Unix "Fork" Command

Aaron Bengochea - ab6503

NYU Tandon Bridge - HW19

Fork() Overview

Modern day computing power allows for the ability to run many programs seamlessly, although, it would not be possible without the evolution of the operating systems that help manage and optimize said computing power. In the pursuit of optimization, modern operating systems commonly handle processes which themselves create new processes. The fork command is the primary method of process creation on Unix and other multitasking operating systems.

The process calling the fork command, which is referred to as the parent process, will first create a copy of itself, called the child process. The fork operation creates a separate address space for the child process. The child process has an exact copy of memory segments of the parent process. Child processes can create new processes themselves which leads to a tree-like structure.

The fork system call takes no arguments and return integer values. If the process creation fails, -1 is returned in the parent. Otherwise, the PID of the child is returned in the parent, and 0 is returned in the child. After the fork operation, both processes will resume the exact instruction right after the fork system call, and they can determine their status by the return value of the call to act accordingly.

In regards to the process state, the parent process is currently in the running state when it makes a call to fork. When the fork call is made by the parent, first the parent's state is set to ready then a child process is created in the new state which eventually enters the ready state. After the creation of the child process, both the parent and child processes are in the ready state. When they are chosen to execute by the operating system, both processes will resume from the point in which the fork call was made.

In practice, the child process may perform the exec system call to overlay itself with the other program. In this case, the child only performs few actions after the fork operation and the complete copy of the parent's memory segments become inefficient. In modern Unix variants that follow the virtual memory model, copy-on-write is implemented to alleviate this effect.