Inodes in Linux File Systems

Understanding the Core of File Management

A Student's Guide to Theory and Practice Command Line File Systems Data Structures Date: June 18, 2025

What is a File System?

- Manages how data is stored and retrieved on a storage device (e.g., hard drive, SSD).
- Provides a hierarchical structure (directories and files).
- Responsible for organizing files, managing metadata, and controlling access.

Common File Systems

Linux: ext4, XFS, Btrfs macOS: APFS, HFS+

Windows: NTFS, FAT32 Network: NFS, SMB

Introducing the Inode: The Unsung Hero

- Inode stands for Index Node.
- A fundamental data structure in Linux/Unix-like file systems.
- Stores metadata (information *about* a file or directory).
 - Crucial Point

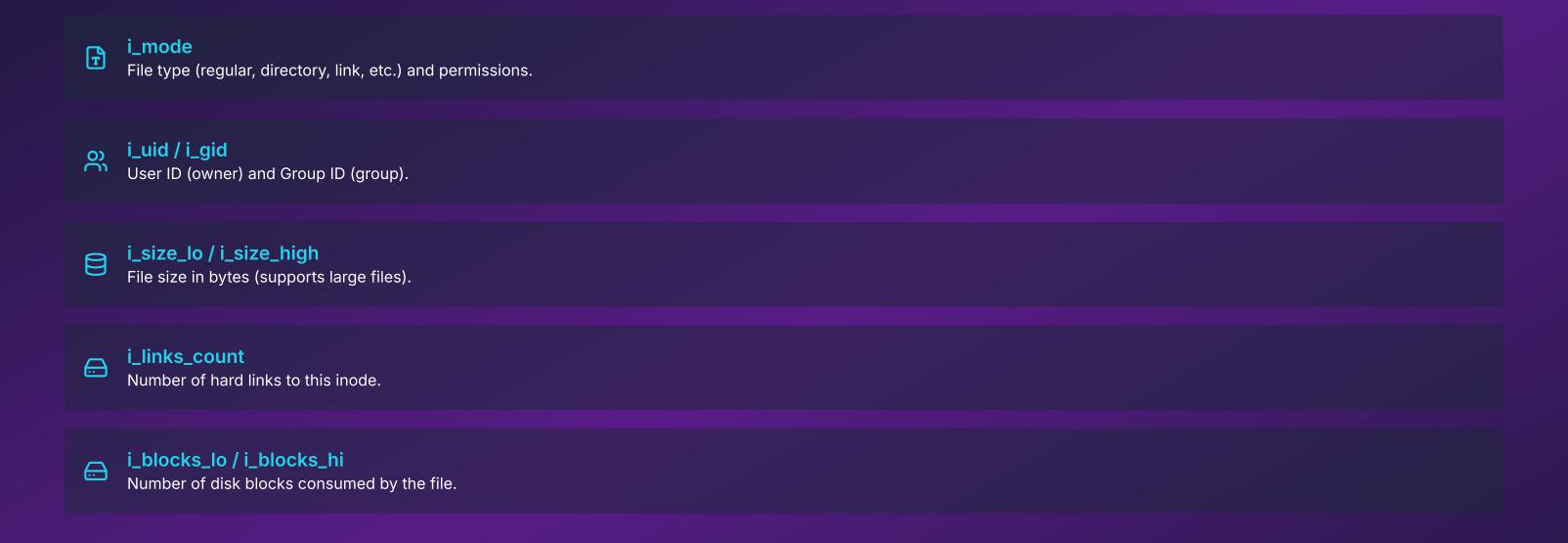
An inode does **not** store the file's name or its actual data content directly.

