CS536 Lab1: Build Your Own Web Server over Sockets

Due: 23:59:59 PM, Feb 8, 2023
Total points: 50 points

1 Goal

This is a programming lab. In this lab, you are going to develop client-server socket programming and build a concurrent Web server in C. Also, we will take the chance to learn a little bit more about how Web browser and web server work behind the scene.

2 Part A: Build Client-Server over Socket Programming (10 points)

- Set up server and client over Socket programming. The sample code and tutorial of socket programming are provided in Brightspace along with lab instructions.

  Basically, the server opens a server process over TCP while each client creates a client process to connect to this server process. After you open the server, you can open the client and you will be prompted to input a message on the client, and then the message will be sent to the server after clicking "Enter". Upon receiving the message, the server will send the message back, which will be displayed on the client terminal. The server will record and display the message on its terminal, too.

- (5 points) You need to slightly revise the provided code to allow the client and server to run on different machines which can be specified by the IP address and port number in the command line. You should run the code using the following commands:

  ```
  ./client [IP Address of the server] [Port number of the server]
  ./server [Port number of the server]
  ```

  You can begin by running both of server and client at the localhost. Namely, use *localhost* as [IP Address of the server]. After it works, you need to finally make the client and server run on two different machines at HAAS G050. Below are the machine names:

  amber01.cs.purdue.edu
  ...
  amber30.cs.purdue.edu

- (5 points) You need to revise the code to support multiple clients on the server using `Pthreads` and save it as `serverMul.c`. You can test the new server with at least two clients running on different machines or one machine with two different terminals.

3 Part B: Develop Your Web Server over HTTP/1.1(25 points)

Please read Chapter 2 of the textbook carefully. The textbook will help you to understand how HTTP works. Instead of using port 80 or 8080 or 6789 for the listening socket, you should pick your own to avoid conflicts. It is suggested not to use port numbers 0-1024.

This simple web server needs to implement the following features:

CS536 @Purdue

- When the client sends GET to request for one .html which exists, it should respond "*200 OK* "and return this .html file.
- When the client sends GET to request for one .html which **doesn't** exist, it should send the response "*404 Not found* ".
- When the client sends GET with the wrong format (e.g. URL String Syntax Error), it should send the response "*400 Bad Request* ".
- When the client sends GET with a different HTTP version, it should send the response "*505 HTTP Version Not Supported* ".

Please save your server code as `server1.c`. If you also revise the client code to test all the above features and save your client code as `client1.c`.

Several files are provided to test with your web server. Of course, please feel free to create and use your own test files.

- text.html: a html file which contains text only.
- picture.html: a html file which contains text and a small picture (< 200KB).
- bigpicture.html a html file which contains text and a big picture (> 1MB).

You need to test your server:

- You can put all the html files in the directory of your server, or more exactly, wherever you run your server.
- (5 points) For your server, you should be able to see the HTTP request format in the console of your server machine. For your client, you should be able to show the specific response given different HTTP requests and the content of the requested file. In the following test, please copy the print on the console at the server side into your lab report.
- (15 points, 5 points per one test html file) Connect to your server from a browser with the URL of http://<machine name>:<port number>/<html file name> and see if it works. For example, you can try `http://localhost:8888/index.html` if your web server uses the port number of 8888 and has a file called index.html in its root folder for all the web contents. Please make sure the server should work with the existing browser (Chrome, Firefox etc) to get the test webpages.
- (5 points) Test with other HTTP responses (404, 400 and 505).

4 Part C: Develop Your Web Server over HTTP/2.0 (15 points)

Please learn how HTTP/2.0 works to GET multiple objects in one webpage. Please modify the code of the client and server to support HTTP/2.0. Save your code as `client2.c` and `server2.c`. Please test your programs with video.html, and compare the page loading process using HTTP/2.0 with the one using HTTP/1.1.

- (10 points) Deploy your server and your client both at localhost and compare how to GET video.html via HTTP/2.0 and HTTP/1.1. Please HTTP/1.1 and HTTP/2.2 one at a time. Please show the order of objects and the time needed on the console. Please copy the print on the console at the client side into your lab report. Please summarize the main difference between HTTP/1.1 and HTPP/2.0 in the report.
- (5 points) Please run HTTP/1.1 and HTTP/2 server at one host (using different port numbers). Please deploy your clients at a different host and run the command to GET video.html almost at the same time. Do you see any changes from the above scenario with the server and clients at the same host? Please briefly explain why. Please include the print on the client console and answer the questions in your lab report.

CS536 @Purdue

5 Materials to turn in

You will submit your assignment at gradescope. You submission should zip all the files into "lab1_PUID.zip", including the following files:

1. All your source code files (`serverMul.c`, `client1.c`, `client2.c`, `server1.c`, `server2.c`)
   - Please add appropriate comments to your code and make it easy to understand.
2. Your project report called `lab1.*` (txt, pdf or word format). In the report,
   - Please include your name and student ID at the top of the first page.
   - Include a brief manual (readme) in the report to explain how to compile and run your source code (if the grader can't compile and run your source code by reading your manual, the project is considered not to be finished).
   - Include the answers and results as specified above.
   - Describe what difficulties you faced and how you solved them, if applicable.

6 More Tips and Support

1. Never submit .rar or other types of compressed file. You must submit .zip file. Otherwise, you will lose some points.
2. Please do not change the file names, otherwise you will lose some points.
3. For all the parts, you will need to support variable ports. The ports are given in the command, and you should not hard code it for your final submission version.
4. If connecting from off-campus, you will first need to connect to the Purdue VPN -https://webvpn.purdue.edu/ and then access the Linux machines.
5. If you want to use any late day, please send an email to cs536-ta@cs.purdue.edu and tell us.

More questions about the assignment should be posted on Campuswire or asked during PSOs.

CS536 @Purdue