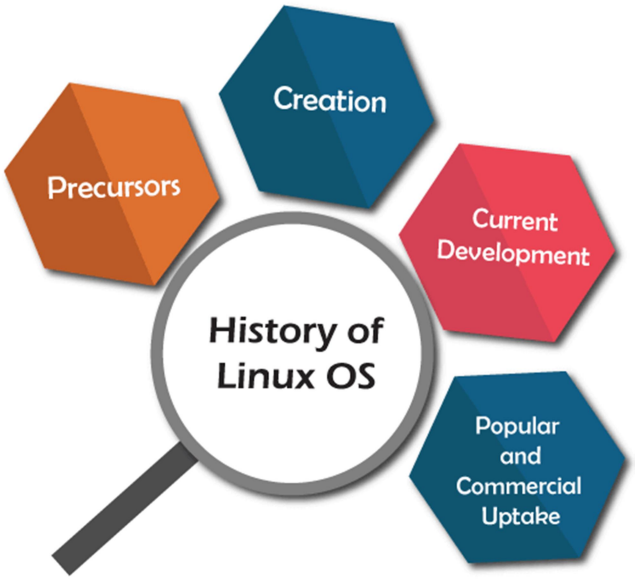
# Unit 6: Linux

**Introduction to Linux**

- Linux is an open-source Unix-like operating system-based family on the Linux kernel, and the OS kernel was first published on 17 September 1991 by Linus Torvalds.
- Originally, Linux was designed for personal computers that were Intel x86 architecture-based, but it has since been moved to more environments than other operating systems.
- Including Android, Linux has the biggest installed base of every general-purpose operating system because of the control of the Linux-based Android over smartphones as of May 2022.
- Also, Linux executes on many embedded systems, i.e., devices whose OS is typically designed into the firmware and is extremely customized to the system.
- It includes spacecraft (Perseverance rover, Dragon crew capsule, and Falcon 9 rocket), automobiles (Toyota, Hyundai, Mercedes-Benz, Audi, and Tesla), televisions (LG and Samsung Smart TVs), video game consoles, smart home devices, automation controls, and routers.
- Linux is one of the most outstanding examples of open-source and free software collaboration. The source code may be distributed, modified, and used non-commercially or commercially by everyone under the conditions of its respective licenses, like the GNU GPL (General Public License). For example, the Linux kernel is licensed upon the GPLv2.

## History of Linux OS

- **Precursors**
  The Unix-based operating system was implemented and conceived in 1969 at AT&T's Bell labs by Joe Ossanna, Douglas McIlroy, Dennis Ritchie, and Ken Thompson in the United States. First published in 1971, Unix was entirely written in assembly language, as was the basic practice at the time. It was updated in the C language by Dennis Ritchie in a key pioneering way in 1973. The availability of a Unix high-level language implementation made its porting to distinct computer platforms convenient.




- **Creation**
  Torvalds registered in a Unix course while visiting the University of Helsinki in the 1990's fall. Torvalds started the Linux kernel development on MINIX, and software written for MINIX was used on Linux as well. Later, Linux was cultivated, and then the development of the Linux kernel appeared on Linux systems.

- **Current development**
  The third-party components are composed of a wide body of work and may contain both user libraries and applications, and kernel modules. Linux community and vendors distribute and combine the kernel, non-GNU components, and GNU components with extra package management software in the fashion of Linux distributions.

- **Popular and commercial uptake**
  In production environments, Linux adoption began to take off initially in the mid-1990s in the supercomputing community instead of being used by only hobbyists, where organizations like NASA began to increasingly replace their expensive machines with inexpensive commodity computer clusters running Linux. Commercial use started when IBM and Dell, pursued by Hewlett-Packard, began providing Linux support for escaping the monopoly of Microsoft in the desktop OS market.

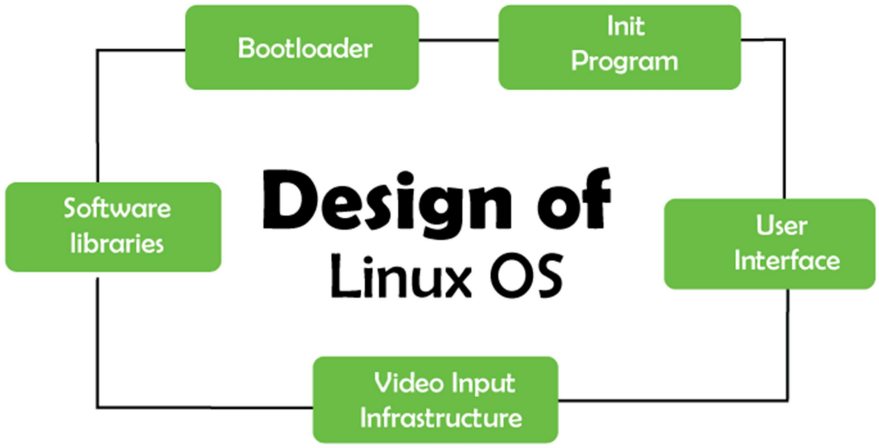
- Linux systems are completely used in computing today, from embedded systems to every supercomputer virtually, and have secured a position in

