

Introduction to Linux

Mukesh Pund

Principal Scientist

NISCAIR, New Delhi, India

History

- In 1969, a team of developers developed a new operating system called "Unix" which was written using C
- Linus Torvalds, a young man studying computer science at the university of Helsinki developed academic version of Unix which is named as Linux.
- Linux is a full UNIX clone.

Linux a powerful OS!

- Today Linux has joined the desktop market.
- On the server side, Linux is well-known as a stable and reliable platform.
- Linux provides many applications like:
 - Databases (MySQL,Postgresql),
 - Network services(Web Servers,DNS, Proxy, firewall etc)
 - Software development tools(C, Java, Python,Perl etc.)
 - Office automation tools
 - And many more...

Is Linux difficult?

- There is excellent and free Internet support and documentation available.
- The graphical user interface (GUI) is similar in design to that on any other system
- A very powerful command line alternative is also available.
- Linux is user friendly.

Properties of Linux

- **It is Open Source**
 - **Today, Linux is ready to accept the challenge of a fast-changing world.**
- **Linux is free:**
 - **If you want to spend absolutely nothing, you don't even have to pay the price of a CD.**
 - **Linux can be downloaded in its entirety from the Internet completely for free.**

Properties of Linux

- Linux is portable to any hardware platform.
- Linux was made to keep on running.
 - **As with UNIX, a Linux system expects to run without rebooting all the time.**
 - **Tasks can be scheduled to run at suitable times.**

Properties of Linux

- **Linux is secure and versatile.**
 - **The security model used in Linux is based on the UNIX idea of security which is robust.**
 - **It is less prone to virus attacks.**
- **Linux is scalable**

Commands..

- Let's have an overview of frequently used commands in Linux.

Note: Some commands can only be executed by super user (example adduser, shutdown etc).

Creating a new user

- Use the `useradd` command
- Use the `passwd` command to set password
- Try it... logon as root

```
[root@mukesh]# useradd sdc1
[root@mukesh]# passwd sdc1
Changing password for user sdc1
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated
successfully
[root@mukesh]#
```

What is a Shell?

- Is a program that takes your commands from the keyboard and gives them to the operating system to perform
- An interface between the Linux system and the user
- Used to call commands and programs
- Many available (bsh; csh; bash; etc.)

