CS3523 OPERATING SYSTEMS – Assignment 2

REPORT :

Working of System calls:

- User processes can't access the services from the kernel of operating system directly. System call is the way in which user processes can contact with the operating systems.
- While invoking a system call the processor changes to kernel mode from the usermode.Also the mode bit is changed from 0 to 1 when switching to kernel mode from usermode.
- In the kernel mode it finishes the request raised by the user process and goes back again to user mode.
- The essientials needed by the user processes are fulfilled by the system calls using application programming interface(API).
- API provides an interface between process and operating system.
- There are different types of system calls for different type of services.
- Each system call has an unique id .
- Fork,exit,wait are the system calls used for process control.
- Write,read,open,close are the system calls used for file manuplation and many others.

What I have learnt from the assignment :

- I have learnt that while booting an operating system many system calls are invoked.
- I have also learned that some system calls such as sys_write,sys_read,sys_close,sys_kill etc have void arguments but they can take arguments through argint,argptr,argstr,argfd.
- Argint is used to retrive the system call argument as an integer and then it calls fetchint to read the value from user memory and write it to *ip. Fetchint casts it into a pointer and kernel verifies the pointer and if it is outside the user part of address space we encounter segmentation fault and it kills the process.
- Argptr is used to retrive the system call argument as a pointer. And then it checks whether it is a valid pointer or not.
- Argstr retrives the system call argument as a string.
- Argfd retrives the system call argument as a file descriptor. Argfd uses argint to fetch file descriptor number and checks whether it is a valid number or not.
- The system call implementations are done in the files such as sysroc.c etc these files decode the arguments using argint,argptr,argstr,argfd and then do the real implementation.

- I have learned that xv6 uses the same code and it avoids special casing the first process.In user mode we can access upto the calling the system call
  and then in the kernel mode xv6 reuses the same code . BY having void argument we can reuse the code So,System calls have void argument instead of having arguments like normal function calls.
- I have learned how to create a new system call.
- Intially there are 21 inbuilt system calls in xv6 OS. I have defined 22nd system call named date in syscall.h file.
- Also declared the 22nd element of the system calls array with sys_date in syscall.c file.Also updated the new system call in user.h file.
- After that I have declared the function that is done by date system call in sysproc.c file and finally called the date system call from user program named mydate.c .

--- END OF REPORT ---

Submitted by,
Kodavanti Rama Sravanth,
CS20BTECH11027.