COURSE NAME : OPERATING SYSTEMS

(COURSE CODE :22516)

SUBJECT TEACHER : MS.T.R.SHINDE

UNIT NO 2 (10Marks)

SERVICES AND COMPONENETS OF OS

Course Outcome : Use operating system tools to perform various functions

- **UNIT OUTCOMES**

2a	Start, stop, and restart the given services in Linux
2b	Explain use of the given System call of specified OS
2c	Explain process the OS follows in managing the given resources
2d	Explain use of the given operating system tool

2.1 Different Services of Operating System :

Operating system services:

The operating system provides the programming environment in which a programmer works on a computer system. The user program requests various resources through the operating system. The operating system gives several services to utility programmers and users. Applications access these services through application programming interfaces or system calls. By invoking those interfaces, the application can request a service from the operating system, pass parameters, and acquire the operation outcomes.

Following are the services provided by an operating system -

- Program execution
- Control Input/output devices
- Program creation
- Error Detection and Response
- Accounting
- Security and Protection
- File Management
- Communication

Program execution

To execute a program, several tasks need to be performed. Both the instructions and data must be loaded into the main memory. In addition, input-output devices and files should be initialized, and other resources must be prepared. The Operating structures handle these kinds of tasks. The user now no longer should fear the reminiscence allocation or multitasking or anything.

Control Input/output devices

As there are numerous types of I/O devices within the computer system, and each I/O device calls for its own precise set of instructions for the operation. The Operating System hides that info with the aid of presenting a uniform interface. Thus, it is convenient for programmers to access such devices easily.

Program Creation

The Operating system offers the structures and tools, including editors and debuggers, to help the programmer create, modify, and debugging programs.

Error Detection and Response

An Error in a device may also cause malfunctioning of the entire device. These include hardware and software errors such as device failure, memory error, division by zero, attempts to access forbidden memory locations, etc. To avoid error, the operating system monitors the system for detecting errors and takes suitable action with at least impact on running applications.

While working with computers, errors may occur quite often. Errors may occur in the:

- **Input/ Output devices:** For example, connection failure in the network, lack of paper in the printer, etc.
- **User program:** For example: attempt to access illegal memory locations, divide by zero, use too much CPU time, etc.
- **Memory hardware:** For example, Memory error, the memory becomes full, etc.

To handle these errors and other types of possible errors, the operating system takes appropriate action and generates messages to ensure correct and consistent computing.

Accounting

An Operating device collects utilization records for numerous assets and tracks the overall performance parameters and responsive time to enhance overall performance. These personal records are beneficial for additional upgrades and tuning the device to enhance overall performance.

Security and Protection

Operating device affords safety to the statistics and packages of a person and protects any interference from unauthorized users. The safety feature counters threats, which are published via way of individuals out of doors the manage of the running device.

For Example:

When a user downloads something from the internet, that program may contain malicious code that may harm the already existing programs. The operating system ensures that proper checks are applied while downloading such programs.

If one computer system is shared amongst a couple of users, then the various processes must be protected from another intrusion. For this, the operating system provides various mechanisms that allow only those processes to use resources that have gained proper authorization from the operating system. The mechanism may include providing unique users ids and passwords to each user.

File management

Computers keep data and information on secondary storage devices like magnetic tape, magnetic disk, optical disk, etc. Each storage media has its capabilities like speed, capacity, data transfer rate, and data access methods.

For file management, the operating system must know the types of different files and the characteristics of different storage devices. It has to offer the proportion and safety mechanism of documents additionally.

Communication

The operating system manages the exchange of data and programs among different computers connected over a network. This communication is accomplished using message passing and shared memory.