Key Functions of an Inode

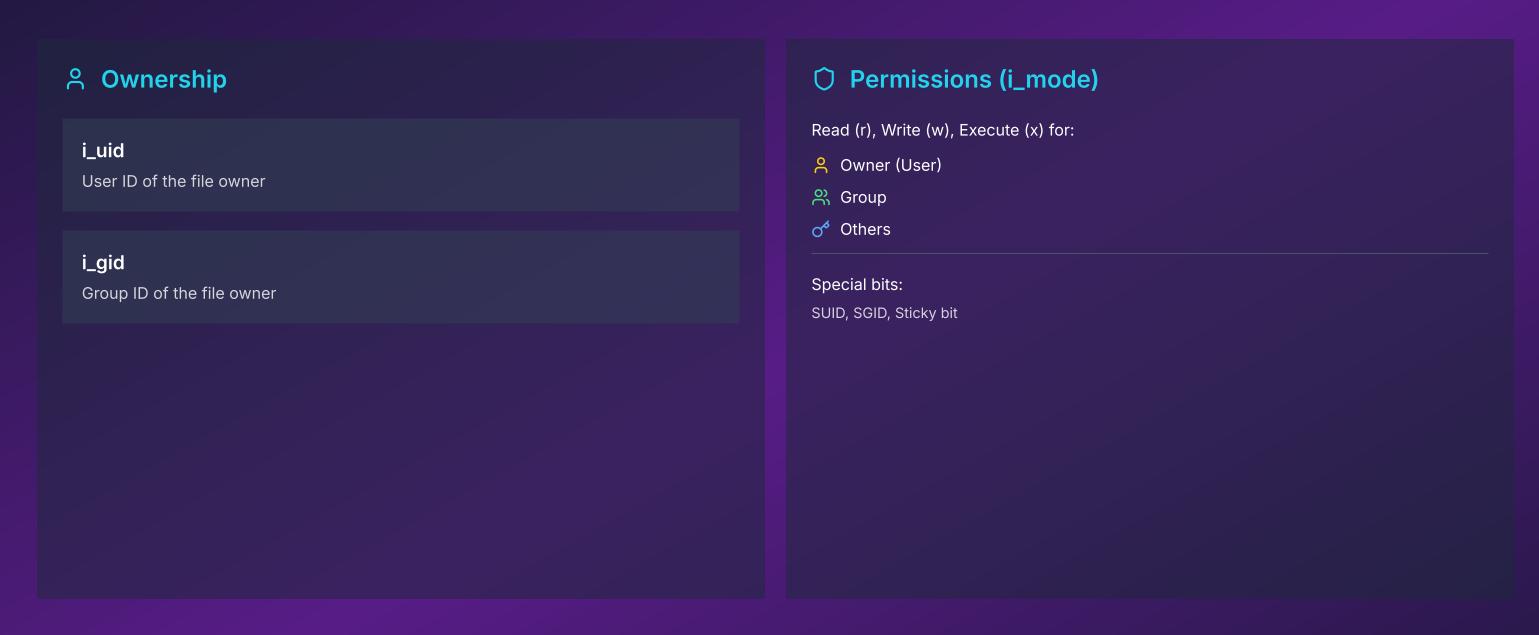
- Uniquely identifies a file or directory within its filesystem.
- Stores attributes: permissions, ownership, timestamps, size.

- Contains pointers to the data blocks where the file's content resides.
- Facilitates file operations: access, modification, deletion.

