**CS118 Project 1**

Gary Zhang 504183404

Stanley Hsu 704338787

**High level server design:**

dostuff

Start

Socket setup

Socket bind

Wait connection

Fork

Delete connection

Parse message

Get file

Respond

Exit

Child

Parent

The server builds on the example code provided. The HTTP handler code is located in a single function “dostuff” that is called by the child process.

dostuff function:

void dostuff **(**int sock**)**

**{**

int n**;**

char temp**[**256**];**

char buffer**[**256**];**

char fname**[**256**];**

char fent**[**256**];**

char type**[**256**];**

bzero**(**buffer**,**256**);**

bzero**(**fname**,** 256**);**

n **=** read**(**sock**,**buffer**,**255**);**

**if** **(**n **<** 0**)** error**(**"ERROR reading from socket"**);**

printf**(**"Here is the message: %s\n"**,**buffer**);**

buffer**[**255**]** **=** 0**;**

/\*

if (n == 256)

{

write(sock, "HTTP/1.1 413 Payload Too Large\r\n", 34);

error("Request too long");

return;

}

\*/

// Check request type

char**\*** fpath\_start **=** strstr**(**buffer**,** "GET /"**);**

**if** **(**fpath\_start **!=** buffer**){**

write**(**sock**,** "HTTP/1.1 400 Bad Request"**,** 25**);**

error**(**"ERROR Bad Request"**);**

**return;**

**}**

// Obtain file path

fpath\_start **+=** 5**;** // "GET /"

char**\*** fpath\_end **=** strstr**(**buffer**,** " HTTP"**);**

int fpath\_length **=** fpath\_end **-** fpath\_start**;**

strncpy**(**fname**,** fpath\_start**,** fpath\_length**);**

fname**[**fpath\_length**]** **=** '\0'**;**

printf**(**"Filename is %s\n"**,** fname**);**

// Detect file extension

char**\*** fent\_start **=** **&**fname**[**fpath\_length**];**

**while** **(**fent\_start **>=** fname**)**

**{**

**if** **(\***fent\_start **==** '.'**)**

**break;**

**else** **if** **(\***fent\_start **==** '/'**)**

**{**

fent\_start **=** 0**;**

**break;**

**}**

fent\_start**--;**

**}**

char**\*** fent\_end **=** **&**fname**[**fpath\_length**];**

int fent\_length **=** fent\_end **-** fent\_start**;**

// No file extension?

**if** **(**fent\_start **<** fname**)**

**{**

printf**(**"No file extension"**);**

fent\_length **=** 0**;**

fent**[**0**]** **=** '\0'**;**

**}**

**else**

**{**

strncpy**(**fent**,** fent\_start**,** fent\_length**);**

fent**[**fent\_length**]** **=** '\0'**;**

printf**(**"File extension is %s\n"**,** fent**);**

**}**

**if** **(!**file\_exist**(**fname**))**

**{**

// printf ("404: Not Found\n");

write**(**sock**,** "HTTP/1.1 404 Not Found\r\n"**,** 24**);**

write**(**sock**,** "Content-Language: en-US\r\n"**,** 25**);**

write**(**sock**,** "Content-Length: 0\r\n"**,** 20**);**

write**(**sock**,** "Content-Type: text/html\r\n"**,** 25**);**

write**(**sock**,** "Connection: close\r\n"**,** 19**);**

**return;**

**}**

// printf("Passed file existence test\n");

