NETWORK LAB

LAB ASSIGNMENT for Week # 7

Socket Programming - Concurrent Server using TCP Sockets

Program #1
String Reversal Concurrent Server using TCP Sockets

Modify your server code written for Program # 1 in Lab Assignment # 6, to change the behaviour of the server program from iterative to concurrent, so that more than one clients can be handled at a time. A concurrent server is a server that waits on the welcoming socket and then creates a new thread or process to handle the incoming request.

In this program use fork() system call to create new processes. There is a single process accepting incoming connections. After a connection is made, the server forks a copy of itself to process the request and then returns to accepting incoming connections. Modify your string reversal server code to fork a new process after accepting a new connection. The child process should handle the actual string reversal task, whereas the parent should continue to accept new connections. Make sure you close sockets in the right places. In particular, note that the file descriptor for each accepted connection needs to be closed in two places! Why?

To test your work, connect to your server with multiple clients at the same time and ensure they all work. When you are done testing your string reversal server, be sure to stop the server.

Program #2
Do the same modification in the server code CalcServerTCP.c written for Program # 2 in Lab Assignment # 6 to create a concurrent server that can handle multiple clients at a time. In this program use POSIX threads to create concurrency. Test your code by connecting multiple clients at the same time.