POSIX Shell - Project Report

Team name - Strangers

Instructor - Prof. Manish Shrivastava

Mentor - Agrima Singh

Team members -

- Aakash Singh (2021201087)
- Prasanth Rao (2021201005)
- Rajat Dave (2021202024)
- Sourabh Patidar (2021201089)

Introduction :

In this project, we worked towards developing a working POSIX compatible shell with basic features provided by the default shell using C/C++ programming language. Our aim was to understand implementation details behind the functionalities of the Linux default shell.

Problem Description :

We were given a problem to develop our own POSIX compatible shell. The problem was divided into two parts - providing the already existing shell features and implementing extended features like history, open commands etc.

Solution Approach :

1. Implementing Non-canonical Mode.
2. Parsing Input and calling appropriate functions.
3. Reading ENVIRONMENT variables from the "myrc.txt" file.
4. Execution of system commands using execvp and handling their I/O using pipes.
5. Implementation of own commands like echo, history, open etc.
6. Implementing logic to execute multiple commands joined by pipes and redirection operators.
7. Implementing background and foreground command execution.
8. Implementing alarms using signal handlers.
9. Implementing history and autocomplete functionality using Trie data structures.

Work Distribution :

- Aakash Singh -
    - Implementing Non-canonical mode.
    - Commands Execution using execvp and managing pipes using dup and pipe.
    - Input/output redirection in case input is separated by pipe or redirection operator.
- Prasanth Rao -
    - Implemented Alarms using signal handling.
    - Implemented Trie for tab autocompletion.
    - Input Parsing for commands.
- Rajat Dave -
    - File Handling for history and I/O redirection.
    - Implemented history command.
    - Implemented background and foreground command execution.

- Sourabh Patidar -
    - Implemented open and echo commands.
    - Working with ENVIRONMENT variables and "myrc.txt" file.
    - Implemented record start and record stop.

Key Challenges :

- Handling file descriptor between parent and child process.
- Working of dup, wait and waitpid commands.
- Keeping track of background processes in implementation of background and foreground command execution.
- Mapping file formats with application's path for open command.
- Handling creation of multiple alarms and keeping track of missed alarms.

Learnings :

- Understood implementation details behind the basic functionalities of Linux shell.
- Understood usage of C/C++ functions like fork, execvp, wait, dup and pipe etc.
- Understood ENVIRONMENT variables and working of .bashrc file.
- Working of alarm function and signals.
- Implementation of Trie Data Structure.

Conclusion :

Successfully implemented POSIX compatible shell with support for basic and extended functionalities.

Working Commands :

- Basic commands - ls, echo, cd, mkdir, touch, cat, grep, pwd, head, tail, chmod, clear, cp
- Extended commands - history, open,alarms, record start, record stop, &, fg
- Extended features - command autocompletion, history searchable via Trie.