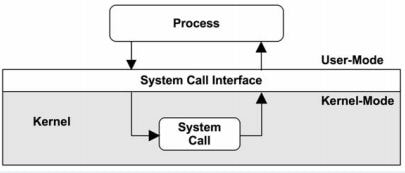
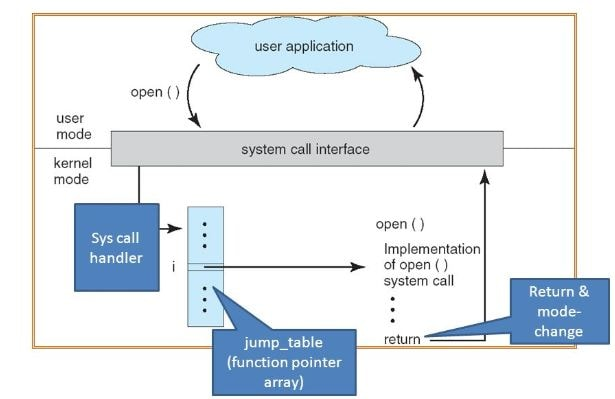
**Chapter 2**

**Services and components of operating system**

**System Calls Implementation:**



* System calls provide an interface between a user program and [**Operating System**](http://sciencerack.com/different-types-of-operating-system/).
* The system calls expose the services offered by the operating system to user programs.
* These are set of functions methods. Most system calls are written in assembly language and are machine dependent.
* Numerous higher level languages such as ‘C’ also permit to make system calls directly.



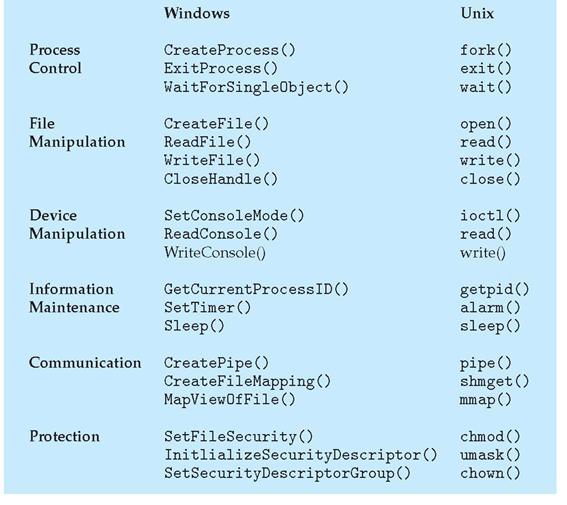
The system call interface layer includes entrance point in kernel code. All system resources are managed and controlled by the kernel. Any request from user or application that occupies access to any system resource must be tackled by kernel code. The user process must not be given untie access to kernel code for security cause. Many opening into kernel code called system calls are provided to the user so that the user processes can raise the execution of kernel code.

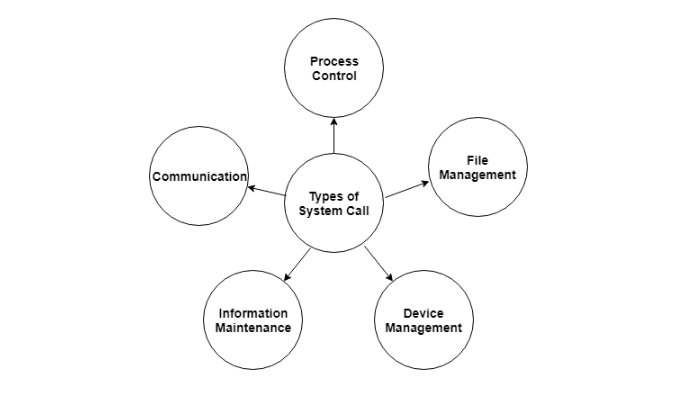
System calls permit processes and users to [**Operating System**](http://sciencerack.com/different-types-of-operating-system/) resources.

There are three universal methods that are used to pass information between a running program and the [**Operating System**](http://sciencerack.com/different-types-of-operating-system/).

* One technique is to store parameters in registers.
* Another is to store parameters in a table in memory and get ahead of the address of table.
* The third way is to push parameters on the stack and let operating system to pop the parameters off the stack.

