Linux File System

Introduction:-

Files reside on physical storage devices such as hard drives, CD-ROMs, or floppy disks.

The files on each storage device are organized into a file system. A **file** is the basic component for data storage. Hard drives can be divided into separate storage devices called partitions, each of which has its own file system. You can perform administrative tasks on your file systems, such as backing them up, attaching or detaching them from your file structure, formatting new devices or erasing old ones, and checking a file system for problems.

A ***file system*** is a set of data structures that exist in on a disk and directories of files.

File systems store user and system data. *File system used to store all of the information and data. i.e. system kernel file, the executable files for the commands supported by the operating system, configuration information files, temporary work files, user data, and various special files that are used to give controlled access to system hardware and operating system functions.*

*A* ***file system*** *is the organizing files on storage devices, such as hard disks and CDs or DVDs.*

 **Physical file system** is a part of the hard disk to hold files. The section of the disk that

holds a file system is called a *partition*.

 Linux consists of *multiple file systems that form virtual storage space for multiple users*.

**Virtual storage** can be allocated using different disks or file systems, but that is clearly

accessible as storage to the operating system and users.

Types of File system:- (File types)

*Linux file system consists of following types*:

**Ordinary files or Regular Files:-**

*Ordinary files can contain text, data, or program information*. *Files cannot contain other files or directories.* File name up to 256 characters long. The special characters (\*? #, &) of shell cannot be used in filenames. It contains only data as a stream of characters.

Ordinary files are of two types –

**Text file:-** It contains only printable characters. i.e. A text files contains lines of characters.

For e.g. All java, C program sources, shell and Perl scripts are text files.

**Binary File:- it** Contains both printable and unprintable characters that cover the entire ASCII

range. Most Linux commands are binary files e.g. Picture, sound and video files are

binary files.

**Device (Special) files:-**

Device or special files are contains information of I/O device and also to provide easy access

to hardware devices. All the operations on the devices are performed by reading or writing the file representing the device. They appear in a file system just like an ordinary file or a directory.

There are two types of devices files in Linux.

***Block- device file:-*** data is transferred in large fixed-size blocks. Handling blocks of data,

including CD/DVD drives, hard disk drives, tape drives, and other storage devices.

**Character device file:-** Data transferred byte-by-byte streams of data i.e. data is transferred

one character at a time, such as through serial or universal serial bus (USB) connections,

including terminals, printers, and network communications.

**Link files:-**

A link is a tool used for having multiple filenames that reference a single file on a physical disk i.e pointer to another file. There are two types of links –

**Hard link** - Hard links don’t actually link to the original file it create their own copy of the

original file's attributes (i.e. location on disk, file access permissions, etc.). If the original file

is deleted, its data can still be accessed using the hard link.

**Soft link (symbolic link)** - provides an indirect pointer or shortcut to a file. Reference to

original file it creates only inked file. If the original file is deleted, its data cannot be

accessed using the symbolic link.

**Named Pipe files:-**

Named pipes are tools that allow two or more system processes to communicate with each other using a file that acts as a pipe between them. This type of communication is known as interprocess communication or IPC for short.

**FIFO Files:-**

FIFO stands for **First In First Out**. These are special files used to move data from one running process on the system to another. A FIFO file is basically a queue where the first chunk of data added to the queue is the first chunk of data removed from the queue. Data can only move in one direction through a FIFO.

**Socket files:-**

Sockets are also tools used for interprocess communication. The difference between sockets and pipes is that sockets will facilitate communication between processes running on different systems, or over the network. Sockets are similar to FIFOs in that they are used to transfer information between sockets. With a socket, however, data can move bi-directionally.

Hierarchy of File system :-

Linux uses the File system Hierarchy Standard (FHS) file system structure, which defines the names, locations, and permissions for many file types and directories.

All the directories in the Linux system comes under the root directory which is represented by a **forward** slash **(/)**. Everything in your system can be found under this root directory even if they are stored in different virtual or physical devices.

A directory is a special kind of file that can contain other files and directories. Regular files

store information, *while a directory stores the names of regular files and other directories, which are called subdirectories.* The subdirectory is called as child of the parent directory because the child directory is created within the parent directory. In Figure the root system directory (/) is the parent of all the other directories, such as /bin, /boot, /dev, /etc, /home, and so on.