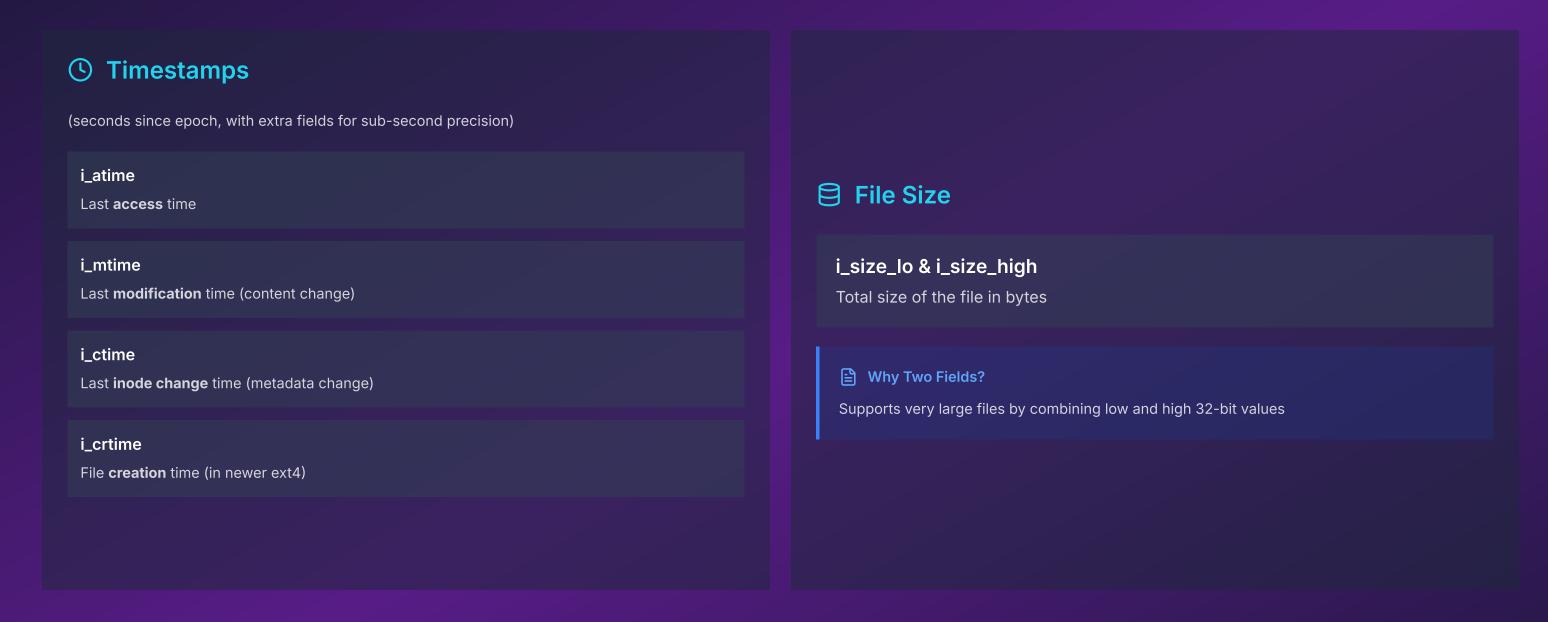
Anatomy of an Inode: Core Metadata Fields (ext4 example)



Inode Metadata: Ownership & Permissions (ext4 i_mode)

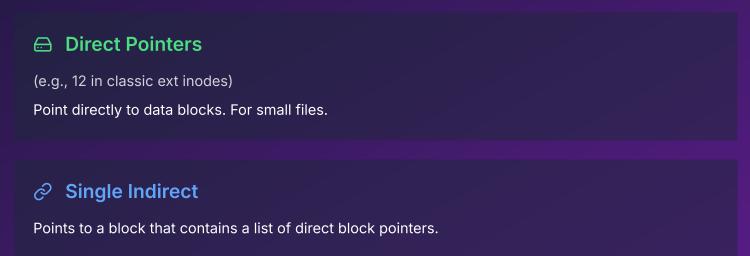


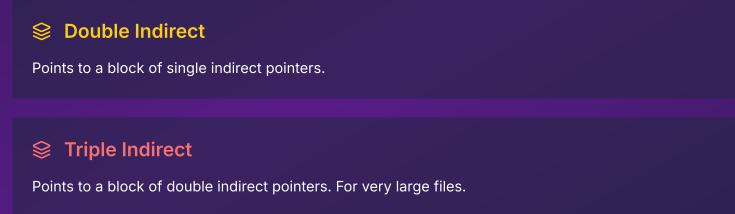
Inode Metadata: Timestamps & Size (ext4 example)



Inode Metadata: Data Block Pointers

Inodes store pointers to the actual data blocks on disk.





Modern Filesystems (ext4)

Often use **Extents**: a contiguous range of blocks, more efficient for large files. (Indicated by EXT4_EXTENTS_FL in i_flags).

Ext4 Inode Structure: Notable Fields

Controls file behavior (e.g., immutable, append-only, extents, inline data, encryption).

i_generation
File version number (useful for NFS).

i_file_acl_lo
Points to extended attribute block (e.g., for ACLs).