server installations like the famous LAMP application stack. The usage of Linux distributions in enterprise and home desktops has been developing.

- Also, Linux distributions have become famous in the netbook market, with several devices moving with installed customized Linux distributions and Google publishing their ChromeOS developed for netbooks.

**Design of Linux OS**

Various open-source developers admit that the Linux kernel was not developed but instead evolved from natural selection. A Linux-based system is a compatible Unix-like OS, derived much of its common design from principles made in Unix during the 1970s and 1980s. Such a system applies the Linux kernel, a monolithic kernel which manages file systems, peripheral access, networking, and process control. Device drivers are either directly integrated with the kernel or included as modules loaded while the device is active.




Installed Linux System components include the below:

- A **bootloader,** for instance, systemd-boot, SYSLINUX, LILO, and GNU GRUB. It is a program that can load the Linux kernel into the main memory of the computer by being run by the computer after the firmware login is performed and when it's turned on.
- An **init program,** like the traditional sysvinit and the newer Upstart, OpenRC, and systemd. It is the first process announced by the Linux kernel and the process tree root. In other words, every process is opened from init. It initiates processes like login prompts and system services (whether in terminal or graphical mode).

- **Software libraries,** which include code that can be applied by running processes. Besides, the most widely used software library is the GNU C Library (glibc) on Linux systems.
- **User interface:** Also, the user interface is called a shell. It is either a GUI (graphical user interface), a CLI (command-line interface), or controls attached to the related hardware, which is normal for embedded systems. The default user interface is graphical for desktop systems. However, the CLI is available by terminal emulator windows or on an isolated virtual console.
- **Video input infrastructure**
  Currently, Linux has two kernel-user space APIs to handle video input devices: DVB API for TV reception and V4L2 API for radio and video streams.

## Interfaces to Linux

- There are plenty of GUI options in Linux, unlike some operating systems that leave you with only one option.
- Linux has exploded with different GUI options in recent years. They are all free and open-source, which is great for the community. However, with so many options available to users it can be a challenge to decide which one you want to use on your system.
- The different desktop GUIs are:

### 1. KDE Plasma

- KDE Plasma is a very popular desktop environment. Its lightweight design and customization options make KDE Plasma very versatile. You have convenient features like mobile phone integration with your Linux system using KDE Connect.
- The KDE Plasma desktop experience gives users a lot of control over the desktop look and feel. Users can choose their color scheme, move panels anywhere they want them to be, or use a different system font. Users can download custom widgets and add anything from clocks to calendars straight on their panel.

### 2. GNOME

- The GNOME desktop environment has been a popular choice for many Linux users over the years. It's popularity is due to the clean, minimalistic look.
- GNOME has been designed with usability in mind and is the perfect setup for people that just need the basics to get some work done on their Linux

machine. All of the features that it offers are tucked away neatly in a desktop dock or application list.

- GNOME is a great desktop environment for those who want to customize their experience, but it can be heavy on resources. Older systems might struggle a little if they don't have enough RAM, or if the processor is a few too many generations behind.

**3. XFCE**

- The XFCE desktop environment is an excellent choice for those who want to have a more lightweight and customizable experience than GNOME offers. The interface can be customized, and the features that you use most are available with one click from your application dock or menu bar, so it's a good choice for PC enthusiasts that enjoy customizing their desktops.

**4. LXDE**

- LXDE is another lightweight desktop environment that uses system resources sparingly, which means it can be used with a cheaper embedded board (like a Raspberry Pi) or an old salvaged computer.
- LXDE provides the user with an easy-to-use interface that is responsive and simple to learn. If you're looking for a free, lightweight desktop environment that is easy to use and provides the basics of what a Linux interface needs, LXDE might be for you.

