Linux System Administration–I
CSE-4043

Chapter 6: File System

NIBEDITA JAGADEV

Department of CSE

Asst. Professor

SOA Deemed to be University, Bhubaneswar, Odisha , India

nibeditajagadev@soa.ac.in

Contents

- File System
- Absolute path and Relative path
- Spaces in file name
- File system mounting and Unmounting
- Organization of file tree

Introduction

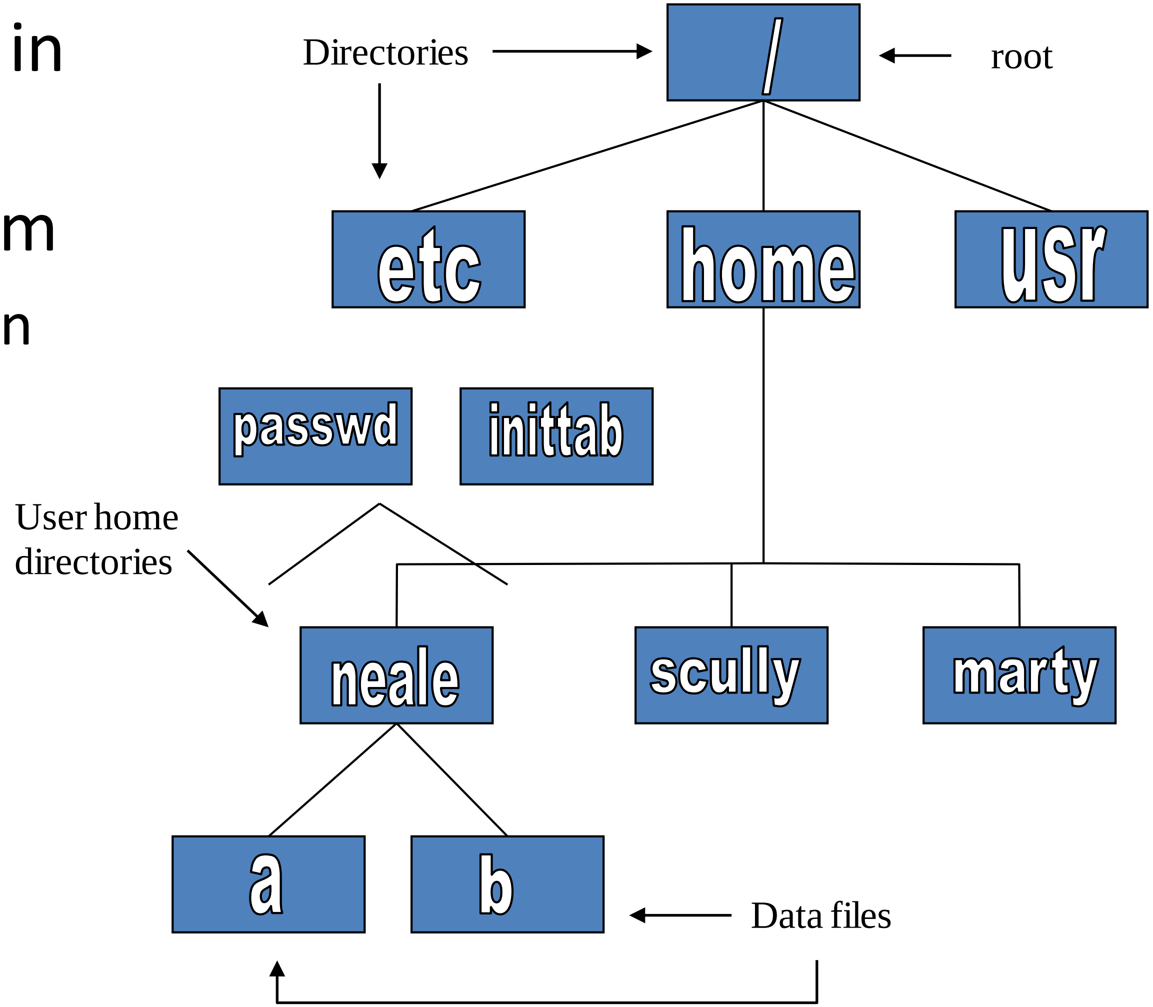
- What is File System?
  - The abstraction used by kernel to represent and organize the storage resources.
- UNIX File System in general
  - File system is organized in tree structure.
  - File tree can be arbitrarily deep.
  - File name must NOT LONGER than 256 chars.
  - Single path name must NOT LONGER than 1023 chars.

