CSE 231 Operating Systems Monsoon 2020

Sambuddho Chakravarty, Arani Bhattacharya

October 8, 2020

Assignment 2 (Total points: 75)

Due date: October 25, 2019; Time: 23:59 Hrs.

Differences Between Processes and Threads (points: 30

As you know by now when `fork()` results in complete duplication of all process memories, while `pthread_create()` results only in partial. Your exercise is to write a program where a global variable is initialized to a value of say 10. At some point, the program `fork()`s and creates a child process. The parent process increments the global variable linearly upto 100 (*i.e.* 11, 12, 13....,100). The child process does the opposite, *i.e.* decrements linearly upto -90. Both the parent and child processes should print the value of the variable.

Repeat the same using `pthread_create()`, where the parent threat increments the global variable linearly upto 100 (*i.e.* 11, 12, 13....,100). The child threat does the opposite, *i.e.* decrements linearly upto -90. Both the parent and child threads should print the value of the variable.

Is there any difference in the value of the global variable as printed by the parent and child process, versus when printed via the parent and child thread? Either ways, please explain in detail the reason behind this difference.

What To Submit

- We have provided the git repository template for code submission.You need to clone the template and use it as specified in README.md. https://github.com/mayank18049/OS-Assignment-2
- The source code of the two programs with their Makefile(s).
- Write-up describing the reasons behind the difference that you observed.

Grading Rubric

- The successful compilation of the two above programs – 10 points.
- Successful demonstration of the two individual cases – 10 points.
- Correct description of the difference of the two cases – 10 points.

Writing Your Own System Call (points: 45)

As described in the class, the OS provides operations to users via system calls. System calls are functions, natively available to users via C library functions, provide access to almost all OS level functionality – *viz.* `open()`, `write()`, `fork()` *etc.*

You have to create your own system call in C, called `sh_task_info()`, which takes argument as PID. It would need to search out the `task_struct()` corresponding to the PID and print out all the fields corresponding to it and also save it in a file. The file name also needs to be supplied as an argument to the system call.

You also would require to handle errors in user inputs, such as incorrect arguments, through appropriate `errno` and function return values (*e.g.* `0` signaling correct input, while `1` signaling incorrect input).

You are supposed to use Linux/kernel v5.9 distribution for the assignment, *i.e.* the system call should be written for the said version. You would need to first download the kernel version from the official repository of the Linux distribution that you are using (*e.g.* Ubuntu) and compile and boot that up. The system call should be written and tested for this version.

What To Submit

- We have provided the git repository template for code submission.You need to clone the template and use it as specified in README.md. https://github.com/mayank18049/OS-Assignment-2
- You need to submit the `diff`, of the originally downloaded kernel source tree and the one with your changes. This patched code MUST match the one you have in our kernel source (running on a VM). We would verify this at the time of the demos.
- Write-up describing the following:
  - Description of your code and how you implemented the function – the logical and implementation details.
  - The inputs the user should give.
  - Expected output (and how to interpret it).
  - Error values and how to interpret them.
- A sample C program to test out your implementation of the system call.

Grading Rubric

- Successful compilation your diff against the base kernel source – 10 points.
- Correct functioning of your system call, testable through the supplied C program – 20 points.
- Correct handling of input errors (atleast two different types of errors should be handled) – 10 points.
- Description of the systems, how to test the system through your supplied C program and what to expect *etc.* – 5 points.

Late Submission Policy

- Submitted on or before Oct 25, 2020 (23:59 hrs) – No points deducted.
- Submitted after Oct 25, 2020 but on or before Oct 27, 2020 (23:59 hrs) – 5 points deducted.
- Submitted after Oct 27, 2020 but on or before Oct 29, 2020 (23:59 hrs) – 15 points deducted.
- Submitted after Oct 29, 2020 – no points shall be awarded.