

CS4375: Theory of Operating Systems - Assignment 2A Report

Building Your Own Shell

By Adalberto Vazquez Villalobos

Instructor: Dr. Deepak Tosh

Spring 2025

Due Date: 02/15/2025, Midnight

1. Introduction

The objective of this assignment was to develop a simple shell called minershell, which reads user commands, executes them using system calls such as fork, exec, and wait, and handles basic command-line interactions.

The shell operates in an interactive while loop, displaying a prompt (minersh\$) and waiting for user input. It supports standard Linux commands, handles errors gracefully, and includes built-in functionality for changing directories using the cd command.

2. Implementation Details

2.1 Command Processing

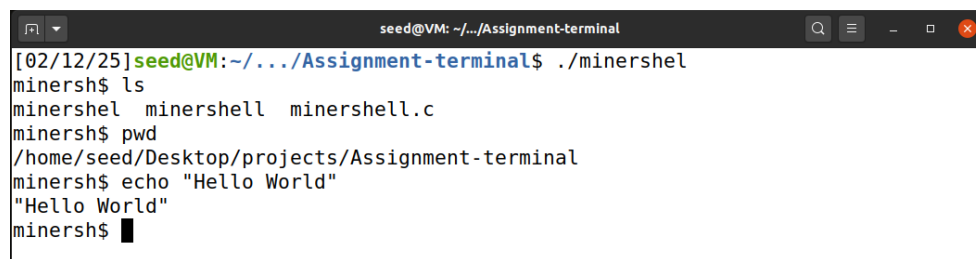
The shell continuously prompts the user for input. The entered command is tokenized and processed accordingly:

- If the command is exit, the shell terminates.
- If the command is cd, the chdir system call is used to change directories.
- For all other commands, a child process is created using fork(), and execvp() is used to execute the command.

2.2 Key Functionalities Implemented

1. Basic Command Execution:

- Commands such as ls, pwd, echo, and cat execute correctly by invoking their respective executables in Linux.
- Example usage:

A screenshot of a terminal window titled 'seed@VM: ~/.../Assignment-terminal'. The terminal shows the execution of the 'minershell' program. The user enters './minershell' at the prompt, which then displays a 'minersh\$' prompt. Subsequent commands and their outputs are: 'ls' (no output), 'minershel minershell minershell.c' (no output), 'pwd' (output: '/home/seed/Desktop/projects/Assignment-terminal'), 'echo "Hello World"' (output: '"Hello World"'), and 'minersh\$' (no output, with a cursor).

```
seed@VM: ~/.../Assignment-terminal
[02/12/25]seed@VM:~/.../Assignment-terminal$ ./minershell
minersh$ ls
minershel minershell minershell.c
minersh$ pwd
/home/seed/Desktop/projects/Assignment-terminal
minersh$ echo "Hello World"
"Hello World"
minersh$
```

2. Graceful Exit:

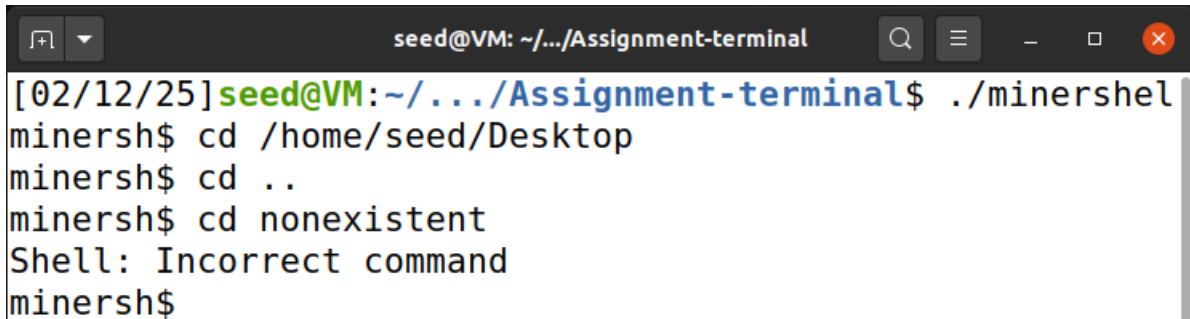
- The shell exits cleanly when the exit command is entered.
- Example usage:



```
seed@VM: ~/.../Assignment-terminal
[02/12/25]seed@VM:~/.../Assignment-terminal$ ./minershell
minersh$ ls
minersh$ pwd
minersh$ echo "Hello World"
minersh$ exit
Exiting shell...
[02/12/25]seed@VM:~/.../Assignment-terminal$
```

3. Handling the cd Command:

- The chdir() system call is used to change the working directory.
- Incorrect usage results in an error message.
- Example usage:



```
seed@VM: ~/.../Assignment-terminal
[02/12/25]seed@VM:~/.../Assignment-terminal$ ./minershell
minersh$ cd /home/seed/Desktop
minersh$ cd ..
minersh$ cd nonexistent
Shell: Incorrect command
minersh$
```

4. Error Handling:

- Invalid commands display appropriate error messages.
- Fork failures are handled to prevent shell crashes.
- Example usage:

```
seed@VM: ~/.../Assignment-terminal
[02/12/25]seed@VM:~/.../Assignment-terminal$ ./minershell
minersh$ invalidcommand
Error: The command could not be executed.
minersh$
```

5. Memory Management:

- Memory allocated for tokenized input is freed after each command to prevent memory leaks.