2.2 System Calls :

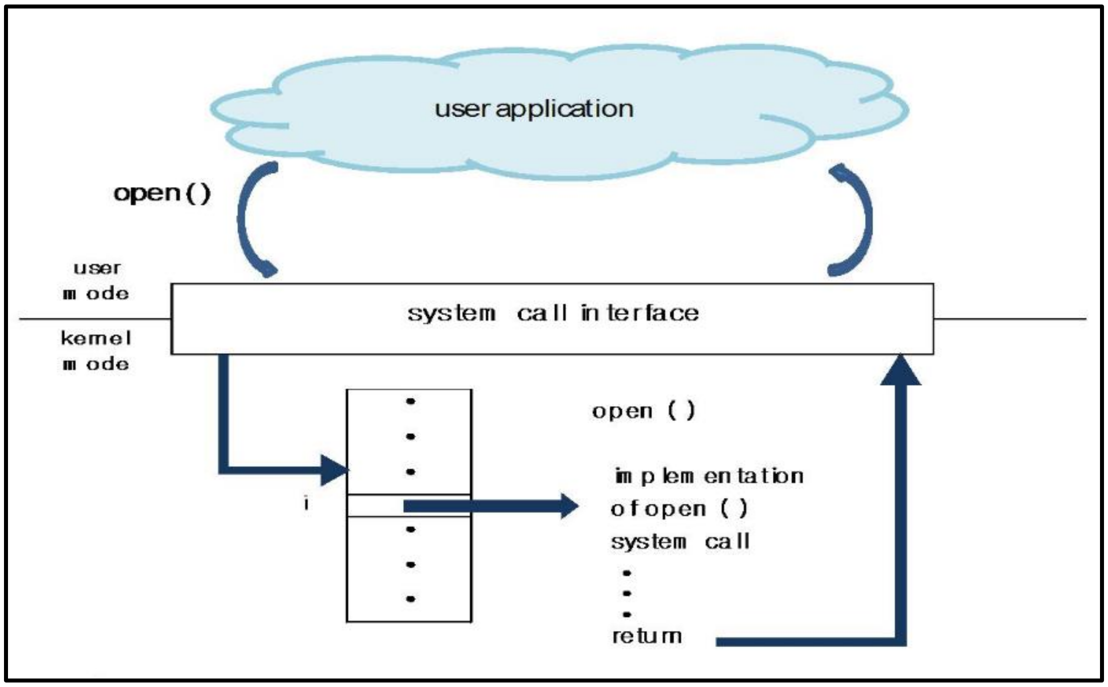
A system call is a way for a user program to interface with the operating system. The program requests several services, and the OS responds by invoking a series of system calls to satisfy the request. A system call can be written in assembly language or a high-level language like **C** or **Pascal**. System calls are predefined functions that the operating system may directly invoke if a high-level language is used.

In this article, you will learn about the system calls in the operating system and discuss their types and many other things.

What is a System Call?

A system call is a method for a computer program to request a service from the kernel of the operating system on which it is running. A system call is a method of interacting with the operating system via programs. A system call is a request from computer software to an operating system's kernel.

The **Application Program Interface (API)** connects the operating system's functions to user programs. It acts as a link between the operating system and a process, allowing user-level programs to request operating system services. The kernel system can only be accessed using system calls. System calls are required for any programs that use resources.



How are system calls made?

When a computer software needs to access the operating system's kernel, it makes a system call. The system call uses an API to expose the operating system's services to user programs. It is the only method to access the kernel system. All programs or processes that require resources for execution must use system calls, as they serve as an interface between the operating system and user programs.

Below are some examples of how a system call varies from a user function.

1. A system call function may create and use kernel processes to execute the asynchronous processing.
2. A system call has greater authority than a standard subroutine. A system call with kernel-mode privilege executes in the kernel protection domain.
3. System calls are not permitted to use shared libraries or any symbols that are not present in the kernel protection domain.
4. The code and data for system calls are stored in global kernel memory.