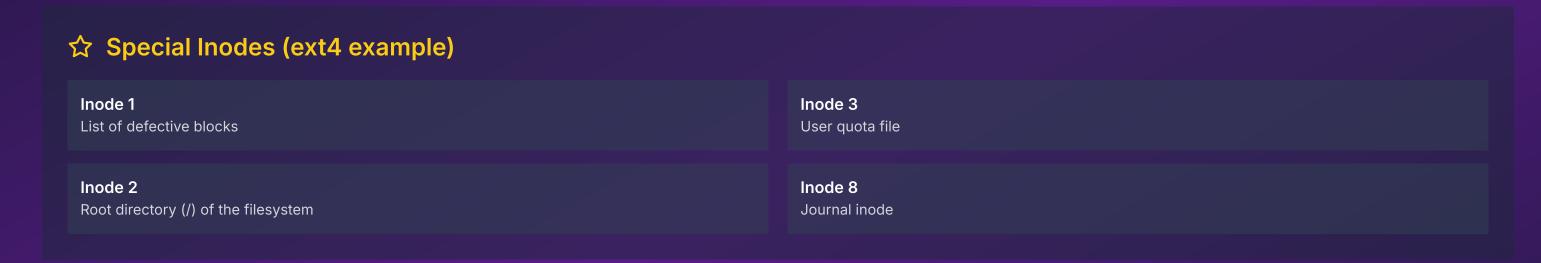
+ i_extra_isize
Size of extended inode fields beyond the original ext2 inode (default 256 bytes for ext4 inode, base was 128).

i_checksum_hi / l_i_checksum_lo

Inode checksum for integrity.

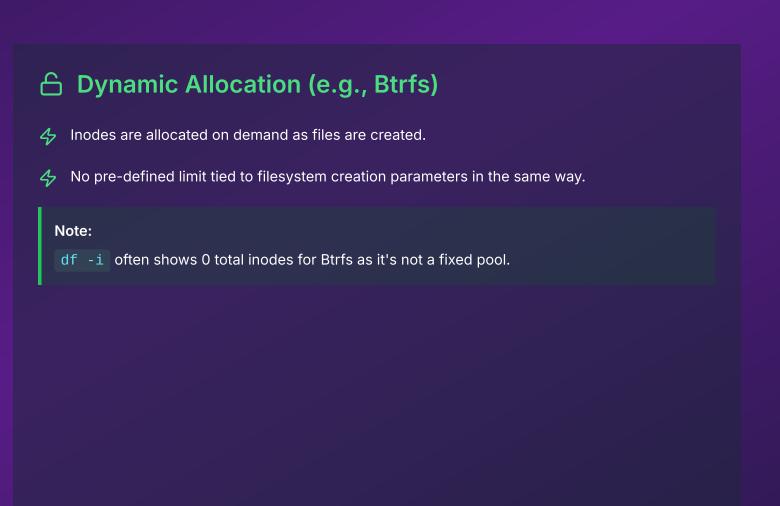
Inode Numbers: Unique Identifiers

- Each file and directory is assigned a unique **inode number** within its filesystem.
- The OS uses this number to locate the inode and its metadata.
- Inode 0 is undefined.



Inode Allocation: When & How?

☆ Fixed Allocation (e.g., ext4) A fixed number of inodes is allocated when the filesystem is created. Often based on a ratio (e.g., 1 inode per 16 KB of disk space). Limitation: This limits the maximum number of files, regardless of free disk space.



Inode Tables: Organization (ext4 example)

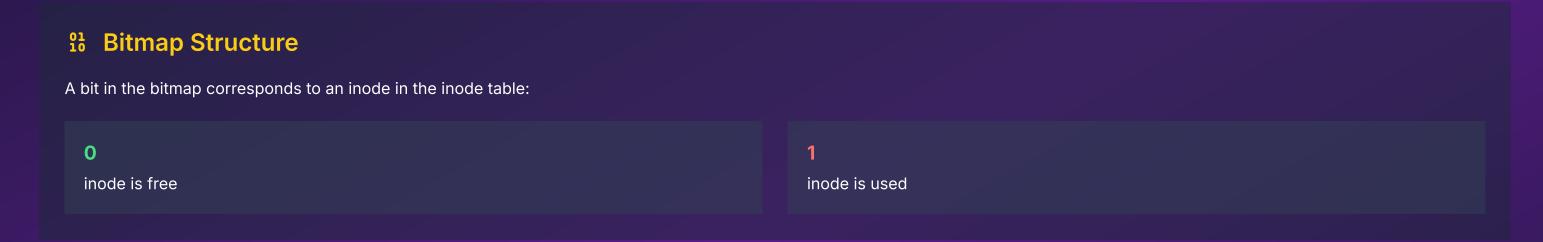
- In ext4, inodes are stored in **inode tables**.
- Each **block group** in an ext4 filesystem has its own inode table.
 - Location Calculation

```
block_group = (inode_number - 1) / inodes_per_group
```

