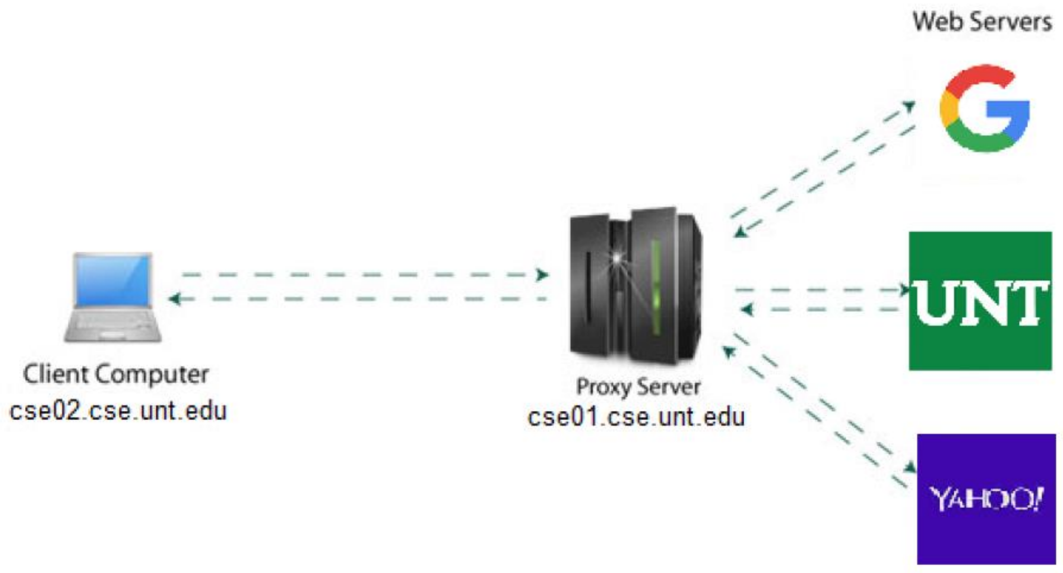

Figure 1. Overall Architecture