1. Can we adapt portions of lecture code that is relevant to the project?

Answer:

Yes, you may adapt it, provided you understand it well.

2. How big we should make the buffer size?

Answer:

There is not a specific size for buffer size. But roughly speaking you shouldn't put it too small as it will incur loop overhead, nor very big because it will consume your system's memory and most of it will remain unused. So you can use for example 256, 512, or 1024 bytes and avoid 1 byte or 21MB bytes buffer sizes (though they all should work perfectly).

3. When the requested file is specified as "file.txt" or "./file.txt", my server opens a file using that exact string as the argument to fopen (and then the server sends the file contents to the client, with no trouble). However, if I request "/file.txt", which makes my server try to use fopen with the string "/file.txt", fopen returns an error, and my server correspondingly returns a 404 error to my client.

Answer:

In Unix you can't open files by passing "/foo.txt" to the fopen function (it will return segmentation error). So, you either pass "./foo.txt" or simply "foo.txt" which they will both be interpreted as the file "foo.txt" which is in the current directory at which the server resides. So, regarding your question, our server should interpret the slash "/" in "/foo.txt" as the root for the absolute path, which begins from the CURRENT DIRECTORY "./" as a reference (NOT from the Unix root "/"). Hence, all files that need to be opened should be opened in a form similar to this fopen("./ece463/sanjay/mohd/foo.txt", "r").

4. Whenever I try to compile using: gcc -o httpclient httpclient.login.c, I get two errors I cannot pinpoint. error: dereferencing pointer to incomplete type warning: assignment makes pointer from integer without a cast Can you tell me what the problem is?

Answer:

Add the netdb.h library

5. Do I have to append a null terminator on every send?

Answer:

If you try to read or write "n" bytes, it should not matter whether there is a null terminator "\0" or not. If you read "n" bytes into a buffer, and then try to print that buffer using a printf, or do stuff like a "strcpy", then the "\0" would come into play. Just an example on this. If your buffer is of size n and you want to print it to the screen (like in the client example) then make sure you read n1 chars from the server stream and let the last char (nth byte) equal to '\0' so that the print knows the end of the string. So, you do only care about the null terminator when you want to use the print. But if you are a server, and you want to send a certain file, then you just read n chars into the buffer and send them to the requestor without any modification (without adding null terminator). And the client will read them as is, and do whatever requested, if requestor wants to: * save the read data to a file, then write data as is. * print the read data to the screen, then make sure your string ends with a null terminator.

6. I have the server working on my machine, but I cannot get it to work when I run it while SSHed into MSEE. Basically I get the same error I would get if I didn't put sudo before the command on my machine. Is there anything wrong?

Answer:

Port numbers from 0 to 1024 are reserved for privileged services. On your machine, the program should run without sudo if you use ports specified in the ‘Port Binding Policy’ on page 2 of the Lab 1 handout.

7. What else do we need to change besides changing SOCK_STREAM to SOCK_DGRAM when creating a UDP socket?

Answer:

In UDP, we bind but we don't listen. So, your code should call socket/bind and then either send or receive data directly. 8. Should the ping responses be generated and sent immediately? Answer: Yes, you don’t have to use fork for them.