The inode table is a contiguous set of blocks.

Tracking Free Inodes: The Inode Bitmap (ext4 example)

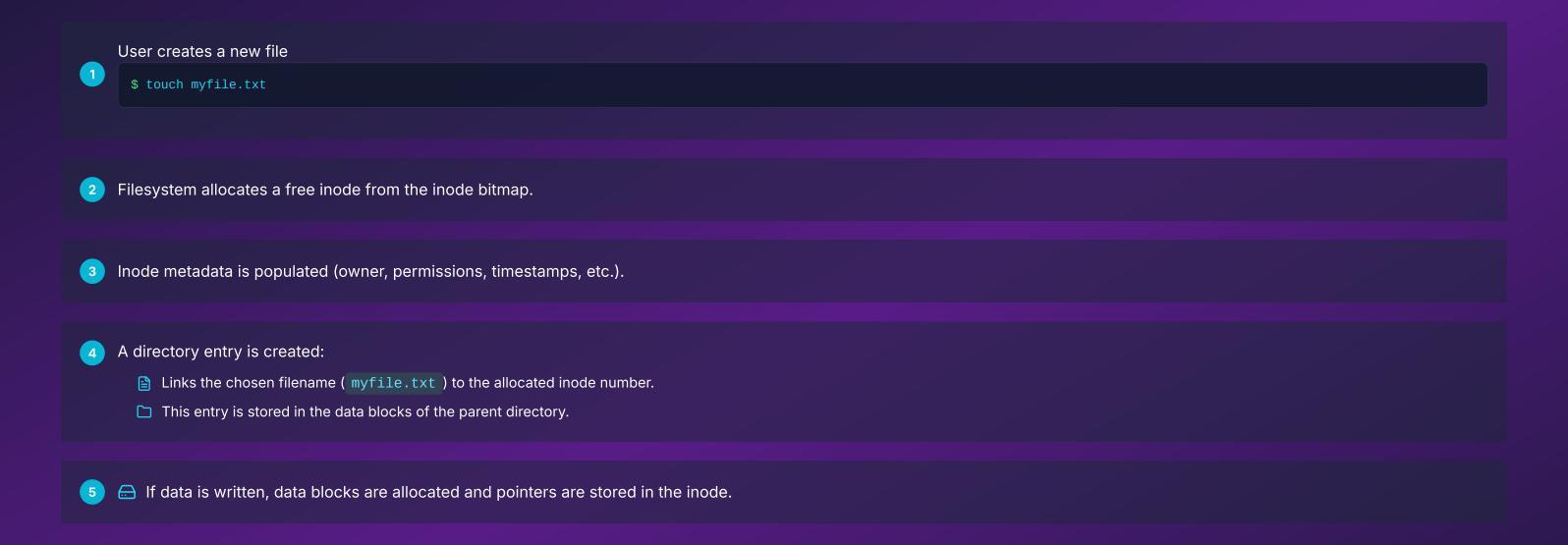
- Filesystems like ext4 use an **inode bitmap** to keep track of free and used inodes.
- Each block group has its own inode bitmap.



