Max Gochenour

CS470 Lab 2 Report

2/10/2025

Introduction

This report explains the implementation of system calls in the provided C program, which creates and manages multiple processes using fork(), execvp(), and wait() system calls. The program demonstrates process creation, execution of different shell commands by child processes, and parent-child process interaction.

Implementation Summary

The program begins by defining an array of shell commands to be executed. The parent process first prints its PID and then enters a loop to create ten child processes using the fork() system call. Each child process executes one command from the predefined list using execvp(). If execvp() fails, an error message is printed. The parent process waits for all child processes to complete before exiting, ensuring proper synchronization.

Results and Observations

Process Creation and Management

Each child process is created using the fork() system call within a loop. If fork() returns a negative value, the program terminates due to failure. The child processes replace their execution image with the specified command using execvp(), which allows them to execute different commands independently. The parent process does not execute commands but waits for all child processes to complete using wait(), ensuring they terminate properly.

Parent-Child Interaction

Once a child process is created, it immediately replaces its execution context with the command provided. The parent process monitors and waits for each child process to finish execution using the wait() system call, ensuring orderly completion. The interaction is one-way; the parent does not send or receive data from the child but ensures they terminate properly and logs their completion.

Conclusion

This program demonstrates system calls for process creation and management in an operating system. By using fork(), execvp(), and wait(), it efficiently executes multiple commands in parallel child processes while ensuring synchronization through the parent process. The implementation provides insight into fundamental process control mechanisms in Unix-based systems.