**Types of System Calls**





**A) Process Control**

Processes need to be controlled as in a running process must be able to halt its execution either normally or abnormally. Also one process may need to run some other process to complete its own execution. So all these system calls come under this category.

1. end, abort
2. load, execute
3. create process, terminate process
4. get process attributes, set process attributes
5. wait for time, wait event, signal event
6. allocate and free memory

**B) File Management**

System calls which deal with operations related to files fall under this type.

1. create file, delete file
2. open, close
3. read, write, reposition
4. get file attributes, set file attributes

**C) Device Management**

A process may need several resources for its execution. So system calls used for asking permission from the kernel to use those resources are included in this type.

1. request device, release device
2. read, write, reposition
3. get device attributes, set device attributes
4. logically attach or detach devices

**D) Information Maintenance**

We need to keep all the information up to date so these system calls help us to do that.

1. get time or date, set time or date
2. get system data, set system data
3. get process, file, or device attributes
4. set process, file, or device attributes

**E) Communication**

Processes need to communicate with each other for many reasons like if they need certain resource which is held by any other process. These system calls assist in doing so.

1. create, delete communication connection
2. send, receive messages
3. transfer status information
4. attach or detach remote devices

**Operating System Component**

**Process Management**

The operating system manages many kinds of activities ranging from user programs to system programs like printer spooler, name servers, file server etc. Each of these activities is encapsulated in a process. A process includes the complete execution context (code, data, PC, registers, OS resources in use etc.).

It is important to note that a process is not a program. A process is only ONE instant of a program in execution. There are many processes can be running the same program. The five major activities of an operating system in regard to process management are

* Creation and deletion of user and system processes.
* Suspension and resumption of processes.
* A mechanism for process synchronization.
* A mechanism for process communication.
* A mechanism for deadlock handling.

**Main-Memory Management**

Primary-Memory or Main-Memory is a large array of words or bytes. Each word or byte has its own address. Main-memory provides storage that can be access directly by the CPU. That is to say for a program to be executed, it must in the main memory.

The major activities of an operating in regard to memory-management are:

* Keep track of which part of memory are currently being used and by whom.
* Decide which process are loaded into memory when memory space becomes available.
* Allocate and deallocate memory space as needed.

**File Management**

A file is a collected of related information defined by its creator. Computer can store files on the disk (secondary storage), which provide long term storage. Some examples of storage media are magnetic tape, magnetic disk and optical disk. Each of these media has its own properties like speed, capacity, data transfer rate and access methods.

A file systems normally organized into directories to ease their use. These directories may contain files and other directions.

The five main major activities of an operating system in regard to file management are

1. The creation and deletion of files.
2. The creation and deletion of directions.
3. The support of primitives for manipulating files and directions.
4. The mapping of files onto secondary storage.
5. The backup of files on stable storage media.

**I/O System Management**

I/O subsystem hides the peculiarities of specific hardware devices from the user. Only the device driver knows the peculiarities of the specific device to whom it is assigned.

**Secondary-Storage Management**

Generally speaking, systems have several levels of storage, including primary storage, secondary storage and cache storage. Instructions and data must be placed in primary storage or cache to be referenced by a running program. Because main memory is too small to accommodate all data and programs, and its data are lost when power is lost, the computer system must provide secondary storage to back up main memory. Secondary storage consists of tapes, disks, and other media designed to hold information that will eventually be accessed in primary storage (primary, secondary, cache) is ordinarily divided into bytes or words consisting of a fixed number of bytes. Each location in storage has an address; the set of all addresses available to a program is called an address space.

The three major activities of an operating system in regard to secondary storage management are:

1. Managing the free space available on the secondary-storage device.
2. Allocation of storage space when new files have to be written.
3. Scheduling the requests for memory access.

**Networking**

A distributed systems is a collection of processors that do not share memory, peripheral devices, or a clock. The processors communicate with one another through communication lines called network. The communication-network design must consider routing and connection strategies, and the problems of contention and security.

**Protection System**

If a computer systems has multiple users and allows the concurrent execution of multiple processes, then the various processes must be protected from one another's activities. Protection refers to mechanism for controlling the access of programs, processes, or users to the resources defined by a computer systems.