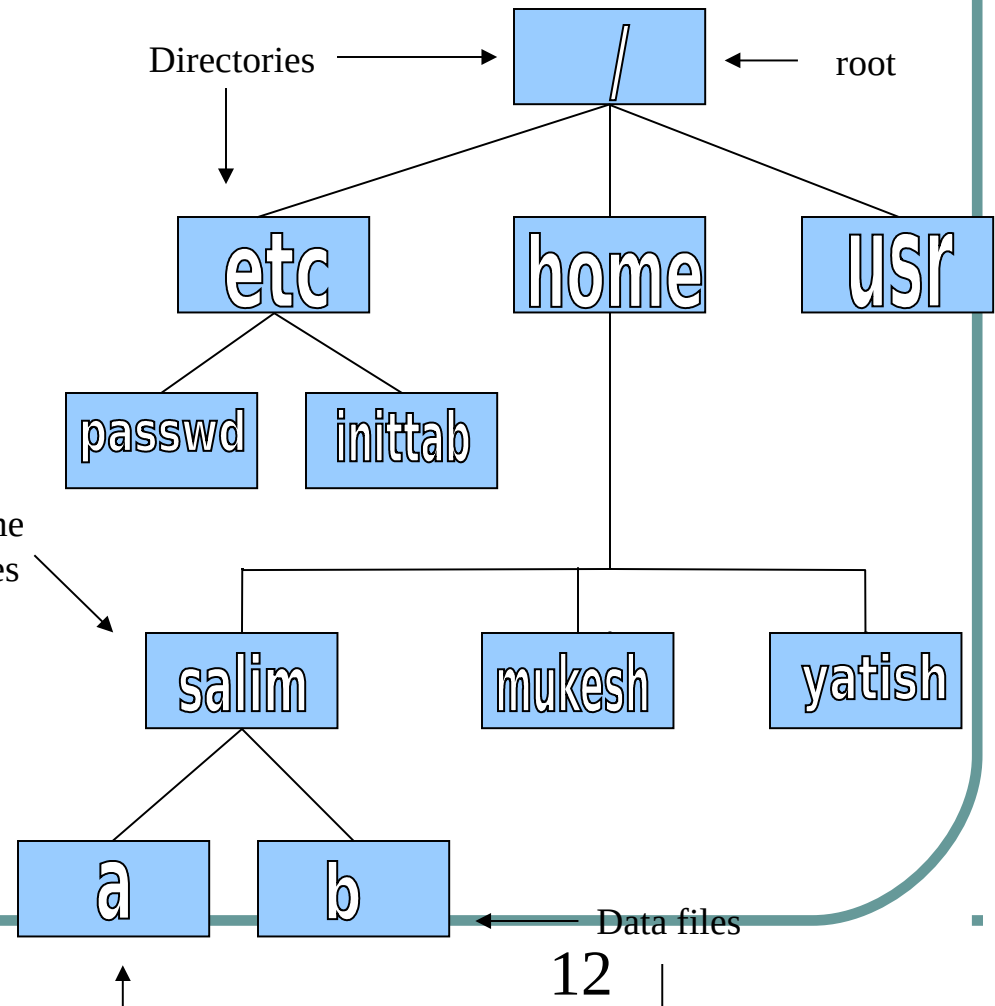
You need help? Add more

- In Linux help can be accessed by command `man` (manual)
 - Use `man <command>` to display help for that command

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



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

Special Files

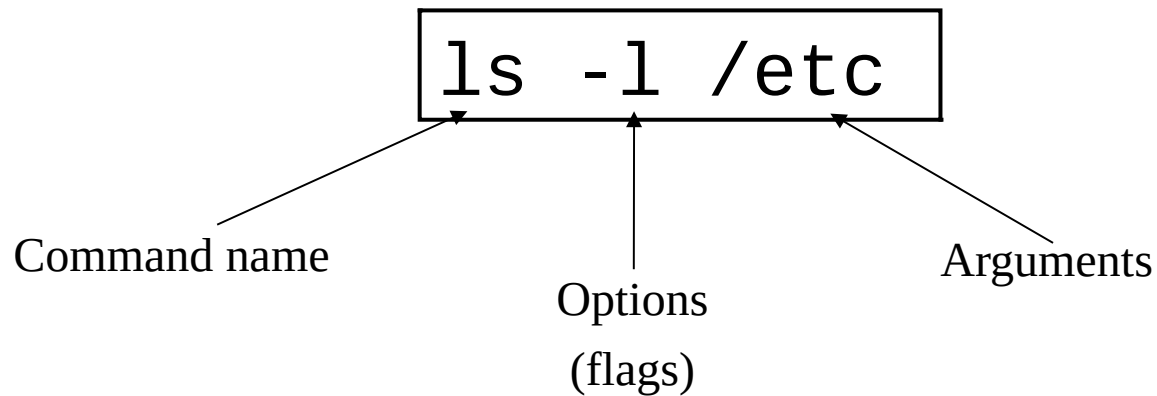
- `/`: The root directory where the file system begins.
- `/boot`: This is where the Linux kernel is kept.
- `/etc`: The `/etc` directory contains the configuration files for the system.
- `/bin`, `/usr/bin`: These two directories contain most of the programs for the system. The `/bin` directory has the essential programs that the system requires to operate, while `/usr/bin` contains applications for the system's users.

Special Files

- `/sbin, /usr/sbin`: The `sbin` directories contain programs for system administration, mostly for use by the superuser.
- `/usr`: The `/usr` directory contains a variety of things that support user applications
- `/lib`: The shared libraries (similar to DLLs in that other operating system) are kept here.
- `/home`: `/home` is where users keep their personal work.
- `/root`: This is the superuser's home directory.

