Code

Initially, we were using C++ to implement the proxy server. However, we decided to switch to Python for the convenient functions that Python provides.

A Server class was created and its constructor takes in a port number, the telemetry flag and blacklist array for initialisation. The port number and telemetry flag are arguments taken from the command line, while the array of blacklisted hostnames is constructed from the provided blacklist file before being passed to the Server constructor.

During the construction of the Server object, it creates a socket and binds it to the specified port and starts to listen for any incoming requests from the browser. A semaphore of initial value 8 is created and used to allow for a maximum of 8 threads running concurrently to communicate with the web server that the user wishes to access. After this initialisation, the Server then starts to accept connections from the client. A thread is used to handle each connection, and the handling of connections is done by a handleConnection function.

Inside the thread, the function will first attempt to acquire the semaphore. This allows us to have only a maximum of 8 connections being handled at any one time. Once the semaphore has been acquired, the thread will acquire the initial CONNECT request from the client. Then, the request is checked to make sure its method is indeed CONNECT and that the hostname requested is not in the blacklist. Lastly, the thread attempts a connection to the requested web server and sends back a confirmation message with code 200 to the client.

Then, using Python's select function, we open 2 read pipes and relay the information back and forth between the web server and the client. This has a timeout duration of 5 seconds. During the relay, we increment the number of bytes received from the website server. Finally, when no more data needs to be transferred, the connection is closed. At this time, telemetry data will be printed to the console if the <flag_telemetry> argument had been set to 1 when the proxy server was started.

Discrepancy between HTTP 1.0/1.1

HTTP 1.0 does not support keep-alive connections as such it times out faster as compared to HTTP 1.1 connections which can stay alive for much longer. This is evident from the telemetry between HTTP 1.0 and 1.1 where certain connections in HTTP 1.1 last much longer than most other connections. This does not happen when HTTP 1.0 is used.