*Linux systems are so versatile is that they support many different file systems. In Linux, the*

*native file system is called the extended file system (ext or extfs), which is installed by default.*

The main reason behind the creation of hierarchical directories structure is

To keep related files collectively and separate them from other group of related files.

 File system can be defined as structure and organization of data on a data storage device.

 It controls format of hard disk or floppy disk when formatting process is carried on storage

device.

 File system decides how files and directories are stored and organized on your storage

media.

 Directories and large numbers of files can be search quickly.

***Feature of Linux File System***

Linux file system was cross developed by using UNIX and minix system. So Linux uses minix’s

file system.

 Linux kernel to allow accessing different file system by loading a kernel module.

 Linux File System is used to store file

 Speed of access to file’s contents

Efficiency of disk storage utilization

***Some Common File System in Linux***

 ***Ext2:*** This is like UNIX file system. It has the concepts of blocks, inodes and directories. File

size can be from 16 GB to 2 TB. Ext2 does not have journaling feature. On flash drives, usb drives, ext2 is recommended, as it doesn’t need to do the over head of journaling.

**Ext3:** It is ext2 file system enhanced with journaling capabilities. Journaling allows fast file

system recovery, where all the changes are tracked. Maximum individual file size can be

from 16 GB to 2 TB.

**Ext4:** Supports huge individual file size and overall file system size. Several other new

features are introduced in ext4: multiblock allocation, delayed allocation, journal checksum.

fast fsck, etc. All you need to know is that these new features have improved the

performance and reliability of the filesystem when compared to ext3. Maximum individual

file size can be from 16 GB to 16 TB.

***NFS:*** Network file system allows many users or systems to share the same files by using a

client/server methodology. NFS allows sharing all of the above file system.

Linux also supports Microsoft NTFS, vfat, and many other file systems. See Linux kernel

source tree Documentation/file system directory for list of all supported file system.



***The /root Directory***

The root (/) file system is mounted by the kernel when the system starts. The root

directory contains all essential programs for file system repair: restoring from a backup,

starting the system, and initializing all devices and operating resources. It also contains the

information for mounting all other file systems.

***The /bin directory***

The /bin directory contains user binaries, or executables, which are the programs

needed to start the system and perform other essential system tasks. This directory holds

many programs that all users need to work with Linux.

***The /boot Directory***

The /boot directory contains the files needed by the bootstrap loader; it also contains

the kernel images.

***The /dev Directory***

The /dev directory contains information of system devices. They access system

devices and resources, such as the hard disks, mice, printers, consoles, modems, memory,

and CD/DVD drives. Linux devices are managed through the use of device special files,

which contain information about I/O devices that are used by the operating system kernel

when a device is accessed. Two types of device special files exist:

**Block device files** are used to manage random access devices that involve

handling blocks of data, including CD/DVD drives, hard disk drives, tape drives,

and other storage devices.

**Character device files** handle byte-by-byte streams of data, such as through serial

or universal serial bus (USB) connections, including terminals, printers, and

network communications. It is used to interface mice, keyboards, monitors,

digital sound cards, disk drives, and other external computer hardware, such as

printers and digital cameras.

***The /etc Directory***

The /etc directory contains binary executable files & configuration files for system

uses when the computer starts. Directory is reserved for the system administrator, and it

contains system-critical information stored in the following files:

**fstab**—The mapping information about file systems to devices (such as hard disks

and CDs/DVDs)

***The /home Directory***

The /home directory is often on the /home partition and is used to offer disk space

for users, such as on a system that has multiple user accounts. For example: the

/home/student

***The /lib Directory***

Library directory contain all kinds of programs needed by the system and the users

i.e. kernel modules, security information, and the shared library images, which are files that programmers generally use to share code in the libraries rather than creating copies of this

code in their programs.

***The /mnt Directory***

Mount points for temporary mounts by the system administrator reside in the /mnt

directory. A temporary mount is used to mount a removable storage medium, such as a

CD/DVD or USB/flash storage so that it can be easily unmounted for quick removal.

