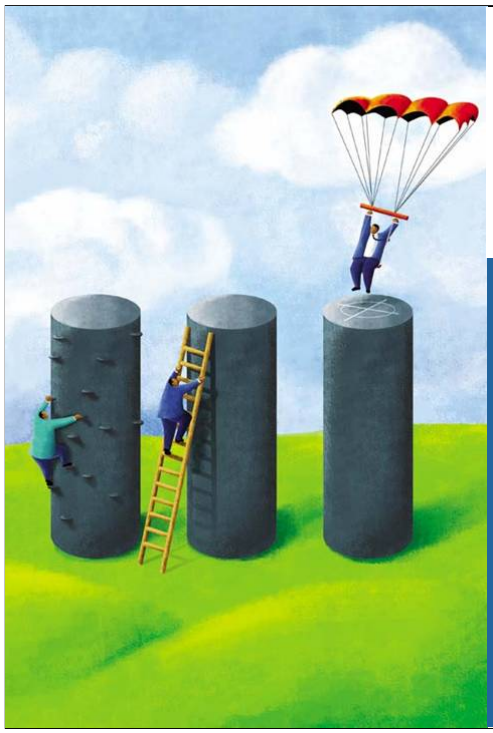


WIPRO
Applying Thought

The Unix Filesystem

By
Unix FCG

Version: 0.1
Date: Oct 2007

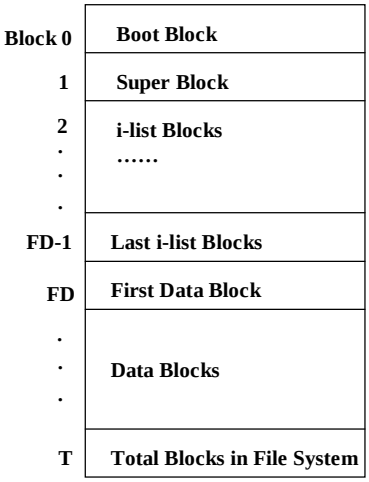
Index

- Unix filesystem Overview
- Internal view of the filesystem
- What is in the i-node?
- Inode explored
- Virtual file system
- External view of the filesystem
- Files and directories – common operations
- Access permissions and ownership
- Special permission bits
- Hard and soft links

Unix filesystem overview

- Everything in Unix is a file
- Filesystem design central to the success of Unix operation system
- Has a hierarchical tree-like structure
  - Single root node – the root
  - Every non-leaf node is a directory
  - Every leaf node is a file – directory, regular file or special device file
- Consistent treatment of file data – all data is treated as streams of bytes in the kernel
- Supports dynamic growth of files, ease of file creation/deletion and protection for file data

Internal view of the unix filesystem



- Filesystem is a sequence of blocks
  - Block size varies – 512 or 1024 bytes or more
- Boot block : Used by kernel for booting
- Super block: Complete catalog of information about filesystem
  - Size/type of filesystem, details of free available blocks, details on inode usage and available inodes
- I-list blocks: Contiguous sequence of inode blocks for the filesystem
- Data blocks: Sequence of data blocks that store file data

Note: Different file systems have different on disk layouts

What is in the inode?

- Inode or index node
  - Every file has an inode
  - List of inodes is the i-list
- Contents of inode
  - File owner identifier
  - File type
  - File access permissions
  - File access times – last accessed, last modified, inode modified
    - The last modified time is the time when the file contents were last updated
    - The inode modified time is the time when an operation on the file results in the inode contents to be updated eg. Changing permissions
  - Number of links
  - File addresses – pointing to the data disk blocks
  - File size

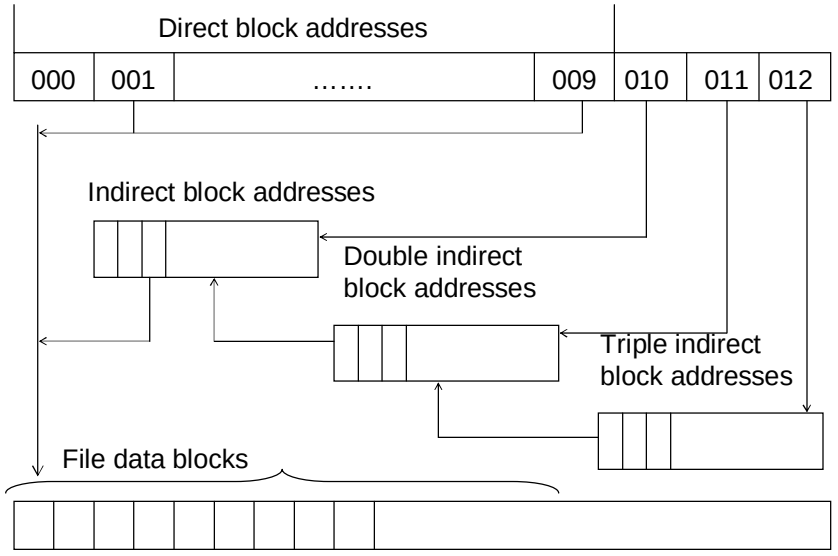
Do you know where the file/directory name is stored ?
Explore what file operations result in the inode contents to be updated

C

Inode explored – File addresses

- Pointers to data disk blocks
- 13 three-byte pointers point either directly or indirectly to the disk blocks containing the data contents of the file.
- Pointers 0-9: direct block addresses
  - addresses of direct blocks containing file data
- Pointer 10: indirect block address
  - address of a single indirect block, a block containing the addresses of direct blocks
- Pointer 11: double indirect block address
  - address of a double indirect block, a block containing the addresses of single indirect blocks which contain the addresses of direct blocks
- Pointer 12: address of a triple indirect block

