



ANUGRAH MEMORIAL COLLEGE, GAYA

A Constituent Unit of Magadh University, Bodh-Gaya

**Department of Computer Application
and Information Technology**

LINUX PROGRAMMING

LECTURE NOTES

**BCA & B.Sc. (IT)
(1st Year) (2024-27)**

PREPARED BY:

Mr. Prince Kumar

Assistant Professor

UNIT-I

LINUX UTILITIES AND SHELL PROGRAMMING

I. Introduction toLinux:

Linux is a Unix-like computer operating system assembled under the model of free and opensource software development and distribution. The defining component of Linux is the Linuxkernel, an operating system kernel first released 5 October 1991 by Linus Torvalds.

Linux was originally developed as a free operating system for Intel x86-based personal computers. It has since been ported to more computer hardware platforms than any other operatingsystem. It isa leadingoperatingsystemon serversandother big ironsystemssuchas mainframe computers and supercomputers more than 90% of today's 500 fastestsupercomputers run some variant of Linux, including the 10 fastest. Linux also runs on embedded systems (deviceswhere the operating systemistypically built into the firmware and highly tailored to the system) such as mobile phones, tablet computers, network routers, televisions and video game consoles; the Android system in wide use on mobile devices isbuilt on the Linux kernel.

Basic Features

FollowingaresomeoftheimportantfeaturesofLinuxOperatingSystem.

- **Portable** - Portability means softwares can works on different types of hardwares in sameway.Linuxkernelandapplicationprogramssupportstheir installationonanykind of hardware platform.
- **Open Source** - Linux source code is freely available and it is community based development project.Multipleteamsworksincollaborationto enhancethecapability of Linux operating system and it is continuously evolving.
- **Multi-User**-Linuxisamultiusersystemmeansmultipleuserscanaccesssystem resources like memory/ ram/ application programs at same time.
- **Multiprogramming** - Linux is a multiprogramming system means multiple applications can run at same time.
- **HierarchicalFileSystem** -Linuxprovidesastandard filestructureinwhichsystem

files/userfiles are arranged.

- **Shell** - Linux provides a special interpreter program which can be used to execute commands of the operating system. It can be used to do various types of operations, call application programs etc.
- **Security** - Linux provides user security using authentication features like password protection/ controlled access to specific files/ encryption of data.

Linux Advantages

1. **Low cost:** You don't need to spend time and money to obtain licenses since Linux and much of its software come with the GNU General Public License. You can start to work immediately without worrying that your software may stop working anytime because the free trial version expires. Additionally, there are large repositories from which you can freely download high quality software for almost any task you can think of.
2. **Stability:** Linux doesn't need to be rebooted periodically to maintain performance levels. It doesn't freeze up or slow down over time due to memory leaks and such. Continuous up-times of hundreds of days (up to a year or more) are not uncommon.
3. **Performance:** Linux provides persistent high performance on workstations and on networks. It can handle unusually large numbers of users simultaneously, and can make old computers sufficiently responsive to be useful again.
4. **Network friendliness:** Linux was developed by a group of programmers over the Internet and has therefore strong support for network functionality; client and server systems can be easily set up on any computer running Linux. It can perform tasks such as network backups faster and more reliably than alternative systems.
5. **Flexibility:** Linux can be used for high performance server applications, desktop applications, and embedded systems. You can save disk space by only installing the components needed for a particular use. You can restrict the use of specific computers by installing for example only selected office applications instead of the whole suite.
6. **Compatibility:** It runs all common Unix software packages and can process all common file formats.
7. **Choice:** The large number of Linux distributions gives you a choice. Each distribution is developed and supported by a different organization. You can pick the one you like best; the

core functionalities are the same; most software runs on most distributions.

8. **Fast and easy installation:** Most Linux distributions come with user-friendly installation and setup programs. Popular Linux distributions come with tools that make installation of additional software very user friendly as well.

9. **Full use of hard disk:** Linux continues to work well even when the hard disk is almost full.

10. **Multitasking:** Linux is designed to do many things at the same time; e.g., a large printing job in the background won't slow down your other work.

11. **Security:** Linux is one of the most secure operating systems. "Walls" and flexible file access permission systems prevent access by unwanted visitors or viruses. Linux users have the option to select and safely download software, free of charge, from online repositories containing thousands of high quality packages. No purchase transactions requiring credit card numbers or other sensitive personal information are necessary.

12. **Open Source:** If you develop software that requires knowledge or modification of the operating system code, Linux's source code is at your fingertips. Most Linux applications are Open Source as well.

Difference between UNIX and LINUX

Features	LINUX	UNIX
Cost	Linux can be freely distributed, downloaded freely, distributed through magazines, Books etc. There are priced versions for Linux also, but they are normally cheaper than Windows.	Different flavors of Unix have different cost structures according to vendors
Development and Distribution	Linux is developed by Open Source development i.e. through sharing and collaboration of code and features through forum etc and it is distributed	Unix systems are divided into various other flavors, mostly developed by AT&T as well as various commercial vendors and non-profit organizations.

	by various vendors.	
Manufacturer	Linux kernel is developed by the community. Linus Torvalds oversees things.	Three biggest distributions are Solaris(Oracle), AIX(IBM)&HP-UX Hewlett Packard. And Apple makes OSX, an Unix based OS..
User	Everyone. From home users to developers and computer enthusiasts alike.	Unix operating systems were developed mainly for mainframes, servers and workstations except OSX, which is designed for everyone. The Unix environment and the client-server program model were essential elements in the development of the Internet
Usage	Linux can be installed on a wide variety of computer hardware, ranging from mobile phones, tablet computers and video game consoles , to mainframes and supercomputers.	The UNIX operating system is used in internet servers, workstations & PCs. Backbone of the majority of finance infrastructure and many 24x365 high availability solutions.
File system support	Ext2, Ext3, Ext4, Jfs, ReiserFS, Xfs, Btrfs, FAT, FAT32 , NTFS	jfs, gpfs, hfs, hfs+, ufs, xfs, zfs format
Text mode interface	BASH (Bourne Again Shell) is the Linux default shell. It can support multiple command interpreters.	Originally the Bourne Shell. Now it's compatible with many others including BASH, Korn & C.
What is it?	Linux is an example of Open Source software development and Free Operating System (OS).	Unix is an operating system that is very popular in universities, companies, big enterprises etc.

