UNIT 2: UNIX File System

UNIX and Shell Programming: BCAC691

Dr. Tumpa Banerjee
Assistant Professor, Department of MCA

Table of Content

File

- Basic structure that stores information on the UNIX system
- A file is a sequence of bytes that is stored somewhere on a storage device
- A file can contain any kind of information that can be represented as a sequence of bytes.
- Image, program, word processing documents, etc, are examples of files.

Filename Convention

- Any ASCII character in a filename except the null character and slash(/)
- The slash acts as a separator between directories and files.
- It is better to use alphanumeric characters for naming files.
- File name is case-sensitive in the UNIX environment. i.e File.txt is different from file.txt.

Type of File

- Ordinary files
- Directories
- Special files
- Device files
- Symbolic links
- Pipes
- Socket

Ordinary file

- An ordinary file or regular file is the most common file type. An ordinary file is divided into two types:
  - ➢ Text file
  - ➢ Binary file

Text File

- A Text file contains only printable characters, and you can often view the contents and make sense out of them.
- All C and Java program sources, shell and parl scripts are text files.
- A new text file contains line of characters where every line is terminated with the newline character also known as line feed.

Binary File

- A binary fine contains both printable and unprintable characters that cover the entire ASCII range.
- Most UNIX commands are binary files and the object code and executables that you produce by compiling C programs are also binary files.
- Picture, sound and video files are binary files as well.

Directories

- directories store both spatial and ordinary files
- a directory file contains an entry for every file and subdirectory that it houses
- each entry has two components (1) the file name and (2) a unique identification number for the file or directory

Spatial files/ Device Files

- used to represent a real physical device such as a printer tape drive or terminal used for input/output operations.
- Device or spatial files are used for device input/output on Unix
- Two flavor of special files for each device: character special files and block special file
- Characters special files: used for input/output, data is transferred one character at a time. This type of access is called raw device access.
- Block special files: used for input/output, data is transferred in large fixed-size blocks. This type of access is called block device access

Spatial files/ Device Files

- It is advantageous to treat devices as files as some of the comments used to access an ordinary file also work with device files.
- Device filenames are generally found inside a single directory structure, $/dev$.
- A device file is indeed special, it is not really a stream of characters.
- Every file has some attributes that are not stored in the file but elsewhere on disk.
- The operation of a device is internally governed by the attributes of its associated file.
- The kernel identifies a device from its attributes and then uses them to operate the device.

Link

- A link is a second name of a file
- with a link only 1 file exist on the disk but it may appear in 2 places in the directory structure
- this allow 2 users to share the same file
- any changes that are made to the file will be seen by both users
- this type of link is called hard link
- hard link cannot be used to give more than one link to directory.
- cannot be used to link a file on different computers

Symbolic Link

- it is a file that only contains the name (including full path name) of another file
- when the operating system operates on a symbolic link it is directed to the file that the symbolic link points to
- the symbolic link is a pointer to the other file
- symbolic link is also known as soft link

Symbolic Link

- it contains text from the path to the file it references
- to an end user symbolic link will appear to have its own name but when you try reading or writing data to the symbolic link it will instead reference these operations to the file it points to
- it will delete the soft link itself, the data file would be there
- if we delete or move it to a different location symbolic file will not function properly

Pipes

- UNIX allows linking commands together using pipes
- The pipe acts as a temporary file that only exist to hold data from one command until it is read by another
- The output/result of first command sequence is used as the input to the second command sequence