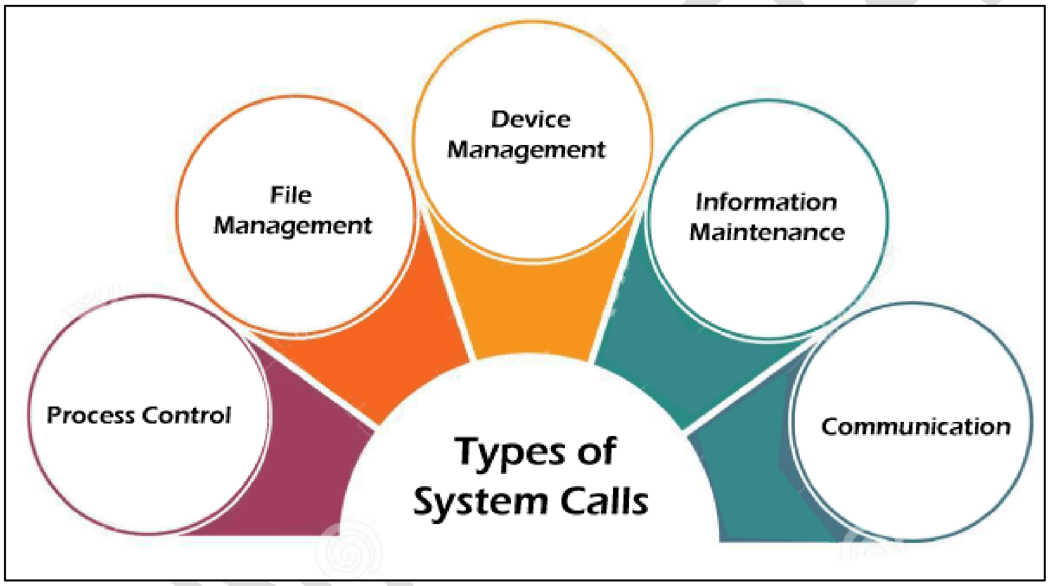
Why do you need system calls in Operating System?

There are various situations where you must require system calls in the operating system. Following of the situations are as follows:

1. It is must require when a file system wants to create or delete a file.
2. Network connections require the system calls to sending and receiving data packets.
3. If you want to read or write a file, you need to system calls.
4. If you want to access hardware devices, including a printer, scanner, you need a system call.
5. System calls are used to create and manage new processes.

Types of System Calls

There are commonly five types of system calls. These are as follows:



1. **Process Control**
2. **File Management**
3. **Device Management**
4. **Information Maintenance**
5. **Communication**

Now, you will learn about all the different types of system calls one-by-one.

- **Process Control**

Process control is the system call that is used to direct the processes. Some process control examples include creating, load, abort, end, execute, process, terminate the process, etc.

- **File Management**

File management is a system call that is used to handle the files. Some file management examples include creating files, delete files, open, close, read, write, etc.

- **Device Management**

Device management is a system call that is used to deal with devices. Some examples of device management include read, device, write, get device attributes, release device, etc.

- **Information Maintenance**

Information maintenance is a system call that is used to maintain information. There are some examples of information maintenance, including getting system data, set time or date, get time or date, set system data, etc.

- **Communication**

Communication is a system call that is used for communication. There are some examples of communication, including create, delete communication connections, send, receive messages, etc.

Examples of Windows and Unix system calls

There are various examples of Windows and Unix system calls. These are as listed below in the table:

Process	Windows	Unix

Process Control	CreateProcess()
	ExitProcess()
	WaitForSingleObject()
		Fork()
		Exit()
		Wait()

File Manipulation	CreateFile()
	ReadFile()
	WriteFile()
	CloseHandle()
		Open()
		Read()
		Write()
		Close()

Device Management	SetConsoleMode()
	ReadConsole()
		Ioctl()
		Read()

	WriteConsole()	Write()

Information Maintenance	GetCurrentProcessID()
	SetTimer()
	Sleep()
		Getpid()
		Alarm()
		Sleep()

Communication	CreatePipe()
	CreateFileMapping()
	MapViewOfFile()
		Pipe()
		Shmget()
		Mmap()

Protection	SetFileSecurity()
	InitializeSecurityDescriptor()
	SetSecurityDescriptorgroup()
		Chmod()
		Umask()
		Chown()

Here, you will learn about some methods briefly:

1. **open()**

The **open()** system call allows you to access a file on a file system. It allocates resources to the file and provides a handle that the process may refer to. Many processes can open a file at once or by a single process only. It's all based on the file system and structure.

