Unix System Architecture, File System, and Shell Commands

Prof. (Dr.) K.R. Chowdhary, Director COE

Email: kr.chowdhary@iitj.ac.in

webpage: http://www.krchowdhary.com

JIET College of Engineering

August 21, 2017

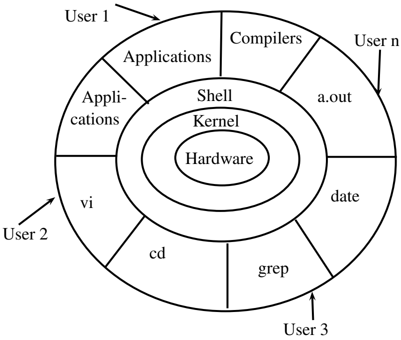
Outlines:

- Functions of an operating system
- Unix system architecture (OS: Kernel, file system, shell)
- Unix File system: Logical Structure
- commands for unix file system
- Disk layout of file system
- Inode structure
- Linux File system: Physical structure
- Updates of file system
- Dtecting and correcting file system
- vi editor
- Shell
- System Administration and monitoring
- Further reading

Functions of an Operating System (OS):

- Unix, Linux, Windows, Mac, Android are OS
- Functions:
  1. Memory Management: When click an icon of a program on screen, OS allocates memory for it and loads that program from disk to memory.
  2. Processor Management: The programs in memory are given times, based on priority, to execute them
  3. IO (Input-output) Management: It provides communication with IO, like Keyboad, dist, display, printer, network, ...
  4. Information Management: Organises the information on disk in the form of files

Unix System Architecture

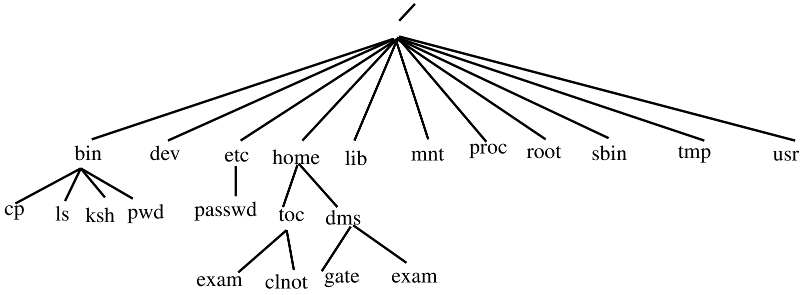


- **Hardware**: CPU, Memory, IO interface

- **Kernel**: It is the core component of Operating System, interacts directly with hardware
- **Shell**: An interface to kernel, hiding complexity of kernel's functions from users.
- **Utilities**: Programs that provide the user most of the functionalities of an operating systems. These are application programs.

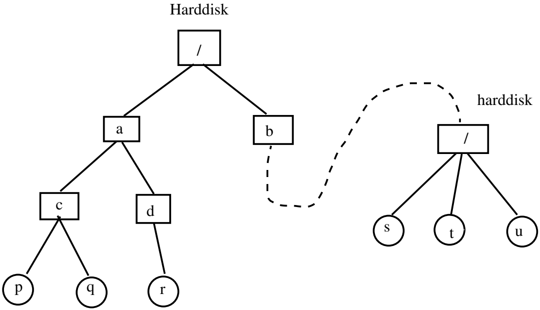
Linux File System

- Two Views: Logical & Physical
- The Unix file system (UFS; also called the Berkeley Fast File System, the BSD Fast File System or FFS) is a file system used by many Unix and Unix-like operating systems. It is a distant descendant of the original filesystem used by Version 7 Unix.



File system structure

- Files have no structure at all, they are only flat sequences of bytes.
- In UNIX a part or all of a disk's file system can be mounted in another disk's file system.
- **Files and inodes.** A file consists of exactly one inode, and zero or more data blocks. An inode is a structure used to maintain information about the file.



Linux File System

- /dev: Contains file representations of peripheral devices and pseudo-devices
- /mnt: Empty directory commonly used by system administrators as a temporary mount point.
- /proc: procfs virtual filesystem showing information about processes as files.
- /sbin: Stands for “system (or superuser) binaries”, contains fundamental utilities, like init,
- /tmp: A place for temporary files not expected to survive a reboot.
- /usr: The “user file system”: originally the directory holding user home directories
- /lib: libraries and data files for programs stored within /usr or elsewhere.

Linux File System commands

- File system Commands: for Directories

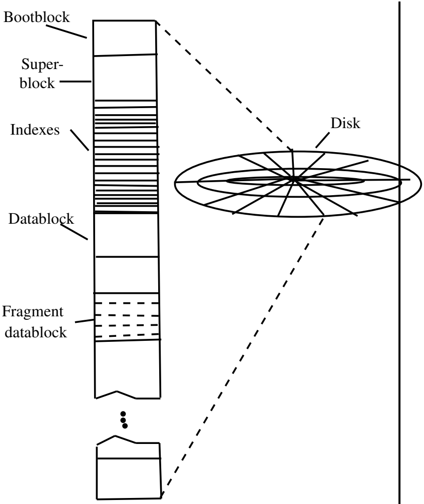
  cd, mkdir, rmdir, pwd (Print Working Directory), cp, mv
- File system Commands: for Files

  ls, cp, mv, ln, touch (Update), rm, cat, more, head, tail, wc, diff, file
- Redirection, and pipe: $>$, $<$, $|$, $/dev/null$
- Devices as files: /dev/sda1, /dev/sda2, /dev/lp, /dev/tty1.../dev/tty63, /dev/tty, /dev/lp,
- Editor: ed, sed, vi
- Misc.: lpr, who, ps, find, sort, cut, paste.

