CS 39006: Lab Test 2

Date: April 4, 2022, Time: 2-15 pm to 3-45 pm

IMPORTANT:
Please write your name and roll no. in a comment as the beginning of every file you write.
Follow the submission instructions exactly, otherwise a small penalty may be imposed.

Problem Statement:

You have to design a TCP concurrent server and client to transfer a file. The details are given below.

Client Functionality:

The client will allow the user to delete a file in the server, or get a specified range of bytes from a file from the server.

For deletion, the client program will take two command line arguments (in this order): the word "del" followed by a string filename. It will then open a TCP connection to the server, and send the command "del" and the filename (both as null-terminated strings) in two **separate** send calls to the server. The server will delete the file and send a null-terminated string "delete success". If there is any error on the server side (for ex., the file cannot be found etc.), the server will just close the connection, nothing will be sent. The client will print either a success message or an error message as appropriate on the display and exit.

For getting contents of a file, the client program will take four things as command line arguments (in this order), the word "getbytes", a filename, and two non-negative integers x and y, $y \geq x$. It will then open a TCP connection to the server, and send the command, the filename, x, and y in four **separate** send calls. Each of them will be sent as a null terminated string. It will then wait to receive the file from the server. The server will send byte x to byte y of the file, and then close the connection (The first byte of a file is to be taken as byte 0 and the last byte as byte N – 1, where N is the number of bytes in the file). If the bytes are received successfully, the client prints them out on the display as they come, and exits when the whole thing is printed. If there is any error (for example, the file does not exist, x or y is larger than the size of the file, or some other error even in the middle of the transfer), the server closes the connection immediately. The client should then print an error message and exit. Note that the file can be of any size, so x and y can be anything. Also, the client will not know the maximum chunk size which the server will use for a send call.

You can assume that all command line arguments will be entered correctly, no need to check. Also, the filename is a simple name with no '/' (so assumed that it will be in current directory of the server if present)

Server Functionality:

The server will be a TCP concurrent server. However, instead of using processes, the server will use one **thread** to handle each connected client. The server will run indefinitely but you can assume that not more than 5 clients will be connected to the server at any one time (though the total number of clients over time can be much more).

The functionality of the server will be as follows:

1. The server will wait on a TCP socket.
2. On receiving a connection request from a client, the server will accept the connection and create a new thread T_C for the client. The main server code will go back to wait for any other client to connect.
3. The thread T_C will do the following:
    a. Wait to receive messages from the client. It will wait till it receives all the messages sent by the client depending on the command sent by client (see client description).
    b. For delete
        i. Check if the filename is present in the current directory. If present, the thread will delete the file and send the message “delete success” to the client. It will then close the connection and exit.
        ii. If there is any error, the thread will close the connection and exit.
    c. For getting file content of a file
        i. Check if the filename is present in the current directory, and if it can be opened for read. If not, close the connection and exit
        ii. Check the size of the file and whether the value of x and y supplied are both consistent with it. If not, close the connection and exit
        iii. Send the required bytes to the client in one or more send calls. You can assume that the file is a text file so send as characters. You are NOT allowed to send any other message (like the size of the file etc.) to the client, and NOT allowed to send a terminating null at the end (if byte y is already a null, that’s ok, but you should not put a null explicitly in your code at the end). Just send the bytes from byte x to byte y of the file. The first byte of a file is to be taken as byte 0 and the last byte as byte N – 1, where N is the number of bytes in the file.
        iv. During the transfer, if any error is encountered, close the connection and exit.
        v. After sending the required bytes, print a message “byte <x> to byte <y> of file <finename> sent” on the display, close the connection, and exit.

You can assume that all files to be read/deleted will be in the current directory only, otherwise an error should be generated. You can use global variables. You should take care of all issues arising out of properties of TCP and concurrent servers as needed for getting full credit.

Submission Instruction:

You have to submit the following two files: <your roll no>`_server.c` containing the server code and <your roll no>`_client.c` containing the client code. For example, if your roll number is 19CS3000, then the files should be named 19CS3000_server.c and 19CS3000_client.c. Upload both the files through the assignment submission link in the Moodle submission page (The assignment will accept up to 2 files).