GPCET
Pioneering Innovative Education

G PULLAIAH COLLEGE OF ENGINEERING & TECHNOLOGY
Approved by AICTE, New Delhi & Permanently Affiliated to JNTUA, Ananthapuramu
(Recognized by UGC under 2(f) & 12(B) & ISO 9001:2008 Certified Institution)
Nandikotkur Road, Venkayapalli, Kurnool-518452

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

13A04806 LINUX PROGRAMMING AND SCRIPTING

YEAR / SEM: IV / II

PREPARED BY:

Mr. A. Siva Krishna Reddy

(13A04806) LINUX PROGRAMMING & SCRIPTING

UNIT 1

LINUX BASICS: Introduction to Linux , File System of the Linux, General usage of Linux kernel & basic commands, Linux users and group , Permissions for file , directory and users, Searching a file & directory, zipping and unzipping concepts

UNIT 2

LINUX NETWORKING: Introduction to Networking in Linux, Network basics & tools, File transfer protocol in Linux, Network file system , Domain Naming Services, Dynamic hosting configuration Protocol & Network information Services.

UNIT 3

PERL SCRIPTING: Introduction to Perl Scripting ,Working with Simple Values, Lists and Hashes, Loops and Decisions, Regular Expressions, Files and Data in Perl Scripting ,References &Subroutines , Running and Debugging Perl, Modules, Object-Oriented Perl.

UNIT 4

TCL/ TK SCRIPTING: Tcl Fundamentals, String and Pattern Matching, Tcl Data Structures ,Control Flow Commands, Procedures and Scope , Evel, Working With UNIX, Reflection and Debugging, Script Libraries, Tk Fundamentals ,Tk by Examples, The Pack Geometry Manager, Binding Commands to X Events, Buttons and Menus, Simple Tk Widgets, Entry and Listbox Widgets Focus, Grabs and Dialogs

UNIT 5

PYTHON SCRIPTING : Introduction to Python, Using the Python Interpreter, More Control Flow Tools, Data Structures, Modules, Input and Output, Errors and Exceptions, Classes, Brief Tour of the Standard Library.

References:

1. Instructor reference material
2. Python Tutorial by Guido van Rossum, and Fred L. Drake, Jr., editor, Release 2.6.4
3. Practical Programming in Tcl and Tk by Brent Welch , Updated for Tcl 7.4 and Tk 4.0
4. Teach Yourself Perl 5 in 21 days by David Till.
5. Red Hat Enterprise Linux 4: System Administration Guide Copyright © 2005 Red Hat,

UNIT-I

I. Introduction to Linux:

Linux is a Unix-like computer operating system assembled under the model of free and open source software development and distribution. The defining component of Linux is the Linux kernel, 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 operating system. It is a leading operating system on servers and other big iron systems such as mainframe computers and supercomputers more than 90% of today's 500 fastest supercomputers run some variant of Linux, including the 10 fastest. Linux also runs on embedded systems (devices where the operating system is typically 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 is built on the Linux kernel.

Basic Features

Following are some of the important features of Linux Operating System.

- **Portable** - Portability means softwares can works on different types of hardwares in same way. Linux kernel and application programs supports their installation on any kind of hardware platform.
- **Open Source** - Linux source code is freely available and it is community based development project. Multiple teams works in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
- **Multi-User** - Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
- **Multiprogramming** - Linux is a multiprogramming system means multiple applications can run at same time.
- **Hierarchical File System** - Linux provides a standard file structure in which system

files/ user files 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 andmuch of its software come with the GNU General Public License. You can start to workimmediately without worrying that your software may stop working anytime because thefree trial version expires. Additionally, there are large repositories from which you canfreely 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. Itdoesn'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 onnetworks. It can handle unusually large numbers of users simultaneously, and can make oldcomputers 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 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 to 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 forums 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 bigest 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 infastructure 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, ect.	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 bug fixing 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.	OS X, 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 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 windows managers (KDE, Gnome)

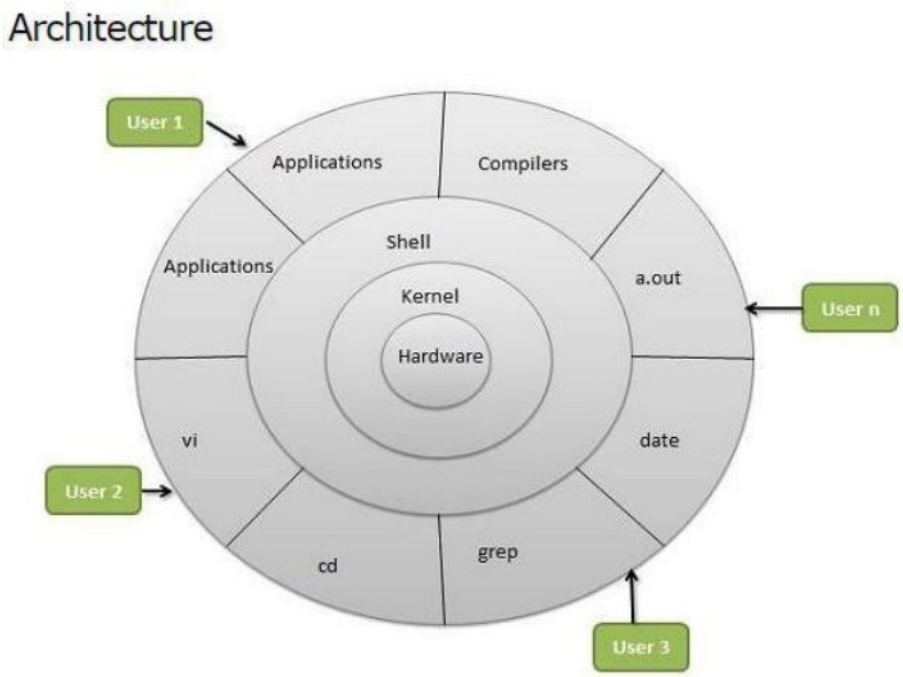
2.Shells (ksh, csh, bash)

