Operating Systems - Monsoon 2021

Sambuddho Chakravarty

October 12, 2021

Assignment 1 (Total points: 100)

Due: Oct 27, 2021. Time: 23:59 Hrs. (Hard Deadline)

1 Process creation and termination system calls (Total points: 65).

The first exercise deals using process creation system call, `fork()`. You need to write a program that spawns a child process, using the `fork()` system call. The child process reads a CSV file, presented with the assignment, that has (fake) student IDs and grades of various assignments. This child process computes the average score of each assignment across students of section 'A', and thereafter prints the details of these assignments (of section 'A', *i.e.*). The parent process does the same operation, on the same CSV file but for assignments given to students of section 'B'.

The parent process must wait for the child process to terminate, by using the system call `waitpid()`. The child process must call the `exit()` system callonce its execution ends. Now create a separate program which repeats exactly the same steps as above, but uses threads using `pthread_create` and `pthread_join`. For the program that uses threads, once the averages are computed, also compute the average score of each assignment across the sections by reusing the result obtained by the two threads.

You would require to refer to the `manpages` for various system calls mentioned.

Note: You are expected to use `read` and `read` system calls, to read and write to the file.

What To Submit

- Program source code with Makefile to compile and pause the compilation at each phase.
- Write-up giving a brief description of how the program works (less than 1 page), with details of each system call, the arguments passed and the expected outcomes and how you handled error/corner cases.

Grading Rubric

- Successful compilation of the program via the Makefile – 10 points.

- Correct output for the program, corresponding to each of the sections – 30 points.
- Correct use the system calls mentioned with proper arguments and error handling – 20 points.
- Descriptive write-up (no more than 1 page) – 5 points.

2 Combining C and Assembly Language Programs (Total points:35)

This second exercise is aimed to serve two objectives – writing assembly language programs and secondly to help combine a C program with and assembly language function. You need to do the following:

1. Write a program with three functions `A()`, `B()` and `C()`.
2. `A()` should call `B()` passing a 64-bit integer as an argument.
3. `B()` should interpret that as a 8-byte ASCII string and print individual characters on screen. You need to call the `write()` system call from assembly language using the `syscall` instruction, passing appropriate arguments.
4. Modify the stack in the function `B()` in such a way that when `B()` executes the `ret` instruction, it jumps to a third function `C()`. `C()` must also be written in C. This MUST happen without an explicit call to function `C()`. A mere `ret` from B, should pass the control to function `C()` instead of `A()`. Finally, the function `C()` needs to terminate the program by using the `exit()` system call.
5. You ofcourse also need a `main()` function from where the program starts. The C file with the `main()` function could also have the function `A()` (*i.e.*, `main()` could call `A()`).
6. The functions `A()`, `B()` and `C()` need to be in three different files. You may use `printf()` or `write()` functions to display which function you are in.

What To Submit

- Program source code with Makefile the compile the program and generate a single binary executable.
- README/Write-up describing the following:
  - Steps to compile and execute the program.
  - Description of the logic used for achieving the above (no more than one page).

Grading Rubric

- Successful compilation of the programs using the Makefile — 5 points.
- Correct output for the program — 20 points.
- Description of how the program works, *viz.* how `A()` calls `B()` and how upon return (`ret`) from `B()`, the control passes to the function `C()` — 10 points