**5. MATE**

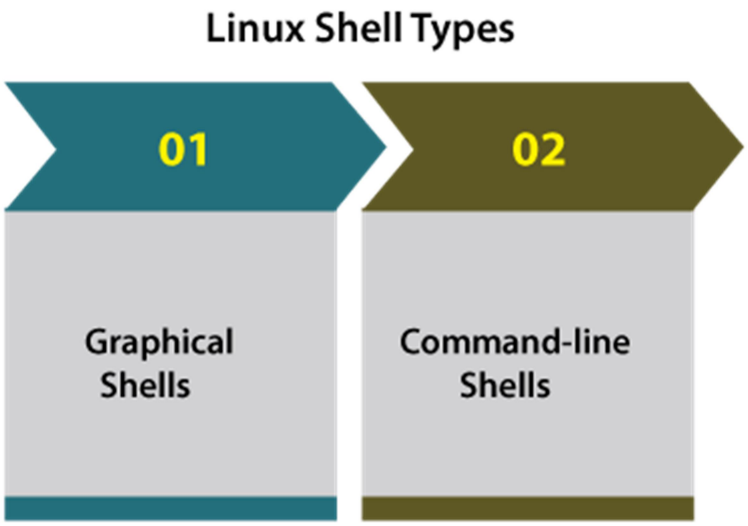
- MATE is a Linux desktop environment that forked from GNOME 2. MATE was created for users that didn't like the direction that GNOME 3 was headed, which means that it has all of the features you would expect from a more polished desktop environment.
- MATE has many applications within it that allow it to work as well as it does. It uses Caja as a filesystem, Pluma as a text editor, Atril for document reading, and much more.

## Linux Shell

- The shell can be defined as a command interpreter within an operating system like Linux/GNU or Unix. It is a program that runs other programs. The shell facilitates every user of the computer as an interface to the Unix/GNU

Linux system. Hence, the user can execute different tools/utilities or commands with a few input data.

- The shell sends the result to the user over the screen when it has completed running a program which is the common output device. That's why it is known as "command interpreter".
- The shell is not just a command interpreter. Also, the shell is a programming language with complete constructs of a programming language such as functions, variables, loops, conditional execution, and many others.
- For this reason, GNU/Unix Linux Shell is stronger than the Windows shell. Broadly, the shell is categorized into two main categories which are explained below:




**Graphical Shells**

- These shells specifies the manipulation of programs that are based on the graphical user interface (GUI) by permitting for operations like moving, closing, resizing, and opening windows and switching focus among windows as well. Ubuntu OS or Windows OS could be examined as a good example that offers a graphical user interface to the user to interact with the program. Various users don't need for typing in any command for all the actions.

**Command-line Shell**

- Various shells could be accessed with the help of a command-line interface by users. A unique program known as Command prompt in Windows or Terminal in macOS/Linux is offered for typing in the human-understandable

commands like "ls", "cat", etc and after that, it is being run. The result is further shown to the user on the terminal.

- Working on a command-line shell is a complicated for many beginners due to it is hard to remember several commands. Command-line shell is very dominant and it permits users for storing commands in a file and run them together. In this way, a repetitive action could be automated easily. Usually, these files are known as Shell scripts in macOS/Linux systems and batch files in Windows.
- There are various types of shells which are discussed as follows:
  - **Bash Shell**

    In the bash shell, bash means Bourne Again Shell. It is a default shell over several distributions of Linux today. It is a sh-compatible shell. It could be installed over Windows OS. It facilitates practical improvements on sh for interactive and programming use which contains:
    - Job Control
    - Command-line editing
    - Shell Aliases and Functions
    - Unlimited size command history
    - Integer arithmetic in a base from 2-64
  - **Csh/Tcsh Shell**

    Tcsh is an upgraded C shell. This shell can be used as a shell script command processor and interactive login shell.

    Tcsh shell includes the following characteristics:
    - C like syntax
    - Filename completion and programmable word
    - Command-line editor
    - Job control
    - Spelling correction
  - **Ksh Shell**

    Ksh means for Korn shell. It was developed and designed by David G. Korn. Ksh shell is a high-level, powerful, and complete programming language and it is a reciprocal command language as well just like various other GNU/Unix Linux shells. The usage and syntax of the C shell are very same as the C programming language.
  - **Zsh Shell**

    Zsh shell is developed to be reciprocal and it combines various aspects of other GNU/Unix Linux shells like ksh, tcsh, and bash. Also, the POSIX shell standard specifications were based on the Korn shell.

Also, it is a strong scripting language like other available shells. Some of its unique features are listed as follows:

- Startup files
- Filename generation
- Login/Logout watching
- Concept index
- Closing comments
- Variable index
- Key index
- Function index and various others that we could find out within the man pages.

All these shells do a similar job but take different commands and facilitate distinct built-in functions.