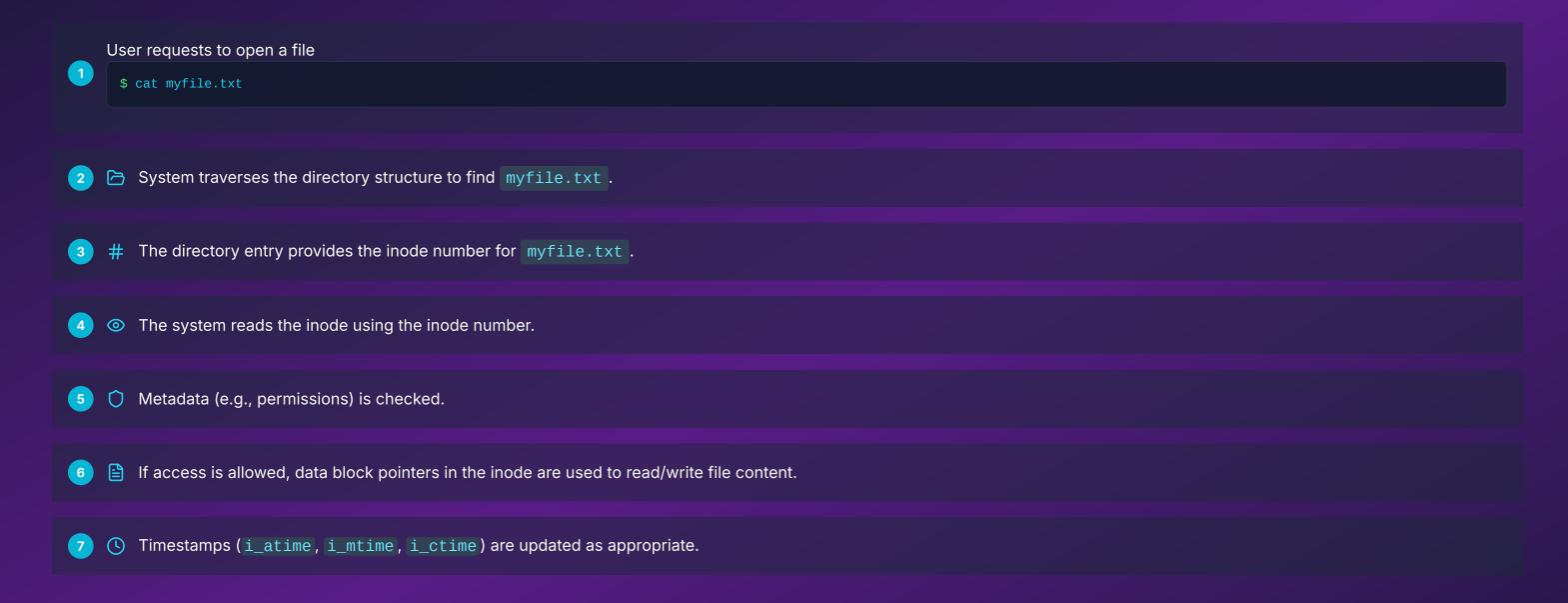
← Efficiency

This allows for quick allocation of a new inode when a file is created.

