Twisha Sharma
CSE 156
Professor M.Parsa

Final Project - Proxy Server

Code Design and Functionality

1. **Server Initialization**
At startup, the server initializes its configuration by loading settings, such as the port it will listen on, paths to any relevant files (like a forbidden sites list), and other operational parameters. This phase is crucial for setting up the environment in which the proxy operates.

2. **Request Handling**
The core functionality revolves around handling incoming client requests. This involves several steps:

- **Listening:** The server listens on a specified port for incoming connections from clients.
- **Connection Handling:** Upon accepting a connection, the server reads the HTTP request from the client, parsing necessary information such as the request type (GET, POST, etc.), the target URL, and any headers.
- **Content Filtering:** If the request URL matches any entry in the forbidden sites list, the server blocks the request and sends an appropriate response back to the client, such as an HTTP 403 Forbidden status.

3. **Forwarding Requests**
For allowed requests, the server then forwards the request to the target web server on the internet. This might involve:

- **DNS Lookup:** Translating the hostname of the target URL into an IP address if necessary.
- **Connecting and Forwarding:** Establishing a connection to the target server, sending the request, and waiting for the response.

4. **Response Handling and Caching**
Once the proxy server receives the response from the target server, it processes it:

- **Caching (Optional):** If the response is cacheable, the proxy might store it for future requests.
- **Response Delivery:** The server then sends the response back to the original client.

5. **Logging and Monitoring**

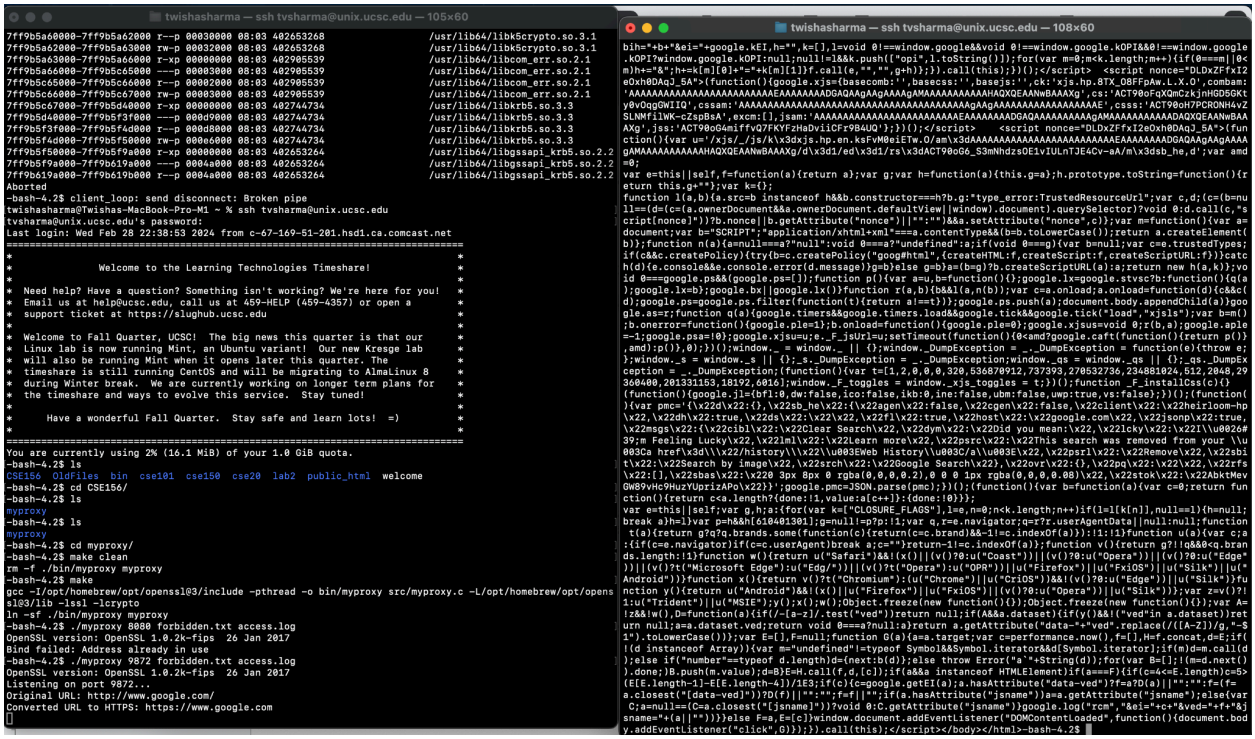
The server keeps logs of requests and responses, including details like request URLs, timestamps, and possibly the size of the data transferred. This is important for monitoring, debugging, and analyzing the traffic going through the proxy.

