**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

* ***Process Management System calls***

fork(), exec(), wait(), exit().

* ***File Management System calls***

open(), read(), write(), close().

* ***Device Management System calls***

read(), write(), ioctl(), select().

* 4. ***Network Management System calls***

socket(), connect(), send(), recv().

* 5. ***System Information Management System calls***

getpid(), getuid(), gethostname(), sysinfo().

**To submit:**

**Exhaustive study of the above mentioned system call categories with their examples :**

**1. Process Management System Calls**

Process management system calls allow the creation, execution, waiting, and termination of processes.

**fork()**

* Creates a new child process.

Example:

#include <stdio.h>

#include <unistd.h>

int main() {

pid\_t pid = fork();

if (pid == 0) printf("Child Process\n");

else printf("Parent Process\n");

return 0;

}

**exec()**

* Replaces the current process with a new program.

Example:

#include <stdio.h>

#include <unistd.h>

int main() {

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

execvp("ls", args);

return 0;

}

**wait()**

* Makes a process wait for a child process to terminate.

Example:

#include <sys/types.h>

#include <sys/wait.h>

#include <stdio.h>

#include <unistd.h>

int main() {

pid\_t pid = fork();

if (pid > 0) wait(NULL);

else printf("Child Process\n");

return 0;

}

**exit()**

* Terminates a process.

Example:

#include <stdlib.h>

int main() {

exit(0);

}

**2. File Management System Calls**

File management system calls are used to interact with files.

**open()**

* Opens a file.

Example:

#include <fcntl.h>

#include <stdio.h>

int main() {

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

if (fd < 0) printf("Error opening file\n");

return 0;

}

**read()**

* Reads data from a file.

Example:

#include <fcntl.h>

#include <unistd.h>

#include <stdio.h>

int main() {

char buffer[100];

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

read(fd, buffer, 100);

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

close(fd);

return 0;

}

**write()**

* Writes data to a file.

Example:

#include <fcntl.h>

#include <unistd.h>

#include <stdio.h>

int main() {

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

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

close(fd);

return 0;

}

**close()**

* Closes a file descriptor.

Example:

#include <fcntl.h>

#include <unistd.h>

int main() {

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

close(fd);

return 0;

}

**3. Device Management System Calls**

Used for interaction with devices.

**read()**

* Reads data from a device (similar to file read).

Example:

#include <fcntl.h>

#include <unistd.h>

#include <stdio.h>

int main() {

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

char data[3];

read(fd, data, sizeof(data));

printf("Read from device: %x %x %x\n", data[0], data[1], data[2]);

close(fd);

return 0;

}

**write()**

* Writes data to a device.

Example:

#include <fcntl.h>

#include <unistd.h>

#include <stdio.h>

int main() {

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

write(fd, "Hello Device!\n", 14);

close(fd);

return 0;

}

**ioctl()**

* Controls device settings.

Example:

#include <sys/ioctl.h>

#include <stdio.h>

#include <fcntl.h>

int main() {

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

int size;

ioctl(fd, FIONREAD, &size);

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

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;

struct timeval timeout;

FD\_ZERO(&set);

FD\_SET(0, &set);

timeout.tv\_sec = 5;

timeout.tv\_usec = 0;

int ret = select(1, &set, NULL, NULL, &timeout);

if (ret) printf("Data available\n");

else printf("Timeout\n");

return 0;

}

**4. Network Management System Calls**

Handles network communication.

**socket()**

* Creates a network socket.

Example:

#include <sys/socket.h>

#include <stdio.h>

int main() {

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

if (sockfd < 0) printf("Socket creation failed\n");

return 0;

}

**connect()**

* Connects a socket to a remote host.

Example:

#include <sys/socket.h>

#include <netinet/in.h>

#include <stdio.h>

#include <string.h>

int main() {

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

struct sockaddr\_in serv\_addr;

memset(&serv\_addr, 0, sizeof(serv\_addr));

serv\_addr.sin\_family = AF\_INET;

serv\_addr.sin\_port = htons(8080);

serv\_addr.sin\_addr.s\_addr = INADDR\_ANY;

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

return 0;

}

**send()**

* Sends data through a socket.

Example:

#include <sys/socket.h>

#include <stdio.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 socket.

Example:

#include <sys/socket.h>

#include <stdio.h>

int main() {

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

char buffer[1024];

recv(sockfd, buffer, 1024, 0);

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

return 0;

}

**5. System Information Management System Calls**

Fetches system-related information.

**getpid()**

* Returns process ID.

Example:

#include <stdio.h>

#include <unistd.h>

int main() {

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

return 0;

}

**getuid()**

* Returns user ID.

Example:

#include <stdio.h>

#include <unistd.h>

int main() {

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

return 0;

}

**gethostname()**

* Retrieves hostname.

Example:

#include <stdio.h>

#include <unistd.h>

int main() {

char name[1024];

gethostname(name, sizeof(name));

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

return 0;

}

**sysinfo()**

* Fetches 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;

}