Types of file system

- Physical file system – we take a disk (or partition!) and we want to arrange files on it.
- Logical file system – file system that demonstrate some logical state of the system such as /proc /dev or /sys (those file systems demonstrate running processes or devices detected by the system or system info.) – Those file system don’t deal with real file and are beyond scope. (but we acknowledge their existence)
- Virtual file system – we take several physical and logical merge them into one file system.

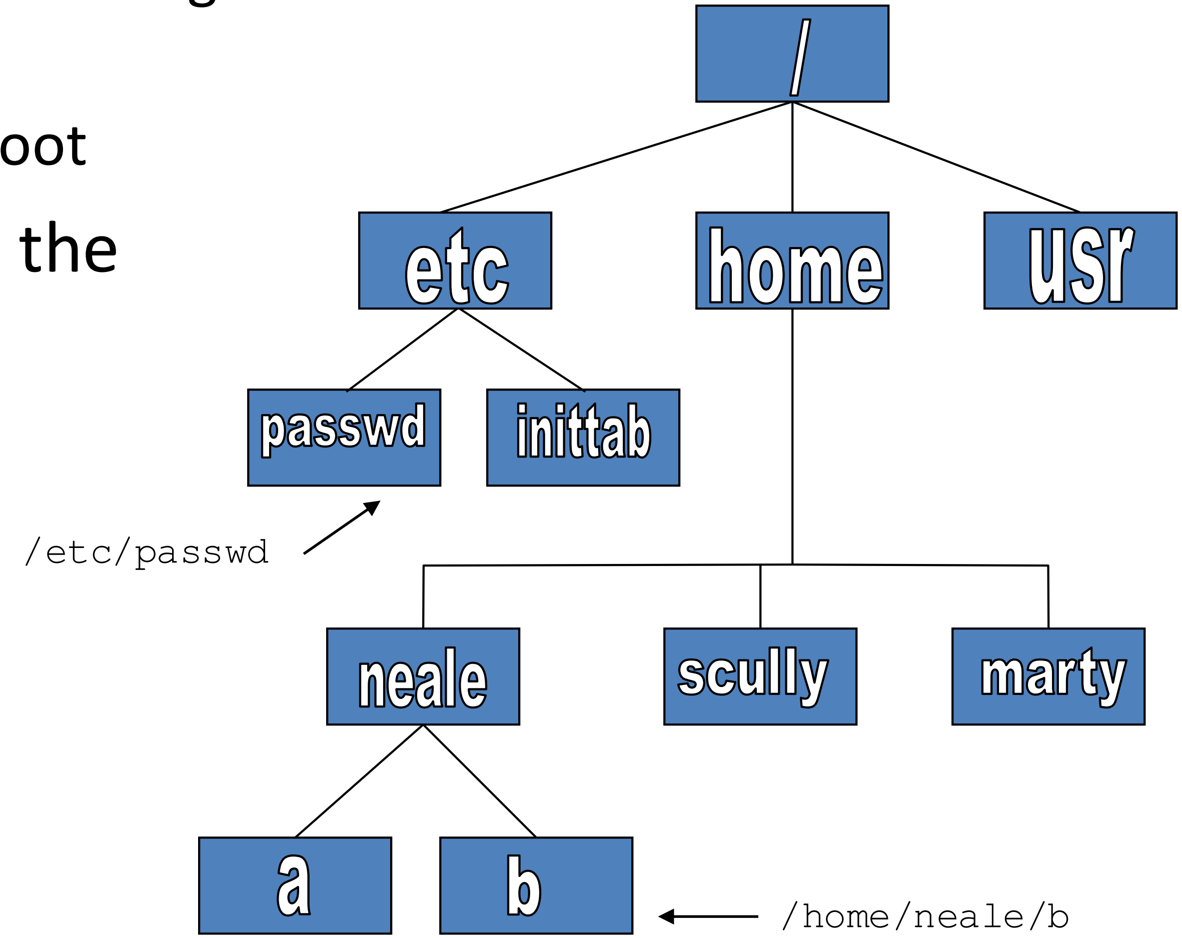
Linux File System Basics

- Linux files are stored in a single rooted, hierarchical file system
  - Data files are stored in directories (folders)
  - Directories may be nested as deep as needed