3. Test Scenarios and Screenshots

To validate the functionality of the shell, the following tests were conducted:

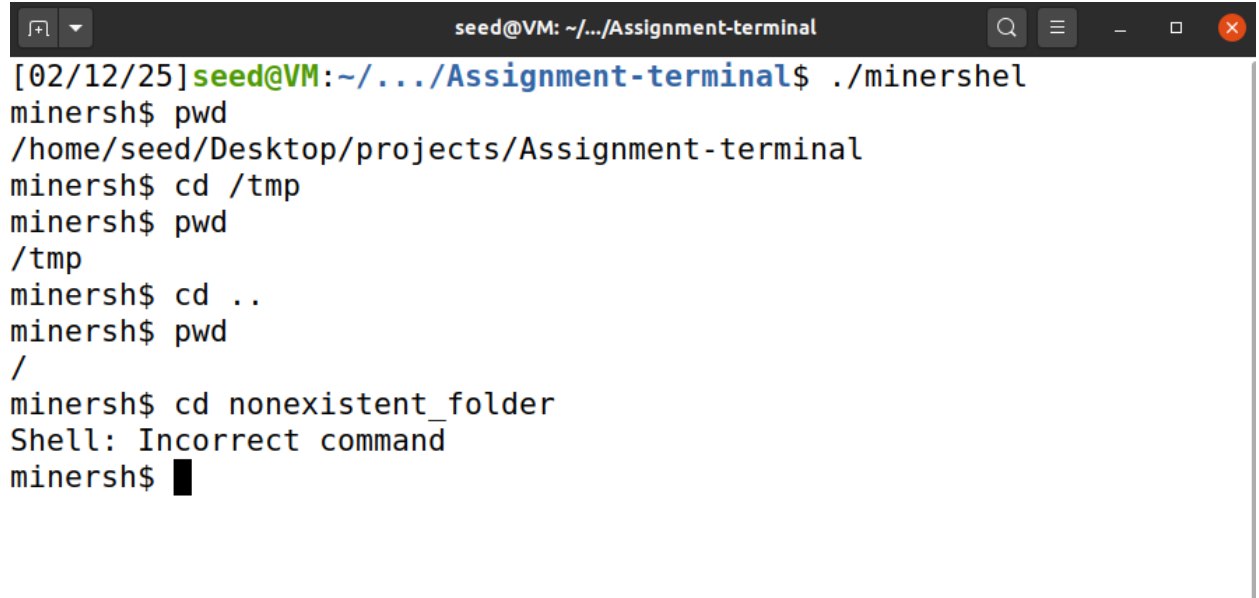
3.1 Basic Command Execution

- **Test:** Running ls in the shell
- **Expected Output:** List of files in the current directory

```
seed@VM: ~/.../Assignment-terminal
[02/12/25]seed@VM:~/.../Assignment-terminal$ ./minershell
minersh$ ls
minershell minershell minershell.c
minersh$ pwd
/home/seed/Desktop/projects/Assignment-terminal
minersh$ echo "Hello miners"
"Hello miners"
minersh$ cat /etc/os-release
NAME="Ubuntu"
VERSION="20.04.1 LTS (Focal Fossa)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 20.04.1 LTS"
VERSION_ID="20.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=focal
UBUNTU_CODENAME=focal
minersh$ █
```