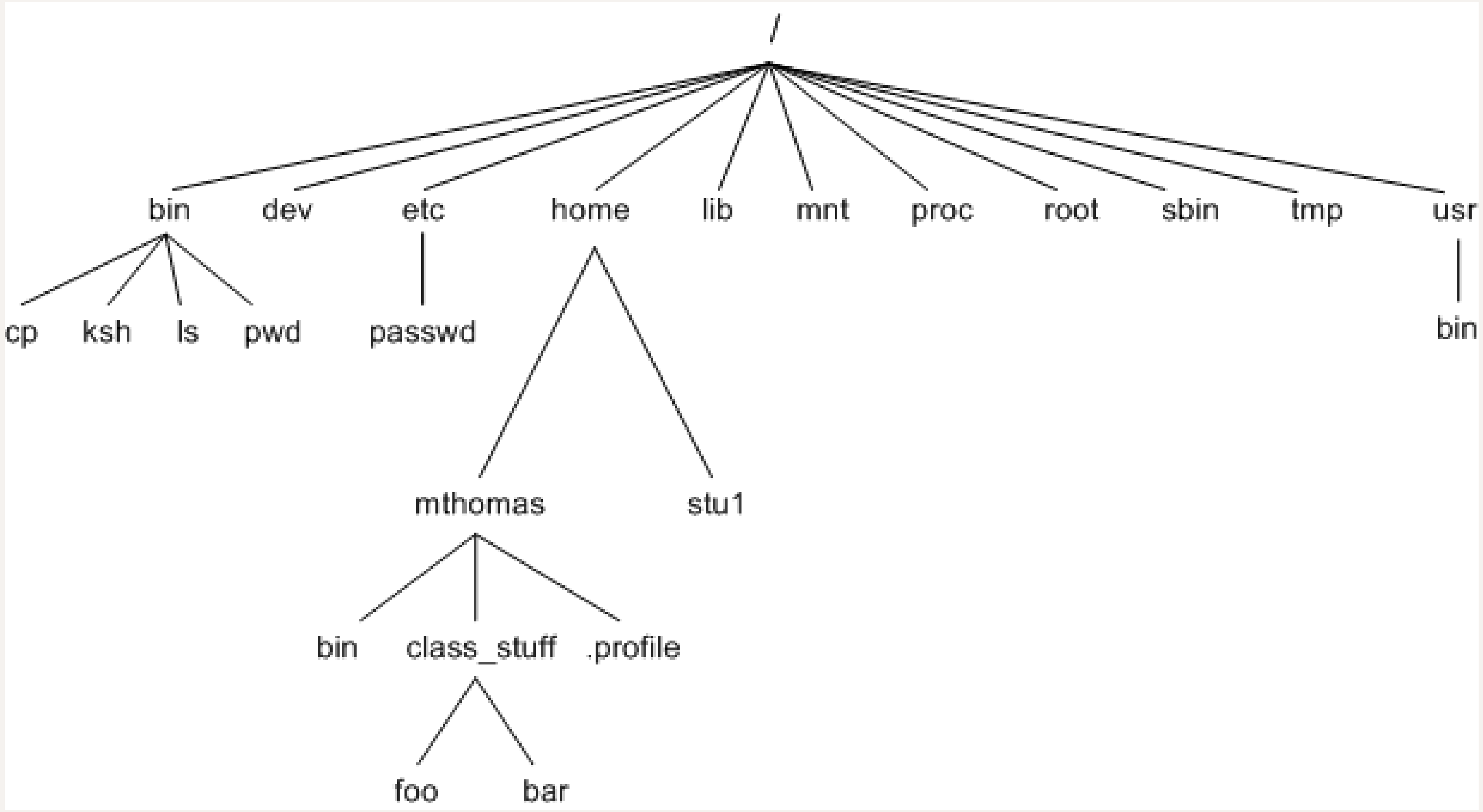
Socket

- UNIX socket is used for client-server application
- Socket is a special file used for inter-process communication
- It is a stream of data, very similar to network stream

The Parent-Child Relationship

- The file system in Unix is a collection of all of the related files (ordinary, directory, and device file) organized in a hierarchical structure.
- The implicit features of every UNIX file system is that there is a top, which serves as the reference point for all files.
- This top is called $root$ and it presented by a / (frontslash). $Root$ is actually a directory.
- It is conceptually different from the user-id root used by the system administrator to login.

The Parent-Child Relationship



The Parent-Child Relationship

- The root directory has a number of subdirectories under it.
- These directories, in turn, have more subdirectories and other files under them.
- Every file apartment route must have a parent, And it should be possible to trace the ultimate percentage of a file to root. The home directory is the parent of stu1 and mthomus.
- The parent is always a directory.

Pathname

- Listing directories that travel through along the path you take to get to the file.
- The path through the file system start at root(/), and the names of directories and files in a pathname are separate by slashes.

Absolute path and Relative path

- Absolute Pathname: path name that trace the path from root to a file are called full or absolute pathname.
- For example: $/home/user/email/inbox$.
- Relative Pathname: Specify a path to a file relative to your present directories
- For example: $cd\ inbox$

inode

- inodes are the data structure used for storing all information about a file except for its name and contents
- In a directory all files exist as entries with file name and the corresponding inode number used to fetch information
- an inode holds the following information:
  - ➢ creation or modification time
  - ➢ size
  - ➢ Permission
  - ➢ Owner

Single dot(.) and double dot(..)