6. **Concurrency Handling**
To efficiently serve multiple clients simultaneously, the server implements concurrency, often through threading or asynchronous I/O. This ensures that the proxy can handle numerous requests in parallel without blocking.

Test Cases

<u>Case 0: Basic GET Request</u>

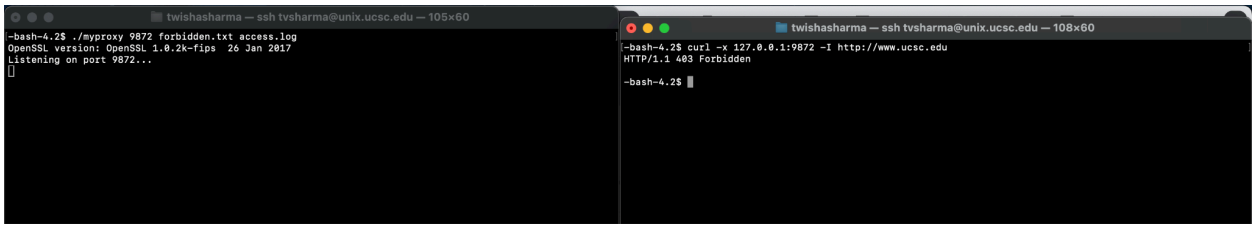
- Objective: Verify that the proxy server correctly handles a basic HTTP GET Request and relays the content from the target server
- Procedure: ./myproxy 9872 forbidden.txt access.log curl -x http://127.0.0.1:9872 http://www.google.com
- Outcome:



<u>Case 1: Forbidden Website Access</u>

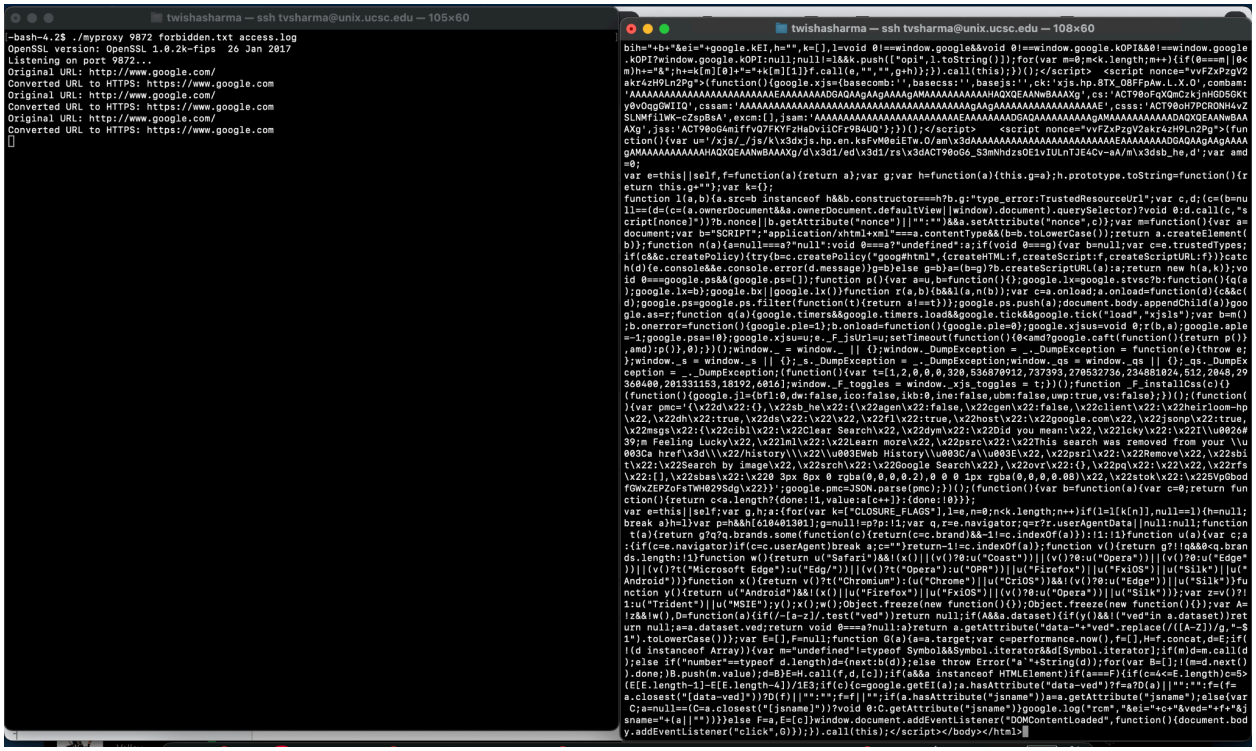
- Objective: Ensure that the proxy server correctly blocks access to the websites listed in the forbidden sites file.

