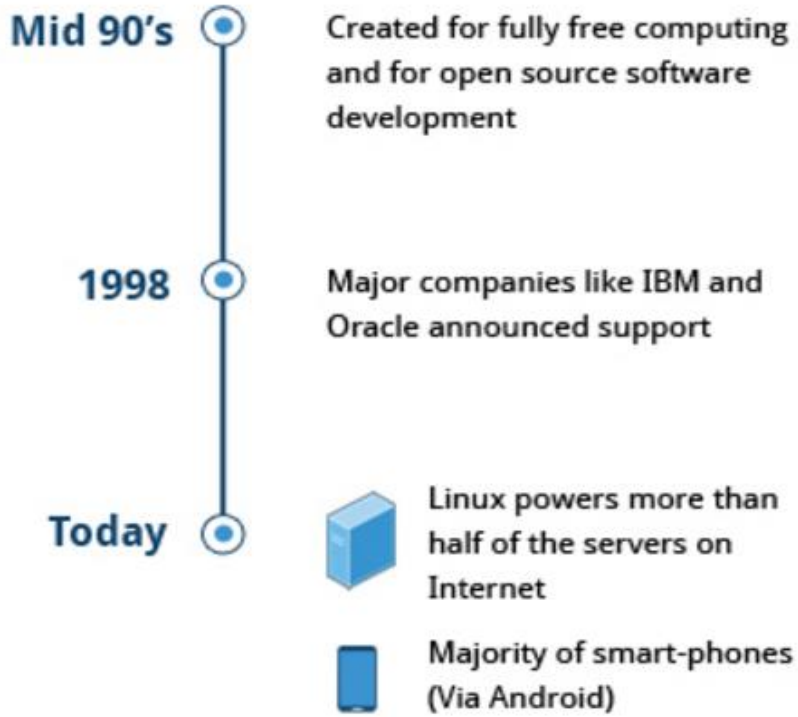
Linux OS Concepts

DevOps Training

Linux History

- Smart phones
- Smart TVs
- Google, Twitter, Facebook, and Amazon are all powered by Linux.
- Built collaboratively across companies, geographies, and market.
- Unmatched Rate of development