2. **read()**

It is used to obtain data from a file on the file system. It accepts three arguments in general:

- A file descriptor.
- A buffer to store read data.
- The number of bytes to read from the file.

The file descriptor of the file to be read could be used to identify it and open it using **open()** before reading.

3. **wait()**

In some systems, a process may have to wait for another process to complete its execution before proceeding. When a parent process makes a child process, the parent process execution is suspended until the child process is finished. The **wait()** system call is used to suspend the parent process. Once the child process has completed its execution, control is returned to the parent process.

4. **write()**

It is used to write data from a user buffer to a device like a file. This system call is one way for a program to generate data. It takes three arguments in general:

- A file descriptor.
- A pointer to the buffer in which data is saved.
- The number of bytes to be written from the buffer.

5. **fork()**

Processes generate clones of themselves using the **fork()** system call. It is one of the most common ways to create processes in operating systems. When a parent process spawns a child process, execution of the parent process is interrupted until the child process completes. Once the child process has completed its execution, control is returned to the parent process.

6. **close()**

It is used to end file system access. When this system call is invoked, it signifies that the program no longer requires the file, and the buffers are flushed, the file information is altered, and the file resources are de-allocated as a result.

7. **exec()**

When an executable file replaces an earlier executable file in an already executing process, this system function is invoked. As a new process is not built, the old process identification stays, but the new process replaces data, stack, data, head, etc.

8. **exit()**

The **exit()** is a system call that is used to end program execution. This call indicates that the thread execution is complete, which is especially useful in multi-threaded environments. The operating system reclaims resources spent by the process following the use of the **exit()** system function.

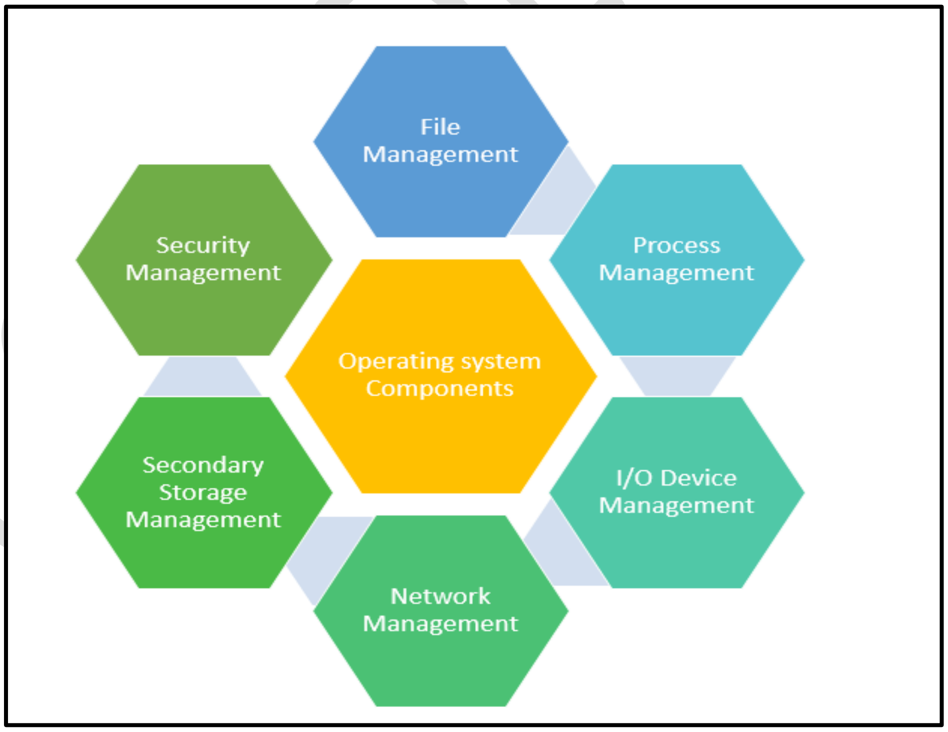
2.3 Operating System Components :