- Procedure : ./myproxy 9872 forbidden.txt access.log curl -x 127.0.0.1:8080 -I http://www.ucsc.edu
- Outcome:



Case 2: Concurrent GET Requests

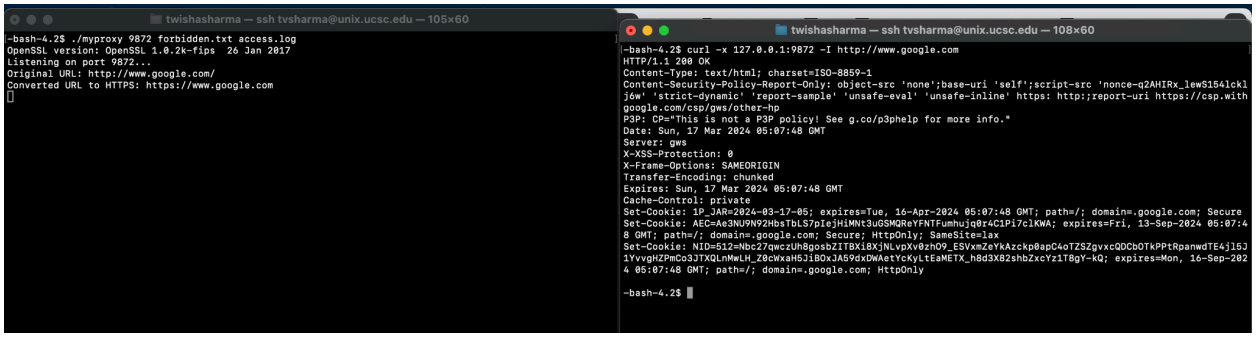
- Objective: Test the server's ability to handle multiple simultaneous GET requests
- Procedure: ./myproxy 9872 forbidden.txt access.log curl -x http://127.0.0.1:8080 http://www.google.com & curl -x http://127.0.0.1:9872 http://www.google.com & curl -x http://127.0.0.1:8080 http://www.google.com
- Outcome:



Case 3: Head Request

- Objective: Verify that the proxy server supports HEAD requests and correctly relays the headers from the target server.
- Procedure: ./myproxy 9872 forbidden.txt access.log curl -x 127.0.0.1:9872 -I http://www.google.com

- Outcome:


twishasharma — ssh tvsharma@unix.ucsc.edu — 105×60
-bash-4.2$ ./myproxy 9872 forbidden.txt access.log
OpenSSL version: OpenSSL 1.0.2k-fips 26 Jan 2017
Listening on port 9872...
Original URL: http://www.google.com/
Converted URL to HTTPS: https://www.google.com
twishasharma — ssh tvsharma@unix.ucsc.edu — 108×60
-bash-4.2$ curl -x 127.0.0.1:9872 -I http://www.google.com
HTTP/1.1 200 OK
Content-Type: text/html; charset=ISO-8859-1
Content-Security-Policy-Report-Only: object-src 'none';base-uri 'self';script-src 'nonce-q2AHIRx_lewS154lckl
j6w' 'strict-dynamic' 'report-sample' 'unsafe-eval' 'unsafe-inline' https: http:;report-uri https://csp.with
google.com/csp/gws/other-hp
P3P: CP="This is not a P3P policy! See g.co/p3phelp for more info."
Date: Sun, 17 Mar 2024 05:07:48 GMT
Server: gws
X-XSS-Protection: 0
X-Frame-Options: SAMEORIGIN
Transfer-Encoding: chunked
Expires: Sun, 17 Mar 2024 05:07:48 GMT
Cache-Control: private
Set-Cookie: 1P_JAR=2024-03-17-05; expires=Tue, 16-Apr-2024 05:07:48 GMT; path=/; domain=.google.com; Secure
Set-Cookie: AEC=Ae3NU9N92HbsTbLS7pIejHiMNt3uGSMQReYFNTFumhujq0r4C1Pi7clKWA; expires=Fri, 13-Sep-2024 05:07:4
8 GMT; path=/; domain=.google.com; Secure; HttpOnly; SameSite=lax
Set-Cookie: NID=512=Nbc27qwczUh8gosbZITBXi8XjNLvpXv0zhO9_ESVxmZeYkAzckp0apC4oTZSZgvxcQDCbOTkPPtRpanwdTE4jl5J
1YvgHZPmCo3JTXQLnMwLH_Z0cWxaH5JiBOxJA59dxDWAetYcKyLtEaMETX_h8d3X82shbZxcYz1T8gY-kQ; expires=Mon, 16-Sep-202
4 05:07:48 GMT; path=/; domain=.google.com; HttpOnly
-bash-4.2$