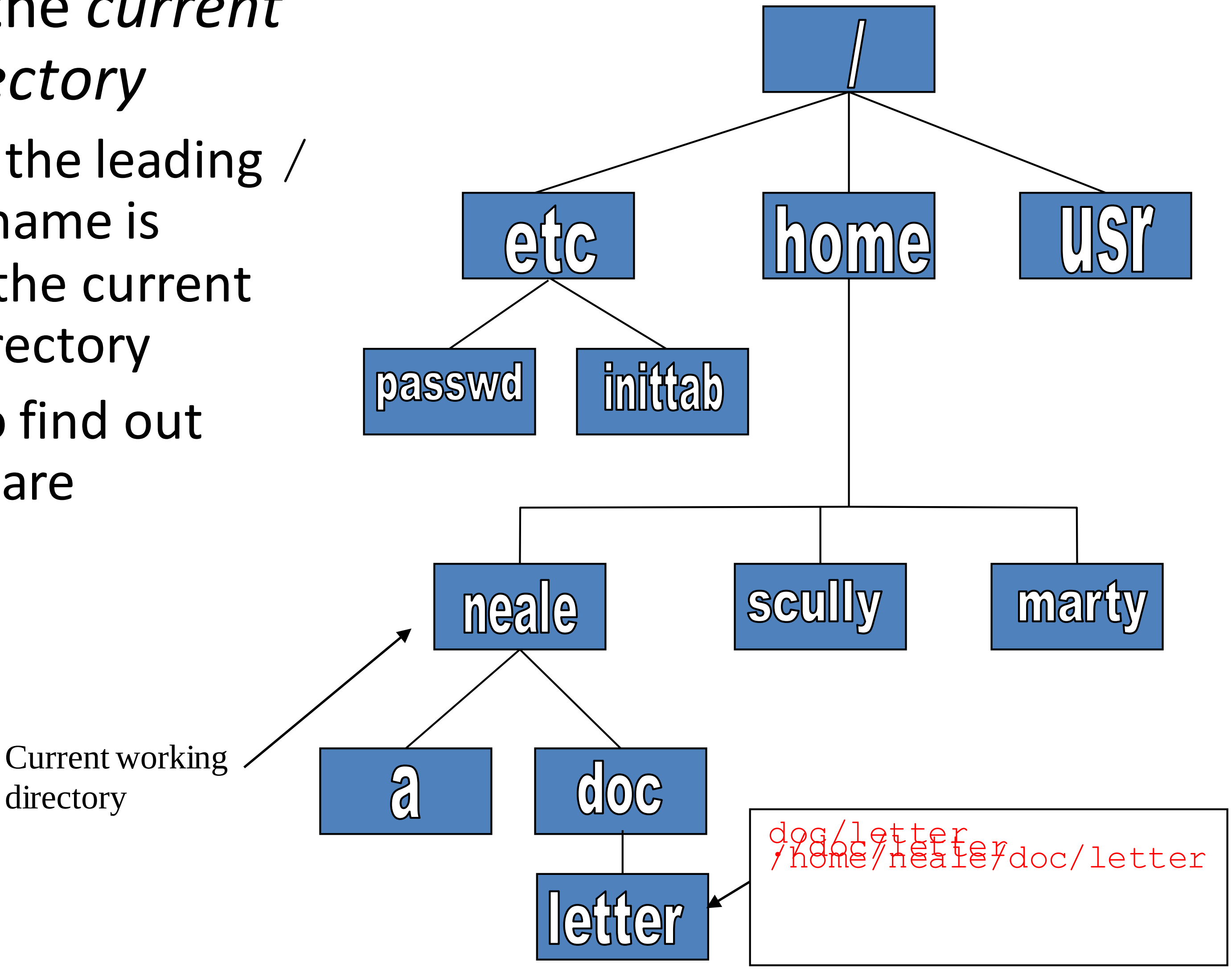
Naming Files

- Files are named by
  - naming each containing directory
  - starting at the root
- This is known as the *pathname*