**Command Interpreter System**

A command interpreter is an interface of the operating system with the user. The user gives commands with are executed by operating system (usually by turning them into system calls). The main function of a command interpreter is to get and execute the next user specified command. Command-Interpreter is usually not part of the kernel, since multiple command interpreters (shell, in UNIX terminology) may be support by an operating system, and they do not really need to run in kernel mode. There are two main advantages to separating the command interpreter from the kernel.

1. If we want to change the way the command interpreter looks, i.e., I want to change the interface of command interpreter, I am able to do that if the command interpreter is separate from the kernel. I cannot change the code of the kernel so I cannot modify the interface.
2. If the command interpreter is a part of the kernel it is possible for a malicious process to gain access to certain part of the kernel that it showed not have to avoid this ugly scenario it is advantageous to have the command interpreter separate from kernel.

**Use of operating System Tools(Linux)**

# **Linux User Management**

* User management includes everything from creating a user to deleting a user on your system. User management can be done in three ways on a Linux system.
* Graphical tools are easy and suitable for new users, as it makes sure you'll not run into any trouble.
* Command line tools includes commands like useradd, userdel, passwd, etc. These are mostly used by the server administrators.
* Third and very rare tool is to edit the local configuration files directly using vi.

## useradd

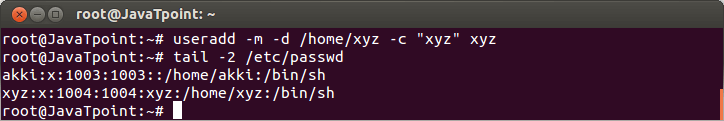
With useradd commands you can add a user.

**Syntax:**

1. useradd -m -d /home/**<userName>** -c "**<userName>**" **<userName>**

**Example:**

1. useradd -m -d /home/xyz -c "xyz" xyz



1. useradd -D

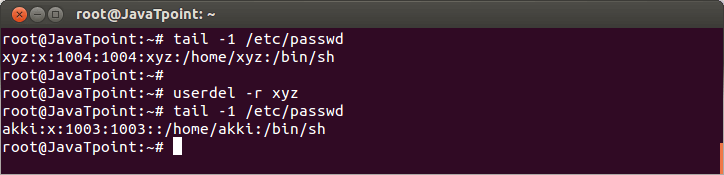


## Userdel

To delete a user account userdel command is used.

**Syntax:**

1. userdel -r **<userName>**



**Example:**

1. userdel -r xyz

Look at the above snapshot, first we have shown the xyz user account with 'tail' command. To delete it, command **"userdel -r xyz"** is passed.

## usermod

The command usermod is used to modify the properties of an existing user.

**Syntax:**

1. usermod -c **<**'newName'**>** **<oldName>**

**Example:**

1. usermod -c 'jhonny' john

## Linux Local User Management6

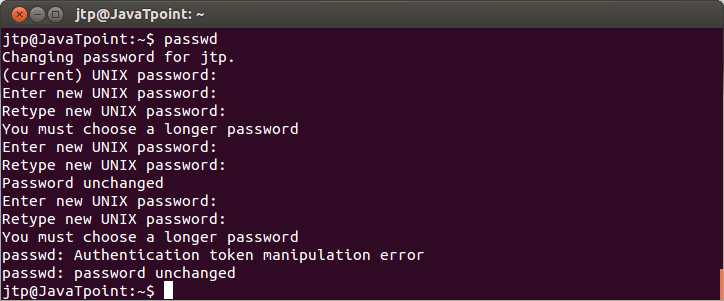
passwd command

**passwd**

A user can set the password with the command **passwd**. Old password has to be typed twice before entering the new one.

**Syntax:**

1. passwd



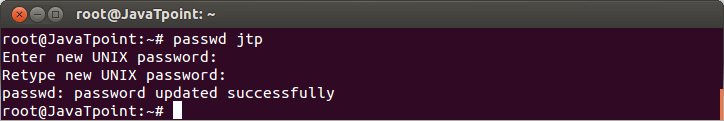
Look at the above snapshot, shell warns the user from creating a simple password. Ultimately, after two or three attempts if password is not changed then the command**passwd fails** and you have to pass the command again.