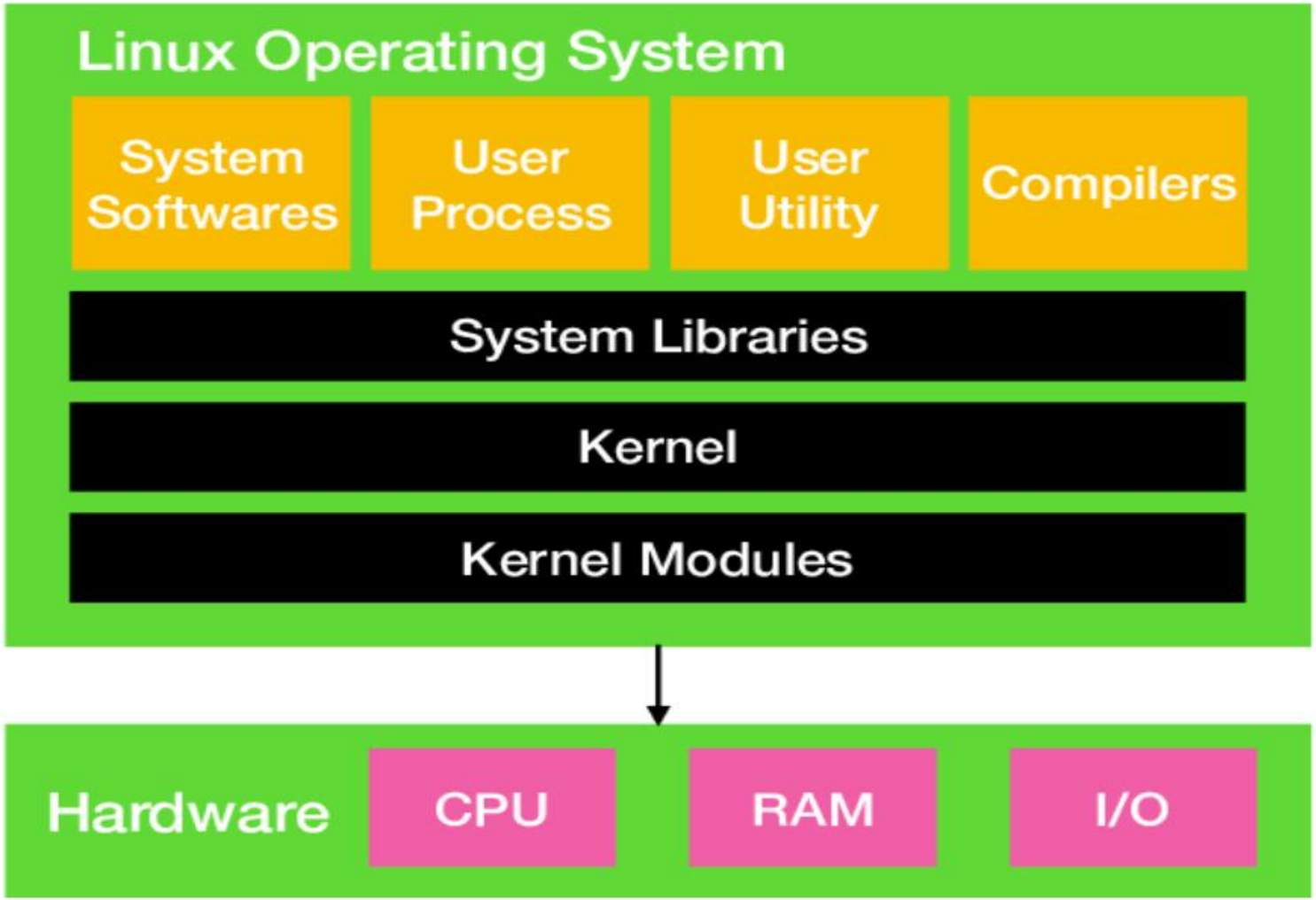
Linus Torvalds



LINUX PHILOSOPHY

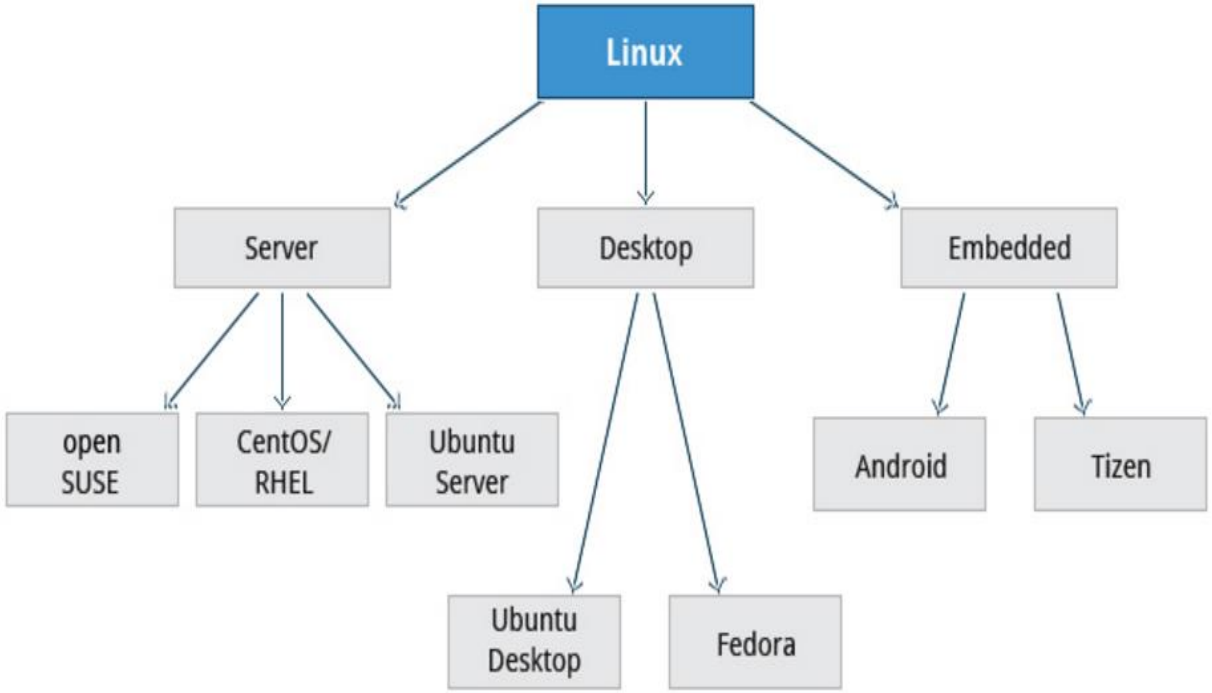
Free and open source version of **UNIX**
Which gives ability of multitasking and gives multiuser functionality on single host