  e.g. $ cut -d: -f1,2 months

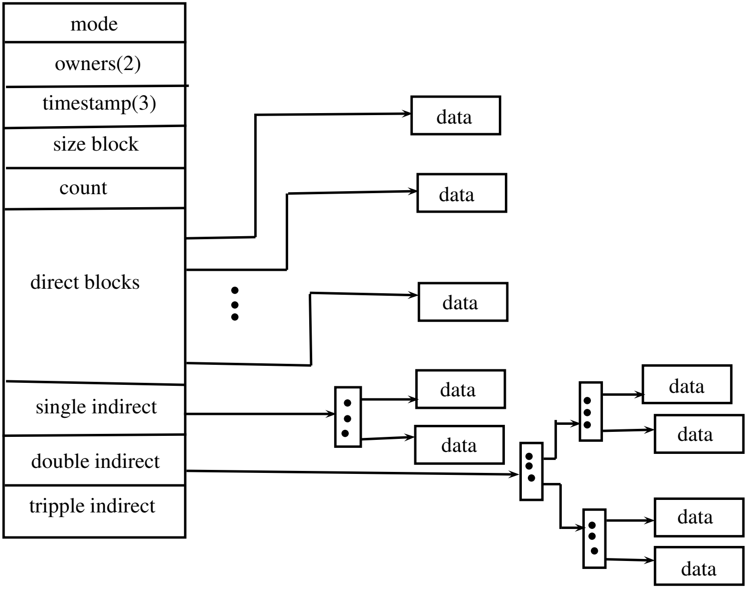
  e.g. $ paste -d: file1 file2

Disk layout of file system



- A file system is created with mkfs. It defines a number of parameters
- a bootblock - contains a primary boot program for the operating system;
- a superblock - static parameters of the file system,
- inodes - stands for index node
- data blocks - each block has typically a size of 4 Kbytes
- fragment data block size - typically of size 512 bytes

Inode structure



Inode structure

- 128 bytes (64 bytes in oldersystems) each
- Statically allocated
- Root inode (inode 2) the root (/) of the filesystem
- Inode contents: Permissions: 9 bits, three time stamps: modification time, access time, status time(when certain changes was made to the inode)
- Through pointers in the inode we can access the file's data blocks.

  12 direct pointers

  1 single indirect pointer

  1 double indirect pointer

  1 triple indirect pointe

Linux File System

- A file system is described by its super-block. It is built when the file system is created and never change.
- Because the super-block contains critical data, newfs replicates it to protect against catastrophic loss.
- The default super block always resides at a fixed offset from the beginning of the file system's disk partition. The redundant super blocks are not referenced
- Within the file system are files. Certain files are distinguished as directories and contain collections of pointers to files that may themselves be directories.
- Every file has a descriptor associated with it called an inode.
- In order to create files with up to 2↑32 bytes, using only two levels of indirection, the minimum size of a file system block is 4096 bytes.

Updates to the file system

- Every working day hundreds of files are created, modified, and removed. Every time a file is modified, the operating system performs a series of file system updates.
- When a user program does an operation to change the file system, such as a write, the data to be written is copied into an internal in-core buffer in the kernel.
- The disk information is updated to reflect the in-core information when the buffer is required for another use
- If all updates are done asynchronously, several serious inconsistencies can arise. One inconsistency is that a block may be claimed by two inodes.
- The problem with asynchronous inode updates can be avoided by doing all inode deallocations synchronously.

Detecting and correcting corruption

- Normally fsck is run non-interactively. In this mode it will only fix corruptions that are expected to occur from an unclean halt.
- **Super-block checking.** The most commonly corrupted item in a file system is the summary information associated with the super-block. The summary information is prone to corruption because it is modified with every change to the file system's blocks or inodes, and is usually corrupted after an unclean halt.
- The super-block is checked for inconsistencies involving file-system size, number of inodes, free-block count, and the free-inode count.

Detecting and correcting corruption...

- **Free block checking.** Fsck checks that all the blocks marked as free in the cylinder group block maps are not claimed by any files.
- If anything is wrong with the block allocation maps, fsck will rebuild them, based on the list it has computed of allocated blocks.

Directories

Directories are files, but they are treated differently. A directory can be identified by its mode bytes. A directory is a file that consists of a number of records, each of which contains the following fields:

- a pointer to the next record
- a number identifying an inode (i.e. another file)
- a number identifying the length of the record
- a string containing the name of the record (max 255 chars). It is this name we usually refer to as a filename. Note that it is part of the directory, and not part of the file.
- (possibly some padding)

Vi advanced features

- The vi editor (short for visual editor) is a screen editor which is available on almost all Unix systems.
- The original code for vi was written by Bill Joy in 1976,

Shell

- Shell is command interpreter as well a programming language (called shell script)
- There are number of shells availabe in unux:
    1. the Bourne shell), Bourne Shell(sh or bash),
    2. Korn Shell (ksh-written by David Korn, of Bell Labs),
    3. C Shell (C shell is the UNIX shell (command execution program, often called a command interpreter) created by Bill Joy at the University of California at Berkeley as an alternative to UNIX's original shell.

System Administration

- top - display Linux processes
- gnome-system-monitor: Shows cpus, memory, and network utilization
- ps: Processes status
- kill: Aborts process(s)

awk

```
pattern { action }
pattern { action }
...
awk 'program' input-file1 input-file2 ...
awk -f program-file input-file1 input-file2 ...
$ awk '{print $3 "\t" $4}' marks.txt
```

Further Reading:

- http://homepages.uc.edu/ thomam/Intro_Unix_Text/File _System.html
- http://www.cis.rit.edu/class/simg211/unixintro /Filesystem.html