GUI	Linux typically provides two GUIs, KDE and Gnome . But there are millions of alternatives such as LXDE, Xfce, Unity, Mate, twm, etc.	Initially Unix was a command based OS, but later a GUI was created called Common Desktop Environment. Most distributions now ship with Gnome.
Price	Free but support is available for a price.	Some free for development use (Solaris) but support is available for a price.
Security	Linux has had about 60-100 viruses listed till date. None of them actively spreading nowadays.	A rough estimate of UNIX viruses is between 85-120 viruses reported till date.
Threat detection and solution	In case of Linux, threat detection and solution is very fast, as Linux is mainly community driven and whenever any Linux user posts any kind of threat, several developers start working on it from different parts of the world.	Because of the proprietary nature of the original Unix, users have to wait for a while to get the proper bugfixing patch. But these are not as common.
Processors	Dozens of different kinds.	x86/x64, Sparc, Power, Itanium, PA-RISC, PowerPC and many others.
Examples	Ubuntu , Fedora , Red Hat , Debian, Archlinux, Android etc.	OSX, Solaris, All Linux
Architectures	Originally developed for Intel's x86 hardware, ports available for over two dozen CPU types including ARM	is available on PA-RISC and Itanium machines. Solaris is also available for x86/x64 based systems. OSX is PowerPC (10.0-

		10.5)/x86(10.4)/x64(10.5-10.8)
Inception	Inspired by MINIX (a Unix-like system) and eventually after adding many features of GUI, Drivers etc, Linus Torvalds developed the framework of the OS that became LINUX in 1992. The LINUX kernel was released on 17th September, 1991	In 1969, it was developed by a group of AT&T employees at Bell Labs and Dennis Ritchie. It was written in "C" language and was designed to be a portable, multi-tasking and multi-user system in a time-sharing configuration

Linux Distribution (Operating System) Names

A few popular names:

1. Redhat Enterprise Linux
2. Fedora Linux
3. Debian Linux
4. Suse Enterprise Linux
5. Ubuntu Linux

Common Things Between Linux & UNIX

Both share many common applications such as:

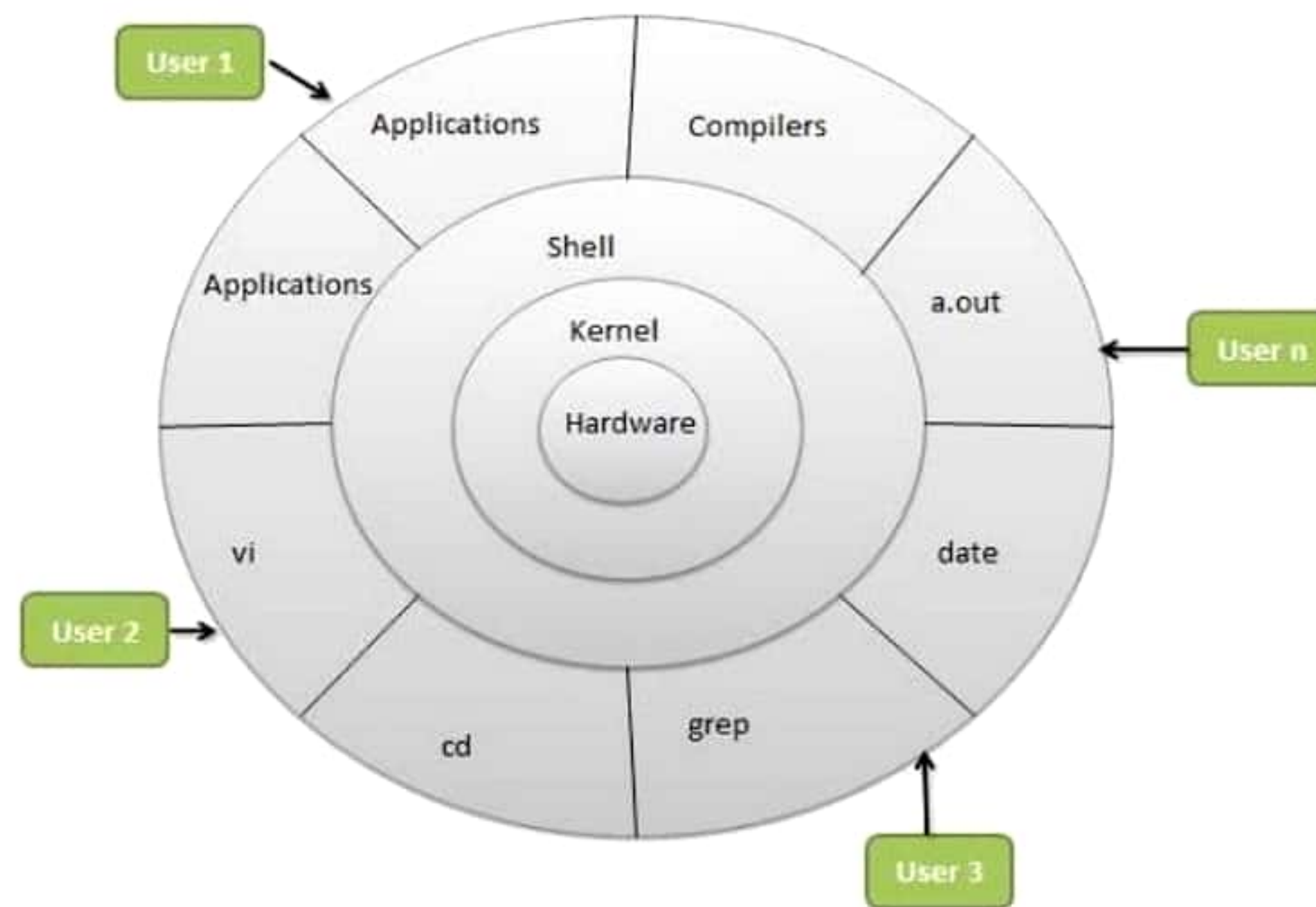
1. GUI, file, and window managers (KDE, Gnome)
2. Shells (ksh, csh, bash)
3. Various office applications such as OpenOffice.org

4. Development tools (perl, php, python, GNU C/C++ compilers)

5. Posix interface

Layered Architecture:

Architecture

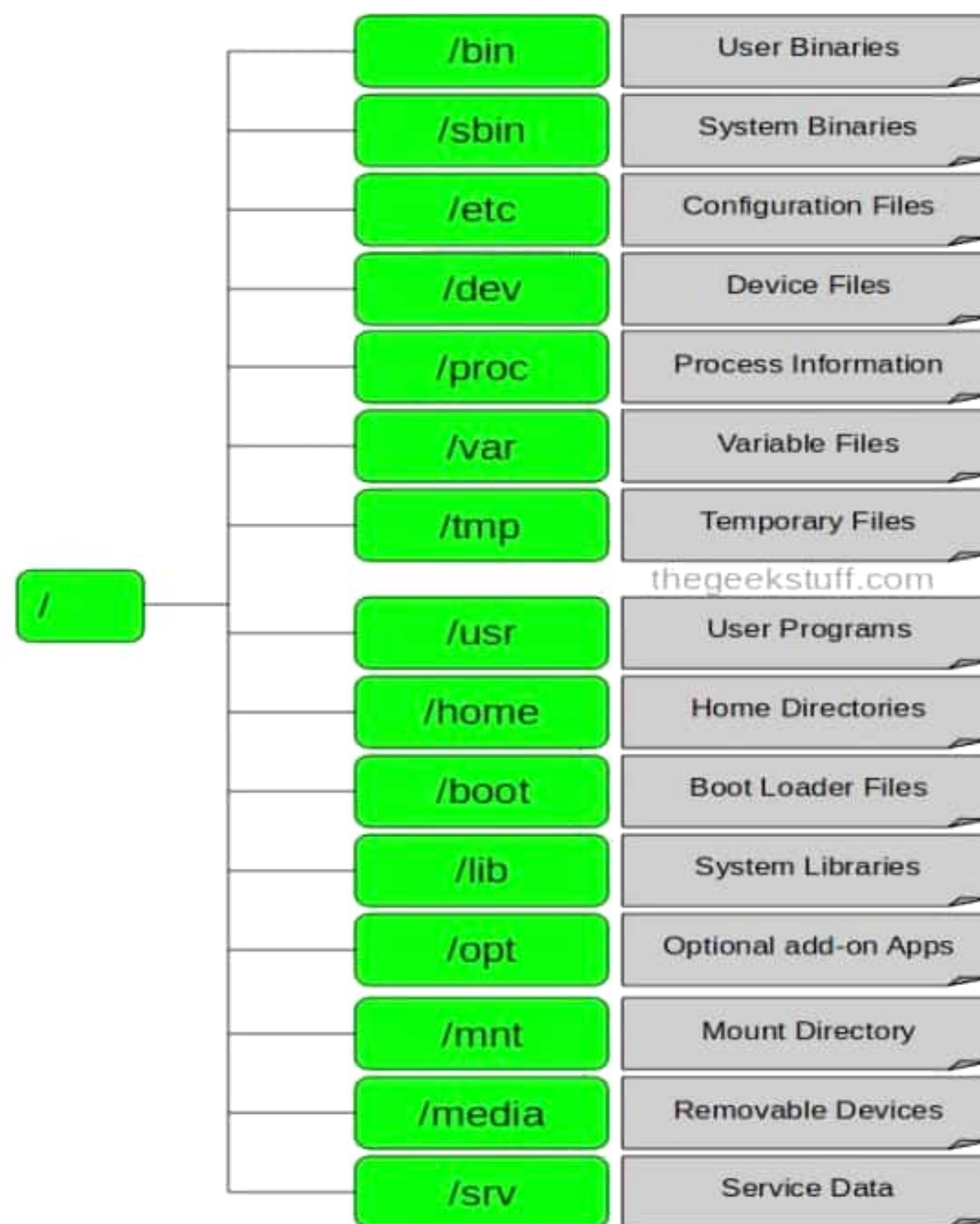


Linux System Architecture consists of the following layers

- **Hardware layer**-Hardware consists of all peripheral devices (RAM/HDD/ CPU etc).
- **Kernel**-Core component of Operating System, interacts directly with hardware, provides low level services to upper layer components.
- **Shell** - An interface to kernel, hiding complexity of kernel's functions from users. Takes commands from user and executes kernel's functions.
- **Utilities**-Utility programs giving user most of the functionalities of an operating system.

LINUX Filesystem

Linux file structure files are grouped according to purpose. Ex: commands, data files, documentation. Parts of a Unix directory tree are listed below. All directories are grouped under the root entry "/". That part of the directory tree is left out of the below diagram.



1. /-Root

- Every single file and directory starts from the root directory.
- Only root user has write privilege under this directory.
- Please note that /root is root user's home directory, which is not same as /.

2. /bin-User Binaries

- Contains binary executables.
- Common linux commands you need to use in single-user modes are located under this directory.
- Commands used by all the users of the system are located here.
- For example: ps, ls, ping, grep, cp.

3. /sbin- System Binaries

- Just like /bin, /sbin also contains binary executables.
- But, the linux commands located under this directory are used typically by system administrator, for system maintenance purpose.
- For example: iptables, reboot, fdisk, ifconfig, swapon

4. /etc -Configuration Files

- Contains configuration files required by all programs.
- This also contains startup and shutdown shell scripts used to start/stop individual programs.
- For example: `/etc/resolv.conf`, `/etc/logrotate.conf`

5. `/dev`—Device Files

- Contains device files.
- These include terminal devices, usb, or any device attached to the system.
- For example: `/dev/tty1`, `/dev/usbmon0`

6. `/proc`—Process Information

- Contains information about system process.
- This is a pseudo file system that contains information about running process. For example: `/proc/{pid}` directory contains information about the process with that particular pid.
- This is a virtual file system with text information about system resources. For example: `/proc/uptime`

7. `/var`—Variable Files

- `var` stands for variable files.
- Content of the files that are expected to grow can be found under this directory.
- This includes — system log files (`/var/log`); packages and database files (`/var/lib`); emails (`/var/mail`); print queues (`/var/spool`); lock files (`/var/lock`); temp files needed across reboots (`/var/tmp`);

8. `/tmp`—Temporary Files

- Directory that contains temporary files created by system and users.
- Files under this directory are deleted when system is rebooted.

9. `/usr`—User Programs

- Contains binaries, libraries, documentation, and source-code for second level programs.
- `/usr/bin` contains binary files for user programs. If you can't find a user binary under `/bin`, look under `/usr/bin`. For example: `at`, `awk`, `cc`, `less`, `scp`
- `/usr/sbin` contains binary files for system administrators. If you can't find a system binary under `/sbin`, look under `/usr/sbin`. For example: `atd`, `cron`, `sshd`, `useradd`, `userdel`
- `/usr/lib` contains libraries for `/usr/bin` and `/usr/sbin`
- `/usr/local` contains user programs that you install from source. For example, when you

install apache from source, it goes under /usr/local/apache2

10. /home—Home Directories

- Home directories for all users to store their personal files.
- For example: /home/john, /home/nikita

11. /boot – Boot Loader Files

- Contains boot loader related files.
- Kernel initrd, vmlinuz, grub files are located under /boot
- For example: initrd.img-2.6.32-24-generic, vmlinuz-2.6.32-24-generic

12. /lib—System Libraries

- Contains library files that support the binaries located under /bin and /sbin
- Library filenames are either ld* or lib*.so.*
- For example: ld-2.11.1.so, libncurses.so.5.7

13. /opt—Optional add-on Applications

- opt stands for optional.
- Contains add-on applications from individual vendors.
- add-on applications should be installed under either /opt or /opt/sub-directory.

14. /mnt—Mount Directory

- Temporary mount directory where sysadmins can mount file systems.

15. /media—Removable Media Devices

- Temporary mount directory for removable devices.
- For examples, /media/cdrom for CD-ROM; /media/floppy for floppy drives; /media/cdrecorder for CD writer

16. /srv—Service Data

- srv stands for service.
- Contains server specific services related data.
- For example, /srv/cvs contains CVS related data.

1. Linux Utilities:

File Handli

ng

utilities:

Cat

Command:

cat linux command concatenates files and prints it on the standard output.

SYNTAX:

The Syntax
is

cat [OPTIONS] [FILE]...

OPTIONS:

-A Show all.

-b Omits line numbers for blank space in the output.

-e A \$ character will be printed at the end of each line prior to a new line. -E Displays a \$ (dollar sign) at the end of each line.

-n Line numbers for all the output lines.

-s If the output has multiple empty lines it replaces it with one empty line. -T Displays the tab characters in the output.

Non-printing characters (with the exception of tabs, new-lines and form-feeds)

-v are printed visibly.

Example:

To Create a new file:

cat > file1.txt

This command creates a new file file1.txt. After typing into the file press control+d (^d) simultaneously to end the file.

1. To Append data into the

file: cat >> file1.txt

To append data into the same file use append operator >> to write into the file, else the file will be overwritten (i.e., all of its contents will be erased).

2. To display a

Linux Programming Notes

file: cat

file1.txt

This command displays the data in the file.