LINUX TOPOLOGY

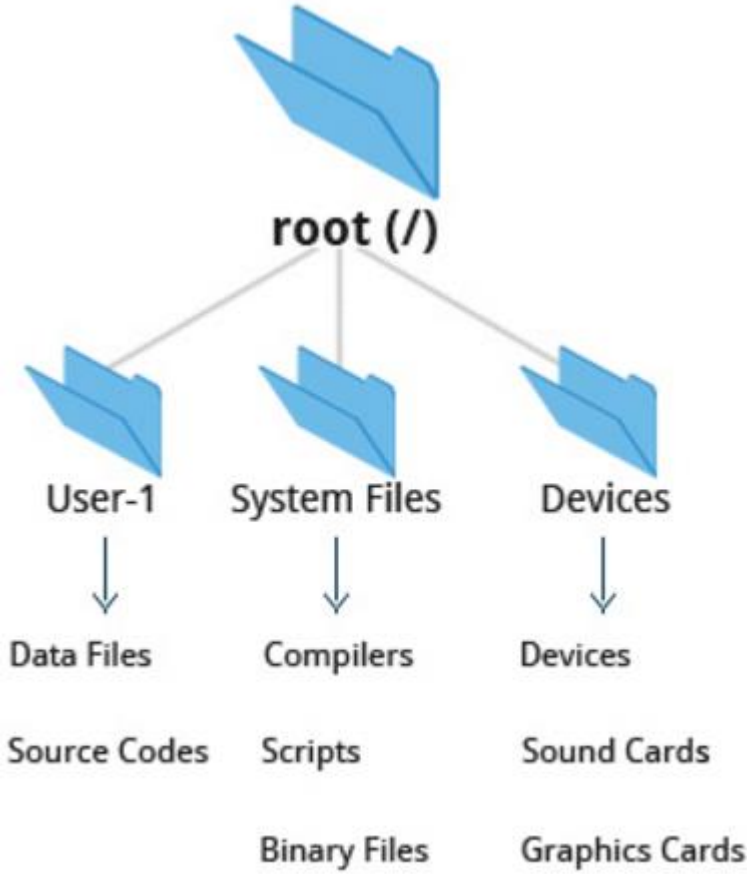


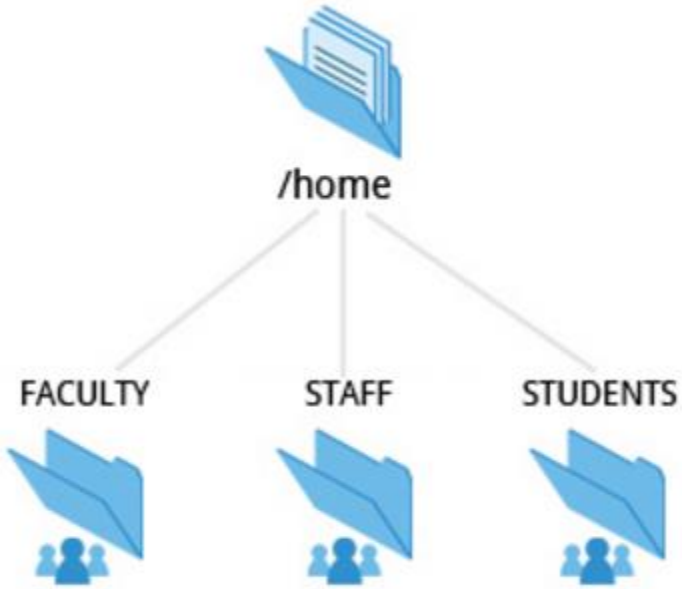
LINUX TERMINOLOGY

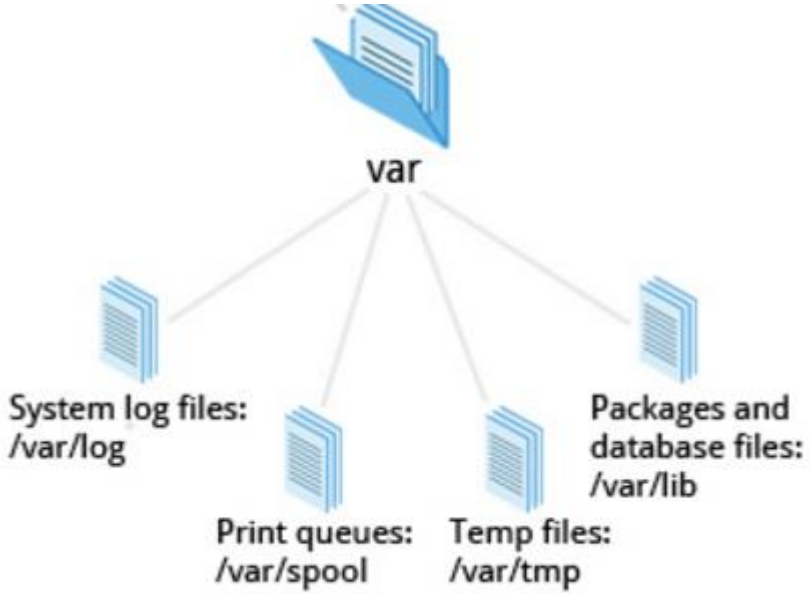
- kernel
- distribution
- service
- filesystem
- command line interface (CLI)
- Shell