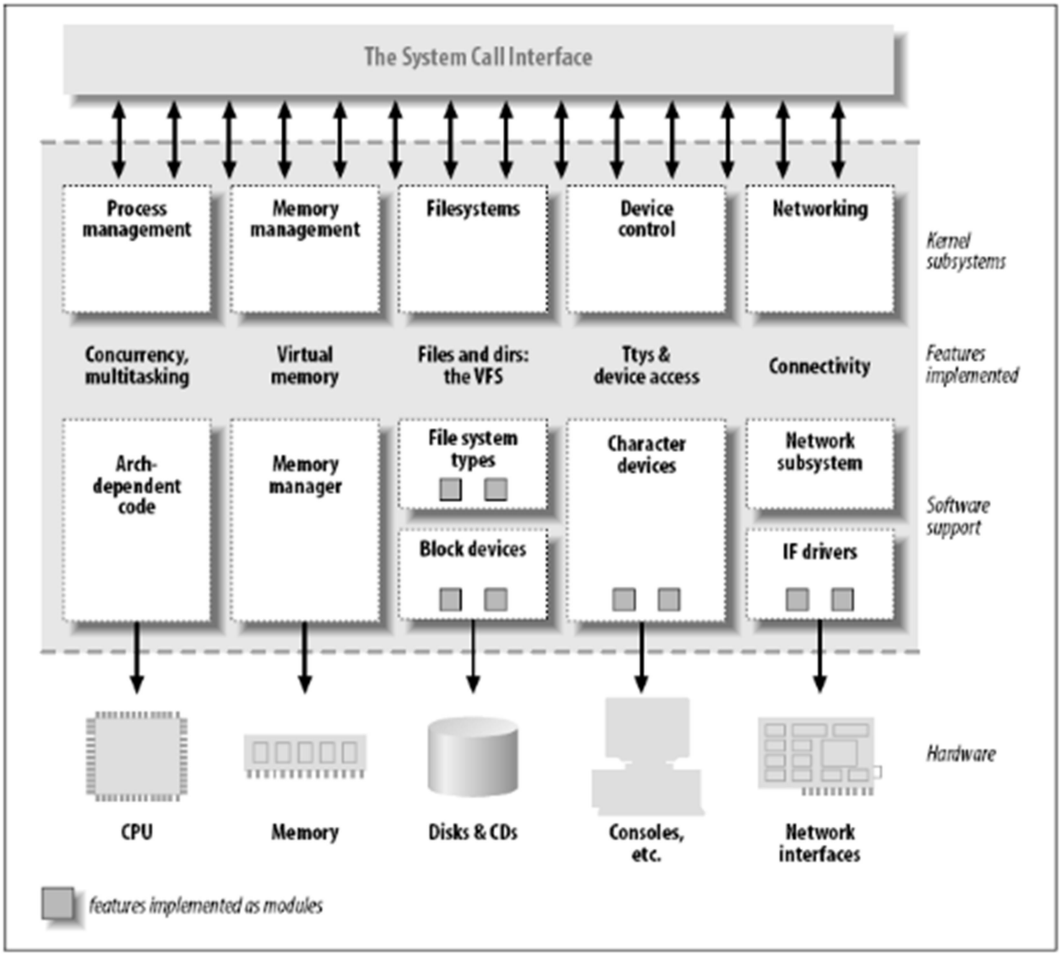
- **Fish**

  Fish stands for "friendly interactive shell". It was produced in 2005. Fish shell was developed to be fully user-friendly and interactive just like other shells. It contains some good features which are mentioned below:

  - Web-based configuration
  - Man page completions
  - Auto-suggestions
  - Support for term256 terminal automation
  - Completely scripted with clean scripts

## Architecture of Kernel

Kernel architecture follows modular approach. Each block in kernel (i.e. file management) is nothing but piece of code written in C language. Each block consists of powerful structure for handling various operations.




**Process management**

This is the primary subsystem of the kernel as it is responsible for distributing the CPU time and resources among all the processes or applications in a fairway. Its purpose is to ensure that no process runs out of CPU resources, allowing multiple applications to run simultaneously without affecting the performance of one. The process scheduler provides fairness, efficiency, and responsiveness by utilizing various scheduling algorithms such as round-robin, priority-based, or multi-level feedback queues.

**Memory management**

This kernel subsystem is responsible for managing and organizing the system's memory resources. It is responsible for ensuring that memory is allocated and distributed appropriately among multiple processes and applications, preventing

issues such as crashes or kernel mode errors caused by insufficient memory. By efficiently managing memory, this subsystem ensures that processes and applications have access to the necessary memory resources and prevents memory-related errors.