**Syntax:**

1. passwd **<userName>**

**Example:**

1. passwd jtp

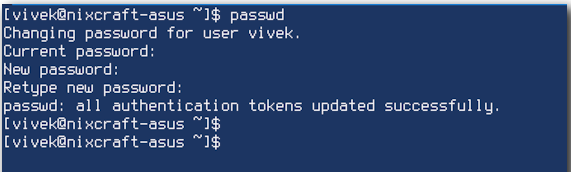


Password Command:

password command is used to set up password for user.

Syntax: # passwd user-name

E.g # passwd vivek



**Security Policy:**

Following are the best practices for securing linux

**1) Keep System updated**

Always keep system updated with latest releases patches, security fixes and kernel when it’s available. Applying security patches is an important part of maintaining Linux server and Linux provides all necessary tools to keep your system updated.

*# yum updates*

*# yum check-update*

**2) Turn Off IPv6**

If you’re not using a IPv6 protocol, then you should disable it

*# vi /etc/sysconfig/network*

***NETWORKING\_IPV6=no***

*IPV6INIT=no*

**3) Enable Iptables (Firewall)**

For best practices it is recommended to enable and configure server firewall to allow only specific ports that are required and block all the remaining ports.

*# service iptables start*

**4) Check Listening Network Ports**

With the help of ‘netstat‘ networking command you can view all open ports and associated programs. As I said above use ‘chkconfig‘ command to disable all unwanted network services from the system.

*# netstat -tulpn*

OR use the ss command as follows:

*$ ss -tulpn*

OR

*nmap -sT -O localhost*

*nmap -sT -O server.example.com*

**5) Enforcing Stronger Passwords**

A number of users use soft or weak passwords and their password might be hacked with a dictionary based or brute-force attacks. The ‘pam\_cracklib‘ module is available in PAM (Pluggable Authentication Modules) module stack which will force user to set strong passwords. Open the following file with an editor.

*# vi /etc/pam.d/system-auth*

And add line using credit parameters as (lcredit, ucredit, dcredit and/or ocredit respectively lower-case, upper-case, digit and other)

*/lib/security/$ISA/pam\_cracklib.so retry=3 minlen=8 lcredit=-1 ucredit=-2 dcredit=-2 ocredit=-1*

**Question Answers:**

Q1) What is process management? State for functions to be performed by OS for process management.

A Program does nothing unless its instructions are executed by a CPU. A program in execution is called a process.

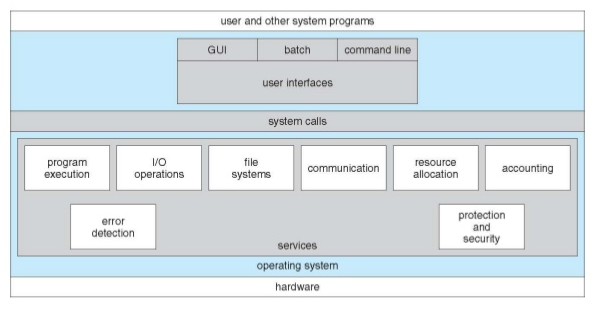
There may exist more than one process in the system which may require the same resource at the same time. Therefore, the operating system has to manage all the processes and the resources in a convenient and efficient way.

Some resources may need to be executed by one process at one time to maintain the consistency otherwise the system can become inconsistent and deadlock may occur.

The operating system is responsible for the following activities in connection with Process Management

1. Scheduling processes and threads on the CPUs.
2. Creating and deleting both user and system processes.
3. Suspending and resuming processes.
4. Providing mechanisms for process synchronization.
5. Providing mechanisms for process communication.

Q2) List any four OS services and describe it.



1. **Program Execution**

The purpose of [computer](http://ecomputernotes.com/fundamental/introduction-to-computer/what-is-computer) system is to allow the users to execute programs in an efficient manner. The operating system provides an environment where the user can conveniently run these programs. The user does not have to worry about the memory allocation or de-allocation or any other thing because these things are taken care of by the operating system.

To run a program, the program is required to be loaded into the RAM first and then to assign CPU time for its execution. Operating system performs this function for the convenience of the user. It also performs other important tasks like allocation and de-allocation of memory, CPU scheduling etc.

2. **I/O Operations**

Each program requires an input and after processing the input submitted by user it produces output. This involves the use of I/O devices. The operating system hides the user from all these details of underlying hardware for the I/O. So the operating system makes the users convenient to run programs by providing I/O functions. The I/O service cannot be provided by user-level programs and it must be provided by the operating system.