3.Various office applications such as OpenOffice.org

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

5.Posix interface

Layered Architecture:

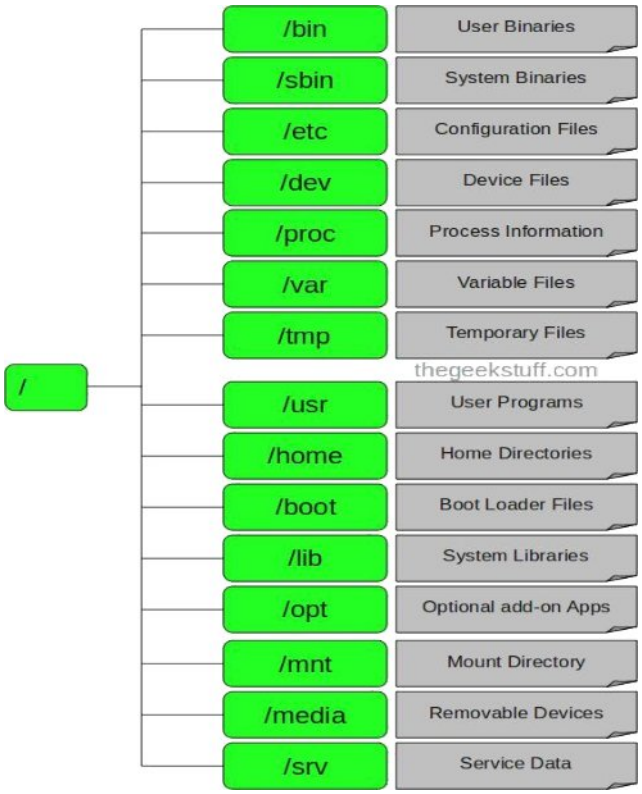


Linux System Architecture is consists of following layers

- **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 systems.

LINUX File system

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 aministrator, 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 filesystem contains information about running process. For example: /proc/{pid} directory contains information about the process with that particular pid.
   - This is a virtual filesystem 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 users 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, vmlinux, 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 supports 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 filesystems.
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:

1.1 File Handling

utilities: Cat

Command:

cat linux command concatenates files and print 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

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 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

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

cd COMMAND:

cd command is used to change the directory.

SYNTAX:

The Syntax is

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

OPTIONS:

-L Use the physical directory structure.

-P Forces 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".

cp COMMAND:

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]... SOURCE DEST

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

cp [OPTIONS]... --target-directory=DIRECTORY SOURCE...

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[=ATTR_LIST]	preserve the specified attributes (default: mode,ownership,timestamps) and security contexts, if possible additional attributes: links, all.
--no-preserve=ATTR_LIST	don't preserve the specified attribute.
--parents	append source path to DIRECTORY.

EXAMPLE:

Copy two files: cp file1 file2

The above cp command copies the content of file1.php to file2.php.

1. To backup the copied

   file: cp -b file1.php

   file2.php

   Backup of file1.php will be created with '~' symbol as file2.php~.

2. Copy folder and

   subfolders: cp -R scripts

   scripts1

   The above cp command copy the folder and subfolders from scripts to scripts1.

ls COMMAND:

ls command lists the files and directories under current working directory.

SYNTAX:

The
Syntax is

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	Puts slash at the end of each directories.
-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
   7373073 book.gif
   7373074 clock.gif
   7373082 globe.gif
   7373078 pencil.gif
   7373080 child.gif

7373081 email.gif

7373076 indigo.gif

The above command displays filename with inode value.

ln COMMAND:

ln command is used to create link to a file (or) directory. It helps to provide soft link for desired files. Inode will be different for source and destination.

SYNTAX:

The Syntax is

ln [options] existingfile(or directory)name newfile(or directory)name

OPTIONS:

-f	Link files without questioning the user, even if the mode of target forbids writing. This is the default if the standard input is not a terminal.
-n	Does not overwrite existing files.
-s	Used to create soft links.

EXAMPLE:

1. ln -s file1.txt file2.txt

   Creates a symbolic link to 'file1.txt' with the name of 'file2.txt'. Here inode for 'file1.txt' and 'file2.txt' will be different.

2. ln -s nimi nimi1

   Creates a symbolic link to 'nimi' with the name of 'nimi1'.

chown COMMAND:

chown command is used to change the owner / user of the file or directory. This is an admin command, root user only can change the owner of a file or directory.

SYNTAX:

The Syntax is

chown [options] newowner filename/directoryname

OPTIONS:

-R	Change the permission on files that are in the subdirectories of the directory that you are currently in.
-c	Change the permission for each file.
-f	Prevents chown from displaying error messages when it is unable to change the ownership of a file.

EXAMPLE:

1. chown hiox test.txt

   The owner of the 'test.txt' file is root, Change to new user hiox.

2. chown -R hiox test

   The owner of the 'test' directory is root, With -R option the files and subdirectories user also gets changed.

3. chown -c hiox calc.txt

   Here change the owner for the specific 'calc.txt' file only.

