Design Document: Mini-Bash Shell Implementation

student: Eva Levy

1. Project Overview

The objective of this project was to develop a simplified command-line interpreter, "Mini-Bash," using C on a Linux environment. This shell supports basic internal commands (`cd`, `exit`) and executes external programs found in the `/bin` or `$HOME` directories.

2. Core Architecture

The shell operates on a standard **Read-Parse-Execute** loop:

- **Read**: The shell captures user input using `fgets()`.
- **Parse**: The input string is cleaned of the newline character and split into individual arguments (tokens) using `strtok()` with space and tab delimiters.
- **Execute**: The shell determines if the command is internal or external and processes it accordingly.

3. System Calls Analysis

To interact with the Linux kernel, the following system calls were implemented:

- **`fork()`**: Used to create a child process. This allows the shell to run external commands in a separate process while the main shell remains active.
- **`execv()`**: Used within the child process to replace its memory image with the executable file of the requested command (e.g., `/bin/ls`).
- **`waitpid()`**: The parent process uses this to suspend execution until the child process finishes, ensuring the prompt only returns after the command is done.
- **`chdir()`**: This system call is used for the internal `cd` command. It is essential because a child process cannot change the working directory of its parent; therefore, the shell must handle it internally.
- **`access()`**: Used to verify if a command exists and is executable in the searched directories before attempting to fork.

4. Error Handling

The shell provides feedback for common issues:

- **Unknown Command**: If a command is not found in the search paths, the shell prints `[command]: Unknown Command`.
- **System Errors**: The `perror()` function is utilized to display descriptive error messages provided by the kernel if a system call (like `chdir` or `fork`) fails.

5. Conclusion

This implementation demonstrates the fundamental mechanics of process management in Unix-like systems, specifically the relationship between parent and child processes and the necessity of internal commands for environment modification.

LINK GITHUB: https://github.com/eva1606/mini-bash.git