**OS Assignment – 4**

**Operating System Lab**

**Aim**

To study and understand various Linux system calls categorized into:

1. Process Management
2. File Management
3. Device Management
4. Network Management
5. System Information Management

**1. Process Management System Calls**

**1.1 fork()**

Creates a new process by duplicating the calling process. The new process is called the 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;

}

**1.2 exec()**

Replaces the current process image with a new process image.

**Example:**

#include <unistd.h>

int main() {

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

execvp(args[0], args);

return 0;

}

**1.3 wait()**

Suspends the calling process until one of its children terminates.

**Example:**

#include <stdio.h>

#include <sys/wait.h>

#include <unistd.h>

int main() {

pid\_t pid = fork();

if (pid == 0)

printf("Child running\n");

else {

wait(NULL);

printf("Child finished\n");

}

return 0;

}

**1.4 exit()**

Terminates the calling process and returns a status code.

**Example:**

#include <stdlib.h>

int main() {

exit(0);

}

**2. File Management System Calls**

**2.1 open()**

Opens a file and returns a file descriptor.

**Example:**

#include <fcntl.h>

int main() {

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

return 0;

}

**2.2 read()**

Reads data from a file descriptor.

**Example:**

#include <unistd.h>

#include <fcntl.h>

#include <stdio.h>

int main() {

char buf[100];

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

read(fd, buf, sizeof(buf));

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

return 0;

}

**2.3 write()**

Writes data to a file descriptor.

**Example:**

#include <unistd.h>

#include <fcntl.h>

int main() {

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

write(fd, "Hello", 5);

return 0;

}

**2.4 close()**

Closes a file descriptor.

**Example:**

#include <unistd.h>

int main() {

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

close(fd);

return 0;

}

**3. Device Management System Calls**

**3.1 read(), write()**

Used the same as file management, but for device files like /dev/tty.

**3.2 ioctl()**

Used to control device parameters.

**Example:**

#include <sys/ioctl.h>

#include <fcntl.h>

int main() {

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

int result;

ioctl(fd, TIOCGWINSZ, &result);

return 0;

}

**3.3 select()**

Monitors multiple file descriptors to see if any are ready for I/O.

**Example:**

#include <sys/select.h>

#include <stdio.h>

int main() {

fd\_set readfds;

FD\_ZERO(&readfds);

FD\_SET(0, &readfds);

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

printf("Input detected\n");

return 0;

}

**4. Network Management System Calls**

**4.1 socket()**

Creates a socket for communication.

**Example:**

#include <sys/socket.h>

int main() {

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

return 0;

}

**4.2 connect()**

Establishes a connection on a socket.

**Example:**

#include <arpa/inet.h>

#include <string.h>

int main() {

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

struct sockaddr\_in serv;

serv.sin\_family = AF\_INET;

serv.sin\_port = htons(80);

inet\_pton(AF\_INET, "93.184.216.34", &serv.sin\_addr);

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

return 0;

}

**4.3 send()**

Sends data over a connected socket.

**Example:**

#include <sys/socket.h>

int main() {

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

// After connect()

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

return 0;

}

**4.4 recv()**

Receives data from a connected socket.

**Example:**

#include <sys/socket.h>

int main() {

char buffer[100];

// After connect()

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

return 0;

}

**5. System Information Management System Calls**

**5.1 getpid()**

Returns the process ID.

**Example:**

#include <stdio.h>

#include <unistd.h>

int main() {

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

return 0;

}

**5.2 getuid()**

Returns the real user ID of the calling process.

**Example:**

#include <stdio.h>

#include <unistd.h>

int main() {

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

return 0;

}

**5.3 gethostname()**

Retrieves the standard host name.

**Example:**

#include <unistd.h>

#include <stdio.h>

int main() {

char name[1024];

gethostname(name, sizeof(name));

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

return 0;

}

**5.4 sysinfo()**

Provides system statistics.

**Example:**

#include <sys/sysinfo.h>

#include <stdio.h>

int main() {

struct sysinfo info;

sysinfo(&info);

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

return 0;

}