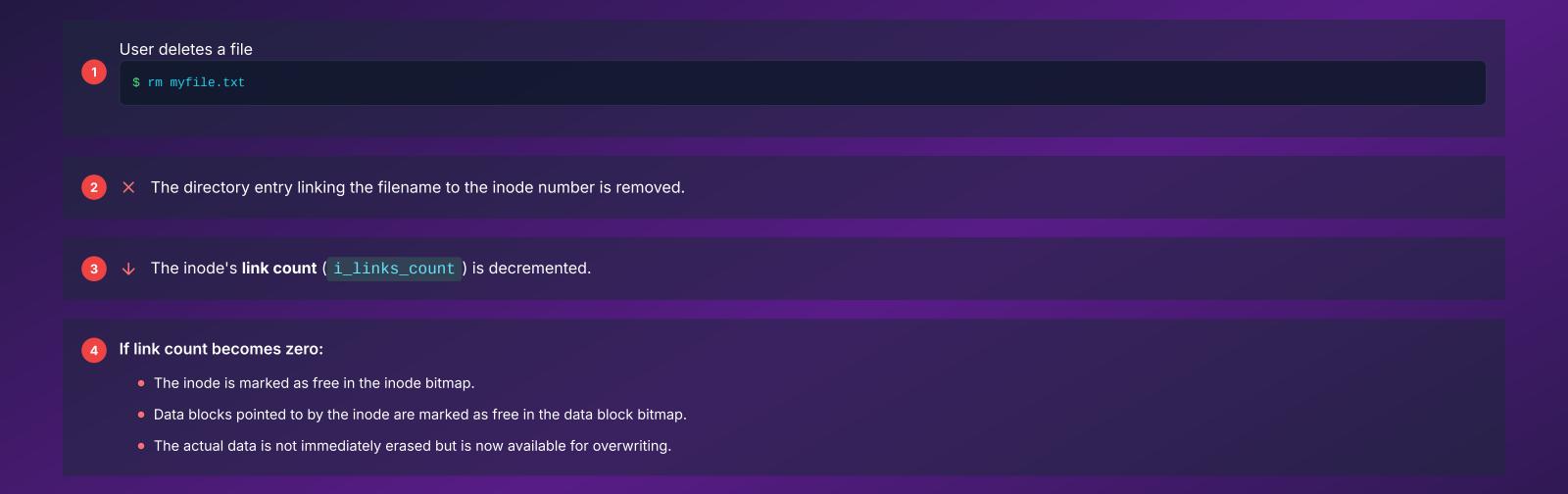
How Inodes Work: File Creation



How Inodes Work: File Access & Modification



How Inodes Work: File Deletion & Inode Reclamation



Directory Structure: Linking Names to Inodes

A directory is a special type of file.

Its data blocks contain a list of directory entries.

Each directory entry consists of:

A filename # The inode number corresponding to that filename

Example: /home/user/file.txt

Root directory (inode 2) contains entry for home

home directory inode points to data blocks with entry for user

user directory inode points to data blocks with entry for file.txt

Hard Links: Multiple Names, One Inode

A hard link creates an additional directory entry (filename) that points to an existing inode.

\$ In original_file.txt hard_link.txt

Characteristics

- Both original_file.txt and hard_link.txt share the same inode number.
- They point to the same metadata and the same data blocks.
- Modifying the file via any hard link affects all names.
- The inode's i_links_count increases for each hard link.
- File data is deleted only when i_links_count drops to 0.

× Cannot span across different filesystems

⚠ Generally, cannot link to directories (to prevent loops and simplify fsck)

Soft (Symbolic) Links: A File Pointing to a Path

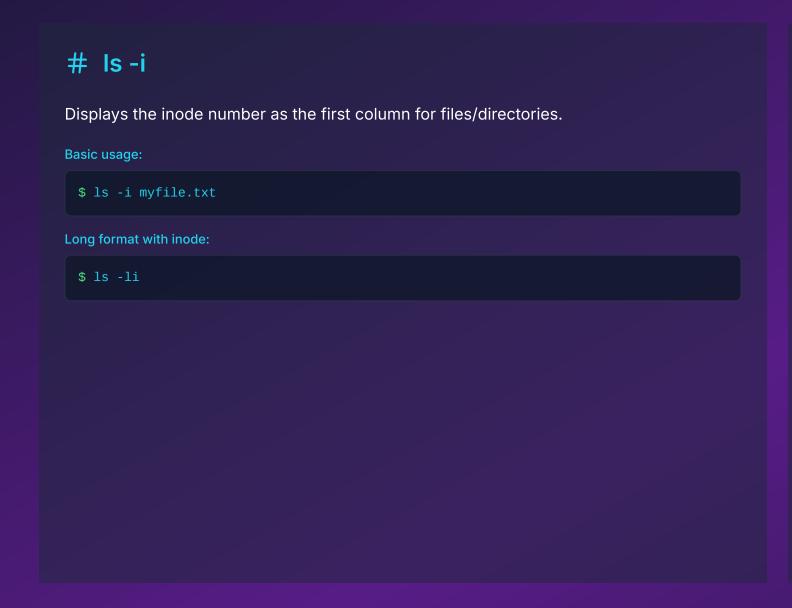
A **soft link** (or symbolic link) is a special file whose content is the *pathname* of another file or directory.

