CS118 Project 1 – Web Server Implementation Using BSD Sockets

Anthony Lai 004445644

Elton Leong 204457607

High Level Description

The server design is based on the ClientServer_Example we were given to demonstrate how to use BSD sockets.

The server runs forever (until Ctrl+C is pressed) and forks itself whenever there is an incoming request. The child process then handles the incoming request while the parent remains to listen on the given port. There is a function `void respondToClient(int sockfd)` that processes and handles each request made to the server. It is called once by each child process. Most of the modifications to the code consist of changes to the function that handles the request.

After reading 511 bytes from the TCP socket, the HTTP request is parsed. The first line consisting of the method, URI, and version numbers is decoded, and this URI is saved for later, to lookup the file to be served. All of the HTTP request headers are parsed, stored in a hash table, and then ignored, as this is a rudimentary HTTP server.

For file lookup in the same directory as the webserver executable, a . is prepended to the request URI (e.g. "." + "/test.jpg" = "./test.jpg"). This method of converting URIs to filepaths is basic but handles files and directories. It is vulnerable to information leaking through a "/../../personaldata.txt" like attack if the client is chooses to send HTTP requests not through a browser but through a custom client, but that wasn't a requirement of the spec to handle properly. If this file cannot be found or if the file is a directory, then a 404 Not Found is the response given by the server. There is a hard coded HTML response accompanying this response. Otherwise, a 200 OK is sent, along with the Content-Type, Content-Length, and Last-Modified headers, followed by the requested file. Additional required headers are added to both 200 and 404 responses (Date, Server, Connection). Adding headers to the response is done via text-based appending to a response string, but in the future an HTTP response class could be created to make the addition of headers less error-prone.

Using a hash table, a hard coded mapping of file extensions to MIME types is made. For each request made to the server, the lower-case form of the extension is looked up in the hash table. If it cannot be found, the default MIME type "application/octet-stream" is given, which instead causes most browsers to download the given file.

Following the writing of all of the data to the TCP socket, the child process closes the socket fd and the child process exits successfully.

Difficulties

Nothing to report.

Manual

To compile the code, type `make`

Run the server using `./webserver $portnum`, where portnum is the desired port, e.g. `./webserver 42069`

To exit the server, hit Ctrl+C.

Sample Output

HTTP Request 1, a successful request of an HTML page

```
elton@TUESDAYSPECIAL:/mnt/c/Users/elton/Desktop/vm-shared/cs118/project1_204457607$ ./webserver 42069

GET /test.html HTTP/1.1

Accept: text/html, application/xhtml+xml, image/jxr, */*

Accept-Language: en-US,en;q=0.5

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/51.0.2704.79 Safari/537.36 Edge/14.14393

Accept-Encoding: gzip, deflate

DNT: 1

Host: localhost:42069

Connection: Keep-Alive
```

HTTP Response 1

```
HTTP/1.1 200 OK

Connection: close

Server: webserver/0.0.1

Content-Type: text/html

Last-Modified: Tue, 31 Jan 2017 22:47:38 GMT

Date: Thu, 02 Feb 2017 08:41:02 GMT

Content-Length: 25


<html>

    Hello

</html>
```

HTTP Request 2, a successful request of a GIF in a directory

```
elton@TUESDAYSPECIAL:/mnt/c/Users/elton/Desktop/vm-shared/cs118/project1_204457607$ ./webserver 42069

GET /dir/test.gif HTTP/1.1

Accept: text/html, application/xhtml+xml, image/jxr, */*

Accept-Language: en-US,en;q=0.5

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/51.0.2704.79 Safari/537.36 Edge/14.14393

Accept-Encoding: gzip, deflate

DNT: 1

Host: localhost:42069

Connection: Keep-Alive
```

HTTP Response 2

```
HTTP/1.1 200 OK

Connection: close

Server: webserver/0.0.1

Content-Type: image/gif

Last-Modified: Tue, 31 Jan 2017 22:47:38 GMT

Date: Thu, 02 Feb 2017 08:43:31 GMT

Content-Length: 154867


<<GIF data follows>>
```

HTTP Request 3, a request of a file that doesn't exist

```
elton@TUESDAYSPECIAL:/mnt/c/Users/elton/Desktop/vm-shared/cs118/project1_204457607$ ./webserver 42069

GET /test HTTP/1.1

Accept: text/html, application/xhtml+xml, image/jxr, */*

Accept-Language: en-US,en;q=0.5

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/51.0.2704.79 Safari/537.36 Edge/14.14393

Accept-Encoding: gzip, deflate

DNT: 1

Host: localhost:42069

Connection: Keep-Alive
```

HTTP Response 3

```
HTTP/1.1 404 Not Found

Connection: close

Server: webserver/0.0.1

Content-Type: text/html

Date: Thu, 02 Feb 2017 08:50:20 GMT

Content-Length: 27


<h1>404 Page Not Found</h1>
```

HTTP Request 4, a request of a directory

```
elton@TUESDAYSPECIAL:/mnt/c/Users/elton/Desktop/vm-shared/cs118/project1_204457607$ ./webserver 42069

GET /dir/ HTTP/1.1

Accept: text/html, application/xhtml+xml, image/jxr, */*

Accept-Language: en-US,en;q=0.5

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/51.0.2704.79 Safari/537.36 Edge/14.14393

Accept-Encoding: gzip, deflate

DNT: 1

Host: localhost:42069

Connection: Keep-Alive
```

HTTP Response 4

```
HTTP/1.1 404 Not Found

Connection: close

Server: webserver/0.0.1

Content-Type: text/html

Date: Thu, 02 Feb 2017 08:49:29 GMT

Content-Length: 27


<h1>404 Page Not Found</h1>
```