Security By File

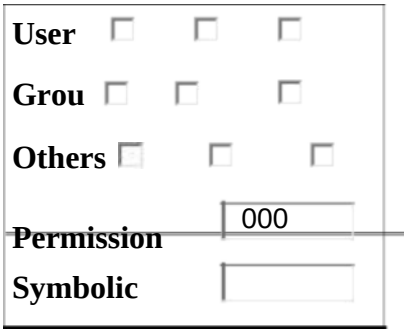
Permissions chmod

Command:

chmod command allows you to alter / Change access rights to files and directories.

File Permission is given for users, group and others as,

	Read Write Execute



SYNTAX:

The Syntax is

chmod [options] [MODE] FileName

File Permission

#	File Permission
0	none
1	execute only
2	write only
3	write and execute
4	read only
5	read and execute
6	read and write
7	set all permissions

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. Create 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 dirname 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.

2. To delete a directory tree: rm -ir

   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 hscripts 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.

diff COMMAND:

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 white space. -c

Use the context output format.

-e Make output that is a valid ed script.

-H Use heuristics to speed handling of large files that have numerous scattered small changes.

-i Ignore changes in case; consider upper- and lower-case letters 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 white space 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
HIOX TEST	HIOX TEST
hscripts.com	HSCRIPTS.com
with friend ship	with friend ship
hiox india	

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 files side by side, ignoring white

   space: diff -by file1.txt file2.txt

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

HIOX TEST HIOX TEST

hscripts.com | HSCRIPTS.com

with friend ship with friend ship

Hioxindia.com <

The third line(with friend ship) 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 printout 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.

HIOX TEST HIOX TEST

hscripts.com HSCRIPTS.com

with friend ship | with friend ship

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	Change the permission on files that are in the subdirectories of the directory that you are currently in.
-c	Change the permission for each file.
-f	Force. Do not report errors.

Hioxindia.com <

EXAMPLE:

1. chgrp hiox test.txt

   The group of 'test.txt' file is root, Change to newgroup hiox.

2. chgrp -R hiox test

   The group of 'test' directory is root. With -R, the files and its subdirectories also changes to newgroup hiox.

3. chgrp -c hiox calc.txt

   They above command is used to change the group for the specific file('calc.txt') only.

About wc

Short for word count, wc displays a count of lines, words, and characters in a file.

Syntax

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

-c	Count bytes.
-m	Count characters.
-C	Same as -m.

-l Count lines.

-w Count words delimited by white space characters or new line characters. Delimiting characters are Extended Unix Code (EUC) characters from any code set defined by iswspace()

File Name of file to word count.

Examples

wc myfile.txt - Displays information about the file myfile.txt. Below is an example of the output.

5 13 57 myfile.txt

5 = Lines
13 = Words
57 = Characters

About split

Split a file into pieces.

Syntax

split [-linecount | -l linecount] [-a suffixlength] [file [name]]

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

-linecount | -l linecount Number of lines in each piece. Defaults to 1000 lines.

-a suffixlength Use suffixlength letters to form the suffix portion of the filenames of the split file. If -a is not specified, the default suffix length is 2. If the sum of the name operand and the suffixlength option-argument would create a filename exceeding NAME_MAX bytes, an error will result; split will exit with a diagnostic message

and no files will be created.

-b n Split a file into pieces n bytes in size.

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

-b n m 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 basename of prefix and suffixlength cannot exceed NAME_MAX bytes; see OPTIONS.

Examples

split -b 22 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 -l 300 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 Do not create a specified file if it does not exist. Do not write any diagnostic messages concerning this condition.

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

-r ref_file Use the corresponding times of the file named by ref_file instead of the current time.

-t time Use the specified time instead of the current time. time will be a decimal number of the form:

[[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].

-f ref_file Use the corresponding times of the file named by ref_file instead of the current time.

File A path name of a file whose times are to be modified.

Examples

settime myfile.txt

Sets the file myfile.txt as the current time / date.

touch newfile.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..

-A or e 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 C STIME TTY TIME CMD
   nirmala 2540 2536 0 15:31 pts/1 00:00:00 bash
   nirmala 2639 2540 0 15:51 pts/1 00:00:00 ps -f

   Displays full information about currently running processes.

kill COMMAND:

kill command is used to kill the background process.

SYNTAX:

The Syntax is

kill [-s] [-l] %pid

OPTIONS:

-s	Specify the signal to send. The signal may be given as a signal name or number.
-l	Write all values of signal supported by the implementation, if no operand is given.
-pid	Process id or job id.
-9	Force to kill a process.

EXAMPLE:

Step by Step process:

- Open a process music player. xmms

  press ctrl+z to stop the process.

- To know group id or job id of the background task. jobs -l
- It will list the background jobs with its job id as,
- xmms 3956 kmail
  3467

- To kill a job or process. kill
  3956

kill command kills or terminates the background process xmms.

About nice

Invokes a command with an altered scheduling priority.

Syntax

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

-increment | -n increment	increment must be in the range 1-19; if not specified, an increment of 10 is assumed. An increment greater than 19 is equivalent to 19. The super-user may run commands with priority higher than normal by using a negative increment such as -10. A negative increment assigned by 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 +13 pico myfile.txt - runs the pico command on myfile.txt with an increment of +13.

About at

Schedules a command to be ran at a particular time, such as a print job late at night.

