**Lab Assignment 4**

**Aim:** To study and learn about various system calls.

**To Perform:** Comprehensive study of different categories of

Linux system calls, categorized as.

**1. Process Management System Calls.**

**fork()**

• The fork() system call creates a new child process by duplicating the

calling process.

Example

#include <stdio.h>

#include <unistd.h> int

main() {

pid\_t pid = fork(); if

(pid == 0) {

printf("This is the child process.\n");

} else {

printf("This is the parent process.\n");

}

return 0;

}

**exec()**

• The exec() family of system calls replaces the current process image with

a new process.

Example:

#include <stdio.h>

#include <unistd.h>

int main() {

char \*args[] = {"/bin/ls", "-l", NULL};

execvp(args[0], args);

return 0;

}

**wait()**

The wait() system call makes a parent process wait for its child process

terminate.

Example:

#include <stdio.h>

#include <sys/types.h>

#include <sys/wait.h

#include <unistd.h>

int main() {

pid\_t pid = fork();

if (pid > 0) {

wait(NULL);

printf("Child process terminated.\n");

} else {

printf("Child process executing.\n");

}

to

return 0;

}

**exit()**

• The exit() system call terminates a process.

Example:

#include <stdlib.h>

int main() {

exit(0);

}

**2. File Management System Calls**

These system calls manage file operations such as opening, reading, writing,

and closing files.

**open()**

• Opens a file and returns a file descriptor.

#include <stdio.h>

int main() {

char buffer[100];

Example:

#include <fcntl.h>

#include <stdio.h>

int main() {

int fd = open("example.txt", O\_CREAT | O\_WRONLY, 0644);

if (fd < 0) {

printf("Error opening file.\n");

}

return 0;

}

**read()**

Reads data from a file.

Example:

#include <fcntl.h>

#include <unistd.h>

int fd = open("example.txt", O\_RDONLY);

read(fd, buffer, sizeof(buffer));

printf("%s\n", buffer);

close(fd);

return 0;

}

**write()**

• Writes data to a file.

Example:

#include <fcntl.h>

#include <unistd.h>

int main() {

int fd = open("example.txt", O\_WRONLY);

write(fd, "Hello, World!", 13);

close(fd);

return 0;

}

**close()**

• Closes an open file.

Example:

#include <fcntl.h>

#include <unistd.h>

int main() {

int fd = open("example.txt", O\_RDONLY);

close(fd);

return 0;

}

**3. Device Management System Calls.**

These system calls manage device input and output operations.

**read()**

• Used to interact with devices like /dev/random (input) and /dev/null

(output).

Example:

int fd = open("/dev/mydevice", O\_RDONLY);

char buffer[100];

ssize\_t bytesRead = read(fd, buffer, sizeof(buffer));

close(fd);

**write()**

• The write() function sends data from user space to a device.

Example:

int fd = open("/dev/mydevice", O\_WRONLY);

char data[] = "Hello, Device!";

write(fd, data, strlen(data));

close(fd);

**ioctl()**

• Performs device-specific operations.

•

Example:

#include <stdio.h>

#include <fcntl.h>

#include <sys/ioctl.h>

int main() {

int fd = open("/dev/tty", O\_RDONLY);

int status;

ioctl(fd, FIONREAD, &status);

printf("Bytes available: %d\n", status);

close(fd);

return 0;

}

**select()**

• Monitors multiple file descriptors.

Example:

#include <sys/select.h>

#include <stdio.h>

#include <unistd.h>

int main() {

fd\_set set;

FD\_ZERO(&set);

FD\_SET(0, &set);

select(1, &set, NULL, NULL, NULL);

printf("Input detected.\n");

return 0;

}

**4. Network Management System Calls.**

These system calls manage network communication.

**socket()**

• Creates a socket.

Example:

#include <sys/socket.h>

int main() {

int sockfd = socket(AF\_INET, SOCK\_STREAM, 0);

return 0;

}

**connect()**

• Connects to a remote host.

Example:

#include <sys/socket.h>

#include <netinet/in.h>

int main() {

int sockfd = socket(AF\_INET, SOCK\_STREAM, 0);

struct sockaddr\_in server;

server.sin\_family = AF\_INET;

server.sin\_port = htons(8080);

connect(sockfd, (struct sockaddr \*)&server, sizeof(server));

return 0;

}

**send()**

• Sends data over a network.

Example:

#include <sys/socket.h>

int main() {

int sockfd = socket(AF\_INET, SOCK\_STREAM, 0);

char message[] = "Hello";

send(sockfd, message, sizeof(message), 0);

return 0;

}

**recv()**

• Receives data from a network.

Example:

#include <sys/socket.h>

int main() {

int sockfd = socket(AF\_INET, SOCK\_STREAM, 0);

char buffer[1024];

recv(sockfd, buffer, sizeof(buffer), 0);

return 0;

}

**5. System Information Management System Calls.**

These system calls retrieve system-related information.

**getpid()**

• Gets the process ID.

Example:

#include <stdio.h>

#include <unistd.h>

int main() {

printf("PID: %d\n", getpid());

return 0;

}

**getuid()**

• Gets the user ID.

Example:

#include <stdio.h>

#include <unistd.h>

int main() {

printf("UID: %d\n", getuid());

return 0;

}

**gethostname()**

• Gets the hostname of the system.

Example:

#include <stdio.h>

#include <unistd.h>

int main() {

char hostname[1024];

gethostname(hostname, sizeof(hostname));

printf("Hostname: %s\n", hostname);

return 0;

}

**sysinfo()**

• Gets system information.

Example:

#include <stdio.h>

#include <sys/sysinfo.h>

int main() {

struct sysinfo info;

sysinfo(&info);

printf("Uptime: %ld seconds\n", info.uptime);

return 0;

}