3. To concatenate several files and

display: cat file1.txt file2.txt

The above cat command will concatenate the two files (file1.txt and file2.txt) and it will display the output in the screen. Some times the output may not fit the monitor screen. In such situation you can print those files in a new file or display the file using less command.

```
cat file1.txt file2.txt | less
```

4. To concatenate several files and to transfer the output to another file. cat

```
file1.txt file2.txt > file3.txt
```

In the above example the output is redirected to new file file3.txt. The cat command will create new file file3.txt and store the concatenated output into file3.txt.

rm COMMAND:

rm linux command is used to remove/delete the file from the directory.

SYNTAX:

The Syntax is

```
rm [options..] [file|directory]
```

OPTIONS:

- f Remove all files in a directory without prompting the user.
- i Interactive. With this option, rm prompts for confirmation before removing any files.
- r(or)-R Recursively remove directories and subdirectories in the argument list. The directory will be emptied of files and removed. The user is normally prompted for removal of any write-protected files which the directory contains.

EXAMPLE:

1. To Remove / Delete a file:

```
rm file1.txt
```

Here the `rm` command will remove/delete the file `file1.txt`.

2. To delete a directory tree:

```
rm -ir tmp
```

This `rm` command recursively removes the contents of all subdirectories of the `tmp` directory, prompting you regarding the removal of each file, and then removes the `tmp` directory itself.

3. To remove more files at once

```
rm file1.txt file2.txt
```

The `rm` command removes `file1.txt` and `file2.txt` files at the same time.

cd COMMAND:

The `cd` command is used to change the directory.

SYNTAX:

The syntax is

```
cd [directory|~|.|./|..|-]
```

OPTIONS:

- L Use the physical directory structure.
- P Force symbolic links.

EXAMPLE:

1. `cd linux-command`

This command will take you to the sub-directory (`linux-command`) from its parent directory.

2. `cd..`

This will change to the parent-directory from the current working directory/sub-directory.

3. `cd~`

This command will move to the user's home directory which is `"/home/username"`.

cpCOMMAND:

`cp` command copy files from one location to another. If the destination is an existing file, then the file is overwritten; if the destination is an existing directory, the file is copied into the directory (the directory is not overwritten).

SYNTAX:

The Syntax is

`cp[OPTIONS]...SOURCEDEST`

`cp[OPTIONS]...SOURCE...DIRECTORY`

`cp[OPTIONS]...--target-directory=DIRECTORYSOURCE...`

OPTIONS:

- `-a` same as `-dpR`.
- `--backup[=CONTROL]` make a backup of each existing destination file
- `-b` like `--backup` but does not accept an argument.
- `-f` if an existing destination file cannot be opened, remove it and try again.
- `-p` same as `--preserve=mode,ownership,timestamps`.

-- preserve the specified attributes (default: mode,ownership,timestamps)andsecuritycontexts,ifpossible
 preserve[=ATTR_LIST] additionalattributes:links,all.
 --no- don'tpreservethespecifiedattribute.
 preserve=ATTR_LIST
 --parents appendsource pathtoDIRECTORY.

EXAMPLE:

Copytwo

files: cp
 file1file2

Theabovecpcommandcopiesthecontentoffile1.phptofile2.php.

1. To backupthecopied

file: cp -b file1.php
 file2.php

Backupoffile1.phpwillbe createdwith'~'symbolasfile2.php~.

2. Copy folder and

subfolders:cp-Rscripts
 scripts1

Theabovecp commandcopythe folderand subfoldersfromscriptstoscripts1.

lsCOMMAND:

lscommandliststhefilesanddirectoriesundercurrentworkingdirectory.

SYNTAX:

The
 Syntaxis

ls[OPTIONS]... [FILE]

OPTIONS:

- l Lists all the files, directories and their mode, Number of links, owner of the file, file size, Modified date and time and filename.
- t Lists in order of last modification time.
- a Lists all entries including hidden files.
- d Lists directory files instead of contents.
- p Put slash at the end of each directory.
- u List in order of last access time.
- i Display inode information.
- ltr List files order by date.
- lSr List files order by file size.

EXAMPLE:

Display root directory

contents: `ls /`

lists the contents of root directory.

1. Display hidden files and

directories: `ls -a`

lists all entries including hidden files and directories.

2. Display inode information:

`ls -i`

7373073book.gif

7373074clock.gif

7373082globe.gif

7373078pencil.gif

7373080child.gif

OPTIONS:

- R Changethepermissiononfilesthatareinthesubdirectoriesofthe directory
thatyouarecurrently in.
- c Changethepermissionforeachfile.
- f Preventschownfromdisplayingerrormessageswhenitisunableto change
theownership ofafile.

EXAMPLE:

1. chownhioxtest.txt

Theowner ofthe'test.txt'fileisroot,Changetonewuserhiox.

2. chown-Rhioxtest

Theownerofthe'test'directoryisroot,With-Roptionthefilesand subdirectories user also gets changed.

3. chown-c hioxcalc.txt

Herechangetheownerforthespecific'calc.txt'file only.

SecurityBy File

Permissionschmod

Command:

chmodcommandallows youtoalter/Changeaccessrightstofilesanddirectories.

FilePermissionisgivenforusers, groupandothersas,

ReadWrite Execute

User	☐	☐	☐
Grou	☐	☐	☐
Others	☐	☐	☐
Permission	000		
Symbolic			

SYNTAX:

TheSyntax is

`chmod[options][MODE]FileName`

FilePermission

#	FilePermission
0	none
1	executeonly
2	writeonly
3	writeandexecute
4	readonly
5	readandexecute
6	readandwrite
7	setallpermissions

OPTIONS:

- c Displays names of only those files whose permissions are being changed
- f Suppress most error messages
- R Change files and directories recursively

-v Output version information and exit.

EXAMPLE:

1. To view your files with what permission they are: ls -

alt

This command is used to view your files with what permission they are.

2. To make a file readable and writable by the group and others.

chmod 066 file1.txt

3. To allow everyone to read, write, and execute the file

chmod 777 file1.txt

mkdir COMMAND:

mkdir command is used to create one or more directories.

SYNTAX:

The Syntax is

mkdir [options] directories

OPTIONS:

-m Set the access mode for the new directories.

-p Create intervening parent directories if they don't exist. -v Print help message for each directory created.

EXAMPLE:

1. Created directory:

mkdir test

The above command is used to create the directory 'test'.

2. Create directory and

set permissions: mkdir -

m 666 test

The above command is used to create the directory 'test' and set the read and write permission.

rmdir COMMAND:

rmdir command is used to delete/remove a directory and its subdirectories.

SYNTAX:

The Syntax is

rmdir [options..] Directory

OPTIONS:

-p Allow users to remove the directory dir name and its parent directories which become empty.

EXAMPLE:

1. To delete/remove a directory

rmdir tmp

rmdir command will remove/delete the directory tmp if the directory is empty.

rm -r tmp

2. To delete a directory tree:

```
rm -r tmp
```

This command recursively removes the contents of all subdirectories of the tmp directory, prompting you regarding the removal of each file, and then removes the tmp directory itself.

mv COMMAND:

mv command which is short for move. It is used to move/rename file from one directory to another. mv command is different from cp command as it completely removes the file from the source and moves to the directory specified, where cp command just copies the content from one file to another.

