Lab 3

Processes

1. Looking at Processes

1.1 Process IDs

Each process in a Linux system is identified by its unique process ID, referred to as *pid*. Process IDs are 1- 32768 numbers that are assigned sequentially by Linux as new process are created.
Every process also has a parent process (except *init*). Thus, we can think the processes in the Linux are arranged in a tree, with the *init* process at its root. The parent process ID, or *ppid*, is simply the process ID of the process's parent.
Most of the process manipulation functions are declared in the header file <unistd.h>. A program can obtain the process ID of the process it's running with the *getpid()* system call, and it can obtain the process ID of its parent process with the *getppid()* system call.

Ex. 3.1: Printing the Process ID

print-pid.c

```
#include<stdio.h>
#include<unistd.h>
int main()
{
int pid, ppid;
pid = getpid()
ppid = getppid();
printf("The Process ID is %d \n", pid);
printf("The Parent Process ID is %d \n", ppid);
return 0;
}
```

1.2 Viewing Processes

ps – display the process running on that system.
pstree – display the all process in process hierarchy.
ps [options] – ps has different options.
e.g.: ps -e -o pid,user,start_time,command
      ps -o pid,ppid,user.

2 Creating a processes

2.1 Using fork

Linux provides one function, fork, that makes the child process that is an exact copy of its parent process. Linux also provides another set of functions, the exec family, that replaces the current process image with a new process image.

Using fork

When programs calls fork, the parent process continues executing the program from the point that fork was called. The child process, too, executes the same process from the same place. The fork return some value for its parent and the child process always has 0 pid.

Ex. 3.3: Using fork

```
    #include<stdio.h>
    int main()
    {
        printf(“ This is to demonstrate the fork\n”);
        fork();
        printf(“Hi!, Everybody\n”)
    }
```

Ex. 3.4: Using fork

```
    #include < stdio.h>
    #include <unistd.h>
    int main()
    {
    int pid;
    printf("The main program process ID is  %d \n", (int) getpid());
    pid = fork();
    if (pid != 0){
            printf("This is the parent process, with id % d \n", (int) getpid());
            printf("The child process ID is %d \n", pid);
            }
    else
            printf(" This is the child process, with id %d\n", (int) getpid());
    return 0;
    }
```

Assignment #L3

1. Execute each of the programs (given in Ex. 3.1 – 3.4) at lest three times, and analyze their outputs.

4. Write the program that creates the the four process using fork() system call.