Hands-on Project #2

CS 447

Instructor: Predrag Tosic
Term: Spring 2024

Do the following set of socket programming exercises; make sure you reference/use the correct exercises in the textbook (and in particular, that you use the current edition of Stallings; see syllabus). Then write a 4-6 page-long report (using Word or similar typesetting software; standard 12-pt font; 1.5-2X spacing) briefly describing your steps, which external resources/references you may have used if any, which challenges you may have encountered as you were going through different socket programming exercises; and a short summary on overall lessons learned. Also include your actual code in the same zipped archive with your 'project/lab report'.

These exercises will use Unix/Linux and whenever you're instructed to 'write a client program', do so in C/C++. You will include your code with your lab report (per above).

Exercise 1. First do **Exercise 2.4 in Chapter 2** of Stallings; then modify your solution of that exercise from the textbook, and write code a *stream-based echo server*, which can simultaneously handle multiple clients connecting to it. No modification of the client code is necessary, but multiple instances of the client should be started to test your server-side code. This program should compile for Winsock. **Hint:** use Windows threads functions. Read on socket programming in Stallings; and use/read-up on references provided therein.

Exercise 2. Write a client program to execute a single HTTP GET to a Web server. This program should conditionally compile for Winsock or BSD sockets.

Exercise 2a. Write a client program to execute a single HTTP GET on any type of content. The requested file will be stored locally. Hint: the HTTP response header fields are of variable number and variable length (depending on the URL requested and the Web server reached). This program should conditionally compile for Winsock or BSD sockets.

Exercise 2b. Write a client program to do serial HTTP GETs to a Web server. This program should conditionally compile for Winsock or BSD sockets.

Exercise 2c. Write a client program to do threaded parallel HTTP GETs to a Web server. This program should compile for Winsock.

Exercise 3. Write a program to accept a connect from a web browser (i.e., acts as an HTTP server) and responds with an HTML message. This program should conditionally compile for Winsock or BSD sockets.

Exercise 4. Write a simple Web server for Windows that serves HTML, text, and GIF images. Hint: to ensure simultaneous processing of multiple clients each GET should spawn its own thread. This program should compile for Winsock.

Exercise 5. Write a program to redirect all GET requests from a browser to a given URL using an HTTP 302 response message to the browser. This program should conditionally compile for Winsock or BSD sockets.