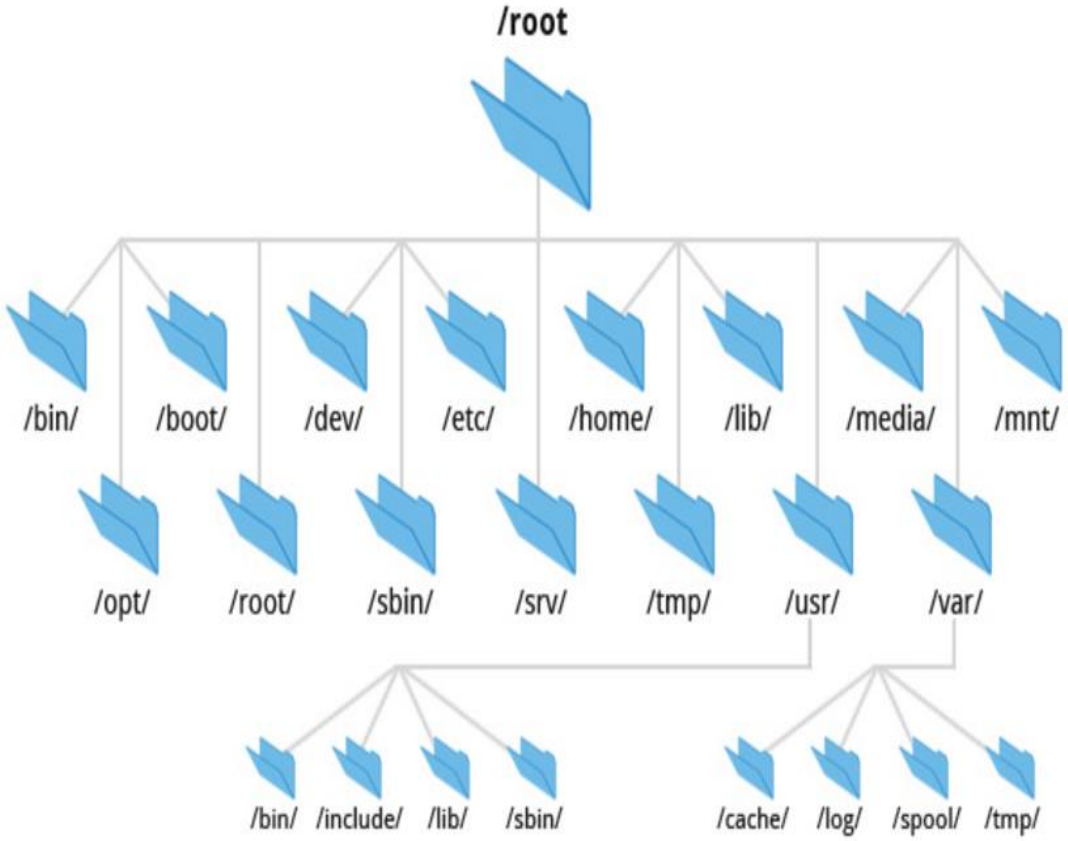
LINUX FILESYSTEM







FILE SYSTEM TREE

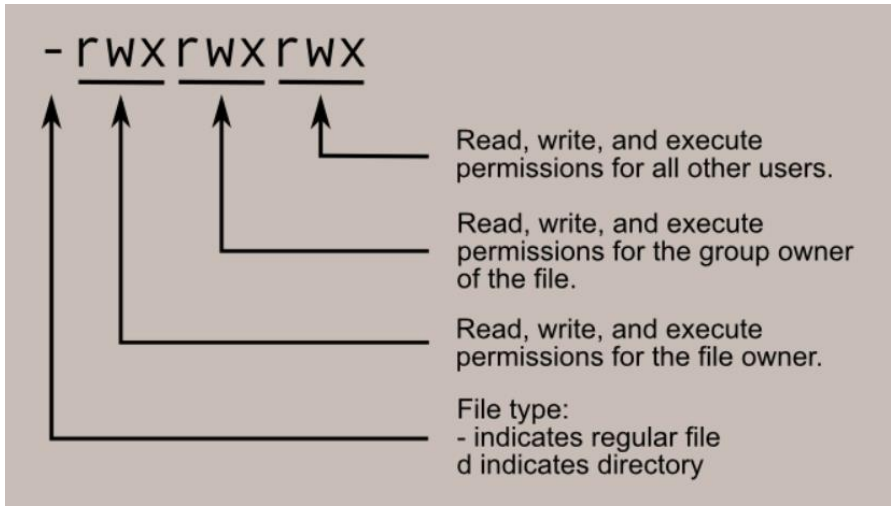


Directory	Description
bin	Essential command binaries
boot	Static files of the boot loader
dev	Device files
etc	Host-specific system configuration
lib	Essential shared libraries and kernel modules
media	Mount point for removeable media
