Operating System Lab

Lab - 06

Objectives:

1. Process Management and Scheduling in Linux. Process Scheduling in Linux

Process Management and Schedtools

- Write a program in C, that receives two integers as command line arguments.
- The main thread creates a child thread and passes the two numbers to it after packaging those two numbers in a struct. The main thread waits for the termination of the child thread.
- The child thread function computes the power (the first number raised to the power of the second number). For Example, If the first number is 2 and the second number is 10 then the answer should be 1024.
    - The thread function should use the pow() library function of the math library.
- The child thread should return the answer to the main thread. The main thread should display the answer and enters the while loop

Compile this program and make an executable with the name of your roll no. Then perform the following tasks:

Task1: Use `readelf(1)`, `od(1)`, and `size(1)` commands to get different attributes of your executable program file. Compile your program file with the `-g` option to `gcc` and rerun the commands to see the differences. Compile your program file with the `--static` option to `gcc` and rerun the commands to see the differences. Note down your observations.

Task2: Execute the program, suspend its execution and then run it in the background. Execute the program in the background, and then bring it to the foreground. Meanwhile, in another terminal keep checking various statistics of your process using `ps(1)` with `-u` and `-l` options. Keep a note of all this on paper.

Task3: Use the `top(1)` command and see confirm statistics that you have noted down using `ps(1)` in the above task. Try changing its nice value and see the behaviour of your process. Using `top(1)` sends different signals to your running process and see whether a core file is generated or not. If not, see what you need to change. Keep a note of all this on paper.

Task4: Learn how you can change the nice value of a running process from the shell using the `renice(1)` command. Also, see how you can run your program with a nice value of something other than the default from the shell using the `nice(1)` command. Keep a note of your observations on paper.

Task5: Use `schedtool(1)`, (install using `sudo apt-get install schedtool` command) to change the following scheduling parameters of your running process and note you're your observations:

- Change scheduling policy SCHED_BATCH, SCHED_IDLEPRIO, SCHED_NORMAL
- Change nice value
- Change static priority
- Change CPU affinity