SYNTAX:

The Syntax is

```
mv[-f][-i]oldname newname
```

OPTIONS:

-f This will not prompt before overwriting (equivalent to --reply=yes). mv -f will move the file(s) without prompting even if it is writing over an existing target. -i Prompts before overwriting another file.

EXAMPLE:

1. To Rename/Move a file:

```
mv file1.txt file2.txt
```

This command renames file1.txt as file2.txt

2. To move a directory

```
mv hscript tmp
```


In the above line mv command moves all the files, directories and sub-directories from hscripts folder/directory to tmp directory if the tmp directory already exists. If there is no tmp directory it rename's the hscripts directory as tmp directory.

3. To Move multiple files/More files into another directory

```
mv file1.txt tmp/file2.txt newdir
```

This command moves the files file1.txt from the current directory and file2.txt from the tmp folder/directory to newdir.

diffCOMMAND:

diff command is used to find differences between two files.

SYNTAX:

The Syntax is

```
diff[options..] from-file to-file
```

OPTIONS:

- a Treat all files as text and compare them line-by-line.
- b Ignore changes in amount of whitespace. -c Use the context output format.
- e Make output that is a valid ed script.
- H Use heuristic to speed handling of large files that have numerous scattered small changes.
- i Ignore changes in case; consider upper- and lower-case letter equivalent.
- n Prints in RCS-format, like -f except that each command specifies the number of lines affected.
- q Output RCS-format diffs; like -f except that each command specifies the number of lines affected.

- r When comparing directories, recursively compare any subdirectories found.
- s Report when two files are the same.
- w Ignore whitespace when comparing lines. -y Use the side by side output format.

EXAMPLE:

Lets create two files file1.txt and file2.txt and let it have the following data.

Data in file1.txt	Data in file2.txt
HIOXTEST	HIOXTEST
hscripts.com	HSCRIPTS.com
withfriendship	withfriendship
hioxindia	

1. Compare files ignoring white

space: `diff -w file1.txt file2.txt`

This command will compare the file file1.txt with file2.txt ignoring white/blank space and it will produce the following output.

```
2c2
<hscripts.com
---
>HSCRIPTS.com
4d3
<Hioxindia.com
```

2. Compare the file side by side, ignoring white space:

`diff -by file1.txt file2.txt`

This command will compare the files ignoring white/blank space, It is easy to differentiate the files.

```
HIOXTEST      HIOXTEST
```

```
hscripts.com  | HSCRIPTS.com
with friend ship  withfriendship
Hioxindia.com   <
```

The third line (withfriendship) in file2.txt has more blank spaces, but still the -b ignores the blank space and does not show changes in the particular line, -y prints out the result side by side.

3. Compare the files ignoring

```
case. diff -iy file1.txt file2.txt
```

This command will compare the files ignoring case (upper-case and lower-case) and displays the following output.

```
HIOXTEST      HIOXTEST
hscripts.com   HSCRIPTS.com
withfriendship |withfriendship
```

chgrp COMMAND:

chgrp command is used to change the group of the file or directory. This is an admin command. Root user only can change the group of the file or directory.

SYNTAX:

The Syntax is

```
chgrp [options] newgroup filename/directoryname
```

OPTIONS:

- R Changethepermissiononfilesthatareinthesubdirectoriesofthedirectory thatyouarecurrently in.
- c Changethepermissionforeachfile.
- f Force.Donotreporterrors.

Hioxindia.com <

EXAMPLE:

1. chgrp hiox test.txt

Thegroupof'test.txt'fileisroot,Change tonewgrouphiox.

2. chgrp -R hiox test

Thegroupof'test'directoryisroot. With -R,the filesand itssubdirectoriesalso changes to newgroup hiox.

3. chgrp -chiox calc.txt

Theabovecommandisusedtochangethegroupforthespecificfile('calc.txt')only.

About wc

Short for wordcount, wcdisplaysaountoflines, words,andcharacters in afile.

Syntax

wc [-c | -m | -C] [-l] [-w] [file...]

- c Countbytes.
- m Countcharacters.
- C Sameas -m.

- l Countlines.
- w Countwordsdelimitedbywhitespacecharactersornewlinecharacters.
DelimitingcharactersareExtendedUnixCode(EUC)charactersfromanycode set defined by iswspace()
- File Nameoffiletoword count.

Examples

wcmyfile.txt-Displays informationaboutthefilemyfile.txt. Belowisanexampleofthe output.

```
5 13 57myfile.txt
```

5=Lines

13=Words

57=Characters

About split

Split afile into pieces.

Syntax

```
split[-linecount|-llinecount][ -asuffixlength][file[name]] split -
```

```
b n [k | m] [ -a suffixlength ] [ file [name]]
```

-linecount|-lNumberoflinesineachpiece.Defaultsto1000lines.

linecount

- a Usesuffixlength lettersto formthesuffixportion ofthe filenamesofthe split
- suffixlength file. If -a is not specified, the default suffix length is 2. Ifthesumofthe name operandandthesuffixlengthoption-argument wouldcreateafilenameexceeding NAME_MAXbytes,anerrorwillresult;splitwill exitwithadiagnosticmessage

and no files will be created.

-bn Split a file into pieces *n* bytes in size.

-bnk Split a file into pieces *n**1024 bytes in size.

-bnm Split a file into pieces *n**1048576 bytes in size.

File The path name of the ordinary file to be split. If no input file is given or file is -, the standard input will be used.

name The prefix to be used for each of the files resulting from the split operation. If no *name* argument is given, *x* will be used as the prefix of the output files. The combined length of the base name of prefix and suffix length cannot exceed NAME_MAX bytes; see OPTIONS.

Examples

split -b22 newfile.txt new- would split the file "newfile.txt" into three separate files called newaa, newab and newac each file the size of 22.

split -l300 file.txt new- would split the file "newfile.txt" into files beginning with the name "new" each containing 300 lines of text each

About settime and touch

Change file access and modification time.

Syntax

touch [-a] [-c] [-m] [-r *ref_file*] [-t *time*] *file* *settime* [-f

ref_file] *file*

-a Change the access time of file. Do not change the modification time unless -m is also specified.

- c Donotcreateaspecifiedfileifitdoesnotexist. Donotwriteanydiagnostic messagesconcerningthiscondition.
- m Changethemodificationtimeoffile.Donotchangetheaccesstimeunless-ais alsospecified.
- r ref_file Usethecorrespondingtimesofthefilenamedbyref_fileinsteadofthecurrent time.
- ttime Usethespecifiedtimeinsteadofthecurrenttime.timewill beadecimalnumber oftheform:

[[CC]YY]MMDDhhmm[.SS]

MM	-	The	month	of	the	year	[01-12].
DD	-	The	day	of	the	month	[01-31].
hh	-	The	hour	of	the	day	[00-23].
mm	-	The	minute	of	the	hour	[00-59].
CC	-	The	first	two	digits	of	the year.
YY	-	The	second	two	digits	of	the year.
SS	-	The	second	of	the	minute	[00-61].

- fref_file Usethecorrespondingtimesofthefilenamedbyref_fileinsteadofthecurrent time.

File Apathname ofa file whosetimesare tobe modified.