mnt	Mount point for mounting a filesystem temporarily
opt	Add-on application software packages
sbin	Essential system binaries
srv	Data for services provided by this system
tmp	Temporary files
usr	Secondary hierarchy
var	Variable data

LINUX FILE PERMISSIONS

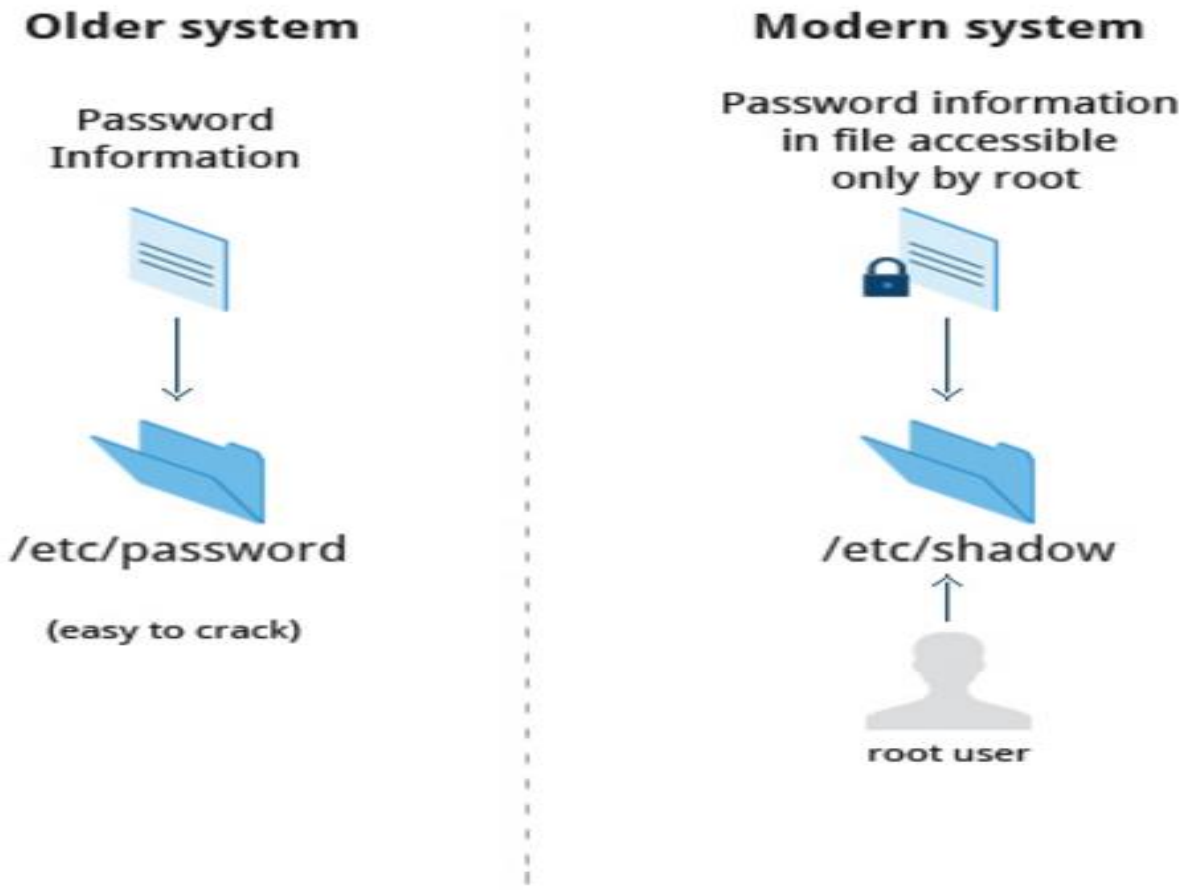
```
[me@linuxbox me]$ ls -l /bin/bash

-rwxr-xr-x 1 root root  316848 Feb 27  2000 /bin/bash
```

