COP 6611
Assignment: Sockets

Due: March 24 2022 at 5:00p.m.

In this project, we will implement a "message server" in a distributed environment (i.e. between processes running on machines that are physically separate) by using BSD sockets.

In this assignment, we will provide a writer program (i.e. you will be given a writer program), and you are required to create a server program, which will communicate with writers by sockets. 3 identical writer programs (except for names, writer 1, writer 2, and writer 3, respectively) will run on one or more machines and the server on another machine (osnode16.cse.usf.edu).

Assignment Specifications:

1. Each writer program will connect and send a message (i.e. message "writer1" will be sent to the server from writer 1 — you do not have to do this, our TA will take care of this part) to the server.

2. The server process will accept this socket connection from a writer, and create a thread process after the socket is accepted.

3. After a thread is created it is responsible for taking care of communication (Step 4) with the writer, and the parent process loops back to listen for a connection — this makes the parent ready for another connection.

4. A message from the writer must be stored by the appropriate thread processes in shared memory. This memory is shared by the server and all its threads, and can store only one message. The message length will be no more than 20 characters (19 chars plus 1 char string terminator in C), such as "writer1" "writer2" and "writer3".

5. A writer process should be invoked (started) with the writer name. For instance, suppose writer program name is "writerp", the first writer should be started by typing "writerp" and will read the message from command line.

6. The shared memory must be properly protected as a critical section so that the stored message will never be corrupted even during the handling of concurrent connections from multiple writers.

7. After a message from a writer is successfully stored in the shared memory by one of the server threads, after a 2 second sleep, the actual contents of the shared memory should be sent back to the original writer by the same server thread.

8. Set the server up to run on osnode16.cse.usf.edu (10.247.53.97 is the IP address). Use port 1050 as the port for listening. You will need to examine the socket, connect, bind, listen, accept, send and recv man pages for the appropriate system calls. (some of them, such as send, gotten via man send). Your server should loop for a total of n connection accepts, where n will be 3 in our trials.
   Look at setsockopt to free the socket after use. For testing use a different port so as to avoid all competing for 1050.

I suggest that first you set up your server so that it listens and responds to 1 writer. You will upload your code into Canvas before the deadline.