

Understanding Linux File System & Linux Text Editors

A Presentation to Systems Administrators 26, 2024

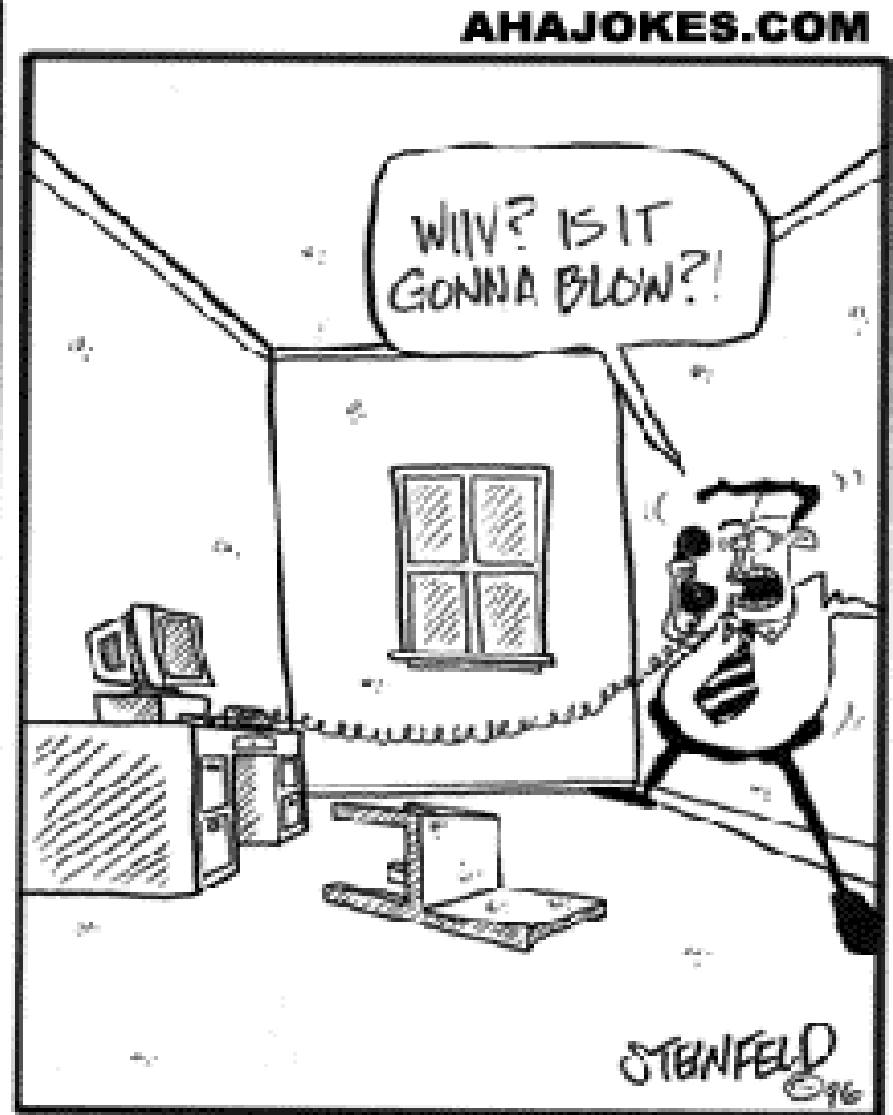
KENET

support@kenet.or.ke

Agenda

- Linux File System Hierarchy
- File System Types
- Directory Structure
- Special Directories & File Permissions
- File System Navigation Commands
- File Manipulation Commands
- File System Maintenance





Introduction

- A file system is a method used by operating systems and software to organize and manage data stored on a storage device, such as a hard drive, solid-state drive (SSD), or flash drive. It provides a hierarchical structure for storing, retrieving, and managing files and directories.
- File systems can vary significantly in terms of their features, performance characteristics, supported file sizes, and compatibility with different operating systems. Some common file systems include **NTFS** (New Technology File System) and **FAT32** (File Allocation Table) on Windows, HFS+ (Hierarchical File System Plus) on macOS, and ext4 (Fourth Extended Filesystem) on Linux.
- Overall, the file system serves as a foundation for storing, organizing, and managing data in an operating system, enabling users to work with files and applications effectively while maintaining data integrity and security.

Linux File System Hierarchy

The Linux File System (FS) follows a hierarchical structure known as the File System Hierarchy Standard (FHS). This standard defines the directory structure and organization of files in Linux-based operating systems. Here's an explanation of the key directories and their purposes within the Linux File System:

1. / (Root):

- The root directory is the top-level directory in the file system hierarchy.
- It serves as the starting point for the entire file system.
- All other directories and files are contained within the root directory.