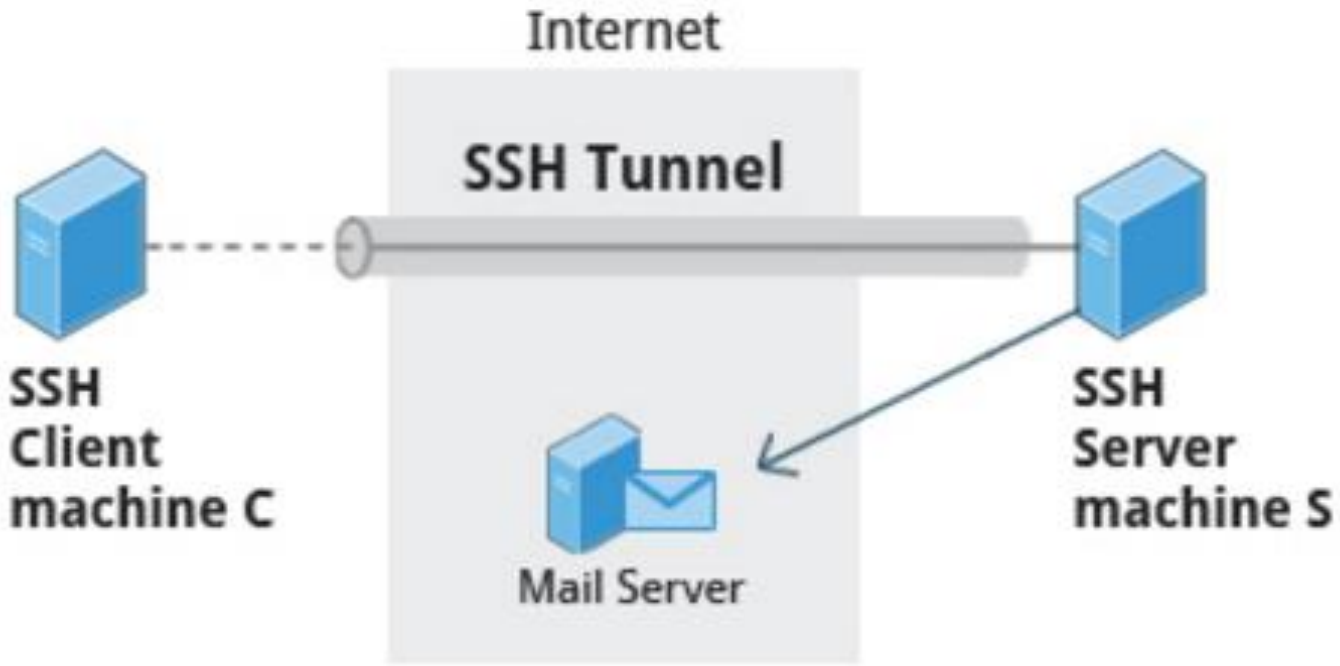


Value	Meaning
777	**(rwxrwxrwx)** No restrictions on permissions. Anybody may do anything. Generally not a desirable setting.
755	**(rwxr-xr-x)** The file's owner may read, write, and execute the file. All others may read and execute the file. This setting is common for programs that are used by all users.
700	**(rwx------)** The file's owner may read, write, and execute the file. Nobody else has any rights. This setting is useful for programs that only the owner may use and must be kept private from others.
666	**(rw-rw-rw-)** All users may read and write the file.
644	**(rw-r--r--)** The owner may read and write a file, while all others may only read the file. A common setting for data files that everybody may read, but only the owner may change.
