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**

These system calls handle the **creation, execution, and termination** of processes.

| **System Call** | **Description** |
| --- | --- |
| fork() | Creates a new process by duplicating the calling process. The new process is called the child process. |
| exec() | Replaces the current process image with a new program. Multiple variants: execl(), execp(), execv(), etc. |
| wait() | Waits for a child process to terminate, blocking the parent process. |
| exit() | Terminates the calling process and returns a status code to the parent. |

**Example:**

c

CopyEdit

pid\_t pid = fork();

if (pid == 0) {

execl("/bin/ls", "ls", NULL); // child process

exit(0);

} else {

wait(NULL); // parent waits

}

**🔹 2. File Management System Calls**

These system calls manage files – **opening, reading, writing, and closing** them.

| **System Call** | **Description** |
| --- | --- |
| open() | Opens a file and returns a file descriptor. |
| read() | Reads data from a file descriptor into a buffer. |
| write() | Writes data from a buffer to a file descriptor. |
| close() | Closes an opened file descriptor. |

**Example:**

c

CopyEdit

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

char buffer[100];

read(fd, buffer, 100);

write(STDOUT\_FILENO, buffer, strlen(buffer));

close(fd);

**🔹 3. Device Management System Calls**

Used for **low-level interaction with hardware devices** (like keyboards, disks).

| **System Call** | **Description** |
| --- | --- |
| read() | Reads data from a device. |
| write() | Sends data to a device. |
| ioctl() | Device-specific input/output operations (control a device). |
| select() | Monitors multiple file descriptors for I/O readiness. |

**Example:**

c

CopyEdit

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

ioctl(fd, TIOCGETA, &termios\_struct); // Get terminal attributes

close(fd);

**🔹 4. Network Management System Calls**

Enable **communication between processes over a network**.

| **System Call** | **Description** |
| --- | --- |
| socket() | Creates a socket for communication. |
| connect() | Connects the socket to a remote address. |
| send() | Sends data over a connected socket. |
| recv() | Receives data from a connected socket. |

**Example:**

c

CopyEdit

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

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

send(sock, "Hello", 5, 0);

recv(sock, buffer, 1024, 0);

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

These calls fetch **information about the system or current process**.

| **System Call** | **Description** |
| --- | --- |
| getpid() | Returns the process ID of the calling process. |
| getuid() | Returns the user ID of the calling process. |
| gethostname() | Gets the name of the host system. |
| sysinfo() | Returns information about the system (memory, uptime, etc.). |

**Example:**

c

CopyEdit

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

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

char hostname[1024];

gethostname(hostname, sizeof(hostname));

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

**✅ Summary Table**

| **Category** | **System Calls** |
| --- | --- |
| **Process Management** | fork(), exec(), wait(), exit() |
| **File Management** | open(), read(), write(), close() |
| **Device Management** | read(), write(), ioctl(), select() |
| **Network Management** | socket(), connect(), send(), recv() |
| **System Info** | getpid(), getuid(), gethostname(), sysinfo() |