LINUX INTRODUCTION

Linux is a Unix-like computer operating system assembled under the model of free and open-source software development and distribution. The defining component of Linux is the Linux kernel, an operating system kernel first released 5 October 1991 by Linus Torvalds. Linux was originally developed as a free operating system for Intel x86-based personal computers. It has since been ported to more computer hardware platforms than any other operating system. It is a leading operating system on servers and other big iron systems such as mainframe computers and supercomputers more than 90% of today's 500 fastest supercomputers run some variant of Linux, including the 10 fastest. Linux also runs on embedded systems (devices where the operating system is typically built into the firmware and highly tailored to the system) such as mobile phones, tablet computers, network routers, televisions and video game consoles; the Android system in wide use on mobile devices is built on the Linux kernel.

Basic Features:

Following are some of the important features of Linux Operating System.

- **Portable -** Portability means softwares can works on different types of hardwares in same way. Linux kernel and application programs supports their installation on any kind of hardware platform.
- **Open Source -** Linux source code is freely available and it is community-based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
- **Multi-User -** Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
- **Multiprogramming -** Linux is a multiprogramming system means multiple applications can run at same time.

- **Hierarchical File System -** Linux provides a standard file structure in which system files/ user files are arranged.
- **Shell -** Linux provides a special interpreter program which can be used to execute commands of the operating system. It can be used to do various types of operations, call application programs etc.
- **Security -** Linux provides user security using authentication features like password protection/ controlled access to specific files/ encryption of data.

Linux Advantages:

- **Low cost:** You don't need to spend time and money to obtain licenses since Linux and much of its software come with the GNU General Public License. You can start to work immediately without worrying that your software may stop working anytime because the free trial version expires. Additionally, there are large repositories from which you can freely download high quality software for almost any task you can think of.
- **Stability:** Linux doesn't need to be rebooted periodically to maintain performance levels. It doesn't freeze up or slow down over time due to memory leaks and such. Continuous uptimes of hundreds of days (up to a year or more) are not uncommon.
- **Performance:** Linux provides persistent high performance on workstations and on networks. It can handle unusually large numbers of users simultaneously, and can make old computers sufficiently responsive to be useful again.
- **Network friendliness:** Linux was developed by a group of programmers over the Internet and has therefore strong support for network functionality; client and server systems can be easily set up on any computer running

Linux. It can perform tasks such as network backups faster and more reliably than alternative systems.

- **Flexibility:** Linux can be used for high performance server applications, desktop applications, and embedded systems. You can save disk space by only installing the components needed for a particular use. You can restrict the use of specific computers by installing for example only selected office applications instead of the whole suite.
- **Compatibility:** It runs all common Unix software packages and can process all common file formats.
- **Choice:** The large number of Linux distributions gives you a choice. Each distribution is developed and supported by a different organization. You can pick the one you like best; the core functionalities are the same; most software runs on most distributions.
- **Fast and easy installation:** Most Linux distributions come with user-friendly installation and setup programs. Popular Linux distributions come with tools that make installation of additional software very user friendly as well.
- **Full use of hard disk:** Linux continues work well even when the hard disk is almost full.
- **Multitasking:** Linux is designed to do many things at the same time; e.g., a large printing job in the background won't slow down your other work.
- **Security:** Linux is one of the most secure operating systems. "Walls" and flexible file access permission systems prevent access by unwanted visitors or viruses. Linux users have to option to select and safely download software, free of charge, from online repositories containing thousands of high-quality packages. No purchase transactions requiring credit card numbers or other sensitive personal information are necessary.

- **Open Source:** If you develop software that requires knowledge or modification of the operating system code, Linux's source code is at your fingertips. Most Linux applications are Open Source as well.