The Current Directory

- One directory is designated the *current working directory*
  - if you omit the leading / then path name is relative to the current working directory
  - Use `pwd` to find out where you are



Current working directory

```
dog/letter
./doc/letter
/home/neale/doc/letter
```

Some Special File Names

- Some file names are special:
  - `/` The root directory (not to be confused with the root user)
  - `.` The current directory
  - `..` The parent (previous) directory
  - `~` My home directory
- Examples:
  - `./a` same as `a`
  - `../jane/x` go up one level then look in directory `jane` for `x`

Special Files

- `/home` - all users’ home directories are stored here
- `/bin`, `/usr/bin` - system commands
- `/sbin`, `/usr/sbin` - commands used by sysadmins
- `/etc` - all sorts of configuration files
- `/var` - logs, spool directories etc.
- `/dev` - device files
- `/proc` - special system files

Absolute path and Relative path

- An absolute path refers to the complete details needed to locate a file or folder starting from the root element and ending with the other subdirectories.
- A relative path is a way to specify the location of a directory relative to another directory.

Spaces in file name

- Names are limited in length and must not contain slash character or null.
- Command line arguments are separated with whitespaces.

File system mounting and Unmounting

- Mounting File System
  - File tree is composed of File System
  - Use *mount* command to map a directory within the existing file tree (mount point) to the root of the new file system.
    - *mount /dev/hda2 /usr*
  - Use *umount* command to detach the file system.
    - Detaching will fail if the file system is busy

Organization of file tree

- "The UNIX file system has never been very well organized."

Organizing of The File System (cont.)

/	The root directory
/bin or /sbin	Commands for basic system operation
/dev	Device entries
/etc	Critical startup and configuration files.
/lib	Library for the C compiler
/tmp	Temporary files
/var/adm or /var/log	Accounting file, log files

Linux System Administration-I
CSE-4043

Chapter 6: File System

NIBEDITA JAGADEV

Department of CSE

Asst. Professor

SOA Deemed to be University, Bhubaneswar, Odisha , India

nibeditajagadev@soa.ac.in

Contents

File types

- File attributes
- Permission bits
- Set UID Set GID bits
- Sticky bits

File types

- Regular Files
  - binary
    - GIF, JPEG, Executable etc.
  - text
    - scripts, program source code, documentation
  - Supports sequential and random access

File types

- Directory
  - Can contain ANY kind of files
  - what is “.” and “..”??

File types

- Device File
  - Allows programs to communicate with hardware.
  - Kernel modules handles device management.

- Device Files (cont.)
  - Character Device
    - Accepts a stream of characters, without regard to any block structure.
    - It is not addressable, therefore no seek operation
  - Block Device
    - Information stored in fixed-sized block
    - It is addressable, therefore seek operation is possible.

File types

- UNIX Domain Sockets (BSD)
  - sockets that are local to a particular host and are referenced through a file system object rather than a network port.
  - X windows

File types

- Named Pipe
  - Allow processes to communicate with each other.

File types

- Symbolic links
- Hard links
  - Linking files by reference
  - System maintains a count of the number of links
  - Does not work across file systems.
- Soft links
  - Linking files by name
  - No counter is maintained
  - Work across file system

File attributes

- The representation of the file is done through different characteristics of that those are known as file attributes.

File Permissions

- The Permission Bit
  - 9 permission bits used to determine 3 types of accesses, READ, WRITE, EXECUTE.
  - Permission can be set based on GROUP, OWNER, ANYONE ELSE.
  - Use *chmod* command to change permission
    - Binary 001 for EXECUTE
    - Binary 010 for WRITE
    - Binary 100 for READ

Set UID and Set GID bits

- The Setuid and Setgid bits
  - Setuid with octal value 4000
  - Setgid with octal value 2000
  - These bits allow programs to access files that processes that would otherwise off limits to the user that runs them.