Syntax

at	executes commands at a specified time.
atq	lists the user's pending jobs, unless the user is the superuser; in that case, everybody's jobs are listed. The format of the output lines (one for each job) is: Job number, 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 queuename] [-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 ran. Format hh:mm. am / pm indication can also follow the time otherwise a 24-hour clock is used. A timezone name of GMT, UCT or ZULU (case insensitive) can follow to specify that the time is in Coordinated Universal Time. Other timezones 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 submit an at-job for potentially immediate execution.

date Specifies the date you wish it to be ran 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 Lists the commands that have been set to run.

-r Cancels 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 enter you should receive a prompt similar to the below example.

commands will be executed using /bin/csh
job 1072250520.a at Wed Dec 24 00:22:00 2003

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 = Deletes the job just created.

or

atrm 23 = Deletes 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 pathname 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, calculate disk usage for link references specified on the command line

-k, show sizes as multiples of 1024 bytes, not 512-byte

-L, calculate disk usage for link references anywhere

-s, report only the sum of the usage in the current directory, not for each file

-x, only traverse files and directories on the device on which the pathname argument is specified.

Other Unix and Unix-like operating systems may add extra options. For example, BSD and GNU du specify a -h option, displaying disk usage in a format easier to read by the user, adding units with the appropriate SI prefix'

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

Sum of directories in human-readable format (Byte, Kilobyte, Megabyte, Gigabyte, Terabyte and Petabyte):

```
 $ du -sh *
 149M directoryOne
 1.8G directoryTwo
```

disk usage of all subdirectories and files including hidden files within the current directory (sorted by filesize) :

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

disk usage of all subdirectories and files including hidden files within the current directory (sorted by reverse filesize) :

```
 $ du -sk .[!.]* *| sort –nr
```

The weight of directories:

```
 $ du -d 1 -c -h
```

df command : Report file system disk space usage

Df command examples - to check free disk space

Type df -h or df -k 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
procbususb     393M  123k  393M  1% /proc/bus/usb
udev           393M  123k  393M  1% /dev
devshm         393M      0  393M  0% /dev/shm
lrm            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
```

du command examples

du shows how much space one ore more files or directories is using. $ du -sh
103M

-s option summarize 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 tools 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 hostname – **telnet**
• outing 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 example of various networking command 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 used 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 machines 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 --udp or netstat –u *will display only UDP connection* **netstat**
-g *will display all multicast network 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 from 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 establish on any port means connectivity between two hosts is working fine.
telnet hostname port *will telnet hostname with the port 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          Flags Mask          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
eth0  UP  BROADCAST  RUNNING  MULTICAST  MTU:1500  Metric:1
      RX packets:126341 errors:0 dropped:0 overruns:0 frame: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
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address               Foreign Address             State
tcp0       0                .230.87:https          ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt FlagsType              State         I-Node Path
unix 10    [ ]         DGRAM                    4970 /dev/log
unix 2     [ ]         DGRAM                    6625 @/var/run/hal/hotplug_socket
unix 2     [ ]         DGRAM                    2952   @udevd
unix 2     [ ]         DGRAM                    100564
unix 3     [ ]         STREAM     CONNECTED     62438 /tmp/.X11-unix/X0
unix 3     [ ]         STREAM     CONNECTED     62437
unix 3     [ ]         STREAM     CONNECTED     10271 @/tmp/fam-root-
unix 3     [ ]         STREAM     CONNECTED     10270
```

```
unix 3  [ ]         STREAM     CONNECTED  9276
unix 3  [ ]         STREAM     CONNECTED  9275
```

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

```
; <<>> DiG 9.2.4 <<>> google.com
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4716
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 4, ADDITIONAL: 4

;; QUESTION SECTION:
;google.com.                IN      A

;; ANSWER SECTION:
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.

;; ADDITIONAL SECTION:
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

;; Query time: 92 msec
;; SERVER: 172.29.36.1#53(172.29.36.1)
;; WHEN: Thu Mar 5 14:38:45 2009
;; MSG SIZE  rcvd: 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         Flags Metric Ref    Use Iface
```

```
54.192.56.321   *          255.255.255.0     U   00            0 eth0
*           255.255.0.0     U     0     0        0 eth0
default     0.0.0.0         UG    0     0        0 eth0
```

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@smashtechl ~]# w
 15:18:22 up 4:38,  3 users, load average: 0.89, 0.34, 0.19
USER     TTYFROM              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 Clear 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 printed in the screen for a given file.

+n Starts up the file from the given number.

EXAMPLE:

1. more -c index.php

   Clears the screen before printing 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 units of lines.
-b	To specify the units 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.

-n number 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 -n 5 index.php

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

4. tail -c 5 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 -c1-3 text.txt

   Output:

   Thi

   Cut the first three letters from the above line.

2. cut -d, -f1,2 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	Sends sorted output to a file.

EXAMPLE:

1. sort test.txt

   Sorts the 'test.txt'file and prints result in the screen.

2. sort -r test.txt

   Sorts the 'test.txt' file in reverse order and prints result in the screen.

About uniq

Report or filter out repeated lines in a file.

Syntax

uniq [-c | -d | -u] [-f fields] [-s char] [-n] [+m] [input_file [output_file]]

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

the input.

-d Suppress the writing of lines that are not repeated in the input.

-u Suppress the writing of lines that are repeated in the input.

-f fields Ignore the first fields fields on each input line when doing comparisons, where fields is a positive decimal integer. A field is the maximal string matched by the basic regular expression:

[[:blank:]]*[^[:blank:]]*

If fields specifies more fields than appear on an input line, a null string will be used for comparison.

-s char Ignore the first chars characters when doing comparisons, where chars is a positive decimal integer. If specified in conjunction with the -f option, the first chars characters after the first fields fields will be ignored. If chars specifies more characters than remain on an input line, a null string will be used for comparison.