Linux Command Basics

- To execute a command, type its name and arguments at the command line



Command Options

- Command options allow you to control a command to a certain degree
- Conventions:
 - Usually being with a single dash and are a single letter (“-l”)
 - Sometimes have double dashes followed by a keyword (“--help”)

Navigation and Looking Around

- `pwd` - print (display) the working directory
- `cd <dir>` - change the current working directory to `dir`
`cd ..`
- `ls` - list the files in the current working directory
- `ls -l` - list the files in the current working directory in long format

File and Directory Manipulation

- `cp <fromfile> <tofile>`
 - Copy from the <fromfile> to the <tofile>
- `mv <fromfile> <tofile>`
 - Move/rename the <fromfile> to the <tofile>
- `rm <file>`
 - Remove the file named <file>
- `mkdir <newdir>`
 - Make a new directory called <newdir>
- `rmdir <dir>`
 - Remove an (empty) directory
- `cat > <file>`
 - Create file <file>

Data display from files

- `cat <file>`
Displays contents of the <file>
- `head -n <fromfile>`
Displays n lines from top of the <fromfile>
- `tail -n <fromfile>`
Displays n lines from bottom of <fromfile>

Standard Files

- UNIX concept of “standard files”
 - standard input (where a command gets its input) - default is the terminal
 - standard output (where a command writes its output) - default is the terminal
 - standard error (where a command writes error messages) - default is the terminal

Redirecting Output

- The output of a command may be sent (piped) to a file:

```
ls -l >output
```

“>” is used to specify the output file

```
ls >>output
```

“>>” is used to append to output

Redirecting Input

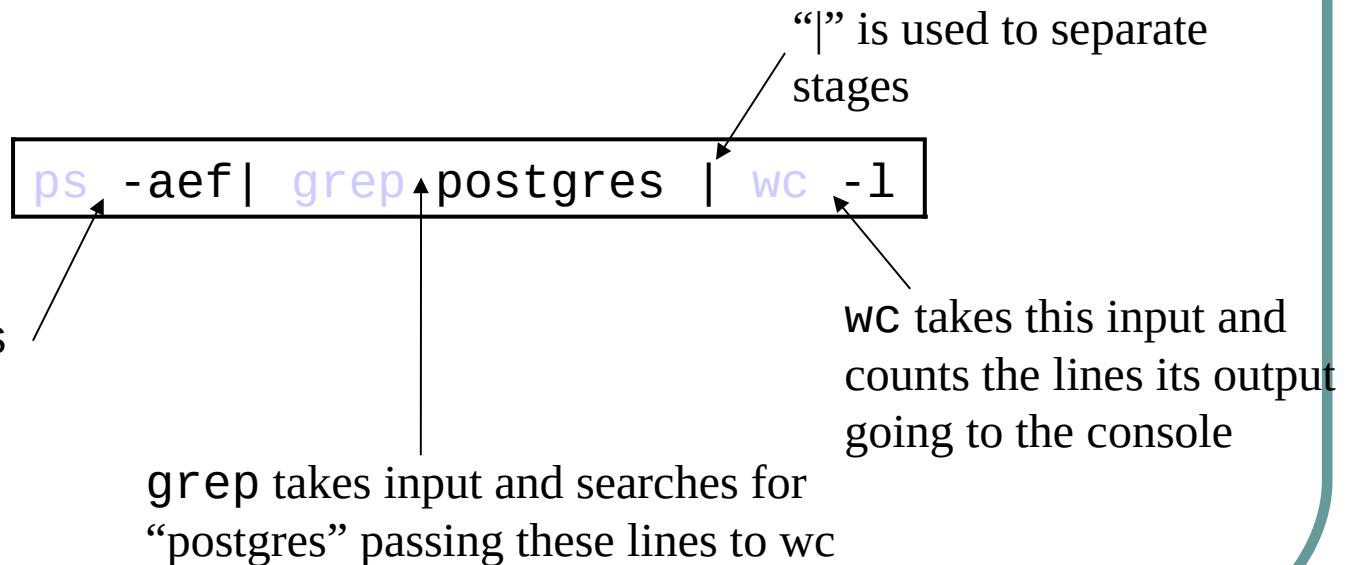
- The input of a command may come (be piped) from a file:

```
wc <input
```