Linux Distribution (Operating System) Names:

A few popular names:

1.Redhat Enterprise Linux

2.Fedora Linux

3.Debian Linux

4.Suse Enterprise Linux

5.Ubuntu Linux

Common Things Between Linux & UNIX:

Both share many common applications such as:

1.GUI, file, and windows managers (KDE, Gnome)

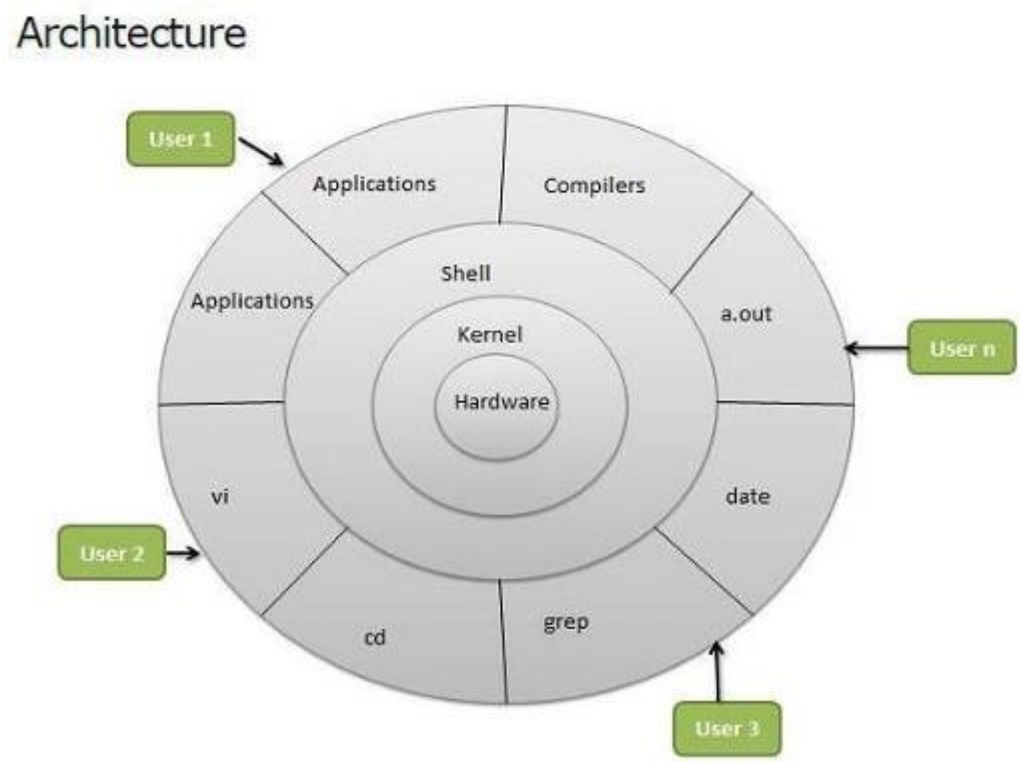
2.Shells (ksh, csh, bash)

3.Various office applications such as OpenOffice.org

4.Development tools (Perl, php, python, GNU c/c++ compilers)

5.Posix interface

Layered Architecture:



1. **Hardware:**

The lowest level of the Linux architecture is the hardware layer. This layer comprises the physical components of a computer, such as the hard drive, RAM, motherboard, CPU, network interfaces, and peripherals. These components are the tangible pieces of your system on which the rest of the architecture is built.

2. Kernel

Directly interfacing with the hardware layer is the kernel, the heart of the Linux operating system. As the core part of the OS, the kernel is responsible for low-level tasks such as disk management, task scheduling, memory management, and controlling peripherals.

The Linux kernel is a monolithic kernel, meaning it encompasses device drivers, file systems, system server calls, and more, all in a single static binary file. Because the kernel directly interacts with the system's hardware, it's crucial in terms of system performance and stability.

3. Shell

One layer up from the kernel is the shell. In simplest terms, the shell is a user interface that allows users to interact with the kernel. In Linux, most interactions with the shell occur in a command-line interface (CLI), where users type commands interpreted by the shell.

There are several different shells available in Linux, each with its unique features and syntax, such as the Bourne Again Shell (bash), the C Shell (csh), and the Z Shell (zsh).

