Lab Assignment 3

CSC317: Computer Networks

Submission instructions: Submit the source files containing your code. All files must be submitted on Moodle.

In this assignment, you will develop a proxy server with simple caching. Consider a system where clients request Web pages via a proxy server. A request from a client includes a hostname (e.g. www.cornellcollege.edu) and a port number (typically 80). Once a request is received by the proxy server, it first checks if the response is already present in cache. A cache entry has a lifetime of 1 minute and the entry is considered to be stale once the lifetime is exceeded. If the proxy server finds the response in cache and if the response is not stale, the server simply responds to the client with the entry from the cache. The server, otherwise, sends a HTTP GET request to the host specified by the request, stores/updates the host's response in cache, and replies the client with the response. The communication diagram is shown in Figure 1.

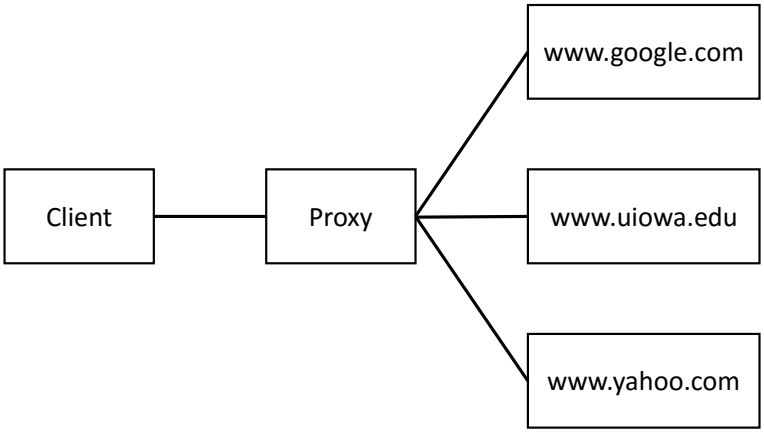

Figure 1: Communication diagram.

Before you start coding, you are advised to get an understanding of HTTP request/response by using telnet. For example, to telnet www.cornellcollege.edu, type the following com-

mand in terminal and hit Enter.

```
telnet www.cornellcollege.edu 80
```

When connected with the host via telnet, type the HTTP GET request (as given below) and hit Enter twice.

```
GET / HTTP/1.0
Host:  www.cornellcollege.edu
Connection:  close
```

You should be able to see the response in terminal. Note the format of the HTTP GET request above. Use this format for the requests in your programs.

Your task is to write programs for both the client and proxy server. The client will send HTTP GET requests to multiple Web servers via the proxy server. Your client should connect to the proxy server and send HTTP GET requests for the following 3 hosts with an interval of 30 seconds.

1. www.uiowa.edu
2. www.google.com
3. www.yahoo.com

Once a response is received from the proxy server, send the exact same HTTP GET request (from the client) directly to the Web server. Check whether the content (excluding header) received from the proxy server matches with the content received directly from the Web server. Output the hostnames for which the contents match and do not match.

To get you started, an example program (filename: **HTTP_GET_test.py**) has been posted on Moodle to show how to send a HTTP GET request using the socket API. The program also includes code for listening and storing the response. Note that the program is written for Python 3.5.x.

In short, you need to implement:

- A proxy server, which receives HTTP GET requests and responds with content from cache (with 1 minute stale interval) or fetches it from the Web server.
- A client, which requests www.uiowa.edu, www.google.com, and www.yahoo.com with 30 seconds gap between two consecutive requests, and verifies the proxy server's response.