“<” is used to specify the input file

Connecting commands with Pipes

- The output of one command can become the input of another:



More Commands

- `who`
 - List who is currently logged on to the system
- `who am i`
 - Report what user you are logged on as
- `ps`
 - List your processes on the system
- `ps -aef`
 - List all the processes on the system
- `echo "A string to be echoed"`
 - Echo a string (or list of arguments) to the terminal

More Commands

- `grep` - Searches files for one or more pattern arguments. It does plain string, basic regular expression, and extended regular expression searching

Example: `ls -l |grep "mukesh"`

`ls` command display the listing of files in current directory. And `grep` command searches for "mukesh" file in that listing.

More Commands

- `kill` - sends a signal to a process or process group
- You can only kill your own processes unless you are root

Example:

```
[root@mukesh log]# ps -aef
```

Above command will display result like:

```
[root@mukesh log]#
```

UID	PID	PPID	C	STIME	TTY	TIME	CMD
root	6715	6692	2	14:34	ttyp0	00:00:00	sleep 10h
root	6716	6692	0	14:34	ttyp0	00:00:00	ps -ef

And one can kill the process by following command:

```
[root@mukesh log]# kill 6715
```

```
[1]+  Terminated                sleep 10h
```

More Commands

- `tar` - manipulates archives
 - An archive is a single file that contains the complete contents of a set of other files; an archive preserves the directory hierarchy that contained the original files.

```
tar -zxvf imap-4.7.tar.gz
imap-4.7/
imap-4.7/src/
imap-4.7/src/c-client/
imap-4.7/src/c-client/env.h
imap-4.7/src/c-client/fs.h
```

Switching Users

- `su <accountname>`
 - switch user accounts. You will be prompted for a password. When this command completes, you will be logged into the new account. Type `exit` to return to the previous account
- `su`
 - Switch to the root user account. Do not do this lightly

Note: The root user does not need to enter a password when switching users. It may become any user desired. This is part of the power of the root account.

PATH Environment Variable

- Controls where commands are found
 - PATH is a list of directory pathnames separated by colons. For example:
`PATH=/bin:/usr/bin:/usr/X11R6/bin:/usr/local/bin:/home/scully/bin`
 - If a command does not contain a slash, the shell tries finding the command in each directory in PATH. The first match is the command that will run

File and Directory Permissions

- **Every file or directory**
 - Is owned by someone
 - Belongs to a group
 - Has certain access permissions for owner, group, and others
 - Default permissions determined by `umask`

File and Directory Permissions

- The long version of a listing (`ls -l`) will display the file permissions:

```
-rwxrwxr-x  1 rvdheij  rvdheij    5224 Dec 30 03:22 hello
-rw-rw-r--  1 rvdheij  rvdheij     221 Dec 30 03:59 hello.c
-rw-rw-r--  1 rvdheij  rvdheij    1514 Dec 30 03:59 hello.s
drwxrwxr-x  7 rvdheij  rvdheij    1024 Dec 31 14:52 posixuft
```