// open file, get filesize

FILE**\*** f **=** fopen**(**fname**,** "rb"**);**

fseek**(**f**,** 0L**,** SEEK\_END**);**

int fsize **=** **(**int**)** ftell**(**f**);**

fseek**(**f**,** 0L**,** SEEK\_SET**);**

printf**(**"Filesize is %d\n"**,** fsize**);**

// read the file into fbuf

char**\*** fbuf **=** **(**char **\*)**malloc**(**fsize**\*sizeof(**char**));**

fread**(**fbuf**,** 1**,** fsize**,** f**);**

write**(**sock**,** "HTTP/1.1 200 OK\r\n"**,** 16**);**

write**(**sock**,** "Content-Language: en-US\r\n"**,** 25**);**

char formatted\_CL**[**256**];**

sprintf**(**formatted\_CL**,** "Content-Length: %d\r\n"**,** fsize**);**

write**(**sock**,** formatted\_CL**,** strlen**(**formatted\_CL**));**

// change content-type according to file extension

**if(**strcmp**(**fent**,** ".jpg"**)** **==** 0 **||** strcmp**(**fent**,** ".jpeg"**)** **==** 0**){**

strncpy**(**type**,** "image/jpeg\0"**,** 11**);**

**}**

**else** **if** **(**strcmp**(**fent**,** ".gif"**)** **==** 0**){**

strncpy**(**type**,** "image/gif\0"**,** 10**);**

**}**

**else** **if** **(**strcmp**(**fent**,** ".html"**)** **==** 0**){**

strncpy**(**type**,** "text/html\0"**,** 10**);**

**}**

**else{**

strncpy**(**type**,** "text/plain\0"**,** 11**);**

**}**

printf**(**"File type is %s\n"**,** type**);**

char formatted\_CT**[**256**];**

sprintf**(**formatted\_CT**,** "Content-Type: %s\r\n"**,** type**);**

write**(**sock**,** formatted\_CT**,** strlen**(**formatted\_CT**));**

write**(**sock**,** "Connection: close\r\n\r\n"**,** 21**);**

write**(**sock**,** fbuf**,** fsize**);**

free**(**fbuf**);**

fclose**(**f**);**

**return;**

**}**

**Difficulties:**

There were difficulties in making sure the correct CR/LF characters are sent, and that overflow was avoided on buffers.

The HTTP protocol and the BSD socket API were new to us. We had to decide which parts of the HTTP protocol to implement.

**Build and Run:**

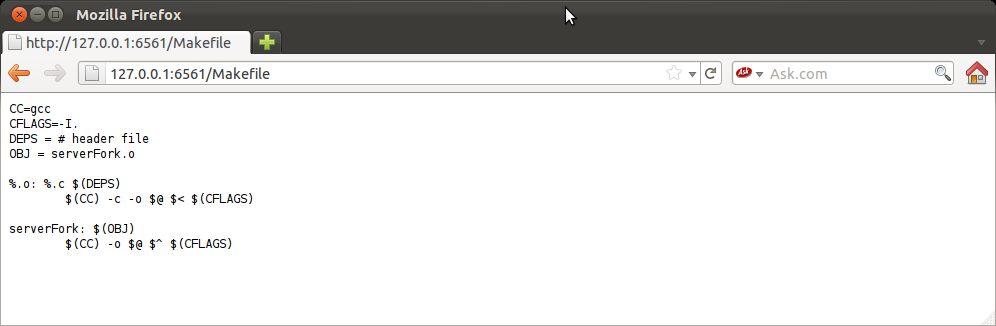
To build: Run make in the source code directory

To run:./serverFork <port>

**Examples:**

In these examples there is a “test2..” directory containing “test.html”, “a.jpg”, and “b.gif”

(1)



Server terminal output: (terminal output includes debug printf outputs)