An operating system is a large and complex system that can only be created by partitioning into small parts. These pieces should be a well-defined part of the system, carefully defining inputs, outputs, and functions.

Although Windows, Mac, UNIX, Linux, and other OS do not have the same structure, most operating systems share similar OS system components, such as file, memory, process, I/O device management.

The components of an operating system play a key role to make a variety of computer system parts work together. There are the following components of an operating system, such as:

1. Process Management
2. File Management
3. Network Management
4. Main Memory Management
5. Secondary Storage Management
6. I/O Device Management
7. Security Management



- **Process Management**

The process management component is a procedure for managing many processes running simultaneously on the operating system. Every running software application program has one or more processes associated with them.

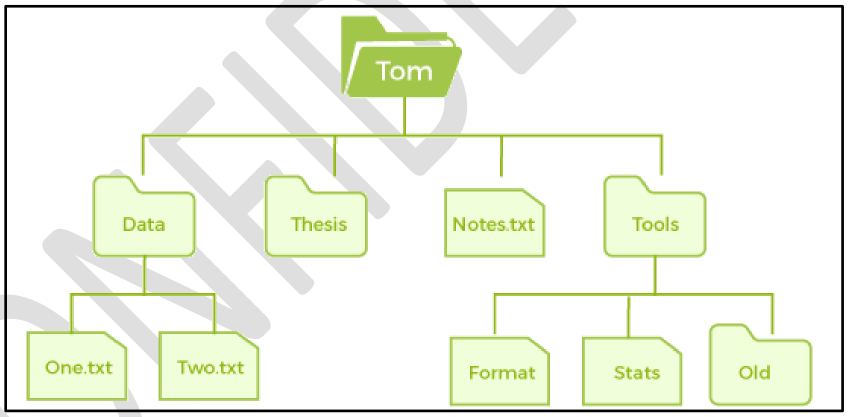
Functions of process management

Here are the following functions of process management in the operating system, such as:

- Process creation and deletion.
- Suspension and resumption.
- Synchronization process
- Communication process

- **File Management**

A file is a set of related information defined by its creator. It commonly represents programs (both source and object forms) and data. Data files can be alphabetic, numeric, or alphanumeric.



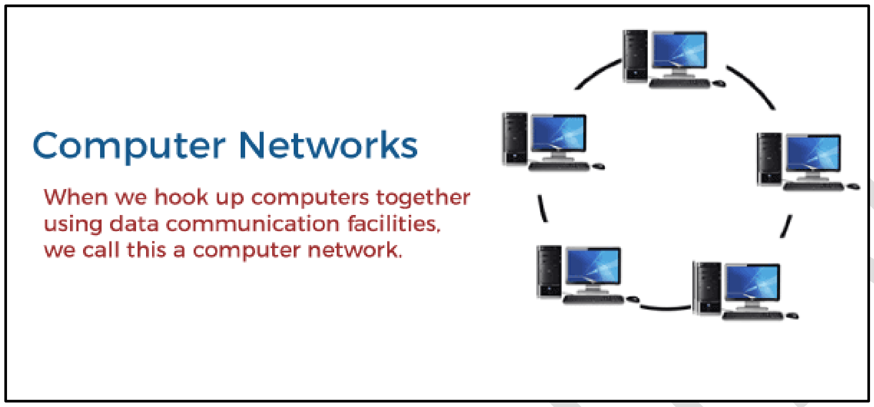
Function of file management

The operating system has the following important activities in connection with file management:

- File and directory creation and deletion.
- For manipulating files and directories.
- Mapping files onto secondary storage.
- Backup files on stable storage media.

