NYU Tandon School of Engineering - Introduction to Operating Systems - Assignment 4

Fall 2024

Assignment 4 (10 points)

This assignment consists of 5 problems.

Problem 1 (2 points)

If you create a main() routine that calls fork() twice, i.e., if it includes the following code:

```
pid_t x=-11, y=-22;
x = fork();
if(x==0) y = fork();
```

Assuming all fork() calls succeed, draw a process tree similar to that of Fig. 3.8 (page 116) in your textbook, clearly indicating the values of x and y for each process in the tree (i.e., whether 0, -11, -22, or larger than 0). The process tree should only have one node for each process. The process tree should be a snapshot just after all forks completed but before any process exits. Each line/arrow in the process tree diagram shall represent a creation of a process, or alternatively a parent/child relationship.

[Insert process tree diagram here - A simple tree with 3 processes. The parent will have x ¿ 0, y = -22. The child will have x = 0, y ¿ 0. The grandchild will have x = 0, y = 0.]

Problem 2 (4 points)

Write a program that creates the process tree shown below:

[Insert process tree diagram here - A tree with a parent and three children. Each child should perform a different simple task, like printing a different message.]

Problem 3 (2 points)

Explain the difference between a zombie process and an orphan process. Provide examples of how each might arise in a typical Unix-like operating system.

Problem 4 (1 point)

Describe a scenario where using `exec()` would be preferable to using `fork()` and `wait()`.

Problem 5 (1 point)

What are the potential problems with creating too many processes? How does the operating system manage this situation?

What to hand in (using Brightspace)

Please submit the following files individually:

1. Source file(s) with appropriate comments. The naming should be similar to “`lab#_$_c`” (# is replaced with the assignment number and $ with the question number within the assignment, e.g., `lab4_b.c`, for lab 4, question b OR `lab5_1a` for lab 5, question 1a).
2. A single pdf file (for images + report/answers to short-answer questions), named “`lab#.pdf`” (# is replaced by the assignment number), containing:
   - Screenshot(s) of your terminal window showing the current directory, the command used to compile your program, the command used to run your program, and the output of your program.
3. Your Makefile, if any. This is applicable only to kernel modules.

RULES

- You shall use kernel version 4.x.x or above. You shall not use kernel version 3.x.x.
- You may consult with other students about general concepts or methods but copying code (or code fragments) or algorithms is NOT ALLOWED and is considered cheating (whether copied from other students, the internet, or any other source).
- If you are having trouble, please ask your teaching assistant for help.
- You must submit your assignment prior to the deadline.