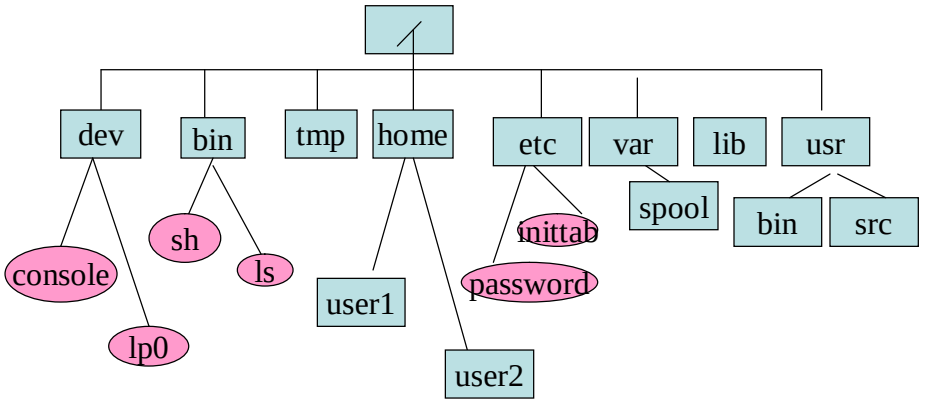
Inode explored – File addresses (contd.)



Virtual File system

- An enhancement to the ufs (unix file system)
- Designed to allow kernel support for multiple file systems
- Virtual File system (VFS) is an abstraction layer
- Handles file-oriented system calls, calls the necessary physical filesystem code to do the I/O
- Each lower-layer filesystem presents an interface that conforms to VFS
- Multiple types of filesystems
  - Disk-based filesystems
    - ext2 (second extended file system, linux native), ufs (unix file system), FAT (file allocation table file system)
  - Network filesystems
    - nfs (network file system), Code, AFS(Andrew file system)
  - Special filesystem
    - /proc file system

External view of filesystem



The top of the tree is “/” known as the root, has a number of subdirectories and files under it

Files system – common operations

Command	Remarks
mkfs	Creates filesystem
fsck	Check and repair filesystems Runs automatically at system boot
mount, unmount	Mounting/unmounting filesystems /etc/fstab contains the details of filesystems to be mounted

Note; Most of the above commands will only be used by sys admins, beware of the Consequences if you are playing around with these in your system
Refer Unix system man pages for more details on commands, options and output

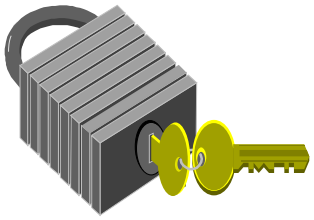
Files and directories – common operations

Command	Remarks
mkdir, rmdir	Create, remove directory
ls, ls –l, ls –t, ls -u	List all files/directories with option to see detailed file information
cp, mv, rm	Copy, move, delete files
cat, head, tail, more	See file contents
du, df	See disk/filesystem usage
cmp, diff	Compare file contents

Refer Unix system man pages for more details on commands, options and output

File and directory - Permissions & ownership

- **Three user categories**
  - **u - User**
  - **g - Group**
  - **o - World (others)**
- **Three levels of permissions**
  - **r – read**
  - **w – write**
  - **x - execute**



ls –l output displays file permissions. Eg.

$ ls –l file1

-rwxrw-r-- 1 owner other 0 Jun 29 12:57 file1

File1 has read/write/execute permission for owner, read/write for group and read permissions for others

File and directory – Permissions and ownership (contd)

Permission	Allowed operations
r	List contents of directory; view contents of file
w	Create/delete/rename files in directory Modify file content
x	Execute files (if binary) Change to directory

chmod command may be used to change file/directory permissions for all user categories
chown/chgrp command may be used to change owner/group.

Special permission bits

- **Setuid**
  - Sets setuid bit; Program assumes user id of owner when run
  - Eg.

```
$ chmod u+s file1
$ ls –l file1
-rwsrw-r--  1  owner  other  0  Jun 29  12:57  file1
```

- **Setgid**
  - Sets setgid bit; Program assumes group of owner when run
  - Eg.

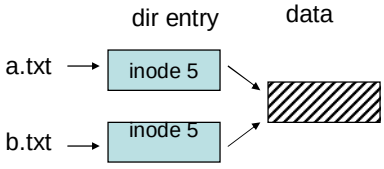
```
$ chmod g+s file1
$ ls –l file1
-rwxrwsr--  1  owner  other  0  Jun 29  12:57  file1
```

- **Stickybit**
  - Sets sticky bit; Typically used for directories i.e /tmp
  - Files in directory can be renamed/deleted only by owner/root, irrespective of permissions for other users
  - Eg.

```
$ chmod +t file1
$ ls –l file1
-rwxr--r-t  1  owner  other  0  Jun 29  12:57  file1
```

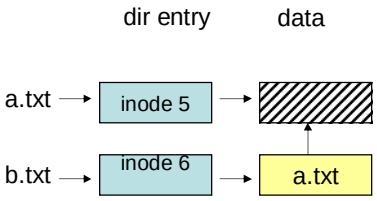
Hard and soft links

- **A link is simply a way to refer to the contents of a file**
- **Two types of links**
  - **Hard links**
  - **Soft links**



Hard Link

• eg. $ ln a.txt b.txt
•hard link points to the same inode
• one copy of data only



Soft Link

• eg.$ ln –s a.txt b.txt
• soft link has a different inode
• two different data files, the link pointing to the other

Go ahead and explore links to directories!

Thank you.