Lab 2 : System Calls

August 6, 2018

Objective :

- Lab 2 is intended to provide the way to use the most common system calls in order to make input-output operations on files, as well as operations to handle files and directories in Linux

Recommended Systems/Software Requirements:

- Any flavour of Linux

References:

1. *Unix concepts and applications*, Fourth Edition, Sumitabha Das, TMH.

Tutorial:

- Tut 1 : Learning how to use the system calls for opening, reading and writing to files - *open*, *read*, *write* and *lseek* system calls
    1. Login to the system, open the *Terminal* and use the **man** command to read the manual pages of *open*, *read*, *write* and *lseek* system calls:
    2. Download the file **open_read_write_with_linux_sys_calls** provided in the helpful resources section of Lab2 and compile using gcc.
       This C program intends to read 100 characters from a file and to print these 100 characters on the terminal. It uses *open* system call to open a file. The *open* call returns a file descriptor *fd* which is then used in the successive *read* and *write* system calls to access the file.
- Tut 2: Writing lines of text using system calls
    1. Download the file **write_lines_of_text_sys_call** provided in the helpful resources section of Lab2 and compile using gcc.
- Tut 3: Simulating the *ls* command
    1. Download the file **simulating "ls" command** provided in the helpful resources section of Lab2 and compile using gcc.
       This C program performs the simulation of the *ls* command in Linux which lists all the folders and sub-folders of its present working directory.

Assignemnts:

1. Using a similar approach as covered in Tut 3 , implement in C the following UNIX commands using System calls : *cat* and *mv*
2. Determine the size of a file using the *lseek* command. Once you found out the size, calculate the number of blocks assigned for the file. Compare these results with the similar results obtained when using the function *stat*.
3. Write a C program that deletes a directory with all its subfolders. The name of the directory should be read from the command line.
4. Write a program that deletes every 5th byte from a file, but without using a temporary file or allocating a buffer in the memory. For adjusting the size of the file you may use the *truncate* function.