4. Applications

The topmost layer of the Linux architecture consists of applications. These are the software programs that you, as the user, interact with directly. They range from system applications like file managers, text editors, and network managers, to user applications like browsers.

History of Linux:

In order to know the popularity of Linux, we need to travel back in time. In earlier days, computers were like a big house, even like the stadiums. So, there was a big problem of size and portability. Not enough, the worst thing about computers is every computer had a different operating system. Software was always customized to serve a specific purpose, and software for one given system didn't run on another system. Being able to work with one system didn't automatically mean that you could work with another. It was difficult, both for the users and the system administrators. Also, those computers were quiet expensive. Technologically the world was not quite that advanced, so they had to live with the size for another decade. In 1960, a team of developers in the Bell Labs laboratories started working on a solution for the software problem, to address these compatibility issues. They developed a new operating system, which was simple, elegant, written in C Programming language instead of Assembly language and most important is it can be able to recycle the code. The Bell Labs developers named this project as " UNIX ".

Unix was developed with small piece of code which is named as kernel. This kernel is the only piece of code that needs to be adapted for every specific system and forms the base of the UNIX system. The operating system and all other functions were built around this kernel and written in a higher programming language, C. This language was especially developed for creating the UNIX system. Using this new technique, it was much easier to develop an operating system that could run on many different types of hardware. So, this naturally affected the cost of Unix operating system, the vendors used to sell the software ten times than the original cost. The source code of Unix, once taught in universities courtesy of Bell Labs, was not published publicly. So, developers tried to find out some solution to provide an efficient solution to this problem.

A solution seemed to appear in form of MINIX. It was written from scratch by Andrew S. Tanenbaum, a US-born Dutch professor who wanted to teach his students the inner workings of a real operating system. It was designed to run on the Intel 8086 microprocessors that had flooded the world market.

As an operating system, MINIX was not a superb one. But it had the advantage that the source code was available. Anyone who happened to get the book 'Operating Systems: Design and Implementation' by Tanenbaum could get hold of the 12,000 lines of code, written in C and assembly language. For the first time, an aspiring programmer or hacker could read the source codes of the operating system, which to that time the software vendors had guarded vigorously. A superb author, Tanenbaum captivated the brightest minds of computer science with the elaborate lively discussion of the art of creating a working operating system. Students of Computer Science all over the world worked hard over the book, reading through the codes to understand the very system that runs their computer.

And one of them was Linus Torvalds. Linus Torvalds was the second-year student of Computer Science at the University of Helsinki and a self-taught hacker. MINIX was good, but still it was simply an operating system for the students, designed as a teaching tool rather than an industry strength one. At that time, programmers worldwide were greatly inspired by the GNU project by Richard Stallman, a software movement to provide free and quality software. In the world of Computers, Stallman started his awesome career in the famous Artificial Intelligence Laboratory at MIT, and during the mid and late seventies, created the Emacs editor.

In the early eighties, commercial software companies lured away much of the brilliant programmers of the AI lab, and negotiated stringent nondisclosure agreements to protect their secrets. But Stallman had a different vision. His idea was that unlike other products, software should be free from restrictions against copying or modification in order to make better and efficient computer programs. With his famous 1983 manifesto that declared the beginnings of the GNU project, he started a movement to create and distribute softwares that conveyed his philosophy (Incidentally, the name GNU is a recursive acronym which actually stands for 'GNU is Not Unix'). But to achieve this dream of ultimately creating a free operating system, he needed to create the tools first. So, beginning in 1984, Stallman started writing the GNU C Compiler (GCC), an amazing feat for an individual programmer. With his smart technical skills, he alone outclassed entire groups of programmers from commercial software vendors in creating GCC, considered as one of the most efficient and robust compilers ever created.

Linus himself didn't believe that his creation was going to be big enough to change computing forever. Linux version 0.01 was released by mid-September 1991, and was put on the net. Enthusiasm gathered around this new kid on the block, and codes were downloaded, tested, tweaked, and returned to Linus. 0.02 came on October 5th.