- Setgid access on directories
  - Files created in this directory will have the same group ownership as the directory itself.
  - Example: when groups of users need to share a lot of files, with s bit, correct group ownership will be set for new files.
  - #chmod g+s files

Sticky Bit

- – Not very popular in today’s system
- – If a directory has sticky bit set, then only the owner can remove file from the directory.
- – /tmp is a good example.

Linux System Administration–I
CSE–4043

Chapter 6: File System

NIBEDITA JAGADEV

Department of CSE

Asst. Professor

SOA Deemed to be University, Bhubaneswar, Odisha , India

nibeditajagadev@soa.ac.in

Contents

- List and inspect files
- Chmod
- Chown and Chgrp
- Umask
- Linux bogus flags
- Access control lists

List and inspect files

- Using “ls –l “to identify file types
  - -	Plain file (hard link)
  - d	Directory
  - lSymbolic link
  - b	Block specific file
  - c	Character special file
  - s	socket
  - p	Named pipe

File ownership

- File Ownership
  - User owner
  - Group owner
    - The group owner is independent of user owner
  - Example: Displaying file ownership “ls –l”

```
drwxrwxr-x     2 pop       403              512 Sep 24 11:47 acm
drwxr-xr-x     9 pop       csdept          1024 Oct 13 15:54 acm.contest
-rw-r-----     1 john      csdept           261 Mar 11  1999 address
```

File permission

- File Access Types

Access	Meaning on File	Meaning on Directory
r	View file contents.	Search directory contents (e.g. use ls, ls *.txt).
w	Alter file contents.	Alter directory contents (e.g. delete files in it).
x	Run executable file	Make it your current directory (cd to it)

- Access classes
  - User access (u)
  - Group access (g)
  - Other access (o)
- Example
  - -rw-rw-r--

File permission

- Setting file protection
  - chmod accessstring_list files
    - Access string has three parts:
      - Access_classes operator access_type
        - » Access classes u, g, o, a
        - » Operator: +, -, =
        - » Access type: any combination of r,w,x
    - For example:
      - #chmod g+w testfile
  - Recursive option: -R
      - #chmod –R go-rwx /home/xyz
  - Read the mannual of chmod for more options

More about file ownership

- Who owns new files?
  - Owner is the one who create it
  - Group owner
    - On System V, current group of the user
    - On BSD-based systems, the group owner of the directory
- Changing file ownership
  - Use chown , chgrp
    - Traditional System V, both superuser and user owner are allowed
    - POSIX-compliant SystemV and BSD, only superuser can change