***The /media Directory***

Mount points for removable storage are in the /media directory.Modern Linux

distributions include both /mnt and /media directories, but automated software to detect

insertion of a CD/DVD typically uses /media. Linux users and programmers are often

encouraged to use /media instead of /mnt as a way to follow the newer FHS

recommendation.

***The /proc Directory***

The /proc directory occupies no space on the disk; it is a virtual file system allocated in

memory only. Files in /proc refer to various processes running on the system as well as details

about the operating system kernel.

***The /sbin Directory***

The /sbin directory is reserved for the system administrator. Programs that start the

system, programs needed for file system repair, and essential network programs are stored

here.

***The /tmp Directory***

Many programs need a temporary place to store data during processing time. A program

may create several temporary data files as it processes. The temporary files might hold data

briefly needed for program. When system shutdown that time temp files automatically

remove.

***The /usr Directory***

This directory is for all user-related programs. The software may be accounting

programs, manufacturing programs, programs for research applications, or office software.

***The /var Directory***

The /var directory holds subdirectories that often change in size. These subdirectories

contain files such as error logs and other system performance logs that are useful to the system

administrator.

**File System parts**

The file system parts contains Boot Block, Super Block, Inode Block, Data Block

A **physical drive** can be divided into several **partitions;** each partition can contain one **filesystem** Each file system contains:

 Boot block(s)

 Superblock

 Inode block

 Data blocks

Partition is divided into data blocks with of fixed size which are used to store content of file. If file is of greater size than of block then it is stored in multiple blocks.





 **Boot Block**

The boot block contains the initial bootstrap program used to load the operating system. Typically, the first sector contains a bootstrap program that reads in a larger bootstrap program from the next few sectors.

 **Super Block**

It contains information about the layout of blocks on a specific partition. This information *is the*

*key to finding anything on the file system, and it should never change. Without the superblock, the file system cannot be accessed*. For this reason, many copies of the superblock are written into the file system at the time the file system is created through partitioning and formatting. If the superblock is destroyed, you can copy one of the superblock copies over the original, damaged superblock to restore access to the file system. Each filesystem has one super block (+ duplicate super block) it contains information about

1. Type of filesystem (ext2, ext3)

2. The total size of the partition

2. The block size

3. Pointers to a list of free blocks

4. The inode number of the root directory

5. Magic number

 **Inode Block**

Every file has one inode number. In Linux every file is recognized with integer number known as

inode number. This structure consists of information of file about

1. File ownership

2. File type (e.g., regular, directory, special device, pipes, etc.)

3. File access permissions. May have setuid (sticky) bit set.

4. Time of last access, and modification

5. Number of links (aliases) to the file

6. Pointers to the data blocks for the file

7. Size of the file in bytes (for regular files)

8. Major and minor device numbers for special devices.

Inodes include pointers to the data blocks. Each inode contains 15 pointers:

 The first 12 pointers point directly to data blocks

 The 13th pointer points to an indirect block, a block containing pointers to data blocks

 The 14th pointer points to a doubly-indirect block, a block containing 128 addresses

of singly indirect blocks

 The 15th pointer points to a triply indirect block (which contains pointers to doubly

indirect blocks, etc.)

 **Data Blocks**

It containing the file data and administrative data, a block is fixed in size. The kernel can only

access the data blocks via the information held in the inode of that file. Data block store actual

user data (contents of file). All allocated can belongs to one and only one file in the file system.

Devices and Drives in Linux

Each piece of hardware that is connected to a computer and it requires software to control it. *A device driver is a software interface between system calls and hardware device. This software is known as a* ***Device Driver****. A driver drives, manages, controls, directs and monitors the entity under its command.* Device drivers—they are used to access system devices and resources, such as keyboards, displays, disks, mice, printers, scanners, and keyboards. They also include devices on the system board such as timers, and graphics chips, and audio chips.

For example, A “pilot” could be a person or an automatic system monitored by a person (an autopilot system in airliners). Similarly, a specific piece of hardware could be controlled by a piece of software (a device driver), or could be controlled by another hardware device, which in turn could be managed by a software device driver. A device driver communicates with the **device controller** to control the device itself. A device controller is the hardware that controls the operation of the device. Device drivers implement mechanism, not policy. A **mechanism** specifies the ways of interacting with the device. The device driver has only to implement the operations needed to interface with the device.