2. /bin (Binary):

- Contains essential executable binaries (programs) needed for system booting and repair.
- Common utilities such as `ls` (list directory contents), `cp` (copy), `mv` (move), and `rm` (remove) reside here.

3. /boot:

- Contains boot loader files, kernel images, and other files required for system booting.
- Configuration files for boot loaders like GRUB are often located here.

Linux File System Hierarchy

4. /dev (Device):

- Contains device files that represent physical and virtual devices connected to the system.
- These files allow user programs and the kernel to communicate with hardware devices.

5. /etc (Editable Text Configuration):

- Contains system-wide configuration files for various applications and services.
- Configuration files for networking, user accounts, system services, and package management are typically stored here.

6. /home:

- Contains user home directories, each corresponding to a specific user account.
- Users typically store their personal files, documents, settings, and configurations in their respective home directories.

7. /lib (Library):

- Contains shared libraries required by programs and applications at runtime.
- These libraries provide essential functions and services used by multiple programs.

Linux File System Hierarchy

8. /mnt (Mount):

- Provides a directory for mounting external file systems or devices temporarily.
- External storage devices such as USB drives, external hard disks, and network shares can be mounted here.

9. /opt (Optional):

- Contains optional or third-party software packages installed on the system.
- Some software vendors may choose to install their applications under this directory.

10. /proc (Process):

- Contains virtual files that represent system and process information.
- Provides real-time information about system resources, processes, and kernel parameters.

11. /root:

- Home directory for the root user (superuser/administrator).
- Root user's personal files and configurations are stored here.

Linux File System Hierarchy

12. /sbin (System Binaries):

- Contains system administration binaries used for system maintenance and management.
- Commands like ifconfig (network interface configuration) and fdisk (disk partitioning) are located here.

13. /var (Variable):

- Contains variable data files that change frequently during system operation.
- Log files, spool directories, cache files, and temporary files generated by programs are stored here.

14. /tmp (Temporary):

- Provides a location for temporary files created by programs and users.
- Files in this directory are typically deleted upon system reboot.

16. /usr (User):

- Contains user-related programs, libraries, documentation, and other non-essential files.
- Commonly used applications, utilities, and libraries are stored here.

File System Types

Linux supports a variety of file system types, each with its own set of features, performance characteristics, and intended use cases. Each file system type has its own strengths and weaknesses, and the choice of file system depends on factors such as the intended use case, performance requirements, scalability needs, and compatibility with other systems.

1. ext4 (Fourth Extended File System):

- ext4 is the default file system for many Linux distributions.
- It is an extension of the earlier ext3 and ext2 file systems and offers improvements in performance, scalability, and reliability.
- Supports large file sizes (up to 16 TB) and partitions (up to 1 EB).
- Features journaling for improved reliability and faster file system checks.
- Supports features like extents (efficient allocation of disk space), delayed allocation (improved write performance), and fast fsck (file system check).

File System Types

2. **XFS (X File System):**

- XFS is a high-performance journaling file system developed by SGI (Silicon Graphics, Inc.).
- Designed for scalability and performance, particularly in large-scale enterprise environments.
- Supports very large file systems (up to 16 EB) and files (up to 8 EB).
- Features delayed allocation and extent-based allocation for improved performance.
- Provides features like online resizing, snapshots, and dynamic inode allocation.

3. **Btrfs (B-tree File System):**

- Btrfs is a modern copy-on-write (COW) file system developed by Oracle.
- Designed for scalability, reliability, and advanced features such as snapshotting, cloning, and RAID-like functionality.
- Supports features like checksums (for data integrity), compression, and deduplication.
- Offers support for subvolumes (independent file system trees), snapshots, and online volume resizing.

File System Types

4. **ZFS (Zettabyte File System):**

- While not native to the Linux kernel, ZFS is a highly advanced file system originally developed by Sun Microsystems (now owned by Oracle).
- Known for its robustness, data integrity features, and support for advanced storage technologies like RAID-Z (similar to RAID but with additional features).
- Supports features like copy-on-write, snapshots, compression, encryption, and deduplication.
- Requires installation via third-party packages on most Linux distributions due to licensing issues.

5. **F2FS (Flash-Friendly File System):**

- F2FS is a file system optimized for NAND flash memory storage devices like SSDs and eMMC.
- Designed to maximize performance and lifespan of flash-based storage devices.
- Features include wear leveling, TRIM support, and log-based transaction mechanism.
- Particularly suitable for use in devices like smartphones, tablets, and embedded systems.