-n Equivalent to -f fields with fields set to n.

+m Equivalent to -s chars with chars set to m.

input_file A path name of the input file. If input_file is not specified, or if the input_file is - , the standard input will be used.

output_file A path name of the output file. If output_file is not specified, the standard output will be used. The results are unspecified if the file named by output_file is the file named by input_file.

Examples

uniq myfile1.txt > myfile2.txt - Removes duplicate lines in the first file1.txt and outputs the results to the second file.

About tr

Translate characters.

Syntax

tr [-c] [-d] [-s] [string1] [string2]

-c	Complement the set of characters specified by string1.
-d	Delete all occurrences of input characters that are specified by string1.
-s	Replace instances of repeated characters with a single character.
string1	First string or character to be changed.
string2	Second string or character to change the string1.

Examples

echo "12345678 9247" | tr 123456789 computerh - this example takes an echo response of '12345678 9247' and pipes it through the tr replacing the appropriate numbers with the letters. In this example it would return *computer hope*.

tr -cd '\11\12\40-\176' < myfile1 > myfile2 - this example would take the file myfile1 and strip all non printable characters and take that results to myfile2.

Text processing utilities and Backup utilities:

Text processing utilities:

cat : concatenate files and print on the standard output

Usage: cat [OPTION] [FILE]...

eg. cat file1.txt file2.txt

cat n

file1.txt

echo : display a line of text

Usage: echo [OPTION] [string] ...

eg. echo I love India

echo $HOME

wc: print the number of newlines, words, and bytes in files

Usage: wc [OPTION]... [FILE]...

eg. wc file1.txt

wc L

file1.txt

sort :sort lines of text files

Usage: sort [OPTION]... [FILE]...

eg. sort file1.txt

sort r

file1.txt

General Commands:

date COMMAND:

date command prints the date and time.

SYNTAX:

The Syntax is

date [options] [+format] [date]

OPTIONS:

Slowly adjust the time by sss.fff seconds (fff represents fractions of a second).

-a

This adjustment can be positive or negative.Only system admin/ super user

can adjust the time.

- date - string Sets the time and date to the value specfied in the datestring. The datestr may contain the month names, timezones, 'am', 'pm', etc.

-u Display (or set) the date in Greenwich Mean Time (GMT-universal time).

Format:

%a Abbreviated weekday(Tue).

%A Full weekday(Tuesday).

%b Abbreviated month name(Jan).

%B Full month name(January).

%c Country-specific date and time format..

%D Date in the format %m/%d/%y.

%j Julian day of year (001-366).

%n Insert a new line.

%p String to indicate a.m. or p.m.

%T Time in the format %H:%M:%S.

%t Tab space.

%V Week number in year (01-52); start week on Monday.

EXAMPLE:

date command

date

The above command will print Wed Jul 23 10:52:34 IST 2008

1. To use tab space:

date +"Date is %D %t Time is %T"

The above command will remove space and print as Date is 07/23/08 Time is 10:52:34

2. To know the week number of the

year, date -V

The above command will print 30

3. To set the date,

date -s "10/08/2008 11:37:23"

The above command will print Wed Oct 08 11:37:23 IST 2008

who COMMAND:

who command can list the names of users currently logged in, their terminal, the time they have been logged in, and the name of the host from which they have logged in.

SYNTAX:

The Syntax is

who [options] [file]

OPTIONS:

am i Print the username of the invoking user, The 'am' and 'i' must be space separated.

-b Prints time of last system boot. -d

print dead processes.

-H Print column headings above the output.

-i Include idle time as HOURS:MINUTES. An idle time of . indicates activity

within the last minute.

-m Same as who am i.

-q Prints only the usernames and the user count/total no of users logged in. -T,-

w Include user's message status in the output.

EXAMPLE:

1. who –Uh

   Output:

   NAME LINE TIME IDLE PID COMMENT

   hiox ttyp3 Jul 10 11:08 . 4578

   This sample output was produced at 11 a.m. The "." indiacates activity within the last minute.

2. who am i

   who am i command prints the user name.

echo COMMAND:

echo command prints the given input string to standard output.

SYNTAX:

The Syntax is

echo [options..] [string]

OPTIONS:

-n do not output the trailing newline

-e enable interpretation of the backslash-escaped characters listed below -E

disable interpretation of those sequences in STRINGs

Without -E, the following sequences are recognized and interpolated:

\NNN	the character whose ASCII code is NNN (octal)
\a	alert (BEL)
\\backslash	
\b	backspace
\c	suppress trailing newline
\f	form feed
\n	new line
\r	carriage return
\t	horizontal tab
\v	vertical tab

EXAMPLE:

echo command

echo "hscripts Hiox India"

The above command will print as hscripts Hiox India

1. To use backspace:

   echo -e "hscripts \bHiox \bIndia"

   The above command will remove space and print as hscriptsHioxIndia

2. To use tab space in echo command

   echo -e "hscripts\tHiox\tIndia"

   The above command will print as hscripts Hiox India

passwd COMMAND:

passwd command is used to change your password.

SYNTAX:

The Syntax is

passwd [options]

OPTIONS:

-a	Show password attributes for all entries.
-l	Locks password entry for name.
-d	Deletes password for name. The login name will not be prompted for password.
-f	Force the user to change password at the next login by expiring the password for name.