Importance Of Linux:

Linux is significant for several reasons, and its importance extends across various domains of computing. Here are key reasons why Linux matters:

- **Open-Source Philosophy:** Linux is built on the open-source philosophy, meaning its source code is freely available for anyone to view, modify, and distribute. This fosters collaboration, innovation, and a sense of community among developers.

- **Stability and Reliability:** Linux is known for its stability and reliability, making it a preferred choice for critical systems such as servers, where up-time is crucial. Many servers and data centers worldwide run on Linux, providing a robust foundation for hosting websites, applications, and services.

- **Security:** Linux's security model is based on strong user permissions, ensuring that users have only the necessary access to system resources. Regular updates and a vigilant open-source community contribute to its reputation as a secure operating system.

- **Customization and Flexibility:** Linux offers a high degree of customization and flexibility. Users can choose from a variety of desktop environments and customize their system according to their preferences. This adaptability makes Linux suitable for a wide range of use cases.

- **Cost-Effective:** Linux is cost-effective as it is free to use and distribute. This is particularly important for businesses and organizations looking to reduce licensing costs associated with proprietary operating systems.

- **Variety of Distributions:** There is a wide range of Linux distributions, each tailored to specific needs. Whether for general desktop use, server environments, embedded systems, or specialized applications, there's likely a Linux distribution suited to the task.

- **Community and Collaboration:** The Linux community is a global network of developers and users who contribute to the continuous improvement of the operating system. This collaborative model promotes knowledge sharing, rapid development, and collective problem-solving.

- **Educational and Learning Opportunities:** Linux provides a valuable learning platform for individuals interested in computer science and system administration. Access to the source code and the ability to experiment with the system make it an excellent educational tool.

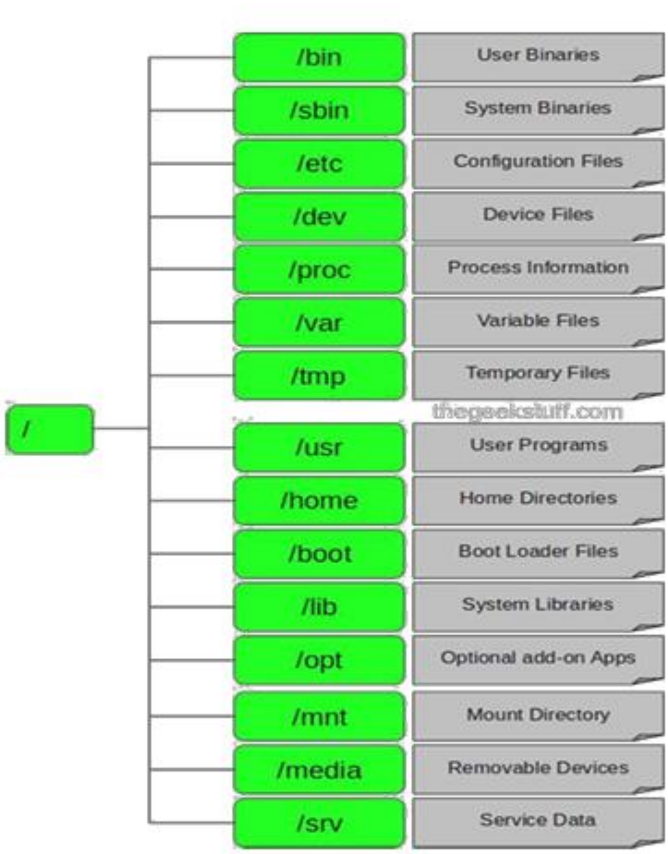
- **Internet and Cloud Infrastructure:** Linux plays a pivotal role in the infrastructure of the internet and cloud computing. Many web servers, cloud services, and networking devices rely on Linux for its performance, reliability, and scalability.

- **Influence on Technology Trends:** Linux has influenced and powered various technology trends, including the rise of open-source software, containerization (e.g., Docker), and the development of the Android operating system for mobile devices.