- **Network Management**

Network management is the process of administering and managing computer networks. It includes performance management, provisioning of networks, fault analysis, and maintaining the quality of service.


Computer Networks
When we hook up computers together using data communication facilities, we call this a computer network.

A distributed system is a collection of computers or processors that never share their memory and clock. In this type of system, all the processors have their local memory, and the processors communicate with each other using different communication cables, such as fibre optics or telephone lines.

The computers in the network are connected through a communication network, which can configure in many different ways. The network can fully or partially connect in network management, which helps users design routing and connection strategies that overcome connection and security issues.

Functions of Network management

Network management provides the following functions, such as:

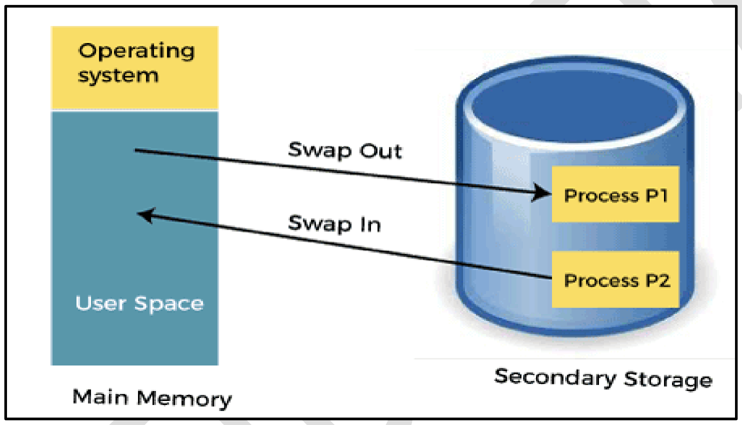
- Distributed systems help you to various computing resources in size and function. They may involve minicomputers, microprocessors, and many general-purpose computer systems.
- A distributed system also offers the user access to the various resources the network shares.
- It helps to access shared resources that help computation to speed up or offers data availability and reliability.

- **Main Memory management**

Main memory is a large array of storage or bytes, which has an address. The memory management process is conducted by using a sequence of reads or writes of specific memory addresses.

It should be mapped to absolute addresses and loaded inside the memory to execute a program. The selection of a memory management method depends on several factors.

However, it is mainly based on the hardware design of the system. Each algorithm requires corresponding hardware support. Main memory offers fast storage that can be accessed directly by the CPU. It is costly and hence has a lower storage capacity. However, for a program to be executed, it must be in the main memory.



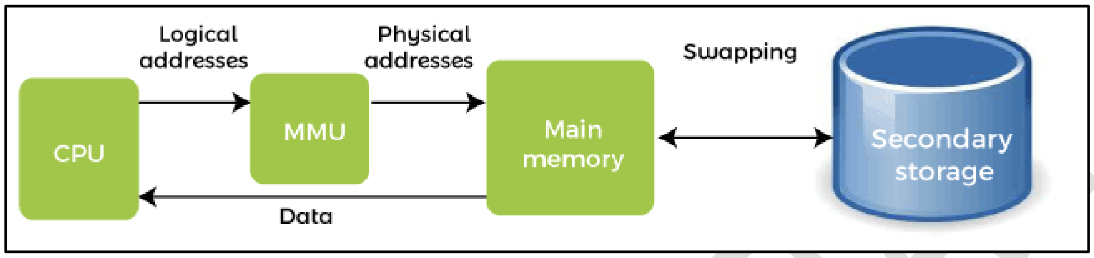
Functions of Memory management

An Operating System performs the following functions for Memory Management in the operating system:

- It helps you to keep track of primary memory.
- Determine what part of it are in use by whom, what part is not in use.
- In a multiprogramming system, the OS decides which process will get memory and how much.
- Allocates the memory when a process requests.
- It also de-allocates the memory when a process no longer requires or has been terminated.

