Lab_2: [Exercising System Calls of UNIX/Linux Operating System]

Objective: System calls are function invocations made from user space into the kernel in order to request some service or resource from the operating system. This assignment is to make students understand and use the different *system call* functions of UNIX/Linux operating system. The system call functions are: {fork, exec, getpid, getppid, exit, close, stat, open, read, wait}

fork( ): used to create a new process called *child* process. The return value is 0 for a child process, negative if creation is unsuccessful, positive for the parent process.

getpid( ): returns process ID of the calling process

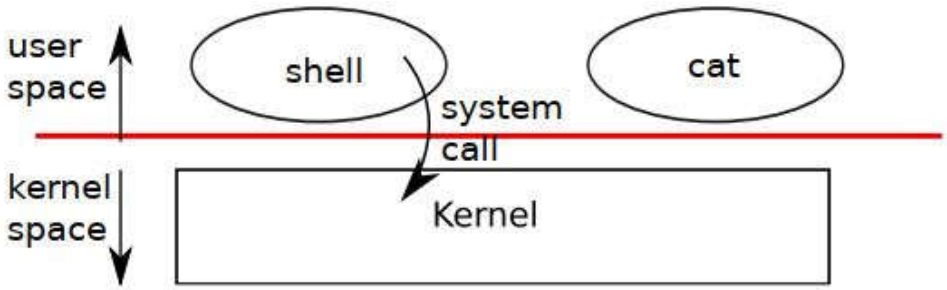
getppID( ): returns parent process ID of the calling process

exec: The exec family of functions (execl, execv, execle, execve, execlp, execvp) is used by the child process to load a program and execute

exit( ): terminate a process either normally or abnormally

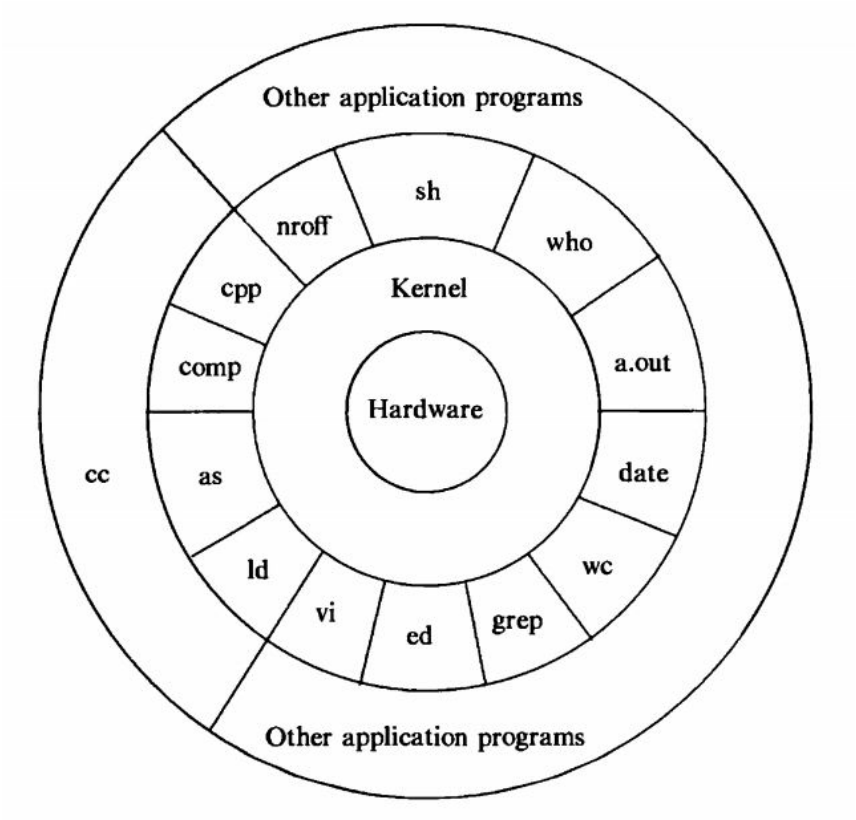
stat( ): return information about a file as a structure

open( ), read( ), close( ), opendir( ), readdir( ), closedir( )



[A kernel and two user process]

Reference: https://pubs.opengroup.org/onlinepubs/7908799/index.html



[Architecture of UNIX Systems]

This architecture is base for every operating system.

Assignment:

1. Write a c program to create a new child process using fork system call.
2. Write a c program to block a parent process until child completes using wait system call.
3. Write a c program to load an executable program in a child processes exec system call.
4. Write a c program or shell script to display file status using stat system call.
5. Write a c program or a shell script to open, display the contents of a directory, and close it using related system calls.

==========%&%++%&%==========