- **Support for Emerging Technologies:** Linux is at the forefront of supporting emerging technologies such as artificial intelligence, machine learning, and the Internet of Things (IoT). Its adaptability makes it a suitable choice for diverse and evolving technological landscapes.

LINUX File system:

Linux file structure files are grouped according to purpose. Ex: commands, data files, documentation. Parts of a Unix directory tree are listed below. All directories are grouped under the root entry "/". That part of the directory tree is left out of the below diagram.



❖ / – Root:

- Every single file and directory start from the root directory.
- Only root user has write privilege under this directory.

- Please note that /root is root user's home directory, which is not same as /.

❖ /bin – User Binaries

- Contains binary executables.
- Common Linux commands you need to use in single-user modes are located under this directory.
- Commands used by all the users of the system are located here.
- For example: ps, ls, ping, grep, cp.

❖ /sbin – System Binaries

- Just like /bin, /sbin also contains binary executables.
- But, the Linux commands located under this directory are used typically by system administrator, for system maintenance purpose.
- For example: iptables, reboot, fdisk, ifconfig, swapon

❖ /etc – Configuration Files

- Contains configuration files required by all programs.
- This also contains startup and shutdown shell scripts used to start/stop individual programs.
- For example: /etc/resolv.conf, /etc/logrotate.conf

❖ /dev – Device Files

- Contains device files.
- These include terminal devices, usb, or any device attached to the system.
- For example: /dev/tty1, /dev/usbmon0

❖ /proc – Process Information

- Contains information about system process.
- This is a pseudo filesystem contains information about running process. For example: /proc/{pid} directory contains information about the process with that particular pid.
- This is a virtual filesystem with text information about system resources. For example: /proc/uptime.

❖ /var – Variable Files

- var stands for variable files.
- Content of the files that are expected to grow can be found under this directory.
- This includes — system log files (/var/log); packages and database files (/var/lib); emails (/var/mail); print queues (/var/spool); lock files (/var/lock); temp files needed across reboots (/var/tmp);

❖ /tmp – Temporary Files

- Directory that contains temporary files created by system and users.
- Files under this directory are deleted when system is rebooted.

❖ /usr – User Programs

- Contains binaries, libraries, documentation, and source-code for second level programs.
- /usr/bin contains binary files for user programs. If you can’t find a user binary under /bin, look under /usr/bin. For example: at, awk, cc, less, scp
- /usr/sbin contains binary files for system administrators. If you can’t find a system binary under /sbin, look under /usr/sbin. For example: atd, cron, sshd, useradd, userdel
- /usr/lib contains libraries for /usr/bin and /usr/sbin
- /usr/local contains users programs that you install from source. For example, when you install apache from source, it goes under /usr/local/apache2

❖ /home – Home Directories

- Home directories for all users to store their personal files.
- For example: /home/john, /home/nikita

❖ /boot – Boot Loader Files

- Contains boot loader related files.
- Kernel initrd, vmlinux, grub files are located under /boot
- For example: initrd.img-2.6.32-24-generic, vmlinuz-2.6.32-24-generic

❖ /lib – System Libraries

- Contains library files that supports the binaries located under /bin and /sbin
- Library filenames are either ld* or lib*.so.*
- For example: ld-2.11.1.so, libncurses.so.5.7

❖ /opt – Optional add-on Applications

- opt stands for optional.
- Contains add-on applications from individual vendors.
- add-on applications should be installed under either /opt/ or /opt/ sub-directory.

❖ /mnt – Mount Directory

- Temporary mount directory where sysadmins can mount filesystems.

❖ /media – Removable Media Devices

- Temporary mount directory for removable devices.
- For examples, /media/cdrom for CD-ROM; /media/floppy for floppy drives; /media/cdrecorder for CD writer

❖ /srv – Service Data

- srv stands for service.
- Contains server specific services related data.
- For example, /srv/cvs contains CVS related data.