- **Secondary-Storage Management**

The most important task of a computer system is to execute programs. These programs help you to access the data from the main memory during execution. This memory of the computer is very small to store all data and programs permanently. The computer system offers secondary storage to back up the main memory.



Today modern computers use hard drives/SSD as the primary storage of both programs and data. However, the secondary storage management also works with storage devices, such as USB flash drives and CD/DVD drives. Programs like assemblers and compilers are stored on the disk until it is loaded into memory, and then use the disk is used as a source and destination for processing.

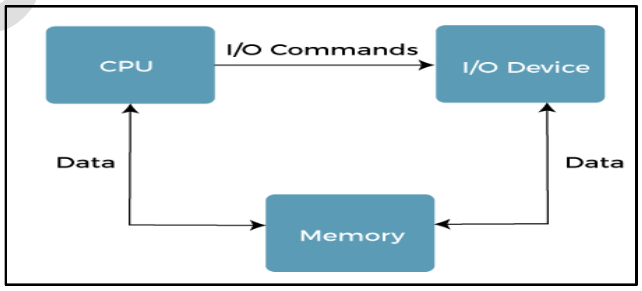
Functions of Secondary storage management

Here are some major functions of secondary storage management in the operating system:

- Storage allocation
- Free space management
- Disk scheduling

- **I/O Device Management**

One of the important use of an operating system that helps to hide the variations of specific hardware devices from the user.



Functions of I/O management

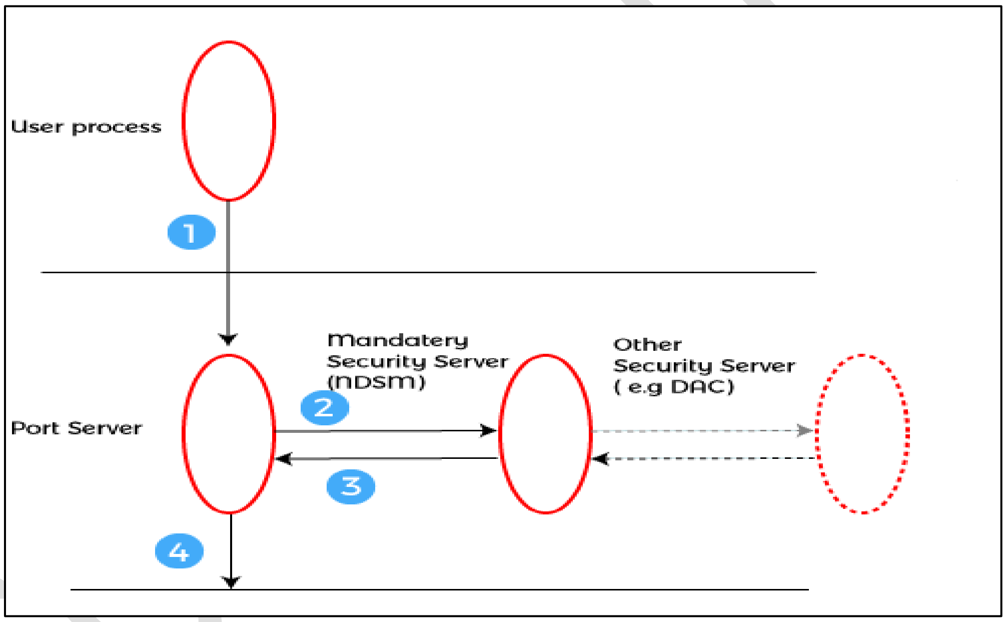
The I/O management system offers the following functions, such as:

- It offers a buffer caching system
- It provides general device driver code
- It provides drivers for particular hardware devices.
- I/O helps you to know the individualities of a specific device.