600	**(rw-------)** The owner may read and write a file. All others have no rights. A common setting for data files that the owner wants to keep private.

HOW PASSWORDS ARE STORED



SSH



ssh user@ip-address
ssh –i key.pem user@ip-address

COPYING FILES SECURELY WITH SCP

```
scp <localfile> <user@remotesystem>:/home/user/
```

scp finename user@ip:/home/vkiran/

scp –i key.pem filename user@ip:/home/vkiran/

scp username@hostname:/location/filename /home/raj/

FILE SYSTEM

	Windows	Linux
Partition	Disk1	/dev/sda1
Filesystem type	NTFS/FAT32	EXT3/EXT4/XFS...
Mounting Parameters	DriveLetter	MountPoint
Base Folder where OS is stored	C drive	/

LINUX COMMANDS

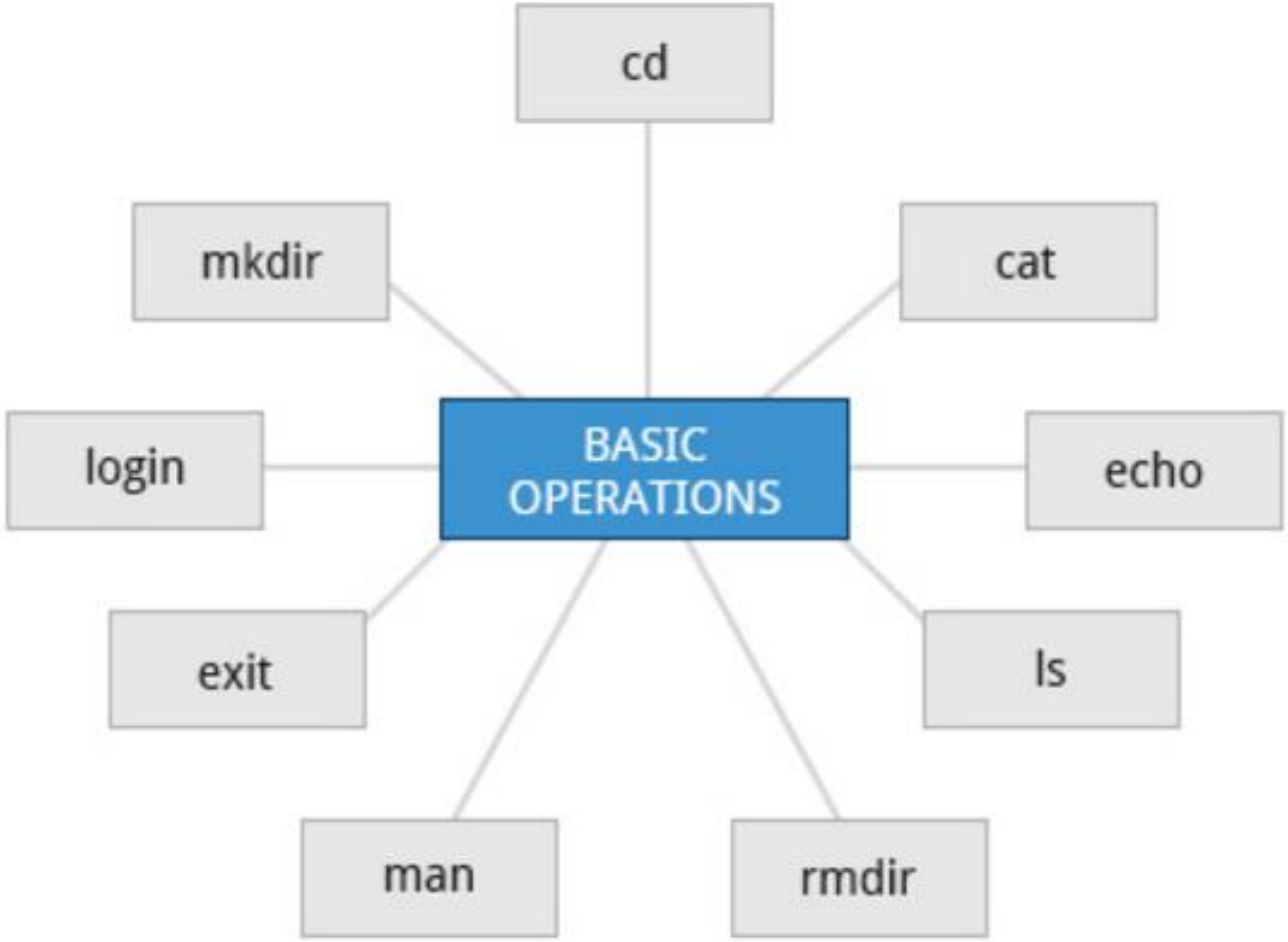
Commands are
CASE SENSITIVE!