- In Unix file system, the double dot(..) used to access the parent directory, whereas the single dot(.) represent the current directory.
- We can prefix a file name with a single dot(.) to hide it.

pwd command

- The PWD stands for print working directory
- PWD is an environment variable that stores the path of the current directory
- Syntax of pwd
- \$ $pwd\ [option]$
- $pwd\ \ -L$ #print the symbolic path
- pwd – P #print the actual path.

cd command

- cd Command is used to change the current directory
- Syntax: $\$\ cd\ [option]$
- $cd\ /$ – go to the root directory
- $cd \sim$ - Navigating directly to the home folder
- $cd\ ..$ – go to the parent directory relative to the current directory
- $cd\ ..\ /\ ..$- changes directory to the parent of the parent of the current directory

$mkdir$ command

- $mkdir$ Comment is used to create a directory
- Syntax: $mkdir\ [option]$

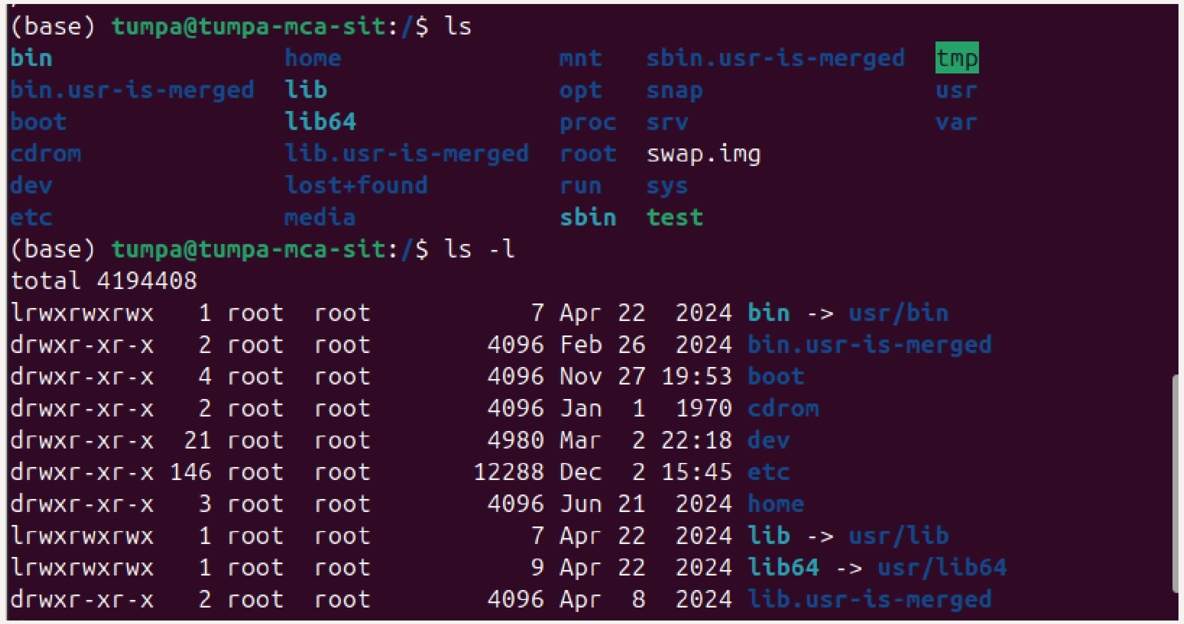
Options	Details
$-help$	Display help related information for the $mkdir$ Command and exit
$--version$	Display the version number
$-v$	Enables verbose mode, displaying a message for every directory created
$-p$	A flag that allows the creation of parent directories as necessary
$-m$	Set file modes or permission for the created directory

ls command

- ls command is used to list all files and directories in the terminal
- Syntax of ls command in unix: $\$ \, ls \, [option] \, [filename]$
- ls display content of current directory

Commonly used options in *ls* command

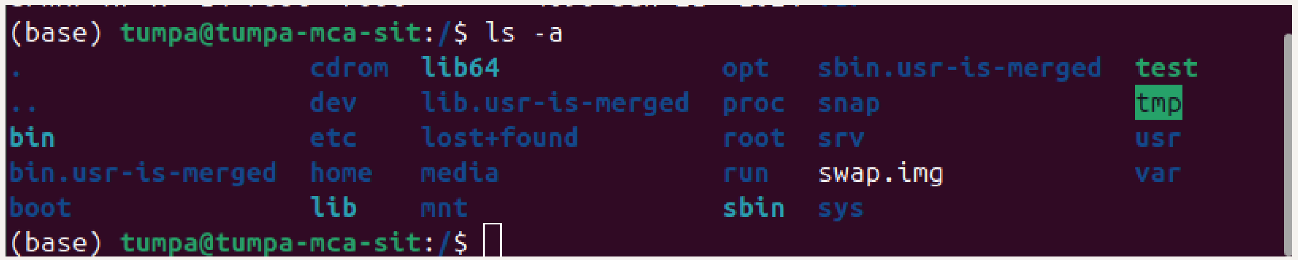
options	Description
$-l$	known as long format that displays detailed information about files and directories
$-a$	Represent all files include hidden files and directories in the listing
$-t$	Sort files and directories by their last modification time, displaying the most commonly recently modified ones first.
$-r$	known as reverse order
$-S$	Sorted files by their size, listing the largest ones first



```
(base) tumpa@tumpa-mca-sit:/$ ls
bin                 home                mnt   sbin.usr-is-merged  tmp
bin.usr-is-merged   lib                 opt   snap                usr
boot                lib64               proc  srv                 var
cdrom               lib.usr-is-merged   root  swap.img
dev                 lost+found          run   sys
etc                 media               sbin  test
(base) tumpa@tumpa-mca-sit:/$ ls -l
total 4194408
lrwxrwxrwx   1 root  root          7 Apr 22  2024 bin -> usr/bin
drwxr-xr-x   2 root  root       4096 Feb 26  2024 bin.usr-is-merged
drwxr-xr-x   4 root  root       4096 Nov 27 19:53 boot
drwxr-xr-x   2 root  root       4096 Jan  1  1970 cdrom
drwxr-xr-x  21 root  root       4980 Mar  2 22:18 dev
drwxr-xr-x 146 root  root      12288 Dec  2 15:45 etc
drwxr-xr-x   3 root  root       4096 Jun 21  2024 home
lrwxrwxrwx   1 root  root          7 Apr 22  2024 lib -> usr/lib
lrwxrwxrwx   1 root  root          9 Apr 22  2024 lib64 -> usr/lib64
drwxr-xr-x   2 root  root       4096 Apr  8  2024 lib.usr-is-merged
```