- **Security Management**

The various processes in an operating system need to be secured from other activities. Therefore, various mechanisms can ensure those processes that want to operate files, memory CPU, and other hardware resources should have proper authorization from the operating system.

Security refers to a mechanism for controlling the access of programs, processes, or users to the resources defined by computer controls to be imposed, together with some means of enforcement.



For example, memory addressing hardware helps to confirm that a process can be executed within its own address space. The time ensures that no process has control of the CPU without renouncing it. Lastly, no process is allowed to do its own I/O to protect, which helps you to keep the integrity of the various peripheral devices.

Security can improve reliability by detecting latent errors at the interfaces between component subsystems. Early detection of interface errors can prevent the foulness of a healthy subsystem by a malfunctioning subsystem. An unprotected resource cannot misuse by an unauthorized or incompetent user.

2.4 Use of OS Tools, User Management, Security Policy, Device Management, Performance Monitor and task Scheduler :

- **User Management :**

  User management includes everything from creating user to deleting user on your system.

  User management can be done in 3 ways on linux os:

  - Graphical tools are easy and suitable for new user.
  - Command line tools includes commands like useradd, userdel, passwd etc. these are mostly used by system administrator.
  - Edit the local configuration files directly using vi.

  Root: root user is super user have all the powers for creating a user, deleting user and can login with the others user account.

  Linux command line tools for managing users and groups:

- useradd :to add new users in linux os.
  Syntax : useradd login_name
- passwd:to setup password for users.
  Syntax: password user_name
- userdel: delete a user account.
  Syntax:userdel –r<username>s

- **Security Policy**

1. keep system updated:

Always keep system updated with latest releases, security fixes and kernel when its available.

2. Use Secure SSH:

   SSH is secure protocol that uses encryption technology during communication with server.
3. deactivate networking ports:

Using 'netstat' networking command you can view all open ports and associated programs.

- Use 'chkconfig' command to disable all unwanted network services from system.

- 4. lock and unlock feature:
  - Instead of removing an account from system,you can lock it for an week or a month.
  - To lock particular user ,use passwd -1 account_name
- 5. turn off IPV6:
  - If you are not using IPV6 protocol, then you should disable it .
- 6.use SFTP ,not FTP:
  - FTP is not safe ,even you encrypt your connection.
  - Secure FTP is encrypts all the data ,credentials and files included.
- 7.use strong password policy:
  - Password must contain both uppercase, lowercase, be mix of numbers, letters and symbols and you be safe.
- 8. install antivirus software:

ClamAV and Maldet these are open source applications that do good job of securing your server from potential threats

- **Device Management**

- Device management is the process of managing the implementation, operation and maintenance of a physical and virtual devices.
- All Linux device files are located in /dev directory, which is an integral part of /root directory. because these device file must be available to OS during boot process.
- Device nodes in the /dev directory provides access to the corresponding kernel devices.
- With udev, the /dev directory reflects the current state
- of the kernel.
- Every kernel device has a device file. if device is disconnected from the system, then device node is removed.

- ## Performance Monitor

  top command :

  - Linux top command is performance monitoring program which is used frequently by many system admin to monitor linux performance.
  - top command is used to display all the running and active real-time processes in ordered list and update it regularly.
  - It display cpu usage ,memory usage, swap memory, cache, size, buffer size, process id, user.
  - top command is much useful for system administrator to monitor and take correct action when required.
  - **Syntax: top**

  vmstat:

  - Used to display statistics of virtual memory, kernel threads ,disk ,system processes, i/o blocks, interrupts, cpu activity and much more.
  - **Syntax : vmstat**

- ## Task Scheduler

- The cron daemon on linux runs tasks in the background at specific times, its like task schedular on windows.
- Add tasks to your system crontab files using the appropriate syntax and cron will automatically run them for you.
- Crontab files can be used to automate backups, system maintenance and other repetitive tasks.