**The Virtual File System (VFS)**

The Virtual File System (VFS) subsystem inside the kernel is responsible for providing an equal interface to all available filesystems on the computer and allowing them to access the stored data on those file systems. It abstracts the details of different file systems, such as ext4, NTFS, or FAT, and provides a consistent file I/O interface to user programs. Regardless of the underlying file system, the VFS layer enables programs to do file-related actions such as opening, reading, writing, and closing files.

**The Networking Unit**

The Networking Unit subsystem in the Linux kernel is an essential component located within the kernel space. It plays a crucial role in facilitating communication between hosts, even if they are not directly connected. In X-Windows, the network subsystem is used for client-server communication, allowing applications to connect over a network. The Linux Kernel's networking stack handles incoming packets, processing them from Layer 2 (data link layer) up to the network layer.

**Inter-Process Communication Unit**

The inter-process communication (IPC) unit facilitates communication and data sharing between distinct operating system processes or threads. Linux supports a number of Inter-Process Communication (IPC) mechanisms. Signals and pipes are two of them, but Linux also supports the System V IPC mechanisms, which are named after the Unix TM release in which they were initially introduced.

**Features of Linux Kernel**

If the portability and efficiency of the Linux kernel weren't enough, it also has a few more features.

- Linux, as an open-source production operating system, serves as an ideal platform for testing and advancing new protocols. It supports a broad number of networking protocols, including the commonly used TCP/IP suite.
- Linux is well-known for its dynamic kernel, which allows for both the addition and removal of software components while the system is running. These components, known as dynamically loadable kernel modules, can be inserted when necessary during system boot (when a specific device is found requiring the module) or at any time by the user.
- Linux's most recent advancement is its usage as an operating system for other operating systems (known as a hypervisor). This is made feasible through a kernel modification known as Kernel-based Virtual Machine (KVM).

## Processes in Linux

A process is an instance of a program currently running on a computer system. In Linux, processes are managed by the operating system's kernel, which allocates system resources and schedules processes to run on the CPU. Understanding and managing processes is a critical skill for Linux administrators and developers.

### Types of Processes

In Linux, processes can be categorized into two types:

1. Foreground Processes: Foreground processes are the kinds of processes that require input from the user and are characterized by their interactivity. For instance, a foreground process would be like you are running an Office application on the Linux system.
2. Background Processes: On the other hand, background processes are non-interactive operations carried out in the background and do not call for any participation from the user. Antivirus software is an example of a Background Process.

Additionally, processes can be system processes or user processes. System processes are initiated by the kernel, while users initiate User processes.

### Process States in Linux

In Linux, a process can be in one of five states:

1. Running: The process is currently executing on the CPU.
2. Sleeping: The process is waiting for a resource to become available.
3. Stopped: The process has been terminated by a user
4. Zombie: The process has completed execution but has not yet been cleaned by the system.
5. Orphan: The parent process of the current process has been terminated.

**What is Process Management in Linux?**

Process management is the task of controlling and monitoring the processes that are running on a Linux system. It involves managing process resources, scheduling processes to run on the CPU, and terminating processes when required.

**Commands for Process Management in Linux**

Linux provides several commands for managing processes, which include:

| Commands | Description |
|---|---|
| ps | Displays information about the processes running currently. |
| top | Provides real-time information about system processes and their resource usage. |
| kill | Terminates a process by sending a signal to it. |
| nice | Adjusts the priority of a process. |
| renice | Changes the priority of a running process. |
| ps PID | Shows the state of an exact process. |
| pidof | Shows the Process ID of a process. |
| df | Shows Disk Management of your system. |
| free | Shows the status of your RAM. |
| bg | For sending a running process to the background. |
| fg | For running a stopped process in the foreground. |

**Practically Managing the Processes**

We have explained some common commands below which are required to manage processes. Let's check them out:

1. Identifying and Terminating Processes in Linux.

Identification of processes and termination of processes can be done by using the below-mentioned commands in Linux.

The ps command is used to find the process ID (PID) of the process you want to manage. As instance, if it's required to kill a process you need to know the process ID (PID) of that exact process.

The kill command in Linux is used to terminate processes by their process IDs (PIDs).

The killall command in Linux is used to terminate processes by their names. It sends a signal to all processes with a specified name, effectively killing them.

2. Managing Priority of Processes in Linux

Managing the priority of a running process is done using "nice" and "renice" commands in Linux.

The nice command in Linux is used to modify the priority of a process. It assigns a lower priority to a process to reduce its resource usage.