Examples

settimemyfile.txt

Setsthefilemyfile.txtasthecurrenttime/date.

touchnewfile.txt

Creates a file known as "newfile.txt", if the file does not already exist. If the file already exists the accessed / modification time is updated for the file newfile.txt

About comm

Select or reject lines common to two files.

Syntax

comm[-1][-2][-3]file1 file2

- 1 Suppress the output column of lines unique to file1.
- 2 Suppress the output column of lines unique to file2.
- 3 Suppress the output column of lines duplicated in file1 and file2. file1
Name of the first file to compare.
- file2 Name of the second file to compare.

Examples

comm myfile1.txt myfile2.txt

The above example would compare the two files myfile1.txt and myfile2.txt.

Process utilities:

ps Command:

ps command is used to report the process status. ps is the short name for Process Status.

SYNTAX:

The Syntax is

ps [options]

OPTIONS:

- a List information about all processes most frequently requested: all those except process group leaders and processes not associated with a terminal..
- Aore List information for all processes.
- d List information about all processes except session leaders.
- e List information about every process now running.
- f Generates a full listing.
- j Print session ID and process group ID.
- l Generate a long listing.

EXAMPLE:

1. ps

Output:

PID	TTY	TIME	CMD
2540	pts/1	00:00:00	bash
2621	pts/1	00:00:00	ps

In the above example, typing ps alone would list the current running processes.

2. ps-f

Output:

UID	PID	PPID	CSTIME	TTY	TIME	CMD
nirmala	2540	2536	015:31	pts/1	00:00:00	bash
nirmala	2639	2540	015:51	pts/1	00:00:00	ps-f

Displays full information about currently running processes.

killCOMMAND:

killcommandisusedtokillthebackground process.

SYNTAX:

TheSyntax is

kill[-s] [-l]%pid

OPTIONS:

- s Specifythesignalto send.Thesignal maybegivenasa signalnameor number.
- l Writeallvaluesofsignalsupportedbytheimplementation,ifnooperandis given.
- pid Processidorjobid.
- 9 Forcetokillaprocess.

EXAMPLE:

StepbyStep process:

- Open a process music player. xmms

pressctrl+z tostopthe process.
- Toknowgroupidorjobidofthebackground task. jobs
-l
- Itwilllistthebackgroundjobswithitsjobidas,
xmms3956
kmail3467
- To kill a job or process. kill 3956

killcommand killsorterminatesthebackground processxmms.

Aboutnice

Invokesacommandwithanalteredschedulingpriority.

Syntax

nice[-increment|-nincrement]command[argument...]

-increment| -increment must be intherange1-19;ifnot specified, anincrementof10is n
increment assumed. An increment greater than 19 is equivalent to 19.

The super-user may run commands with priority higher than normal by using a
negativeincrementsuchas-10.Anegativeincrementassignedby an
unprivileged user is ignored.

command The name of a command that is to be invoked. If command names any of the
special built-in utilities, the results are undefined.

argument Any string to be supplied as an argument when invoking command.

Examples

nice+13picomyfile.txt -runsthepico command onmyfile.txt withanincrementof+13. About

at

Schedulesacommandtoberanataparticulartime, suchasaprintjoblateatnight.

Syntax

at executescommandsataspecifiedtime.

atq liststheuser'spendingjobs,unless theuser isthesuperuser;inthat case,everybody'sjobs
arelisted.Theformatoftheoutputlines(oneforeachjob)is:Jobnumber,date,hour, job

class.

atrm deletes jobs, identified by their job number.

batch executes commands when system load levels permit; in other words, when the load average drops below 1.5, or the value specified in the invocation of atrun.

at [-c|-k|-s] [-f filename] [-q queue name] [-m] -t time [date] [-l] [-r]

-c C shell.csh(1) is used to execute the at-job.

-k Korn shell.ksh(1) is used to execute the at-job.

-s Bourne shell.sh(1) is used to execute the at-job.

-f filename Specifies the file that contains the command to run.

-m Sends mail once the command has been run.

-t time Specifies at what time you want the command to be run. Format hh:mm.am/pm
indication can also follow the time otherwise a 24-hour clock is used. A time zone
name of GMT, UCT or ZULU (case insensitive) can follow to specify that the
time is in Coordinated Universal Time. Other time zones can be specified using
the TZ environment variable. The below quick times can also be entered:

midnight - Indicates the time 12:00 am (00:00).

noon - Indicates the time 12:00 pm.

now - Indicates the current day and time. Invoking at-now will submit an at-job for potentially immediate execution.

date Specifies the date you wish it to be run on. Format month, date, year. The
following quick days can also be entered:

today - Indicates the current day.

tomorrow - Indicates the day following the current day.

- l List the commands that have been set to run.
- r Cancel the command that you have set in the past.

Examples

at -m 01:35 < atjob = Run the commands listed in the 'atjob' file at 1:35AM, in addition all output that is generated from job mail to the user running the task. When this command has been successfully entered you should receive a prompt similar to the below example.

```
commands          will          be          executed          using          /bin/csh
job1072250520.a atWed Dec 2400:22:002003
```

at -l = This command will list each of the scheduled jobs as seen below. 1072250520.a Wed
Dec 24 00:22:00 2003

at-r 1072250520.a = Delete the job just created. or

atrm 23 = Delete job 23.

If you wish to create a job that is repeated you could modify the file that executes the commands with another command that recreates the job or better yet use the crontab command.

Note: Performing just the **at** command at the prompt will give you an error "Garbled Time", this is a standard error message if no switch or time setting is given.

Disk utilities:

du (abbreviated from *disk usage*) is a standard Unix program used to estimate file space usage—space used under a particular directory or files on a file system.

du takes a single argument, specifying a path name for du to work; if it is not specified, the current directory is used. The SUS mandates for du the following options:

- a, display an entry for each file (and not directory) contained in the current directory

- H,calculatediskusageforlinkreferencesspecifiedonthecommandline
- k,showsizesas multiplesof1024bytes, not512-byte
- L,calculatediskusage forlinkreferencesanywhere
- s,reportonlythesumoftheusageinthecurrentdirectory,notforeachfile
- x,onlytraverse filesanddirectoriesonthedeviceonwhichthepathnameargument is specified.

Other Unix and Unix-like operating systems mayadd extra options. For example, BSD and GNUdu specify a -h option, displayingdiskusage in a format easiertoreadbytheuser, adding units with the appropriate SI prefix'

```
$du-sk*
152304directoryOne
1856548directoryTwo
```

Sumofdirectoriesinhuman-readableformat(Byte,Kilobyte,Megabyte,Gigabyte,Terabyteand Petabyte):

```
$du-sh*
149M directoryOne
1.8GdirectoryTwo
```

diskusageofallsubdirectoriesand filesincludinghiddenfileswithinthecurrent directory (sorted by filesize) :

```
$du-sk.[!.]**|sort-n
```

diskusageofallsubdirectoriesand filesincludinghiddenfileswithinthecurrent directory (sorted by reverse filesize) :

```
$du-sk.[!.]**|sort-nr
```

Theweightofdirectories:

```
$ du-dl-c-h
```


dfcommand: Report filesystem disk space usage

Dfcommand examples-to check free disk space

Type `df -h` to list free disk

space: `$ df -h`

OR

`$ df -k`

Output:

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sdb1	20G	9.2G	9.6G	49%	/
varrun	393M	144k	393M	1%	/var/run
varlock	393M	0	393M	0%	/var/lock
procusb	393M	123k	393M	1%	/proc/bus/usb
udev	393M	123k	393M	1%	/dev
devshm	393M	0	393M	0%	/dev/shm
lrn	393M	35M	359M	9%	/lib/modules/2.6.20-15-generic/volatile
/dev/sdb5	29G	5.4G	22G	20%	/media/docs
/dev/sdb3	30G	5.9G	23G	21%	/media/isomp3s
/dev/sda1	8.5G	4.3G	4.3G	51%	/media/xp1
/dev/sda2	12G	6.5G	5.2G	56%	/media/xp2
/dev/sdc1	40G	3.1G	35G	9%	/media/backup

ducommand examples

`du` shows how much space one or more files or directories is using. `$`

`du -sh`

103M

`-s` option summarizes the space a directory is using and `-h` option provides "Human-readable" output.