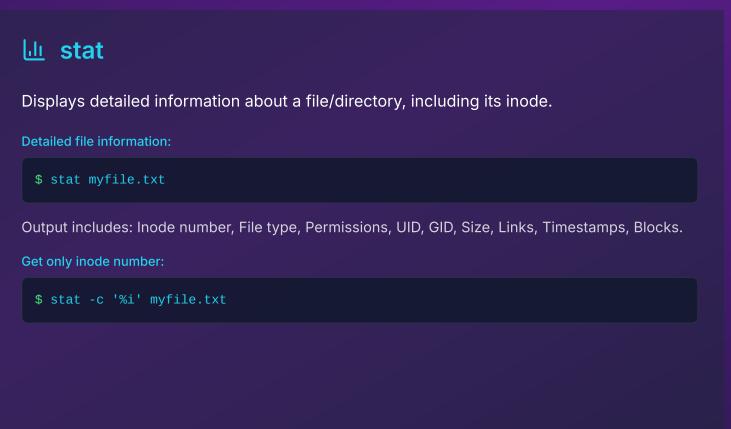
\$ ln -s target_file.txt soft_link.txt **Characteristics** • soft_link.txt has its own, separate inode number. • The inode for soft_link.txt stores the path to target_file.txt. • If target_file.txt is deleted or moved, the soft link becomes "broken" or "dangling." ✓ Can span across different filesystems ✓ Can link to directories

Hard Links vs. Soft Links: Key Inode Differences

Feature	Hard Link	Soft Link (Symbolic Link)
Inode Number	Shares inode with target	Has its own, distinct inode
Data Reference	Points directly to data (via shared inode)	Stores the <i>pathname</i> of the target
Target Deletion	Link remains valid; data persists if other links exist	Link becomes broken/dangling
Cross Filesystem	× No	✓ Yes
Link to Directory	× Generally No (restricted)	✓ Yes
1s -1 Link Count	Reflects shared inode's link count	Link count of soft link file itself (usually 1)

Viewing Inodes: Is -i and stat





Checking Inode Usage: df -i

The df -i command displays inode usage statistics for mounted filesystems. \$ df -i \$ df -i /path/to/filesystem **# Output Columns** Filesystem IFree Number of free inodes Device name IUse% Inodes Total number of inodes in the filesystem Percentage of inodes used **IUsed** Number of used inodes

Running out of inodes (IFree is 0 or very low) prevents new file creation, even if disk space is available.

Finding Files by Inode & Other find Uses

Q find /path -inum <inode_number>

Locates all files (hard links) associated with a specific inode number within /path.

Example:

\$ find /home -inum 123456

Useful for finding all names of a hard-linked file or troubleshooting.

find /path -xdev -inum <inode_number>

Searches only within the filesystem of /path (does not cross mount points).

Example:

\$ find /home -xdev -inum 123456

⚠ Dangerous but Useful

Delete files to free inodes:

\$ find /path -type f -delete

Can be used to delete files, e.g., to free up inodes if many small, unnecessary files exist.

⚠ Use with extreme caution!

Inode Limitations & Advanced Concepts

⚠ Inode Exhaustion (Fixed Allocation FS)

Running out of inodes prevents new file creation. Critical for systems with many small files (e.g., mail servers, caches).



Inline Data (ext4 EXT4_INLINE_DATA_FL)

For very small files, data can be stored directly within the inode structure itself, avoiding data block allocation and improving access speed.

Btrfs Dynamic Inode Allocation

Btrfs allocates inodes dynamically, not from a fixed pool set at mkfs time. df -i shows 0 total inodes.

☆ Special Inodes

Reserved inode numbers for filesystem internal structures (e.g., root dir, journal, bad blocks list).



🔇 Inode Size (i_extra_isize)

ext4 default inode size is 256 bytes, extendable for more metadata.

Summary & Key Takeaways

Inodes are central to Linux file management, storing metadata (not names or data). # Each file/directory has a unique inode number per filesystem. Inode structure includes permissions, ownership, timestamps, size, and data block pointers. Hard links share an inode; soft links have their own inode pointing to a path. >_ Commands like ls -i, stat, df -i, and find -inum are vital for inspection and troubleshooting. Understanding inode allocation (fixed vs. dynamic) and limitations is key for system administration.

Thank You!

Questions & Discussion