- ✓ Commands tell the operating system to perform set of operations
- ✓ The syntax form of the commands are
  Command options arguments

Getting Help:

- ✓ In LINUX/UNIX whenever you need help with a command type "man" followed by the command name
- ✓ The Syntax is
  man [options] command
- ✓ Common options are
  -M Keyword path to man pages
  -k Keyword list command for all keyword matches
- ✓ We can use help command also
  command - -help

BASIC OPERATIONS



EXPLORING THE FILESYSTEM

Command	Usage
`cd /`	**C**hanges your current **d**irectory to the root (/) directory (or path you supply)
`ls`	**Li**st the contents of the present working directory
`ls -a`	**Li**st **all** files including **hidden** files and directories (those whose name start with .)
`tree`	Displays a **tree** view of the filesystem

Absolute path: $ cd /usr/bin

Relative path: $ cd ../../usr/bin

SED

Command	Usage
`sed s/pattern/replace_string/ file`	Substitute first string occurrence in a line
`sed s/pattern/replace_string/g file`	Substitute all string occurrences in a line
`sed 1,3s/pattern/replace_string/g file`	Substitute all string occurrences in a range of lines
`sed -i s/pattern/replace_string/g file`	Save changes for string substitution in the same file

AWK (ALFRED AHO, PETER WEINBERGER, AND BRIAN KERNIGHAN)

Command	Usage
`awk '{ print $0 }' /etc/passwd`	Print entire file
`awk -F: '{ print $1 }' /etc/passwd`	Print first field (column) of every line, separated by a space
`awk -F: '{ print $1 $6 }' /etc/passwd`	Print first and sixth field of every line

GREP

Command	Usage
`grep [pattern] <filename>`	Search for a pattern in a file and print all matching lines
`grep -v [pattern] <filename>`	Print all lines that do **not** match the pattern
`grep [0-9] <filename>`	Print the lines that contain the numbers 0 through 9
`grep -C 3 [pattern] <filename>`	Print context of lines (specified number of lines above and below the pattern) for matching the pattern. Here the number of lines is specified as 3.

EXECUTING PREVIOUS COMMANDS

Syntax	Task
`!`	Start a history substitution
`!$`	Refer to the last argument in a line
`!n`	Refer to the n^{th} command line
`!string`	Refer to the most recent command starting with string

ARCHIVE FILES

zip

unzip

tar → c – create v – verbose f – file x – extract

gzip

dd → dd

ADDITIONAL COMMANDS

Download

- wget
- curl

Memory

- du
- df
- free
- vmstat

CRONTAB

```
*     *     *   *    *        command to be executed
-     -     -   -    -
|     |     |   |    |
|     |     |   |    +----- day of week (0 - 6) (Sunday=0)
|     |     |   +------- month (1 - 12)
|     |     +--------- day of        month (1 - 31)
|     +----------- hour (0 - 23)
+------------- min (0 - 59)
```

min	hour	day/month	month	day/week	Execution time
30	0	1	1,6,12	*	— 00:30 Hrs on 1st of Jan, June & Dec.
0	20	*	10	1-5	–8.00 PM every weekday (Mon-Fri) only in Oct.
0	0	1,10,15	*	*	— midnight on 1st ,10th & 15th of month
5,10	0	10	*	1	— At 12.05,12.10 every Monday & on 10th of every month

THANK YOU!

training@laksans.com