EXAMPLE:

1. passwd

   Entering just passwd would allow you to change the password. After entering passwd you will receive the following three prompts:

   Current Password:
   New Password:
   Confirm New Password:

   Each of these prompts must be entered correctly for the password to be successfully changed.

pwd COMMAND:

pwd - Print Working Directory. pwd command prints the full filename of the current working directory.

SYNTAX:

The Syntax is

pwd [options]

OPTIONS:

-P The pathname printed will not contain symbolic links.

-L The pathname printed may contain symbolic links.

EXAMPLE:

1. Displays the current working directory.

pwd

If you are working in home directory then, pwd command displays the current working directory as /home.

cal COMMAND:

cal command is used to display the calendar.

SYNTAX:

The Syntax is

cal [options] [month] [year]

OPTIONS:

-1 Displays single month as output.

-3 Displays prev/current/next month output.

-s Displays sunday as the first day of the week. -m Displays Monday as the first day of the week.

-j Displays Julian dates (days one-based, numbered from January 1).

-y Displays a calendar for the current year.

EXAMPLE:

1. cal

Output:

```
   September 2008
Su Mo Tu We Th Fr Sa
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30
```

cal command displays the current month calendar.

2. cal -3 5 2008

Output:

```
     April 2008          May 2008            June 2008
Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa
 1  2  3  4  5                    1  2  3   1  2  3  4  5  6  7
 6  7  8  9 10 11 12   4  5  6   7  8  9 10   8  9 10 11 12 13 14
13 14 15 16 17 18 19  11 12 13  14 15 16 17  15 16 17 18 19 20 21
20 21 22 23 24 25 26  18 19 20  21 22 23 24  22 23 24 25 26 27 28
27 28 29 30          25 26 27 28 29 30 31   29 30
```

Here the cal command displays the calendar of April, May and June month of year 2008.

login Command

Signs into a new system.

Syntax

login [-p] [-d device] [-h hostname | terminal | -r hostname] [name [environ]]

-p	Used to pass environment variables to the login shell.
-d device	login accepts a device option, device. device is taken to be the path name of the TTY port login is to operate on. The use of the device option can be expected to improve login performance, since login will not need to call ttyname. The -d option is available only to users whose UID and effective UID are root. Any other attempt to use -d will cause login to quietly exit.
-h hostname | terminal	Used by in.telnetd to pass information about the remote host and terminal type.
-r hostname	Used by in.rlogind to pass information about the remote host.

Examples

login computerhope.com - Would attempt to login to the computerhope domain.

uname command

Print name of current system.

Syntax

uname [-a] [-i] [-m] [-n] [-p] [-r] [-s] [-v] [-X] [-S systemname]

-a	Print basic information currently available from the system.
-i	Print the name of the hardware implementation (platform).

-m	Print the machine hardware name (class). Use of this option is discouraged; use uname -p instead.
-n	Print the nodename (the nodename is the name by which the system is known to a communications network).
-p	Print the current host's ISA or processor type.
-r	Print the operating system release level.
-s	Print the name of the operating system. This is the default.
-v	Print the operating system version.
-X	Print expanded system information, one information element per line, as expected by SCO Unix. The displayed information includes: • system name, node, release, version, machine, and number of CPUs. • BusType, Serial, and Users (set to "unknown" in Solaris) • OEM# and Origin# (set to 0 and 1, respectively)
-S systemname	The nodename may be changed by specifying a system name argument. The system name argument is restricted to SYS_NMLN characters. SYS_NMLN is an implementation specific value defined in <sys/utsname.h>. Only the super-user is allowed this capability.

Examples

uname -arv

List the basic system information, OS release, and OS version as shown below.

SunOS hope 5.7 Generic_106541-08 sun4m sparc SUNW,SPARCstation-10

uname -p

Display the Linux platform.

Files and Directories

Working with Files

In this chapter we learn how to create, open, read, write, and close files.

UNIX File Structure

In UNIX, everything is a file.

Programs can use disk files, serial ports, printers and other devices in the exactly the same way as they would use a file.

Directories, too, are special sorts of files.

File types

Most files on a UNIX system are regular files or directories, but there are additional types of files:

1. **Regular files**: The most common type of file, which contains data of some form. There is no distinction to the UNIX kernel whether this data is text or binary.
2. **Directory file:** A file contains the names of other files and pointers to information on these files. Any process that has read permission for a directory file can read the contents of the directory, but only the kernel can write to a directory file.
3. **Character special file**: A type of file used for certain types of devices on a system.
4. **Block special file**: A type of file typically used for disk devices. All devices on a system are either character special files or block special files.
5. **FIFO**: A type of file used for interprocess communication between processes. It's sometimes called a named pipe.
6. **Socket:** A type of file used for network communication between processes. A socket can also be used for nonnetwork communication between processes on a single host.
7. **Symbolic link:** A type of file that points to another file.

The argument to each of different file types is defined as follows_

Macro	Type of file
S_ISREG()	Regular file
S_ISDIR()	Directory file
S_ISCHR()	Character special file
S_ISBLK()	Block special file
S_ISFIFO()	Pipe or FIFO
S_ISLNK()	Symbolic link
S_ISSOCK()	Socket

Searching files and directories

- **find** Find files (find <start directory> -name <file name> -print)

Ex: *find /home –name readme -print*

(Search for readme starting at home and output full path.)

“/home" = Search starting at the home directory and proceed through all its subdirectories

"-name readme" = Search for a file named readme

"-print" = Output the full path to that file

- **locate** File locating program that uses the slocate database.

