

EXPERIMENT N0-3

Aim: Create a child process in Linux using the fork system call. From the child process obtain the process ID of both child and parent by using getpid and getppid system call. Explore wait, getpid, getuid, getgid, getegid, geteuid and waitpid before termination of process.

code:

```
#include<stdio.h>

#include<unistd.h>

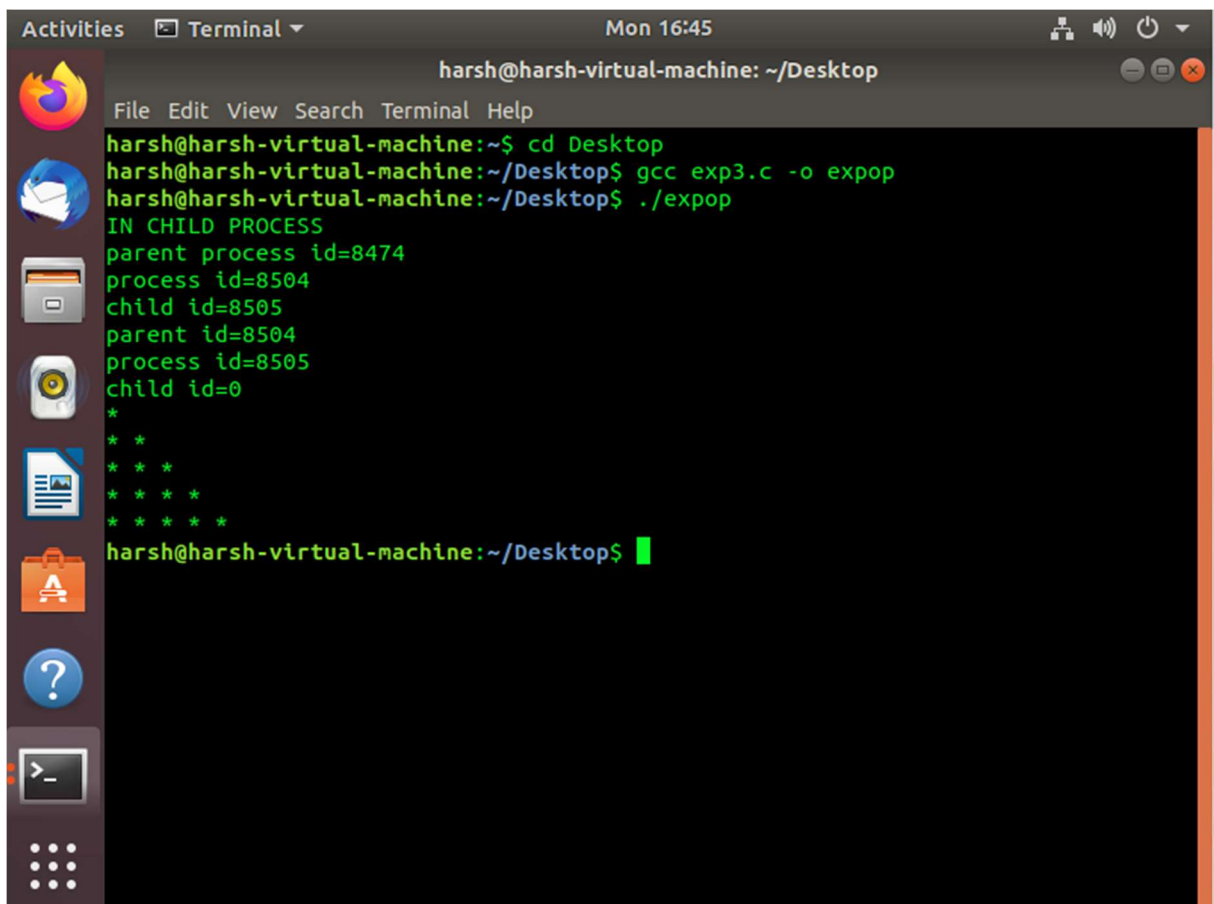
void main()
{
    int pid,ppid,cpid;
    pid=fork();
    if(pid<0){
        printf("ERROR\n");
    }
    else if (pid>0)
    {
        printf("parent process id=%d\n",getppid());
        printf("process id=%d\n",getpid());
        printf("child id=%d\n",pid);
    }
    else{
        printf("IN CHILD PROCESS\n");
        printf("parent id=%d\n",getppid());
        printf("process id=%d\n",getpid());
        printf("child id=%d\n",pid);
        for(int i=1;i<=5;i++)
```

```

    {
        for(int j=1;j<=i;j++)
        {
            printf("* ");
        }
        printf("\n");
    }
}

```

output:



The screenshot shows a terminal window titled "harsh@harsh-virtual-machine: ~/Desktop". The user has executed the following commands:

```

harsh@harsh-virtual-machine:~$ cd Desktop
harsh@harsh-virtual-machine:~/Desktop$ gcc exp3.c -o expop
harsh@harsh-virtual-machine:~/Desktop$ ./expop

```

The output of the program is as follows:

```

IN CHILD PROCESS
parent process id=8474
process id=8504
child id=8505
parent id=8504
process id=8505
child id=0
*
* *
* * *
* * * *
* * * * *

```

The terminal window also shows a sidebar with various application icons and a status bar at the bottom.