The renice command is used to modify the priority of an already running process. It can increase or decrease the priority of a process, depending on the specified value.

3. Analyzing Resource Usage in Linux.

The top command in Linux is used to display real-time information about processes running on a system, including CPU and memory usage. It provides an interactive interface that allows users to monitor and manage processes.

## Process Scheduling in Linux

Scheduling of processes is one of the most important aspects or roles of any operating system. A Process Scheduler deals with process scheduling in Linux. Process Scheduler uses Scheduling Algorithms that help in deciding the process to be executed.

### The Linux Scheduling Algorithm

Process scheduling is one of the most important aspects or roles of any operating system.

- Process Scheduling is an important activity performed by the process manager of the respective operating system.
- Scheduling in Linux deals with the removal of the current process from the CPU and selecting another process for execution.

Similar to the other UNIX-based operating systems, in LINUX a Process Scheduler deals with process scheduling.

- Process Scheduler chooses a process to be executed.
- Process scheduler also decides for how long the chosen process is to be executed.

Hence, Scheduling Algorithms is nothing but a kind of strategy that helps the process scheduler in deciding the process to be executed.

### Scheduling Process Types in Linux

In the LINUX operating system, we have mainly two types of processes namely - Real-time Process and Normal Process. Let us learn more about them in detail.

1. Realtime Process

Real-time processes are processes that cannot be delayed in any situation. Real-time processes are referred to as urgent processes.

There are mainly two types of real-time processes in LINUX namely:

- SCHED_FIFO
- SCHED_RR.

A real-time process will try to seize all the other working processes having lesser priority.

For example, A migration process that is responsible for the distribution of the processes across the CPU is a real-time process. Let us learn about different scheduling policies used to deal with real-time processes briefly.

SCHED_FIFO

FIFO in SCHED_FIFO means First In First Out. Hence, the SCHED_FIFO policy schedules the processes according to the arrival time of the process.

SCHED_RR

RR in SCHED_RR means Round Robin. The SCHED_RR policy schedules the processes by giving them a fixed amount of time for execution. This fixed time is known as time quantum.

2. Normal Process

Normal Processes are the opposite of real-time processes. Normal processes will execute or stop according to the time assigned by the process scheduler. Hence, a normal process can suffer some delay if the CPU is busy executing other high-priority processes. Let us learn about different scheduling policies used to deal with the normal processes in detail.

Normal (SCHED_NORMAL or SCHED_OTHER)

SCHED_NORMAL / SCHED_OTHER is the default or standard scheduling policy used in the LINUX operating system. A time-sharing mechanism is used in the normal policy. A time-sharing mechanism means assigning some specific amount of time to a process for its execution. Normal policy deals with all the threads of processes that do not need any real-time mechanism.

Batch (SCHED_BATCH)

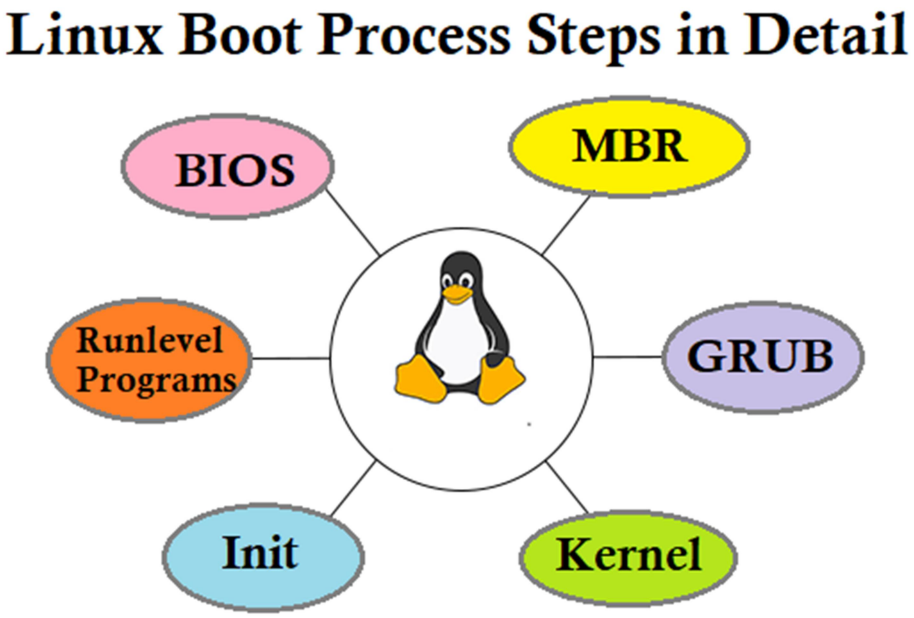
As the name suggests, the SCHED_BATCH policy is used for executing a batch of processes. This policy is somewhat similar to the Normal policy. SCHED_BATCH policy deals with the non-interactive processes that are useful in optimizing the CPU throughput time. SCHED_BATCH scheduling policy is used for a group of processes having priority: 0.

