Lab 3

Echo Client and Server

Due September 26 at 10am

This lab will gives hands on practice with inter-process communication and client server applications. Processes that communicate over a network transfer data via sockets, an abstraction provided by the operating system. This lab will give you some practice working with C's standard socket API.

Echo Client-Server

A simple echo server echoes (repeats back) a message sent by a client. You will build an echo client that connects to an echo server. The client will connect to the server and send a message, the server will receive the message and immediately send the same message back to the client.

Instructions

1. You will need 2 files - echo_server.c (echo_server.c has been given to you) and echo_client.c. **You will create echo_client.c.**
2. The server listens for connections from a client on the localhost at port 22000. When the server receives a message it simply sends it back to the client. The server does not terminate after sending back the message. It continues to run and service requests from clients until terminated at the command line with cntl c.
3. The client program you need to write should prompt a user to enter a message. The client will send this one message to the server. After sending the message, the client will wait for a response from the server. After receiving the message echoed back by the server, the client should print to standard output: "Echo response received from server: *insert message here*". For example, if a client sends "hello" to the server, after receiving

the message back from the server, the client should print – “Echo response received from server: hello”. The client should then terminate.

Important Notes: You may assume that neither the message to the server nor the response will be longer than 15 bytes. You can statically allocate memory for the message like so:

buffer[16];

Because these messages are so short, you may assume that the full message is sent or received in every write or read call.

It is important that the only output written to standard output be from the client as described above or the grading test cases may fail.

Submit echo_client.c only since echo_server.c will be given to you.

Grading

80 points - Implementation

40 points if the client connects to the server.

80 points for complete implementation.

10 points

The code is in the correct file – echo_client.c and echo_server.c.

10 points

Your name: Add your name in the comments section at the top of echo_client.c

Readability: Make sure your code is indented and neatly commented. Do not leave commented out code in your submission. Comments describing what your code does is ok. Don’t leave old code in the source file.

Compilation: If your code does not compile, you will receive a 0 on the lab. Once notified, you will have 3 days to fix the lab to receive partial credit, up to 75%.