More about file ownership

– More about changing file ownership

- Use recursive option –R

```
#chown –R harvey /home/xyz
```

- Use find command

```
# find /home/xyz –print | xargs chown test
```

– Change owner and group using chown

```
#chown new-owner:new-group files
```

– Change group owner

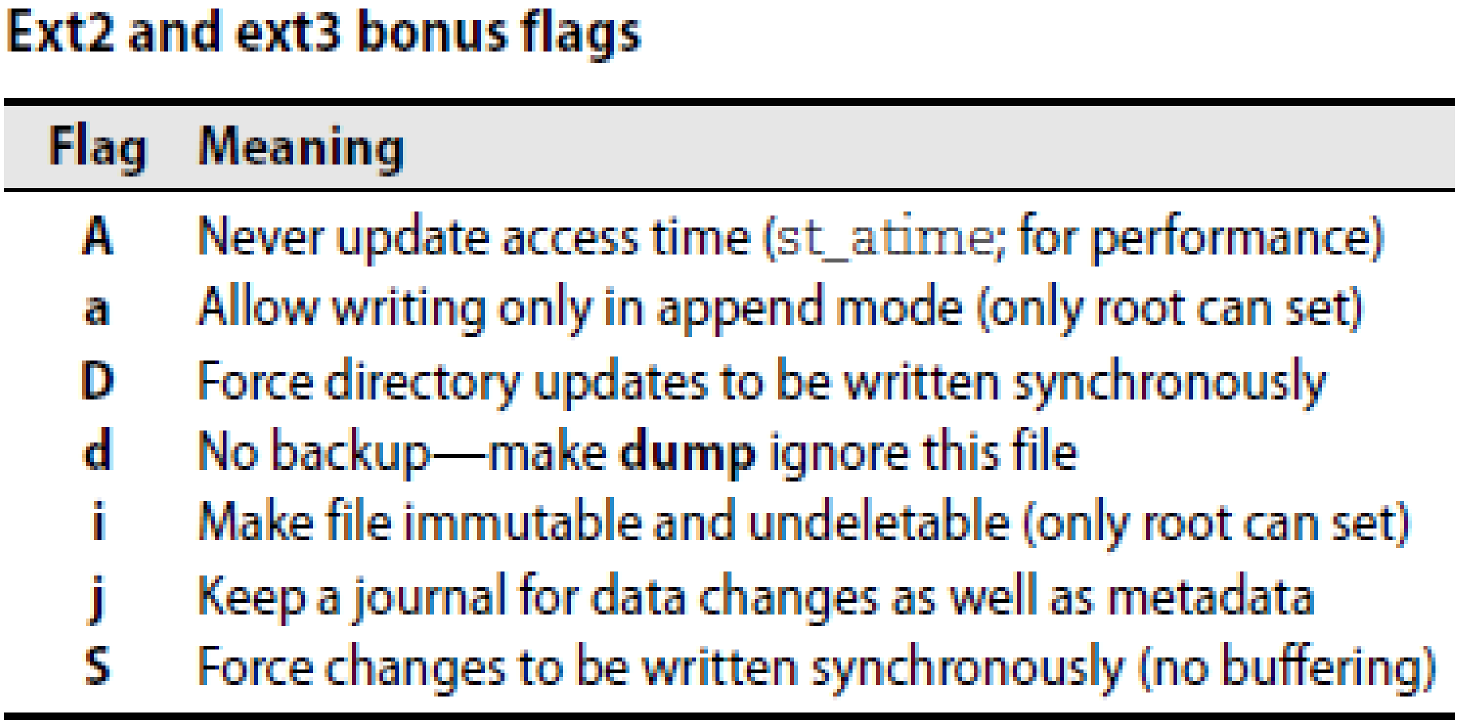
```
#chgrp newgroup file
```

BSD: non-root users of chgrp must be

» The owner of the file and

» A member of the new group to change

Linux bogus flags



Ext2 and ext3 bonus flags

Flag	Meaning
A	Never update access time (st_atime; for performance)
a	Allow writing only in append mode (only root can set)
D	Force directory updates to be written synchronously
d	No backup—make **dump** ignore this file
i	Make file immutable and undeletable (only root can set)
j	Keep a journal for data changes as well as metadata
S	Force changes to be written synchronously (no buffering)

Access Control List

- Traditional UNIX 9bit permission
- Windows has a much more complicated way – Access control lists
- POSIX started ACL for Unix
  - Extension to support multiple user/groups
- NSFv4 ACL
  - A union of all (UNIX + Windows) ACLs

ACL in Linux

- Entries:
  - User::perms
  - User:username:perms
  - Group::perm
  - Group:groupname:perm
  - Other::perms
  - Mask::perms
- Implemented at the file system level
  - Disabled by default
    - Enable by mounting as –o acl
  - Try to match the single most appropriate entry
- Set/Get access control list – setfacl,getfacl

NFSv4

- Linux does not support it
  - Can map POSIX acl to NFSv4 acl
- Solaris does support it
- See the textbook for more info

INODES

- Reference to a file
- Points directly and indirectly to blocks
- Contain the OS info on a file
- Does not contain the file name
- Each Inode is identified by unique number

INODES

- Kernel maintains file information in a structure called *inode.*
    - Creation, modification time stamps
    - Ownership, file size etc.
- Commonly used INODE information can be found by using *ls* command
- Group information and be modified by using *chgrp* command.

Summary

- All UNIX file system are very similar.
- All file system have this concept of file tree.
  - Transparent to user even mount point is mapped to a remote file system.
- To communicate with devices, special device files are used.
- More information check out the man pages.

Thank You