Networking commands:

These are most useful commands in my list while working on Linux server, this enables you to quickly troubleshoot connection issues e.g. whether other system is connected or not, whether other host is responding or not and while working for FIX connectivity for advanced trading system this tool saves quite a lot of time.

This article is in continuation of my article [How to work fast in Unix and Unix Command tutorials](#) and Examples for beginners.

- finding host/domain name and IP address - **hostname**
- test network connection - **ping**
- getting network configuration - **ifconfig**
- Network connections, routing tables, interface statistics - **netstat**
- query DNS lookup name - **nslookup**
- communicate with other host name - **telnet**
- outlining steps that packets take to get to network host - **traceroute**
- view user information - **finger**
- checking status of destination host - **telnet**

Example of Networking commands in Unix

Let's see some examples of various networking commands in Unix and Linux. Some of them are quite basic e.g. ping and telnet and some are more powerful e.g. nslookup and netstat. When you use these commands in combination of find and grep you can get anything you are looking for e.g. hostname, connection end points, connection status etc.

hostname

hostname with no options displays the machine's host name

hostname -d displays the domain name the machine belongs to

hostname -f displays the fully qualified host and domain name

hostname -i displays the IP address for the current machine

ping

It sends packets of information to the user-defined source. If the packets are received, the destination device sends packets back. Ping can be used for two purposes

1. To ensure that a network connection can be established.
2. Timing information as to the speed of the connection.

If you do **ping www.yahoo.com** it will display its IP address. Use **ctrl+C** to stop the test.

ifconfig

View network configuration, it displays the current network adapter configuration. It is handy to

determine if you are getting transmit (TX) or receive (RX) errors.

netstat

Most useful and very versatile for finding connection to and from the host. You can find out all the multicast groups (network) subscribed by this host by issuing "**netstat -g**"

netstat -nap | grep port will display process id of application which is using that port
netstat -a or **netstat -all** will display all connections including TCP and UDP
netstat -tcp or **netstat -t** will display only TCP connection
netstat -u or **netstat -u** will display only UDP connection
netstat -g will display all multicast networks subscribed by this host.

nslookup

If you know the IP address it will display hostname. To find all the IP addresses for a given domain name, the command nslookup is used. You must have a connection to the internet for this utility to be useful.

E.g. **nslookup blogger.com**

You can also use nslookup to convert hostname to IP address and from IP address to hostname.

traceroute

A handy utility to view the number of hops and response time to get to a remote system or web site is traceroute. Again you need an internet connection to make use of this tool.

finger

View user information, displays a user's login name, real name, terminal name and write status. This is pretty old unix command and rarely used now days.

telnet

Connects destination host via telnet protocol, if telnet connection establishes on any port means connectivity between two hosts is working fine.

telnet hostname port will telnet hostname with the ports specified. Normally it is used to see whether host is alive and network connection is fine or not.

10 Most important linux networking commands

Linux is most powerful operating system which often needs to use commands to explore it effectively. Some of the commands are restricted to normal user groups as they are powerful and has more functionality involved in it. Here we summarized most interesting and useful networking commands which every linux user are supposed to be familiar with it.

1. Arp manipulates the kernel's ARP cache in various ways. The primary options are clearing an address mapping entry and manually setting up one. For debugging purposes, the arp program also allows a complete dump of the ARP cache. ARP displays the IP address assigned to particular ETH card and mac address

```
[fasil@smashtech]# arp
Address      HWtype HWaddress      FlagsMask      Iface
59.36.13.1   ether  C              eth0
```

2. Ifconfig is used to configure the network interfaces. Normally we use this command to check the IP address assigned to the system. It is used at boot time to set up interfaces as necessary. After that, it is usually only needed when debugging or when system tuning is needed.

```
[fasil@smashtech~]# /sbin/ifconfig
eth0UPBROADCASTRUNNINGMULTICASTMTU:1500Metric:1
    RXpackets:126341errors:0dropped:0overruns:0frame:0
    TX packets:44441 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
```

3. Netstat prints information about the networking subsystem. The type of information which is usually printed by netstat are Print network connections, routing tables, interface statistics, masquerade connections, and multicast.

```
[fasil@smashtech~]# netstat
ActiveInternetconnections(w/o servers)
ProtoRecv-QSend-QLocalAddress      ForeignAddress      State
tcp0      0      .230.87:https      ESTABLISHED

ActiveUNIXdomainsockets(w/o servers)
ProtoRefCntFlagsType      State      I-Node Path
unix10    []      DGRAM      4970/dev/log
unix2     []      DGRAM      6625@/var/run/hal/hotplug_socket
unix2     []      DGRAM      2952 @udev
unix2     []      DGRAM      100564
unix3     []      STREAM    CONNECTED  62438/tmp/.X11-unix/X0
unix3     []      STREAM    CONNECTED  62437
unix3     []      STREAM    CONNECTED  10271@/tmp/fam-root-
unix3     []      STREAM    CONNECTED  10270
```



```
unix3      []    STREAM    CONNECTED    9276
unix3      []    STREAM    CONNECTED    9275
```

4. ping command is used to check the connectivity of a system to a network. Whenever there is a problem in network connectivity, we use ping to ensure the system is connected to the network.

```
[root@smashtech~]#ping google.com
PING google.com (74.125.45.100) 56(84) bytes of data:
64 bytes from yx-in-f100.google.com (74.125.45.100): icmp_seq=0 ttl=241 time=295 ms
64 bytes from yx-in-f100.google.com (74.125.45.100): icmp_seq=1 ttl=241 time=277 ms
64 bytes from yx-in-f100.google.com (74.125.45.100): icmp_seq=2 ttl=241 time=277 ms

---google.com ping statistics---
3 packets transmitted, 3 received, 0% packet loss, time 6332ms
rttmin/avg/max/mdev = 277.041/283.387/295.903/8.860 ms, pipe 2
```

5. Nslookup is a program to query Internet domain name servers. Nslookup has two modes: interactive and non-interactive. Interactive mode allows the user to query name servers for information about various hosts and domains or to print a list of hosts in a domain. Non-interactive mode is used to print just the name and requested information for a host or domain.

```
[fasil@smashtech~]#nslookup google.com
Server:      server ip
Address:     gateway ip 3
```

```
Non-authoritative answer:
Name: google.com
Address: 209.85.171.100
Name: google.com
Address: 74.125.45.100
Name: google.com
Address: 74.125.67.100
```