Idle (SCHED_IDLE)

SCHED_IDLE policy deals with the processes having extremely Low Priority. Low-priority tasks are the tasks that are executed when there are absolutely no tasks to be executed. SHED_IDLE policy is designed for the lowest priority tasks of the operating systems.

## Booting Process of Linux

An OS is low-level software that handles resources, gives basic services to other software, and controls peripherals. We will explain each boot process in detail:



- **BIOS**

  BIOS is an acronym for Basic Input/Output System. In other words, the BIOS can load and run the MBR (Master Boot Record) boot loader. When we first turn on our system, the BIOS first implements a few integrity checks of the SSD or HDD.

  After that, the BIOS finds, loads, and runs the boot loader function, which can be detected in the MBR. Sometimes, the MBR is on a CD-ROM or USB stick, like with a live Linux installation. Then, the boot loader function is loaded into memory, and BIOS provides system control to it once it is detected.

- **MBR**

  MBR is an acronym for Master Boot Record and is liable to load and run the GRUB boot loader. MBR is placed in the first bootable disk sector, which is generally /dev/sda, relying on our hardware. Also, the MBR includes details of GRUB, or LILO is an older system.

- **GRUB**

  GRUB is sometimes known as GNU GRUB, which stands for GNU GRand Unified Bootloader. It is the classic bootloader for almost all the latest Linux systems. The splash screen of GRUB is often the initial thing we see when we boot our system. It contains a general menu where we can choose some portions.

  We can use our keyboard to choose the one we wish our system to initiate with if we have multiple installed kernel images. The latest kernel image is chosen by default. The splash screen will delay for some seconds for us to choose options. It will load the kernel image (default) if we don't. In several systems, we can see the GRUB configuration file at /etc/grub/conf or /boot/grub/grub.conf.

- Kernel

  Often, the kernel is called the code of an operating system. It contained full control on everything in our system. In this boot process stage, the kernel mounts the base file system that was chosen that is set up in the file, i.e., grub.conf. Then, it runs the /sbin/init function, which is always the initial function to be run. We can confirm it with its PID (process id), which should be always 1. Then, the kernel creates a temporary base file system with the help of initrd (Initial RAM Disk) until the actual file system is mounted.

- Init

At this stage, our system runs runlevel programs. It would find an init file, generally detected at /etc/inittab, to determine the run level of Linux. Modern Linux systems utilize systemd to select a run level rather. Then, systemd will start running runlevel programs.

There are six run labels in the Linux operating system:

- 0-halt
- 1-single-user mode
- 2-multiuser, without NFS
- 3-Full multiuser mode
- 4-unused
- 5-X11
- 6-reboot

- Runlevel programs

  We can see distinct services getting started depending on which distribution of Linux we have installed. They are called runlevel programs and are run from distinct directories depending on our run level. All six runlevels mentioned above contain its directories:

  - Run level 0- /etc/rc0.d/
  - Run level 1- /etc/rc1.d/
  - Run level 2- /etc/rc2.d/
  - Run level 3- /etc/rc3.d/
  - Run level 4- /etc/rc4.d/
  - Run level 5- /etc/rc5.d/
  - Run level 6- /etc/rc6.d/

  If we look in distinct run level directories, we will find programs that begin with either a "K" or "S" for kill and startup, respectively. These start up programs are run at the time of the system startup and kill several programs at the time of shutdown.

## Uses of Linux OS

1. Web servers: W3Cook releases stats that utilize the top 1,000,000 Alexa domains, which estimate that 96.55% of web servers use Linux, 1.73% use Windows, and 1.72% use FreeBSD as of May 2015.
2. Laptops and desktops: As of May 2022, the estimated Linux market share is around 2.5% on desktop computers, according to web server statistics. Microsoft Windows include a market share of approximately 75.5%, while macOS has around 14.9%.