In Linux, every device needs to have device driver so that it can be made accessible by the

OS to the users. The device driver actually provides the standard input/output calls to access the device and interact with it.

A device driver provides the following features:

 A set of routines that communicate with a hardware device and provide a uniform interface

to the operating system kernel.

 A self-contained component that can be added to, or removed from, the operating system

dynamically.

 Management of data flow and control between user programs and a peripheral device.

 A user-defined section of the kernel that allows a program or a peripheral device to appear

as a `` /dev '' device to the rest of the system's software.

**Categories of I/O Devices & Drivers**

There are four broad categories of I/O:

 block devices

 character devices

 file systems

 network (sockets)

 **Block Devices**

Block Devices are devices, as the name suggests, deal with blocks of data. Therefore, block device drivers are implemented to read/write block data and also supports random access data. However, since data is to be handled in blocks, therefore its implementation involves an extra intermediate buffer to store data between reads and writes. It has block device operations such as

– open()

– release()

– direct\_access()

– revalidate\_disk() etc. Examples: A floppy, hard disks.

 **Character Devices**

Character Devices are devices where data is a stream of bytes i.e. the input/output happens byte by byte. Therefore, the character device drivers would be in such a way to implement such kind of streamed data. The interactive operations (often called the file operations) which would be most vital to implement are:

– open() , close()

– read() , write() Examples: A terminal, Keyboard, etc

 **Network devices**

Network device are the ones which are used to transmit data from one machine to another. The network device driver’s implementation is entirely different to those of char devices and block devices. Here, they are exchange data in the form of data packets with another remote machine, using a standard protocol. They also deal with setting up ip addresses, configurations and modifying transmission parameters, traffic, etc.

Their I/O operations are specific to the data packet transmission protocol, as in, TCP/UDP.

Examples: NIC

 **File systems**

The file system determines how data is organized on a block device in order to present higher-level software with a hierarchy of directories and files, with access permissions for both.

A file system is *not* a device driver. It is a software driver within the operating system that maps

between low-level and high-level data structures.

**Mounting devices**

*Files reside on physical storage devices such as hard drives, CD-ROMs, or floppy disks. All the drives and devices in your computer are one big filesystem.* The files on each storage device are organized into a file system. When you want to add a new storage device, you need to format it as a file system and then attach it to your Linux file structure. Hard drives can be divided into separate storage devices called *partitions*, each of has its own file system. You can perform administrative tasks on your file systems, such as backing them up, attaching or detaching them from your file structure, formatting new devices or erasing old ones, and checking a file system for problems.

*To attach these devices and drives to the filesystem for access files on a device, you attach*

*its file system to a specified directory. This is called mounting the file system of device*

For example, to access files on a floppy disk, you first mount its file system to a particular

directory. With Linux, you can mount a number of different types of file systems. You can even

access a Windows hard drive partition or tape drive, as well as file systems on a remote server.

*So that you can access them, you have to use the mount and umount commands. Some devices are automatically mounted when you boot up your computer. These are listed in the /etc/fstab file.*

**The mount Command**

The mount command takes two arguments: the *storage device* through which Linux accesses the file system and *mount point* is the directory on your main directory tree where you want the on storage device attached. If the directory does not yet exist, you have to create it. The device is a special device file that connects your system to the hardware device.

***# mount device mount point***

***# umount device or mount point***

File system can be mounted on hard disk partition only by root user. Where CD-ROM, floppies and any other devices can be mounted by any user.

***Mounting File System***

Mounting a filesystem simply means making the particular filesystem accessible at a certain

point in the Linux directory tree. When mounting a filesystem it does not matter if the filesystem is a hard disk partition, CD-ROM, floppy, or USB storage device. You simply need to know the device name associated with the particular storage device and a directory you would like to mount it to. When mounting a particular filesystem or device you need to know the special device file associated with it. A device file is a special file in Unix/Linux operating systems that are used to allow programs and the user to communicate directly with the various partitions and devices on your computer. These device files are found in the */dev* folder.

