Roll No : 33231



Date
Page

# ASSIGNMENT 10

**Problem Statement** : Implement a new system call add this new system call in LINUX Kernel (any kernel source, any architecture and any Linux kernel distribution) and demonstrate use of same.

**Theory :**

## Adding a Simple System Call

Simple Example –
Say we wanted to add our own version of system call getpid(). Lets call our version mygetPid(). The implementation of my getPid() is

```
asmlinkage long sys = getPid (void)
{
    return current → tgid ;
}
```

assemlinkage must appear before every system call. It tells the compiler to only look on the stack for function argument

## Steps needed to follow to add this System Call –

1) Implement function call and put it in appropriate file.

2) Add an entry to end of system call table.

3) Define system call number in include / asm / unistd.h

4) Recompile your kernel and boot to it.

33231

Date
Page

## Accessing the system call from User - Space -

### Compile command -

gcc -I /The /Location /of /your /linux /include
test syscall . c

### Adding swipe () system call

Swipe () that steals the collective time slice for a specified set of processes and adds it to target process timeslice . The swipe () system call takes a target process id , an integer that provides type of set of processes and an integer that provides the process set .

The system call returns the total amount of the timeslice swiped on a negative number if an error occured . Therefore , the prototype of function is :

int swipe (Pid -t target , int which , int who)

### Using Swipe () :-

1) Create a process that monopolizes the CPU
2) Create a wrapper program that takes a simple command starting with the collective timeslices of all processes in the set

### Conclusion :

I have succesfully understood the concpt of new system call .