3. **File System Manipulation**

While working on the computer, generally a user is required to manipulate various types of files like as opening a file, saving a file and deleting a file from the storage disk. This is an important task that is also performed by the operating system.

Thus operating system makes it easier for the user programs to accomplish their task by providing the file system manipulation service. This service is performed by the 'Secondary Storage Management' a part of the operating system.

4. **Communication**

Operating system performs the communication among various types of processes in the form of shared memory. In multitasking environment, the processes need to communicate with each other and to exchange their [information](http://ecomputernotes.com/fundamental/information-technology/what-do-you-mean-by-data-and-information). These processes are created under a hierarchical structure where the main process is known as parent process and the sub processes are known as child processes.

5. **Error Detection**

Operating system also deals with hardware problems. To avoid hardware problems the operating system constantly monitors the system for detecting the errors and fixing these errors (if found). The main function of operating system is to detect the errors like bad sectors on hard disk, memory overflow and errors related to I/O devices. After detecting the errors, operating system takes an appropriate action for consistent computing.

This service of error detection and error correction cannot be handled by user programs because it involves monitoring the entire computing process. These tasks are too critical to be handed over to the user programs. A user program, if given these privileges; can interfere with the corresponding operation of the operating systems.

6. **Resource allocation**

In the multitasking environment, when multiple jobs are running at a time, it is the responsibility of an operating system to allocate the required resources (like as CPU, main memory, tape drive or secondary storage etc.) to each process for its better utilization. For this purpose various types of algorithms are implemented such as process scheduling, CPU scheduling, disk scheduling etc.

7. **Accounting**

Operating system keeps an account of all the resources accessed by each process or user. In multitasking, accounting enhances the system performance with the allocation of resources to each process ensuring the satisfaction to each process.

8. **Protection System**

If a computer system has multiple users and allows the concurrent execution of multiple processes, then the various processes must be protected from one another's activities.

**9. Accounting**

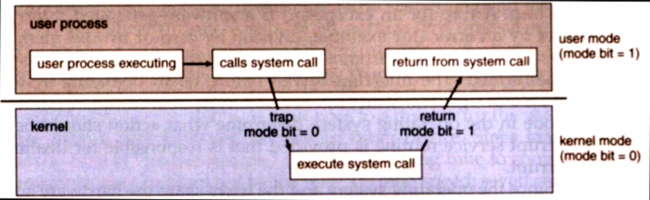
This service of the operating system keeps track of which users are using how much and what kinds of computer resources have been used for accounting or simply to accumulate usage statistics.

**10. Protection and Security**

Protection includes in ensuring all access to system resources in a controlled manner. For making a system secure, the user needs to authenticate him or her to the system before using (usually via login ID and password).

Q3) What is system call? List type of system calls with example.

**System Calls:** System calls are programming interface to the services provided by the operating system.



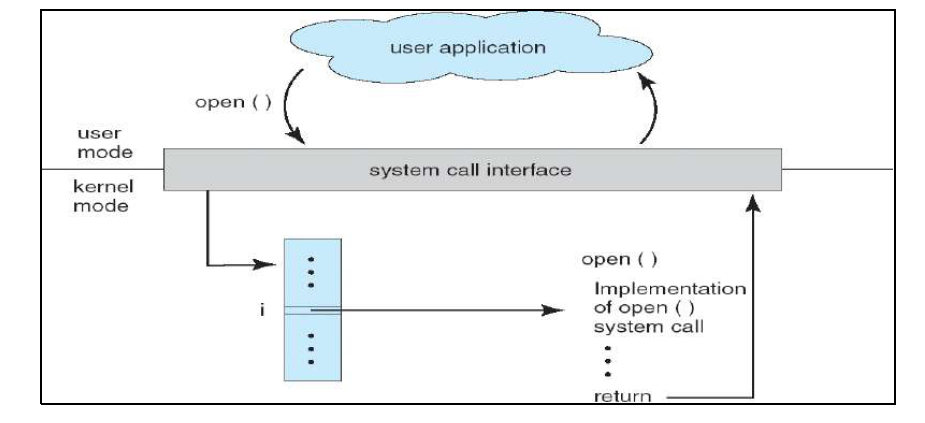
1. Each system call associated with a particular number.