**# mkdir /media/flashdisk**

This command will have now created a directory called */media/flashdisk.* The next step would be to mount the filesystem to that folder or mount point.

**# mount -t vfat /dev/sdc1 /media/flashdisk**

You have now mounted an msdos filesystem, which is indicated by the -t (type) option. The device is recognized by the */mnt/flashdisk* point. Now you can access *vfat* formatted disks as you would any other directory.

A file system can be "mounted" on your Linux system interactively or automatically at

startup. Then the file system is just as accessible as any other file system on your computer. The

file system is mounted to an empty directory on the main directory tree.

**unmount a filesystem**

When you are done using a particular filesystem, you should unmount. The command to unmount a filesystem is the umount command.When unmounting a filesystem you simply type umount followed by the mount point. For example:

**# umount /media/flashdisk**

**# umount /dev/sdc1**

***Mounting Floppy Disks***

Mounting floppy disks: To access a file on floppy disks. The disk first has to be mounted on

Linux file system. The device name for your floppy drive is ' fdo ' & it is located in the directory

/dev. The /dev/fd0 reference your floppy device. More than one floppy disks are connected it

represented fd1, fd2 , fd3 & so on. You can mount to any directory you want.

***# mount /dev/fd0 /mnt/floppy***

*Mounts the floppy disk in your floppy drive to the /media/floppy.*

If you want to replace one mounted file system with another. You have mounted floppy disks

and now you want to take it out & put in a new one. You must unmount those floppy disks before you can put in and mount the new one.

***# umount /dev/fd0 & # unmount /mnt/floppy***

***When you shutdown you system, any disk you have mounted is automatically unmounted. Linux file system type “ ext , ext2 , ext3 ”.***

***Formatting Floppy Disks***

Before it can be used to store data. i.e. floppy must be formatted.

***# mkfs -t ext3 /dev/fd0***

The [ -t ] option specifies that the ext2,ext3 is a file system builder to create a file system of

that type. Options are listed before the device name. The floppy is formatted you can mount it.

***Mounting CD/DVD***

To mount a CD/DVD, all you have to do is insert it into the drive. HAL (Hardware Abstraction Layer) will detect it and mount it. It automatically in the /disk directory. You will have to first decide on a directory to mount it to. The directory is created dynamically when a disk is inserted and detect when the disk is removed.

***# mount /dev/cddvd /mnt/cddvd***

***# mount /mnt/cddvd***

Since it mount directory root /mnt/cddvd is reserved cdrom files system on Linux. One or

most helpful ways to do created and managed on writable CD/DVD is use the loop drive.

***# mount -t iso9660 -o loop = /dev/loopo my.cdimage /mnt/mystuff***

It is for create a CD image file. i.e. brun

***Mounting for Hard Drive Partitions:***

To mount automatically using the /etc/f stab file. Linux partition hard disk you created during

installation are already automatically mounted for you. Mount command with the device name of the partition and the directory to which you want to mount it.

***# mount -t ext3 /dev/hda4 /mnt/mydata***

You can also a mount a window partition and the directory access files on it. As with a Linux

partition but you also have to specify the file system types as window. Use a [-t] option & type

vfat for window 95/98/me,msdos for MS-DOS. User mount window hard disk partition /dev/hda1 to Linux file structure at directory /mnt/window then for MS-DOS /mnt/dos

***# mount -t vfat /dev/hda1 /mnt/window***

***# mount -t vfat /dev/hda1 /mnt/window***

***Mounting USB drive***

After you plug in your USB device to your USB port, linux will add new block device into /dev/

directory. At this stage you are not able to use this device as the USB filesystem needs to be

mouted before you are able to retrieve any data. To find out what name your block device file

have you can run fdisk command:

# fdisk –l

You will get more information about USB drive.

**Device Boot Start End Blocks Id System**

/dev/sda1 1 7301 58645251 b W95 FAT32

$ mkdir -p /mnt/myusb

# mount /dev/sda1 /media/usb

**OR**

$ mount -t vfat -o rw, users /dev/sda1 /mnt/myusb

Filesystem Acess Permission Device name mountpoint

Type