Kyle jang
ID: 11752589

Implementation Summary

My implementation of the HTTP proxy server uses a sequential approach to handling the web requests between user and server. The proxy acts like a middleman, accepting connections from users and forwarding the request to the server.

Connection Handling

-The Proxy listens on port 48062 in my test
-Using a socket interface it accepts the users connections request
- All connections are handled sequentially in a loop

Request Processing

-Parses the inputted HTTP requests to extract the method
-Converts HTTP/1.1 to HTTP/1.0 for compatibility
-Implements proper Header handling

Server Communication

-Establishes a connection to the targeted web servers
-Forwards the modified requests to the servers
-Takes the servers responses and relays them to the user

Testing Results

-When using Curl, it successfully retrieved local content (HTML and the cougar image) which shows handling of basic HTTP requests

-When testing NetCat (nc -l 48064) its showed proper GET request syntax and correct header implementations.

-When using FireFox the localhost displayed the image and posted successfully
-Some websites like google loaded properly
-Some websites like wiki.com failed to load

Analysis

Most modern websites cannot be accessed using the proxy due to
-Proxy only implementing HTTP protocol
-Most modern websites redirect HTTPS
-Needing HTTP/1.1 instead of /1.0
-Complex features like multiple simultaneous connections
-Complex redirect chains

Conclusion
Due to the limitation of HTTP/1.0 most modern day websites are inaccessible because of the complex features HTTP/1.1 adds, Only simple HTTP requests work properly demonstrating core network concepts, while also showing the limitation and evolution of web development with HTTP/1.0 and /1.1