Commonly used options in ls command

Option	Description
$-R$	List files and directories recursively
$-i$	Known as inode which displays the index number of each file
$-g$	Known as group, displays group ownership
$-h$	Print file size in human readable format
$-d$	List directories themselves



```
(base) tumpa@tumpa-mca-sit:/$ ls -a
.                   cdrom  lib64                opt   sbin.usr-is-merged  test
..                  dev    lib.usr-is-merged    proc  snap                tmp
bin                 etc    lost+found           root  srv                 usr
bin.usr-is-merged   home   media                run   swap.img            var
boot                lib    mnt                  sbin  sys
(base) tumpa@tumpa-mca-sit:/$ 
```

- Field 1: file permission
- Field 2: number of link
- Field 3: owner
- Field 4: group
- Field 5: size
- Feel 6: last modified date and time
- Field 7: file name

Important File System in UNIX

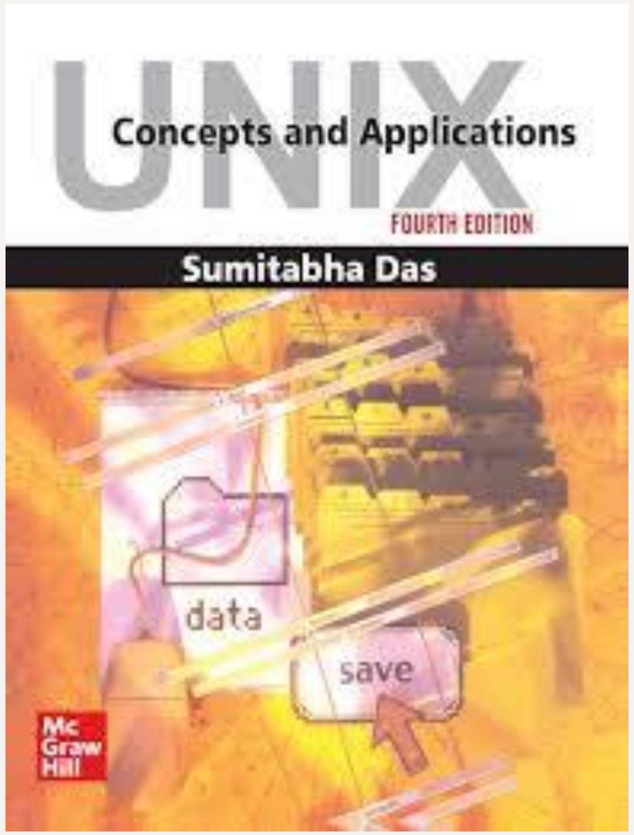
- $/bin$ and $/user/bin$: These are the directories where all commonly used unix commands are found. Note that path variable always shows these.
- $/sbin$ and $/usr/sbin$: If there is a command that you cannot execute but the system administrator then it would be in one of these directories. Only the system administrator's path shows these directories.
- $/etc$: This directory contains configuration files of the system. Your login name and password are stored in files
- $/dev$:This directory contains all device files. These files do not occupy space on disk. There would be more subdirectories like pts, dsk and rdsk in this directory.

Important File System in UNIX

- $/lib$ and $/usr/lib$ - contain all library files in binary form, You will need to link your C programs with files in these directories.
- $/usr/include$ - Contains the standard header files used by C programs. The statement #include<stdio.h> used in most C programs refers to the files stdio.h in this directory.
- $/usr/share/man$ -this is where man pages are stored. There are separate subdirectories here that contain pages for each section.

QUIZ

Reference


UNIX
Concepts and Applications
FOURTH EDITION
Sumitabha Das
data
save
Mc Graw Hill