Ex: locate –u to create the database,

locate <file/directory> to find file/directory

Zipping and Unzipping

File compression, backing up and restoring

- compress Compress data.
- uncompress Expand data.
- cpio Can store files on tapes. to/from archives.
- gzip - zip a file to a gz file.
- gunzip - unzip a gz file.
- tar Archives files and directories. Can store files and directories on tapes.

Ex: tar -zcvf <destination> <files/directories> - Archive copy groups of files. tar –zxvf <compressed file> to uncompress

- zip – Compresses a file to a .zip file.
- unzip – Uncompresses a file with .zip extension.

Two Marks Questions

1) What is Linux?

Linux is an operating system based on UNIX, and was first introduced by Linus Torvalds. It is based on the Linux Kernel, and can run on different hardware platforms manufactured by Intel, MIPS, HP, IBM, SPARC and Motorola. Another popular element in Linux is its mascot, a penguin figure named Tux.

2) What is the difference between UNIX and LINUX?

Unix originally began as a propriety operating system from Bell Laboratories, which later on spawned into different commercial versions. On the other hand, Linux is free, open source and intended as a non-propriety operating system for the masses.

3) What is BASH?

BASH is short for Bourne Again SHell. It was written by Steve Bourne as a replacement to the original Bourne Shell (represented by /bin/sh). It combines all the features from the original version of Bourne Shell, plus additional functions to make it easier and more convenient to use. It has since been adapted as the default shell for most systems running Linux.

4) What is Linux Kernel?

The Linux Kernel is a low-level systems software whose main role is to manage hardware resources for the user. It is also used to provide an interface for user-level interaction.

5) What is LILO?

LILO is a boot loader for Linux. It is used mainly to load the Linux operating system into main memory so that it can begin its operations.

6) What is a swap space?

A swap space is a certain amount of space used by Linux to temporarily hold some programs that are running concurrently. This happens when RAM does not have enough memory to hold all programs that are executing.

7) What is the advantage of open source?

Open source allows you to distribute your software, including source codes freely to anyone who is interested. People would then be able to add features and even debug and correct errors that are in the source code. They can even make it run better, and then redistribute these enhanced source code freely again. This eventually benefits everyone in the community.

8) What are the basic components of Linux?

Just like any other typical operating system, Linux has all of these components: kernel, shells and GUIs, system utilities, and application program. What makes Linux advantageous over other operating system is that every aspect comes with additional features and all codes for these are downloadable for free.

9) Does it help for a Linux system to have multiple desktop environments installed?

In general, one desktop environment, like KDE or Gnome, is good enough to operate without issues. It's all a matter of preference for the user, although the system allows switching from one environment to another. Some programs will work on one environment and not work on the other, so it could also be considered a factor in selecting which environment to use.

10) What is the basic difference between BASH and DOS?

The key differences between the BASH and DOS console lies in 3 areas:
– BASH commands are case sensitive while DOS commands are not;
– under BASH, / character is a directory separator and \ acts as an escape character. Under DOS, / serves as a command argument delimiter and \ is the directory separator
– DOS follows a convention in naming files, which is 8 character file name followed by a dot and 3 character for the extension. BASH follows no such convention.

11) What is the importance of the GNU project?

This so-called Free software movement allows several advantages, such as the freedom to run programs for any purpose and freedom to study and modify a program to your needs. It also allows you to redistribute copies of a software to other people, as well as freedom to improve software and have it released to the public.

12) Describe the root account.

The root account is like a systems administrator account, and allows you full control of the system. Here you can create and maintain user accounts, assigning different permissions for each account. It is the default account every time you install Linux.

13) What is CLI?

CLI is short for Command Line Interface. This interface allows user to type declarative commands to instruct the computer to perform operations. CLI offers an advantage in that there

is greater flexibility. However, other users who are already accustom with using GUI find it difficult to remember commands including attributes that come with it.

14) What is GUI?

GUI, or Graphical User Interface, makes use of images and icons that users click and manipulate as a way of communicating with the computer. Instead of having to remember and type commands, the use of graphical elements makes it easier to interact with the system, as well as adding more attraction through images, icons and colors.

15) How do you open a command prompt when issuing a command?

To open the default shell (which is where the command prompt can be found), press Ctrl-Alt-F1. This will provide a command line interface (CLI) from which you can run commands as needed.

16) How can you find out how much memory Linux is using?

From a command shell, use the “concatenate” command: cat /proc/meminfo for memory usage information. You should see a line starting something like: Mem: 64655360, etc. This is the total memory Linux thinks it has available to use.

17) What is typical size for a swap partition under a Linux system?

The preferred size for a swap partition is twice the amount of physical memory available on the system. If this is not possible, then the minimum size should be the same as the amount of memory installed.

18) What are symbolic links?

Symbolic links act similarly to shortcuts in Windows. Such links point to programs, files or directories. It also allows you instant access to it without having to go directly to the entire pathname.

19) Does the Ctrl+Alt+Del key combination work on Linux?

Yes, it does. Just like Windows, you can use this key combination to perform a system restart. One difference is that you won’t be getting any confirmation message and therefore, reboot is immediate.

20) How do you refer to the parallel port where devices such as printers are connected?

Whereas under Windows you refer to the parallel port as the LPT port, under Linux you refer to it as /dev/lp . LPT1, LPT2 and LPT3 would therefore be referred to as /dev/lp0, /dev/lp1, or /dev/lp2 under Linux.

21) Are drives such as harddrive and floppy drives represented with drive letters?

No. In Linux, each drive and device has different designations. For example, floppy drives are referred to as /dev/fd0 and /dev/fd1. IDE/EIDE hard drives are referred to as /dev/hda, /dev/hdb, /dev/hdc, and so forth.