3.2 Change Directory (cd)

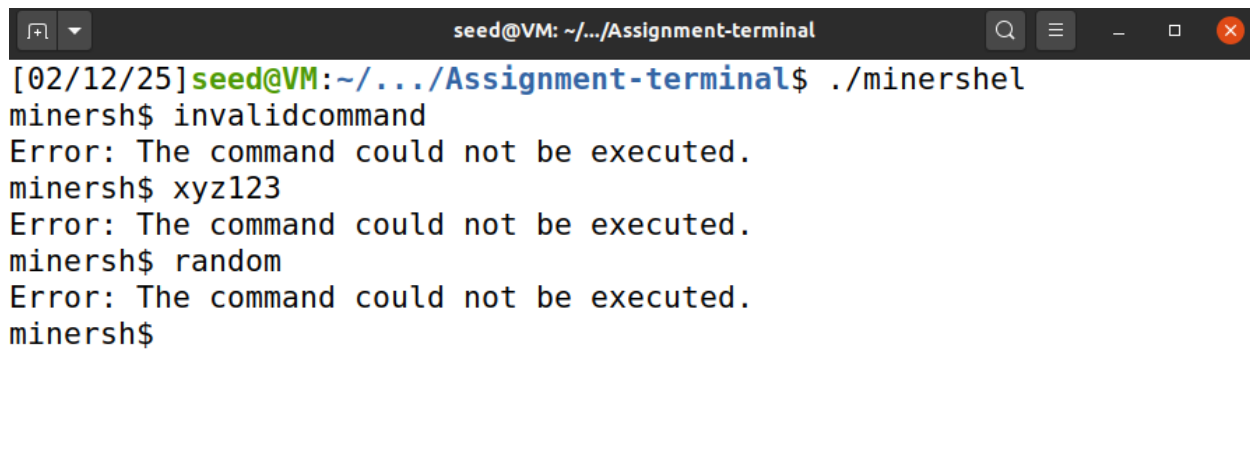
- **Test:** Running cd to navigate directories
- **Expected Output:** No output, but pwd should reflect the change

A terminal window titled 'seed@VM: ~/.../Assignment-terminal' showing a sequence of commands and their outputs. The user runs './minershell' to start a shell. Inside the shell, they use 'pwd' to show the current directory, then 'cd /tmp' to change to the tmp directory, 'pwd' to confirm, 'cd ..' to go back to the root, 'pwd' to confirm, and finally 'cd nonexistent_folder' which results in an 'Incorrect command' error.

```
[02/12/25]seed@VM:~/.../Assignment-terminal$ ./minershell
minersh$ pwd
/home/seed/Desktop/projects/Assignment-terminal
minersh$ cd /tmp
minersh$ pwd
/tmp
minersh$ cd ..
minersh$ pwd
/
minersh$ cd nonexistent_folder
Shell: Incorrect command
minersh$
```

3.3 Error Handling for Invalid Commands

- **Test:** Running an invalid command
- **Expected Output:** Error: The command could not be executed.

A terminal window titled 'seed@VM: ~/.../Assignment-terminal' showing three attempts to run invalid commands in the minershell. Each attempt results in an 'Error: The command could not be executed.' message.

```
[02/12/25]seed@VM:~/.../Assignment-terminal$ ./minershell
minersh$ invalidcommand
Error: The command could not be executed.
minersh$ xyz123
Error: The command could not be executed.
minersh$ random
Error: The command could not be executed.
minersh$
```

3.4 Handling exit Command

- **Test:** Running exit
- **Expected Output:** Shell terminates

```
seed@VM: ~/.../Assignment-terminal
[02/12/25]seed@VM:~/.../Assignment-terminal$ ./minershell
minersh$ exit
Exiting shell...
[02/12/25]seed@VM:~/.../Assignment-terminal$
```

3.5 Ensuring No Zombie Processes

- **Test:** Running ps before and after executing commands
- **Expected Output:** No leftover child processes

```
seed@VM: ~/.../Assignment-terminal
[02/12/25]seed@VM:~/.../Assignment-terminal$ ./minershell
minersh$ ps
  PID TTY          TIME CMD
 2487 pts/0    00:00:00 bash
 3907 pts/0    00:00:00 minersh
 3908 pts/0    00:00:00 ps
minersh$ ls
minersh minershell minershell.c
minersh$ ps
  PID TTY          TIME CMD
 2487 pts/0    00:00:00 bash
 3907 pts/0    00:00:00 minersh
 3913 pts/0    00:00:00 ps
minersh$
```

4. Challenges Faced

- Handling the `cd` command without forking a new process required adjusting logic to modify the parent process's working directory.
- Ensuring that memory was properly freed to avoid leaks.
- Implementing error handling for unknown commands without crashing the shell.

5. Conclusion

- The minershell successfully implements a simple shell capable of executing basic Linux commands. The shell correctly handles `cd`, prevents zombie processes, manages memory efficiently, and gracefully handles errors. Further enhancements could include support for piping (`|`), redirections (`>`, `<`), and background processes (`&`).

6. References

- *An A-Z index of the Linux Command Line: Bash + Utilities*. An A-Z Index of the Linux command line - SS64.com. (n.d.). <https://ss64.com/bash/>
- Köhntopp, K. (2021, January 4). *Fork, exec, wait and exit*. Percona. <https://percona.community/blog/2021/01/04/fork-exec-wait-and-exit/>
- (N.d.). <https://pubs.opengroup.org/onlinepubs/009695299/basedefs/sys/wait.h.html>