File System Types

6. NTFS (New Technology File System):

- While not native to Linux, NTFS is a file system developed by Microsoft and is commonly used in Windows operating systems.
- Linux provides limited read and write support for NTFS partitions via the ntfs-3g driver.
- Useful for accessing files stored on Windows partitions from Linux systems.

File Permissions

In Linux, file permissions are a crucial aspect of the security model, governing who can access, modify, or execute files and directories. Permissions are assigned to three categories of users: the file owner, the group associated with the file, and others (everyone else). Here's an overview of Linux file permissions:

1. **Read (r):**

- Allows users to view the contents of a file or view the list of files within a directory.
- For directories, the read permission allows users to list the contents of the directory.

2. **Write (w):**

- Allows users to modify the contents of a file or create, rename, or delete files within a directory.
- For directories, the write permission allows users to create, delete, or rename files within the directory.

3. **Execute (x):**

- For files: Allows users to execute the file if it is a program or script.
- For directories: Allows users to access files and subdirectories within the directory.
- File permissions are represented using a symbolic notation or numeric notation.

File Permissions

➤ Symbolic Notation:

Each permission category (user, group, others) is represented by a combination of letters: **r** for read, **w** for write, and **x** for execute.

The permissions are displayed in the order: user, group, others.

An example permission string might be ***rw-r--r--***, which means the file owner has read and write permissions, while the group and others have only read permissions.

➤ Numeric Notation:

Each permission has a corresponding numeric value: ***read (4), write (2), and execute (1)***.

The sum of these values represents the permission set for each category.

For example, ***rw-r--r--*** in numeric notation would be represented as **644**, where the first digit (6) corresponds to the user's permissions, the second digit (4) to the group's permissions, and the third digit (4) to others' permissions.

File permissions can be viewed and modified using the **ls** command to display permissions and the **chmod** command to change permissions. Additionally, the **chown** and **chgrp** commands are used to change the owner and group of a file, respectively.

File System Navigation Commands

- In Linux, navigating the file system involves using various commands in the terminal to move between directories, list directory contents, create and remove directories, and perform other file system-related operations. Here are some of the most commonly used file system navigation commands:

1. **cd (Change Directory):**

- Used to change the current working directory.
- Syntax: ***cd [directory]***.

Examples: ***cd /*** - Changes to the root directory & ***cd /home/user*** - Changes to the ***"/home/user"*** directory.

2. **pwd (Print Working Directory):**

- Displays the current working directory.
- Syntax: ***pwd***.

Example: ***pwd*** - Displays the current directory path.

File Navigation Commands

3. ls (**List Directory Contents**):

- Lists the files and directories in the current directory.
- Syntax: ***ls [options] [directory]***.
- Common options:
 - l***: Long listing format.
 - a***: Include hidden files (those starting with a dot).
 - h***: Human-readable sizes.

4. find:

- Searches for files and directories matching specified criteria.
- Syntax: ***find [directory] [options] [criteria]***.

Example: ***find /home/user -name "*.txt"*** - Searches for all text files in the ***"/home/user"*** directory.

5. locate:

- Searches a pre-built database of files for those matching specified criteria.
- Syntax: ***locate [pattern]***.

Example: ***locate myfile.txt*** - Searches for all files named ***"myfile.txt"*** on the system.

File Manipulation Commands

1. rm (Remove):

- Deletes files or directories.
- Syntax: ***rm [options] [file/directory]***.
- Common options:
 - r***: Recursively delete directories and their contents.
 - f***: Force deletion without confirmation.

Example: ***rm file.txt*** - Deletes the file named "***file.txt***".

2. cp (Copy):

- Copies files or directories.
- Syntax: ***cp [options] [source] [destination]***.
- Common options:
 - r***: Copy directories recursively.
 - v***: Verbose mode (display detailed output).

Example: ***cp file.txt /path/to/destination*** - Copies "***file.txt***" to the specified destination.

File Manipulation Commands

3. mv (Move):

- Moves or renames files or directories.
- Syntax: ***mv [options] [source] [destination]***.
- Common options:
 - i: Prompt before overwriting existing files.

Example: ***mv file.txt new_location/*** - Moves "***file.txt***" to the "***new_location***" directory.

4. touch:

- The touch command is used to create empty files or update the access and modification timestamps of existing files.
- Syntax: ***touch [options] [file]***.

Example: ***touch newfile.txt*** - Creates a new empty file named "***newfile.txt***" in the current directory.

- If the specified file already exists, touch updates the access and modification timestamps to the current time by default. This can be useful for updating the timestamps of files without modifying their contents.
- ***touch*** can also be used to create multiple files simultaneously. For example, ***touch file1.txt file2.txt*** will create both ***file1.txt*** and ***file2.txt*** in the current directory.

