ID: __________________ Name:________________________
Tutorial Day:_________ Time: ________Room:_________
Tutor's Name:_______________________Date: __________

ICT60001 Operating System Management - Week 9

Goals:

- Review lecture 9
- Practice looking up Linux C code
- Understand client-server architecture, file I/O, buffers and socket programming (TCP) in a Linux OS

Resources:

- Lecture notes (week 9)
- Computer running Kali Linux
- Sample C programs

Lab:
Download and unzip the Lab file (Week 9 Linux programming.zip) from the week 9 lecture part of Canvas onto your H: drive, laptop or portable drive.
Lab Task:
Today you are going to write a file server and file transfer client programs in Linux C.

For each program that you write, you must add block comments which describe the functions you create. Add the following sections:

- Description
- @param – describe each formal parameter
- @return – describe what the function returns
- @pre-conditions – describe any operations or declarations which are assumed to have been done before the function is called. (optional)
- @post-conditions – describe any changes to the program state or variables after the function has returned. (optional)

Examine the iptalk.c program.
Identify the server code.

1. What function calls are required for a server to accept a TCP connection from a client?

2. What function calls are required for a client to initiate a TCP connection to a server?

ID: ___________________ Name:__________________________
Tutorial Day:__________ Time: _________Room:__________
Tutor's Name:___________________________Date: ___________

The functioning of a file server follows these steps:

1. The server creates a TCP socket bound to a port, in a listening state ready to accept a connection.
2. The client creates a TCP socket and connects to the server
3. The server reads a list of the current directory and sends it to the client and goes into an idle state while waiting for a file request.
4. The client selects (or types) the name of a file on the server
5. The client requests the file from the server
6. The server checks the file size and sends it to the client
7. The client creates an empty file with the same name as the requested file
8. The server opens the file, and copies it byte by byte into the socket while counting the bytes sent
9. The client copies the bytes from the socket to the file, while counting the bytes received
10. Both server and client close their respective files once the counters reach the file size.
11. The server resumes an idle state, waiting for a request from the client
12. The client either requests another file or issues the exit command and shuts down
13. If the server receives an exit command it closes the connection and waits for another connection.

Identify the code in iptalk.c which will create a server socket and wait for a connection

ID: ________________ Name:______________________
Tutorial Day:_________ Time: ________Room:_________
Tutor's Name:______________________Date: __________

Identify the code in iptalk.c which will connect to the server.

Identify the code in dirlist.c which will return a list of files

Identify the code in iptalk.c that (suitable modified) will send a list of files to the client

Identify the code in iptalk.c that will send a filename (from the client section of the code) to the server

Identify the code in filesize.c that will return the size of a file

ID: __________________ Name:________________________
Tutorial Day:_________ Time: ________Room:_________
Tutor's Name:_________________________Date: __________

Identify the code in filecopy that will open a file and copy it into a buffer (char array)

Identify the code in iptalk.c that will copy a byte from a buffer into the socket

Identify in filecopy.c the read() and write() functions and where they return the number of bytes sent or received.

Identify the code in filecopy.c that creates a file and copies a buffer into it

ID: _________________ Name:______________________
Tutorial Day:_________ Time: ________Room:_________
Tutor's Name:________________________Date: __________

In assignment 4 you will combine these snippets of code into one program, including the required #includes, and variable declarations.

Add in the appropriate struct code (addr_in), port number, IP address (127.0.0.1), file name c-strings, counting variables, buffers (char arrays) and cleanup code (closing files, sockets) and make it work. Use read() the requested file in the server and write() to the file in the client. Use send() and recv() to send bytes through the socket. Use common char buffers for these commands.

You do not have to create or adapt any multithreaded code, call fork() or dup2(). Both the server and the client can be single-threaded.

//If you use Leafpad on the Kali GUI this will be easier than mastering the various vim cut and paste commands.

Save, compile, edit... until you can get the code to compile without error and run (after you chmod +x).

To test, run your server on one directory and your client from another, and copy a file which is in the server directory but not in the client directory.

If you copy a binary and the copy runs there has been no error in the copying.