Here is the message: GET /Makefile HTTP/1.1

Host: 127.0.0.1:6561

User-Agent: Mozilla/5.0 (X11; Linux i686; rv:7.0.1) Gecko/20100101 Firefox/7.0.1

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8

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

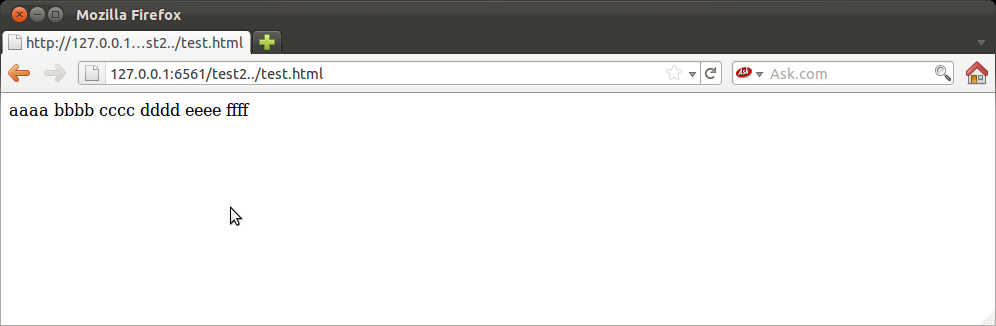
Accept-Encoding: gzip

Filename is Makefile

No file extensionFilesize is 154

File type is text/plain

(2)



Server terminal output:

Here is the message: GET /test2../test.html HTTP/1.1

Host: 127.0.0.1:6561

User-Agent: Mozilla/5.0 (X11; Linux i686; rv:7.0.1) Gecko/20100101 Firefox/7.0.1

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8

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

Accept-Encod

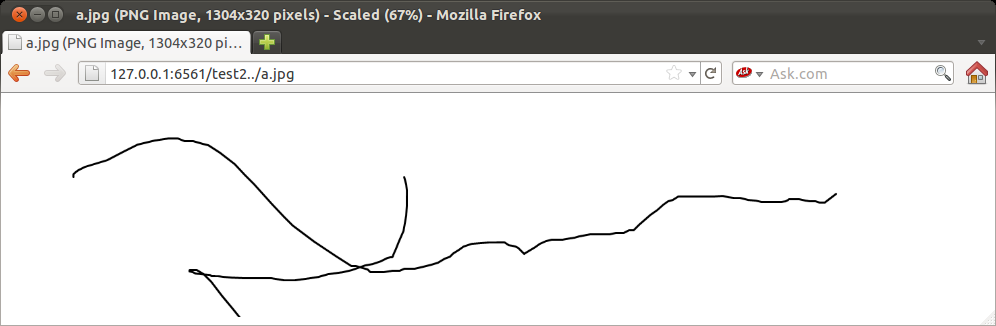
Filename is test2../test.html

File extension is .html

Filesize is 30

File type is text/html

(3)



Server terminal output:

Here is the message: GET /test2../a.jpg HTTP/1.1

Host: 127.0.0.1:6561

User-Agent: Mozilla/5.0 (X11; Linux i686; rv:7.0.1) Gecko/20100101 Firefox/7.0.1

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8

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

Accept-Encoding:

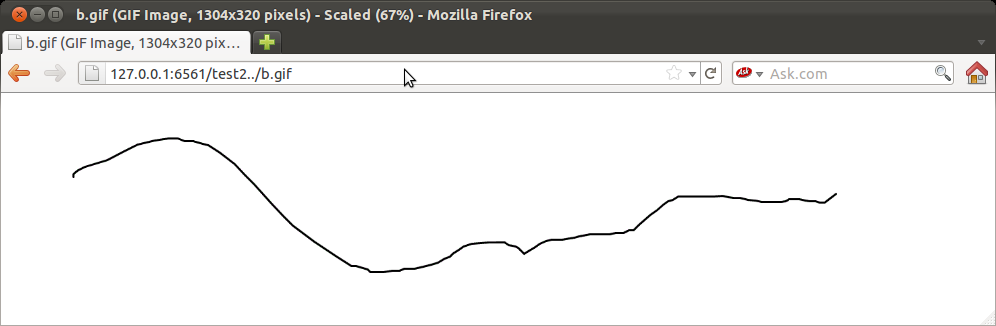
Filename is test2../a.jpg

File extension is .jpg

Filesize is 12439

File type is image/jpeg

(4)



Server terminal output:

Here is the message: GET /test2../b.gif HTTP/1.1

Host: 127.0.0.1:6561

User-Agent: Mozilla/5.0 (X11; Linux i686; rv:7.0.1) Gecko/20100101 Firefox/7.0.1

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8

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

Accept-Encoding:

Filename is test2../b.gif

File extension is .gif

Filesize is 3938

File type is image/gif