3. Mobile devices: Android has become the leading OS for smartphones which is Linux kernel-based. In July 2022, 71.9% of smartphones worldwide using the internet used Android. Also, Android is a famous OS for tablets, being liable for more than 60% of table sales as of 2013.
4. Film production: Linux has been the preferred platform in the film industry for years. The first big film released on a Linux server was 1997's Titanic. Since then, big studios, including Industrial Light & Magic, Weta Digital, Pixar, and DreamWorks Animation, have relocated to Linux.
5. Government use: Linux distros have also got popularity in several national and local governments. Kerala has gone to the mandating extent that every state high school use Linux on their systems. China utilizes Linux exclusively as the OS for its Loongson processor family for achieving technology independence.

## Linux OS Working

The Linux operating system follows of standard design that's the key to its several distributions and variations. Every Linux distribution is based on the Linux kernel but can differ based on factors like:

- **Kernel version:** Distros can be set up with more recent releases to add new aspects or with previous releases to be more balanced.
- **Kernel modules:** It is software that can be unloaded and loaded into the kernel develop functionality without restarting. Often, kernel modules are used for supporting:
    - Device drivers, which utilize code that manages how linked devices work.
    - File system drivers, which utilize code that manages how the kernel operates with distinct file systems.
    - System calls, which utilize code that manages how programs claim services through the kernel.
- **Configuration options:** Kernels unified with configuration options configured to add only file system or device drivers are used for a few specialized distributions; for instance, compiling the kernel for any wireless device without wired network device drivers.

  The kernel is the one thing that every system has in common running Linux. Linux operates by:
    - Booting and loading the Linux kernel.

    - The kernel handles every system output and input once booted. The system is booted, and processes can be initialized.
    - The system can be utilized for processes that contain commands interactively entered by the command line, network server functions, desktop applications, or any program or application as system processes are booted.
- The user experience can widely vary, relying on how the Linux system is being utilized while the kernel may almost be identical with some compilation differences and divergence for configuration. For example, a few use cases of Linux with distinct user experiences are:
    - **Desktop productivity** systems, like those utilized by software developers or several other professionals. The workstations of software development may be enhanced for performance, while desktops may be enhanced for the utilization of desktop productivity tools for administrative professionals.
    - **Network servers** might not even add a command line window for direct access. Remotely, these headless servers are handled by Windows sessions or network terminals. Servers may be utilized by several but should directly be accessed by authorized system admins only.
    - **Thin clients** let users utilize a rich desktop environment with a lightweight device. It includes Google Chromebooks and Raspberry Pi single-card systems.
- Linux operates much similar to any GUI-based operating system when using it as a GUI with a desktop environment. Applications and many other resources can be launched by pressing icons, and files can be deleted, copied, or moved using a trackpad or mouse.

## Pros and Cons of Linux OS

Some benefits of using Linux are listed and explained below:

- **Open source:** The Linux kernel is published under the open-source software license of GNU GPL. Most distributions contain several applications with various options in almost all categories. Also, several distributions contain proprietary software, like device drivers offered by manufacturers, to support hardware.
- **Reliability:** Linux is treated as a reliable operating system, and it is well-supported with several security patches. Also, Linux is treated as a stable OS,

which means it can execute in almost every circumstance. Linux can also handle errors when running unexpected input and software.

- **Licensing costs:** Linux has no accurate licensing fees, unlike Apple macOS or Microsoft Windows. While system support is present for a fee from several Linux vendors, the operating system itself is free to use and copy. A few IT organizations have enhanced their savings by moving their server software to Linux from a commercial operating system.
- **Backward compatibility:** Linux and many open-source software tend to be frequently updated for functional and security patches while having core functionality. Shell scripts and configurations are likely to operate unchanged even if software updates are used. Generally, Linux and other open-source applications do not alter their operation modes with new versions, unlike economic software vendors that mount new releases of their operating systems with new forms of work.
- **Several choices:** Between almost all infinite options, several available distros, and many application options to configure, compile, and run Linux on almost all hardware platforms, it's possible to develop Linux for almost all applications.

A few drawbacks of Linux are:

- **Lack of standard:** No standard version is available for Linux, which may be nice to optimize Linux for specific applications, but less so to deploy desktop images and standardized servers. The huge variety of options can convolute support as an outcome.
- **Support costs:** Support isn't free, while an organization can freely acquire Linux without licensing fees. Almost all enterprise Linux distributors, such as Red Hat and SUSE, provide support contracts. These license fees can significantly decrease savings depending on the situation.
- **Proprietary software:** PC productivity software, such as Microsoft Office, can't be utilized on Linux desktops, and many proprietary software may not be available for Linux platforms.
- **Steep learning curve:** Several users battle to learn to use Linux-based applications and Linux desktops.
- **Unsupported hardware:** Several hardware manufacturers enable the device drivers of Linux accessible for their products, but several don't.