File System Maintenance

File system maintenance is essential for ensuring the integrity, performance, and reliability of a system's storage. It involves various tasks aimed at optimizing, monitoring, and repairing the file system. Here are some common file system maintenance tasks in Linux:

1. Regular Backups:

- Regularly backing up important data is crucial for protecting against data loss due to hardware failures, accidental deletions, or system corruption.
- Use backup tools like `rsync`, `tar`, or specialized backup software to create and maintain backups of critical files and directories.

2. File System Checking (fsck):

- Use the ***fsck*** command to check and repair file system inconsistencies and errors.
- Running ***fsck*** can help identify and fix issues such as orphaned inodes, bad blocks, and corrupted metadata.
- Schedule periodic file system checks, especially after unexpected shutdowns or system crashes, using tools like `systemd` or `cron`.

File System Maintenance

3. Disk Usage Analysis:

- Monitor disk usage to identify files and directories consuming excessive space.
- Use utilities like `du` (disk usage) and `df` (disk free) to analyze disk space usage and identify large files or directories.
- Consider using graphical tools like `baobab` or `Disk Usage Analyzer` for a visual representation of disk usage.

4. Defragmentation (Optional):

- While modern Linux file systems like `ext4`, `XFS`, and `Btrfs` perform automatic fragmentation management, fragmentation may still occur over time.
- Use tools like `e4defrag` (for `ext4`), `xfs_fsr` (for `XFS`), or `btrfs filesystem defragment` (for `Btrfs`) to defragment file systems if needed.
- Note that `SSDs` do not require defragmentation and may even be negatively impacted by it due to wear leveling algorithms.

File System Maintenance

5. Partition and File System Resizing:

- Resize partitions and file systems as needed to accommodate changing storage requirements.
- Use tools like parted, fdisk, or gdisk for partitioning, and resize2fs (for ext2/3/4), xfs_growfs (for XFS), or btrfs filesystem resize (for Btrfs) for resizing file systems.

6. Monitoring and Performance Tuning:

- Monitor file system performance metrics like I/O throughput, disk latency, and file system utilization.
- Use monitoring tools like sar, iostat, or atop to analyze system performance and identify potential bottlenecks.
- Tune file system parameters such as journaling mode, mount options, and cache settings to optimize performance for specific workloads.
- Regular Updates and Patches:

Text editors are essential tools for creating, editing, and manipulating text files. In the Linux environment, several text editors are available, each with its own features, capabilities, and user interfaces. This presentation will explore some of the most popular text editors used in Linux and highlight their key features.

1. Vi and Vim

- Vi: A classic text editor available on most Unix-like operating systems.
- Vim (***Vi Improved***): An enhanced version of Vi with additional features and functionality.
- Known for its modal editing capabilities, extensive customization options, and powerful command-line interface.
- Vim supports syntax highlighting, multiple buffers, split windows, macros, and plugins.

2. Nano

- Nano: A simple and user-friendly text editor for beginners and casual users.
- Features a straightforward interface with basic navigation and editing commands.
- Supports syntax highlighting, search and replace, line numbering, and auto-indentation.
- Nano is often preferred for quick edits and simple text manipulation tasks.

3. Emacs

- Emacs: A highly extensible and customizable text editor with a built-in Lisp interpreter.
- Offers a wide range of features, including syntax highlighting, code completion, version control integration, and an extensive package ecosystem.
- Known for its powerful scripting capabilities and the ability to transform into a full-fledged Integrated Development Environment (IDE).
- Emacs users often tout its flexibility and productivity-enhancing features.

Conclusion

- In conclusion, the Linux file system is a vital component of the operating system, providing a hierarchical structure for organizing and managing data stored on storage devices. Throughout this presentation, we have explored various aspects of the Linux file system, including its structure, supported file system types, and important file system navigation commands.
- Overall, the Linux file system plays a critical role in facilitating data storage, organization, and access in Linux-based operating systems. By understanding its structure, supported file system types, and navigation commands, users can effectively manage their data and optimize the performance and reliability of their Linux systems.

What Does It Take?

*What do you think it takes to
understand linux file system?*

Ready or Not?

Do you believe you are ready?

Just Begin

*‘Chack Achaka’ or ‘**Just begin**’, the philosophy behind Dr. Okinyi Nyawade*

"A journey of a thousand miles begins with a single step" - Chinese Proverb



QUESTIONS **And** **Answers**

*Transforming education
through ICT*

Thank You

www.kenet.or.ke

Jomo Kenyatta Memorial
Library, University of Nairobi
P. O Box 30244-00100, Nairobi.
0732 150 500 / 0703 044 500