22) How do you change permissions under Linux?

Assuming you are the system administrator or the owner of a file or directory, you can grant permission using the chmod command. Use + symbol to add permission or – symbol to deny permission, along with any of the following letters: u (user), g (group), o (others), a (all), r (read), w (write) and x (execute). For example the command chmod go+rw FILE1.TXT grants read and write access to the file FILE1.TXT, which is assigned to groups and others.

23) In Linux, what names are assigned to the different serial ports?

Serial ports are identified as /dev/ttyS0 to /dev/ttyS7. These are the equivalent names of COM1 to COM8 in Windows.

24) How do you access partitions under Linux?

Linux assigns numbers at the end of the drive identifier. For example, if the first IDE hard drive had three primary partitions, they would be named/numbered, /dev/hda1, /dev/hda2 and /dev/hda3.

25) What are hard links?

Hard links point directly to the physical file on disk, and not on the path name. This means that if you rename or move the original file, the link will not break, since the link is for the file itself, not the path where the file is located.

26) What is the maximum length for a filename under Linux?

Any filename can have a maximum of 255 characters. This limit does not include the path name, so therefore the entire pathname and filename could well exceed 255 characters.

27)What are filenames that are preceded by a dot?

In general, filenames that are preceded by a dot are hidden files. These files can be configuration files that hold important data or setup info. Setting these files as hidden makes it less likely to be accidentally deleted.

28) Explain virtual desktop.

This serves as an alternative to minimizing and maximizing different windows on the current desktop. Using virtual desktops, each desktop is a clean slate where you can open one or more programs. Rather than minimizing/restoring all those programs as needed, you can simply shuffle between virtual desktops with programs intact in each one.

29) How do you share a program across different virtual desktops under Linux?

To share a program across different virtual desktops, in the upper left-hand corner of a program window look for an icon that looks like a pushpin. Pressing this button will “pin” that application in place, making it appear in all virtual desktops, in the same position onscreen.

30) What does a nameless (empty) directory represent?

This empty directory name serves as the nameless base of the Linux file system. This serves as an attachment for all other directories, files, drives and devices.

31) What is the pwd command?

The pwd command is short for print working directory command. It’s counterpart in DOS is the cd command, and is used to display the current location in the directory tree.

32) What are daemons?

Daemons are services that provide several functions that may not be available under the base operating system. Its main task is to listen for service request and at the same time to act on these requests. After the service is done, it is then disconnected and waits for further requests.

33) How do you switch from one desktop environment to another, such as switching from KDE to Gnome?

Assuming you have these two environments installed, just log out from the graphical interface. Then at the Log in screen, type your login ID and password and choose which session type you wish to load. This choice will remain your default until you change it to something else.

34) What are the kinds of permissions under Linux?

There are 3 kinds of permissions under Linux:
– Read: users may read the files or list the directory
– Write: users may write to the file of new files to the directory
– Execute: users may run the file or lookup a specific file within a directory

35) How does case sensitivity affect the way you use commands?

When we talk about case sensitivity, commands are considered identical only if every character is encoded as is, including lowercase and uppercase letters. This means that CD, cd and Cd are three different commands. Entering a command using uppercase letters, where it should be in lowercase, will produce different outputs.

36) What are environmental variables?

Environmental variables are global settings that control the shell's function as well as that of other Linux programs. Another common term for environmental variables is global shell variables.

37) What are the different modes when using vi editor?

There are 3 modes under vi:
– Command mode – this is the mode where you start in
– Edit mode – this is the mode that allows you to do text editing
– Ex mode – this is the mode wherein you interact with vi with instructions to process a file

38) Is it possible to use shortcut for a long pathname?

Yes, there is. A feature known as filename expansion allows you do this using the TAB key. For example, if you have a path named /home/iceman/assignments directory, you would type as follows: /ho[tab]/ice[tab]/assi[tab] . This, however, assumes that the path is unique, and that the shell you're using supports this feature.

39) What is redirection?

Redirection is the process of directing data from one output to another. It can also be used to direct an output as an input to another process.

40) What is grep command?

grep a search command that makes use of pattern-based searching. It makes use of options and parameters that is specified along the command line and applies this pattern into searching the required file output.

41) What could possibly be the problem when a command that was issued gave a different result from the last time it was used?

One highly possible reason for getting different results from what seems to be the same command has something to do with case sensitivity issues. Since Linux is case sensitive, a command that was previously used might have been entered in a different format from the present one. For example, to lists all files in the directory, you should type the command ls, and not LS. Typing LS would either result in an error message if there is no program by that exact name exist, or may produce a different output if there is a program named LS that performs another function.

42) What are the contents in /usr/local?

It contains locally installed files. This directory actually matters in environments where files are stored on the network. Specifically, locally-installed files go to /usr/local/bin, /usr/local/lib, etc.). Another application of this directory is that it is used for software packages installed from source, or software not officially shipped with the distribution.

43) How do you terminate an ongoing process?

Every process in the system is identified by a unique process id or pid. Use the kill command followed by the pid in order to terminate that process. To terminate all process at once, use kill 0.

44) How do you insert comments in the command line prompt?

Comments are created by typing the # symbol before the actual comment text. This tells the shell to completely ignore what follows. For example: “# This is just a comment that the shell will ignore.”

45) What is command grouping and how does it work?

You can use parentheses to group commands. For example, if you want to send the current date and time along with the contents of a file named OUTPUT to a second file named MYDATES, you can apply command grouping as follows: (date cat OUTPUT) > MYDATES