Experiment Number: 4

Problem Statement: **System Calls**

NAME: **Manoj Dhanraj Mule**  ROLLNO: 71

CLASS: IT-B BATCH: B3

DATE OF PERFORMANCE: 26/08/2024

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Design menu driven application demonstrating use of different system calls.**

1. **Process related system call: fork, exit, wait,**
2. **File related system call: open, read, write, close, link, unlink, stat**
3. **Communication system call: pipe, fifo**
4. **Information related system call**

**Code:**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/stat.h>

#include <sys/types.h>

#include <sys/wait.h> #include <string.h> void processRelated(); void fileRelated(); void communicationRelated(); void informationRelated();

int main() { int choice; while (1) {

printf("\nMenu:\n");

printf("1. Process-related system calls\n"); printf("2. File-related system calls\n"); printf("3. Communication-related system calls\n"); printf("4. Information-related system calls\n"); printf("5. Exit\n"); printf("Enter your choice: "); scanf("%d", &choice); switch (choice) { case 1:

processRelated();

break; case 2: fileRelated(); break; case 3:

communicationRelated();

break;

case 4:

informationRelated();

break; case 5: exit(0); default:

printf("Invalid choice!\n");

}

}

return 0;

}

void processRelated() {

pid\_t pid = fork();

if (pid < 0) { perror("fork failed");

exit(1); } else if (pid == 0) {

printf("Child process: PID = %d\n", getpid()); exit(0); } else { wait(NULL);

printf("Parent process: Child terminated, PID = %d\n", getpid());

}

}

void fileRelated() { int fd; char buffer[100]; ssize\_t bytesRead;

fd = open("example.txt", O\_CREAT | O\_RDWR, 0644); if (fd < 0) { perror("open failed");

return;

}

write(fd, "Hello, World!\n", 14); lseek(fd, 0, SEEK\_SET);

bytesRead = read(fd, buffer, sizeof(buffer) - 1);

if (bytesRead < 0) { perror("read failed");

} else { buffer[bytesRead] = '\0';

printf("Read from file: %s\n", buffer);

}

struct stat fileStat;

if (stat("example.txt", &fileStat) == 0) { printf("File size: %ld bytes\n", fileStat.st\_size);

} else {

perror("stat failed");

}

close(fd);

if (link("example.txt", "example\_link.txt") == 0) { printf("Hard link created.\n");

} else {

perror("link failed");

}

if (unlink("example\_link.txt") == 0) {

printf("Hard link removed.\n");

} else {

perror("unlink failed");

}

}

void communicationRelated() { int pipefd[2];

pid\_t pid;

char write\_msg[] = "Hello from parent"; char read\_msg[20];

if (pipe(pipefd) == -1) { perror("pipe failed");

return;

}

pid = fork(); if (pid < 0) { perror("fork failed");

exit(1); } else if (pid == 0) { close(pipefd[1]); // Close writing end read(pipefd[0], read\_msg, sizeof(read\_msg)); printf("Child read from pipe: %s\n", read\_msg);

close(pipefd[0]); exit(0);

} else { close(pipefd[0]);

write(pipefd[1], write\_msg, strlen(write\_msg) + 1);

close(pipefd[1]);

wait(NULL);

}

if (mkfifo("myfifo", 0666) == 0) {

printf("FIFO created.\n");

} else {

perror("mkfifo failed");

}

if (unlink("myfifo") == 0) {

printf("FIFO removed.\n");

} else {

perror("unlink failed");

}

}

void informationRelated() { pid\_t pid = getpid(); uid\_t uid = getuid();

printf("Current Process ID: %d\n", pid);

printf("Current User ID: %d\n", uid);

}

**Output:**

ubuntu@ubuntu:~$ nano lab4.c ubuntu@ubuntu:~$ gcc lab4.c -o menu\_syscalls

ubuntu@ubuntu:~$ ./menu\_syscalls

Menu:

1. Process-related system calls
2. File-related system calls
3. Communication-related system calls
4. Information-related system calls
5. Exit

Enter your choice: 1

Child process: PID = 59298

Parent process: Child terminated, PID = 59296

Menu:

1. Process-related system calls
2. File-related system calls
3. Communication-related system calls
4. Information-related system calls
5. Exit

Enter your choice: 1

Child process: PID = 59302

Parent process: Child terminated, PID = 59296

Menu:

1. Process-related system calls
2. File-related system calls
3. Communication-related system calls
4. Information-related system calls
5. Exit Enter your choice: 2 Read from file: Hello, World!

File size: 14 bytes

Hard link created.

Hard link removed.

Menu:

1. Process-related system calls
2. File-related system calls
3. Communication-related system calls
4. Information-related system calls
5. Exit

Enter your choice: 3

Child read from pipe: Hello from parent FIFO created.

FIFO removed.

Menu:

1. Process-related system calls
2. File-related system calls
3. Communication-related system calls
4. Information-related system calls
5. Exit

Enter your choice: 4

Current Process ID: 59296

Current User ID: 1000

Menu:

1. Process-related system calls
2. File-related system calls
3. Communication-related system calls
4. Information-related system calls
5. Exit

Enter your choice: 5