Ameer Khan
ID: 1476214
3/1/19

CMPE 156: Lab 5 Documentation

In order to compile the program, first go to src folder which has the makefile, proxy.c and forbidden-site.txt. First run make clean, then run make which will create an executable called proxy in the bin folder. When you compile the program in the src folder, run the program by doing ../bin/proxy <listening port number> <forbidden-sites.txt>. I tested the program using curl -v –proxy IP address:port url.

Now I will explain **the protocol description**. The main proxy thread sets up and waits for a TCP connection from a client. Once the proxy program receives a connection, it gets the information for the correct browser/host requested. Then it checks if it is a valid GET or HEAD request and checks if it is HTTP 1.1/1.0. If it is not valid, a 501 message of Not implemented is sent. Then I check if the website that is requested is forbidden or not. If it is, then I send a 403 message of Forbidden URL. After that, I tried to implemented concurrency. I have a thread spawn two more thread/children. This is done by two functions I created client_browser and browser_client. Both these threads keep track of forwarding information such as request information from client to browser and html body information from browser to client. If the information is sent successfully, then a code of 200 ok is sent.

Now I will explain the message formats. The user has to insert two argument which is the port number and name of the forbidden text file. Once the arguments have been entered on the proxy server, I have the request headers printed out with the added forwarded header and then I print out the buffer with the actual html page. This is printed on the proxy server to help me keep track that the right information is being sent. In between these two messages, I print out the host name and IP address of the host.

Now I will talk about error handling done in the proxy program. The error checking done is to check the correct number of arguments input by the user, error checking for opening the socket, error checking for checking if the IP address is valid, error checking for connecting to the server/client, error checking for binding to a socket, error checking for creating a thread, error checking for sending/receiving information, error checking for opening a file and erroring checking for writing to the log file. I also have conditions to send the right error codes when it needs to send such as 400 bad request, 403 forbidden urn and 501 Not implemented.