2. System call interface maintains a table indexed according to these numbers.

3. The system call interface invokes intended system call in operating system kernel & returns status of the system call and any return values.

4. The caller needs to know nothing about how the system call is implemented. Just needs to obey API and understand what OS will do as a result call.

5. Most details of operating system interface hidden from programmers by API. It is managed by run-time support library.



System calls-

**System calls related to process control**: End, Abort Load, Execute Create process, Terminate process Ready process, Dispatch process Suspend, Resume Get Process attribute, set attribute Wait for time Wait event, signal event

**System calls Related to File management**: Create file, delete file Open file , Close file Create directory Read, write, Reposition Get file attribute , set file attribute Create a link Change the working directory

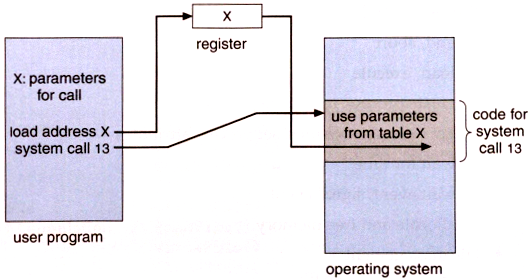
**System calls Related to Device Management**: Request a device, Release a device Read, Write, Reposition Get device attribute, set device attribute

**System calls Related to Information Maintenance**: Get Time or Date, Set Time or Date Get System data, Set system data Get process, file or device attributes Set process, file or Device attributes.

**Q4) Explain how parameter passing is done while implementing system calls.**

Three general methods exist for passing parameters to the OS:

1. Parameters can be passed in registers.
2. When there are more parameters than registers, parameters can be stored in a block and the block address can be passed as a parameter to a register.
3. Parameters can also be pushed on or popped off the stack by the operating system.



Q5) **What are the activities involved in secondary storage management?**

**The three major activities of an operating system in regard to secondary storage management**

**are:**

**a.** Managing the free space available on the secondary-storage device.

**b.** Allocation of storage space when new files have to be written.

**c.** Scheduling the requests for memory access.

|  |
| --- |
| Q6) **Describe activities of I/O system and secondary storage Management.** |

I/O System: Input / Output device management provides an environment for the better interaction between system and the I / O devices such as printers, scanners, tape drives etc. To interact with I/O devices in an effective manner, the operating system uses some special programs known as device driver. The device drivers take the data that operating system has defined as a file and then translate them into streams of bits or a series of laser. A device driver is a specific type of computer software that is developed to allow interaction with hardware devices. Typically this continues an interface for communicating with the I/O device, through the specific computer bus or communication subsystem that the hardware is connected with. The device driver is a specialized hardware dependent computer program that enables another program, typically an operating system to interact transparently with a hardware device, and usually provides the required interrupt handling necessary for the time dependent hardware interfacing.

Activities:

 Providing interfaces to other system components.

 Managing devices

 Transferring data

 Detecting I/O completion

Secondary storage Management: The computer system provides secondary storage to back up main memory. Secondary storage is required because main memory is too small to accommodate all data and programs, and the data that it holds is lost when power is lost. Most modern computer systems use disks as the principal on-line storage medium for programs and data. Most of the programs including compilers, assemblers, word processors, editors, and formatters are stored on a disk until loaded into memory. Secondary storage consists of tapes drives, disk drives, and other media.

Activities:

 Free space management

 Storage allocation

 Storage de-allocation

 Disk scheduling.

To read a piece of data that is stored at the end of the file, one has to read all of the data that comes before it-you cannot jump directly to the desired data. This is similar to the way cassette tape players work. If one wants to listen to the last song on a cassette tape, he has to either fast-forward over all of the songs that come before it or listen to them. There is no way to jump directly to a specific song.

**Q) Write use of following system calls.**

1) fork( )

2) exec( )

3) abort( )

4) end( )

1. fork ( ) This system call is used to create a new process.

2. exec( ) This system call is used to replace the process’s memory space with a new program. It loads a binary file into memory and starts its execution.

3. abort( ) This system call is used to halt process execution abnormally.

4. end ( ) This system call is used to halt process execution normally.

**Open ( ) system call**

For most file systems, a program initializes access to a file in a file system using the open system call. This allocates resources associated to the file (the file descriptor), and returns a handle that the process will use to refer to that file.