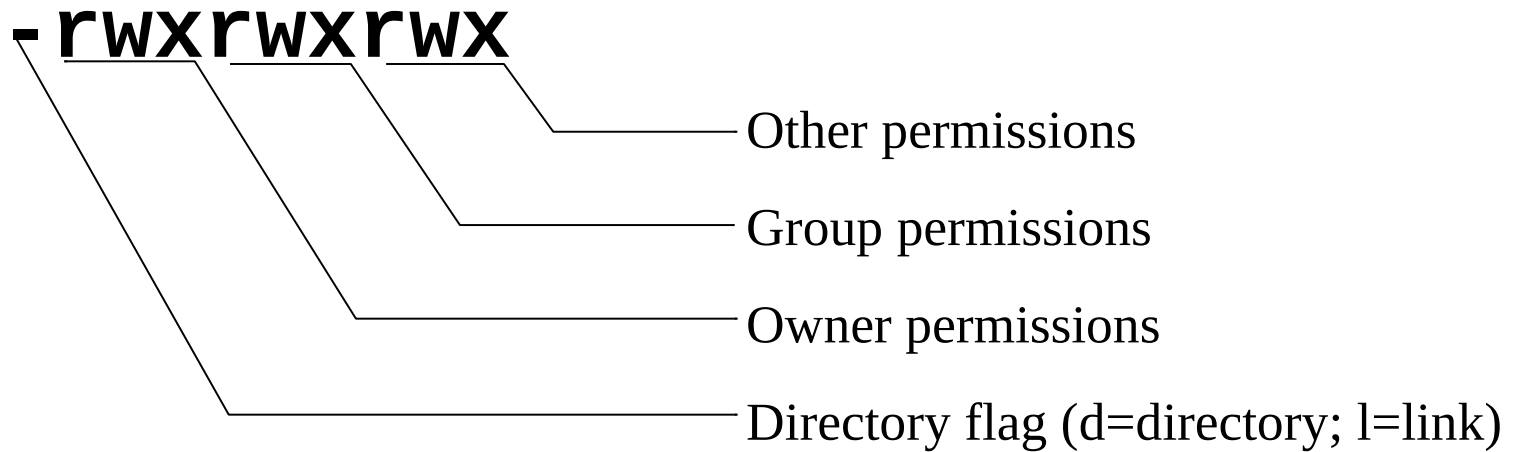
Permissions

Owner

Group

Interpreting Permissions

- rwxrwxrwx



Changing Permissions

- Use the `chmod` command to change file or directory permissions

```
rwX rwX rwX = 111 111 111 = 777
```

```
rw- rw- rw- = 110 110 110 = 666
```

```
rwX --- --- = 111 000 000 = 700
```

```
chmod 755 file # Owner=rwx Group=r-x Other=r-x
```

```
chmod 500 file2 # Owner=r-x Group=--- Other=---
```

```
chmod 644 file3 # Owner=rw- Group=r-- Other=r--
```

```
chmod +x file # Add execute permission to file for all
```

```
chmod o-r file # Remove read permission for others
```

```
chmod a+w file # Add write permission for everyone
```

Changing ownership

- `chown` - change file ownership

```
chown name some_file
```

- `chgrp` - change a file's group ownership

```
chgrp new_group some_file
```

Processes

- As with any multitasking operating system, Linux executes multiple, simultaneous processes.
- Processes are created in a hierarchical structure whose depth is limited only by the virtual memory available to the virtual machine
- A process may control the execution of any of its descendants by suspending or resuming it, altering its relative priority, or even terminating it
- Termination of a process by default causes termination of all its descendants; termination of the root process causes termination of the session
- Linux assigns a *process ID* (PID) to the process

Processes

- **Foreground**
 - When a command is executed from the prompt and runs to completion at which time the prompt returns is said to run in the foreground
- **Background**
 - When a command is executed from the prompt with the token “&” at the end of the command line, the prompt immediately returns while the command continues is said to run in the background

Process Control Commands

- `ps` - list the processes running on the system
- `kill` - send a signal to one or more processes (usually to "kill" a process)

Process Control Commands

```
$ ps
PID TTY TIME CMD
1280 pts/5 00:00:00 bash
1293 pts/5 00:00:00 xload
1294 pts/5 00:00:00 ps
```

```
$ kill -9 1293
[2]+ Terminated xload
```

Processes

& causes process to be run in "background"

```
[root@mukesh log]# sleep 10h &
[1] 6718 ←
[root@mukesh log]# ps
UID          PID    PPID  C  STIME TTY          TIME CMD
root         6718   6692  0  14:49 ttys0        00:00:00 sleep 10h
```

Job Number

Process ID (ID)

Parent Process ID

Editors

- **Several choices available:**
 - `vi` Standard UNIX editor
 - `xedit` X windows text editor
 - `emacs` Extensible, Customizable Self-Documenting Display Editor
 - `pico` Simple display-oriented text editor

Thanks