6. dig (domain information groper) is a flexible tool for interrogating DNS name servers. It performs DNS lookups and displays the answers that are returned from the name server(s) that were queried. Most DNS administrators use dig to troubleshoot DNS problems because of its flexibility, ease of use and clarity of output. Other lookup tools tend to have less functionality than dig.

```
[fasil@smashtech~]#dig google.com
```



```

;<<>> DiG9.2.4<<>>google.com
;;globaloptions:printcmd
;;Gotanswer:
;;->>HEADER<<- opcode:QUERY,status:NOERROR,id:4716
;;flags:qr rdra;QUERY:1, ANSWER:3, AUTHORITY:4,ADDITIONAL:4

;;QUESTIONSECTION:
google.com.          IN      A

;;ANSWERSECTION:
google.com.          122 IN    A    74.125.45.100
google.com.          122 IN    A    74.125.67.100
google.com.          122 IN    A    209.85.171.100

;;AUTHORITY SECTION:
google.com. 326567 IN NS ns3.google.com. google.com.
326567 IN NS ns4.google.com. google.com. 326567 IN
NS ns1.google.com. google.com. 326567 IN NS
ns2.google.com.

;;ADDITIONALSECTION:
ns1.google.com. 152216 IN A 216.239.32.10
ns2.google.com. 152216 IN A 216.239.34.10
ns3.google.com. 152216 IN A 216.239.36.10
ns4.google.com.152216 IN A 216.239.38.10

;;Querytime:92msec
;;SERVER:172.29.36.1#53(172.29.36.1)
;;WHEN: Thu Mar5 14:38:45 2009
;;MSGSIZErcvd: 212

```

7. Route manipulates the IP routing tables. Its primary use is to set up static routes to specific hosts or networks via an interface after it has been configured with the ifconfig program. When the add or del options are used, route modifies the routing tables. Without these options, route displays the current contents of the routing tables.

```

[fasil@smashtech~]#route
Kernel IP routing table
Destination  Gateway      Genmask      FlagsMetricRef  Use Iface

```



```

54.192.56.321*          255.255.255.0 U00          0eth0
*          255.255.0.0   U  0  0  0eth0
default      0.0.0.0   UG  0  0  0eth0

```

8. Traceroute: Internet is a large and complex aggregation of network hardware, connected together by gateways. Tracking the route one's packets follow (or finding the miscreant gateway that's discarding your packets) can be difficult.

Traceroute utilizes the IP protocol 'time to live' field and attempts to elicit an ICMP TIME_EXCEEDED response from each gateway along the path to some host. The only mandatory parameter is the destination host name or IP number. The default probe datagram length is 40 bytes, but this may be increased by specifying a packet length (in bytes) after the destination host name.

```

[fasil@smashtech~]# traceroute google.com
traceroute: Warning: google.com has multiple addresses; using 209.85.171.100
traceroute to google.com (209.85.171.100), 30 hops max, 38 byte packets
1***

```

9. W-displays information about the users currently on the machine, and their processes. The header shows, in this order, the current time, how long the system has been running, how many users are currently logged on, and the system load averages for the past 1, 5, and 15 minutes.

```

[fasil@smashtech1~]# w
15:18:22 up 4:38, 3 users, load average: 0.89, 0.34, 0.19
USER  TTY FROM          LOGIN@ IDLE JCPU PCPU WHAT
root  :0    -          10:41 ?xdm? 24:53  1.35s /usr/bin/gnome-session
root  pts/1  :0.0       10:58  1.00s 0.34s 0.00s w
root  pts/2  :0.0       12:10 23:32 0.03s 0.03s bash

```

Filters:

more COMMAND:

more command is used to display text in the terminal screen. It allows only backward movement.

SYNTAX:

The Syntax is

more[options]filename

OPTIONS:

- c Clears the screen before displaying.
- e Exit immediately after writing the last line of the last file in the argument list.
- n Specify how many lines are reprinted in the screen for a given file.
- +n Start up the file from the given number.

EXAMPLE:

1. more -c index.php

Clears the screen before reprinting the file.

2. more -3 index.php

Prints first three lines of the given file. Press **Enter** to display the file line by line.

head COMMAND:

head command is used to display the first ten lines of a file, and also specifies how many lines to display.

SYNTAX:

The Syntax is

head[options]filename

OPTIONS:

- n To specify how many lines you want to display.
- n number The number option-argument must be a decimal integer whose sign affects

the location in the file, measured in lines.

-c number The number option-argument must be a decimal integer whose sign affects the location in the file, measured in bytes.

EXAMPLE:

1. head index.php

This command prints the first 10 lines of index.php'.

2. head -5 index.php

The head command displays the first 5 lines of index.php'.

3. head -c 5 index.php

The above command displays the first 5 characters of index.php'.

tail COMMAND:

tail command is used to display the last or bottom part of the file. By default it displays last 10 lines of a file.

SYNTAX:

The Syntax is

tail [options] filename

OPTIONS:

-l To specify the unit of lines.

-b To specify the unit of blocks.

-n To specify how many lines you want to display.

-c number The number option-argument must be a decimal integer whose sign affects the location in the file, measured in bytes.

-nnumber The number option-argument must be a decimal integer whose sign affects the location in the file, measured in lines.

EXAMPLE:

1. tail index.php

It displays the last 10 lines of index.php'.

2. tail -2 index.php

It displays the last 2 lines of index.php'.

3. tail -n5 index.php

It displays the last 5 lines of index.php'.

4. tail -c5 index.php

It displays the last 5 characters of index.php'.

cut COMMAND:

cut command is used to cut out selected fields of each line of a file. The cut command uses delimiters to determine where to split fields.

SYNTAX:

The Syntax is

cut[options]

OPTIONS:

-c Specifies character positions.

-b Specifies byte positions.

-d flags Specifies the delimiters and fields.

EXAMPLE:

1. `cut -c 1-3 text.txt`

Output:

Thi

Cut the first three letters from the above line.

2. `cut -d, -f 1,2 text.txt`

Output:

This is, an example program

The above command is used to split the fields using delimiter and cut the first two fields.

paste COMMAND:

paste command is used to paste the content from one file to another file. It is also used to set column format for each line.

SYNTAX:

The Syntax is

`paste [options]`

OPTIONS:

- s Paste one file at a time instead of in parallel.
- d Reuse characters from LIST instead of TABs.

EXAMPLE:

1. `paste test.txt > test1.txt`

Paste the content from 'test.txt' file to 'test1.txt' file.

2. ls|paste----

List all files and directories in four columns for each line.

sort COMMAND:

sort command is used to sort the lines in a text file.

SYNTAX:

The Syntax is

sort[options] filename

OPTIONS:

- r Sorts in reverse order.
- u If line is duplicated display only once.
- o filename Send sorted output to a file.

EXAMPLE:

1. sorttest.txt

Sort the 'test.txt' file and print result in the screen.

2. sort-rtest.txt

Sort the 'test.txt' file in reverse order and print result in the screen.

About uniq

Report or filter out repeated lines in a file.

Syntax

uniq[-c|-d|-u][-ffields][-schar][-n] [+m][input_file[output_file]]

- c Precede each output line with a count of the number of times the line occurred in