Operating System Lab

Lab - 04

Objectives:

1. Understanding the concept of Input Output redirection. Lecture-08
2. Understanding the concept of Inter Process Communication(IPC) Lecture-09
3. Understanding the concept of Signals and their working. Lecture-10

execlp() & wait()

Task 01:

```
#include<stdio.h>
#include<unistd.h>
#include<fcntl.h>
int main(void){
    write(1, "I am learning OS", 17);
    write(1, "I know what is syscall", 23);
    write(1, "I am going to run the echo command", 35);
    execl("/usr/bin/echo", "echo", "i am here", NULL);
    write(1, "Should i be printed on screen or not", 37);
    return 0;
}
```

Write down output of the above program.

Task 02: Write down a C program which will print the contents of present working directories using `execlp`.

Task 03:

```
#include<stdio.h>
#include<unistd.h>
#include<fcntl.h>
#include<wait.h>
int main(void) {
    printf("I am Parent\n");
    int cp = fork();
    if(!cp){
        printf("%d", cp);
        execl("/usr/bin/echo", "I am the child", NULL);
    }
    else{
        printf("%d", cp);
        wait(NULL);
        execlp("/usr/bin/echo", "I am Parent child is terminated", NULL);
    }
    return 0;
}
```

Predict the output of the above program. What is it doing?

I/O Redirection

Task 01: Perform the following tasks:

a) Write a single command to copy the contents of **`/etc/passwd`** into **`out.txt`** without using **`cp`** command. (Hint: I/O Redirection)

b) Find all the files named `*libc.so*` in your root directory (using `find` command) and redirect the output to `libc_locations.txt,` and errors to `/dev/null`.

Task 02: Perform the following tasks:

a) Write a single command to add **"`Hi, I am <Your Name>`"** into `myself.txt`. (Hint: Use `echo` command)

b) Append **"`My Roll No is : <Your Roll No.>`"** using IO redirection.

Task 03:

```
#include<stdio.h>
#include<unistd.h>
#include<fcntl.h>
int main(void) {
   int fd = open("/tmp/fake", O_RDONLY);
   perror("ARM: Can't open file");
   printf("Ever wanted to be a Hacker?\n");
   printf("If Yes, Work hard and learn how OS throws errors to other files.\n");
   return 0;
}
```

Save the above given source code as **hacking.c**. Compile and make executable of the **hacking.c** and perform I/O redirection operations as described below:

a) Redirect the output to a file named `work_hard.txt`.

b) Redirect the error to a file named `failed.txt.`

c) Redirect the stdout and stderr to a file called `screen_copy.txt` using **copy descriptor**.

Pipes/ Fifos

Task 01:

a) Write a single command to display all the lines containing **`kali`** in **`/etc/passwd`** counts the number of lines in the output.
b) Write a single command to count the occurrences of word **`root`** in **`/etc/passwd.`**
c) Draw PPFDT of each process that was created in the above questions.

Task 03: Write a single command which will copy the source code **hacking.c** (I/O Redirection: Task 03) on **stdout** and in the following files: **advice1.c, advice2.c, advice3.c.**
(Hint: use `tee` command)

Task 04: Create a fifo called `transporter.` Open two shells and display the contents of the **advice1.c (**created in the above task) on both shells.
(Hint: Use tee command to save data in `transporter` and on second shell use any command to read `transporter`)

Signals

Task 01: Ignore the signal no `2`, `3` and `15`.

Task 02: Using `trap` command, create a disposition for `SIGQUIT` that echoes "**DEAD**!".

Task 03: Ignore signal number `9`. Run `sleep` command for `50` seconds. Now go onto another terminal and try sending `SIGKILL` to the `sleep` process.