Intentionally left blank for writing solutions to the provided lab exercises.

4 Experiment 4: Directory Manipulation Using System Calls

4.1 Objective

In this lab, we'll explore how shell commands like mkdir (used to create directories), rmdir (used to remove directories) and ls used to list directory contents) actually work behind the scenes. These commands rely on special functions called system calls such as mkdir, opendir, and readdir. By learning about these functions, you'll be able to understand how these commands function and even write your own code using them.

4.2 Reading Material[1][3][2]

4.2.1 Directory Management Related System Calls

System Call	Description	Programming Syntax
`opendir`	Opens a directory stream corresponding to the given directory name.	`DIR *opendir(const char *dirname);`
`readdir`	Reads the next directory entry from the directory stream.	`struct dirent *readdir(DIR *dir_stream);`
`closedir`	Closes the directory stream.	`int closedir(DIR *dir_stream);`
`chdir`	Changes the current working directory.	`int chdir(const char *path);`
`mkdir`	Creates a new directory with the specified name and permission mode.	`int mkdir(const char *pathname, mode_t mode);`
`rmdir`	Removes a directory.	`int rmdir(const char *pathname);`
`getcwd`	Gets the pathname of the current working directory.	`char *getcwd(char *buf, size_t size);`

Table 3: System calls for directory management

4.2.2 The dirent structure

The dirent structure in C is used to store information about directory entries when working with directory-related system calls. It's commonly associated with functions like readdir and is defined in the <dirent.h> header file in Linux systems.
The structure `dirent` typically contains the following members:

- `ino_t d_ino`: This member represents the inode number of the directory entry.

- `off_t d_off`: It stores the offset of the next `readdir` call within the directory stream.
- `unsigned short int d_reclen`: It denotes the length of this record.
- `unsigned char d_type`: This member identifies the type of the file. For example, `DT_DIR` for directories, `DT_REG` for regular files, and others based on the file type.
- `char d_name[]`: This member holds the name of the directory entry. It is a character array representing the name of the file or directory.

4.3 Sample Programs

1. A C program that prints the contents of a directory using system calls like opendir, readdir, and closedir

```
#include <stdio.h>
#include <dirent.h>

int main() {
    DIR *dir;
    struct dirent *entry;

    dir = opendir(".");

    if (dir) {
        printf("Contents of the directory:\n");
        while ((entry = readdir(dir)) != NULL) {
            printf("%s\n", entry->d_name);
        }
        closedir(dir);
    }

    return 0;
}
```

2. Write a C program to create a new directory named "NewDirectory" within the file system

```
#include <stdio.h>
#include <sys/stat.h>
#include <sys/types.h>

int main() {
    const char *dirname = "NewDirectory";

    // Creating a new directory named "NewDirectory"
    mkdir(dirname, 0777);
```

```
    return 0;
}
```

Video Reference:

https://youtube.com/playlist?list=PLWjmN065fOfGdAZrlP6316HVHh8jlve

References

[1] Greg Gagne Abraham Silberschatz, Peter B. Galvin. *Operating System Concepts.* Wiley, 10 edition, 2018.

[2] Sumitabha Das. *Unix Concepts And Applications.* Wiley, 4 edition, 2006.

[3] Richard Stones Neil Matthew. *Beginning Linux Programming.* Wiley, 4 edition, 2007.

4.4 Lab Exercises

Exercise 1: Create a C program that prompts the user to enter a directory name and uses the `mkdir` system call to create the directory.

Exercise 2: Write a program that opens the current directory using `opendir` and reads its contents using `readdir`, then displays the list of directory entries.

Exercise 3: Create a C program to delete a directory specified by the user using the `rmdir` system call.

Exercise 4: Write a program that uses the `getcwd` system call to retrieve the current working directory and displays it to the user.

4.5 Sample Viva Questions

Question 1: How is the mkdir system call used in C programming?

Question 2: Discuss the significance of the rmdir system call.

Question 3: How is the opendir system call used in C programming?

Question 4: Explain the chdir system call and its significance.

Question 5: What are the main fields or members of the dirent structure?