Linux Commands

Importance of Linux:

Most of the Projects Build is deployed in Linux Servers. So, it is responsibility to understand Deployment instructions in Linux.

Understanding Linux will give added advantage to understand the Functionality of more security applications like Banking and Insurance.

All Product based companies recruiting Engineers based on Linux knowledge.

Introduction

- Linux development started around 1991 by Linus Torlvads.
- Linux is a multi-user, multitasking Operating System.
- Linux is Open Source not like Unix.
- Linux is "case-sensitive" ls is different of LS.
- Below are few of the Linux distributions(flavors).

1) Redhat

2) Ubuntu.

3) CentOS

4) SuSe Linux

5) Fedora

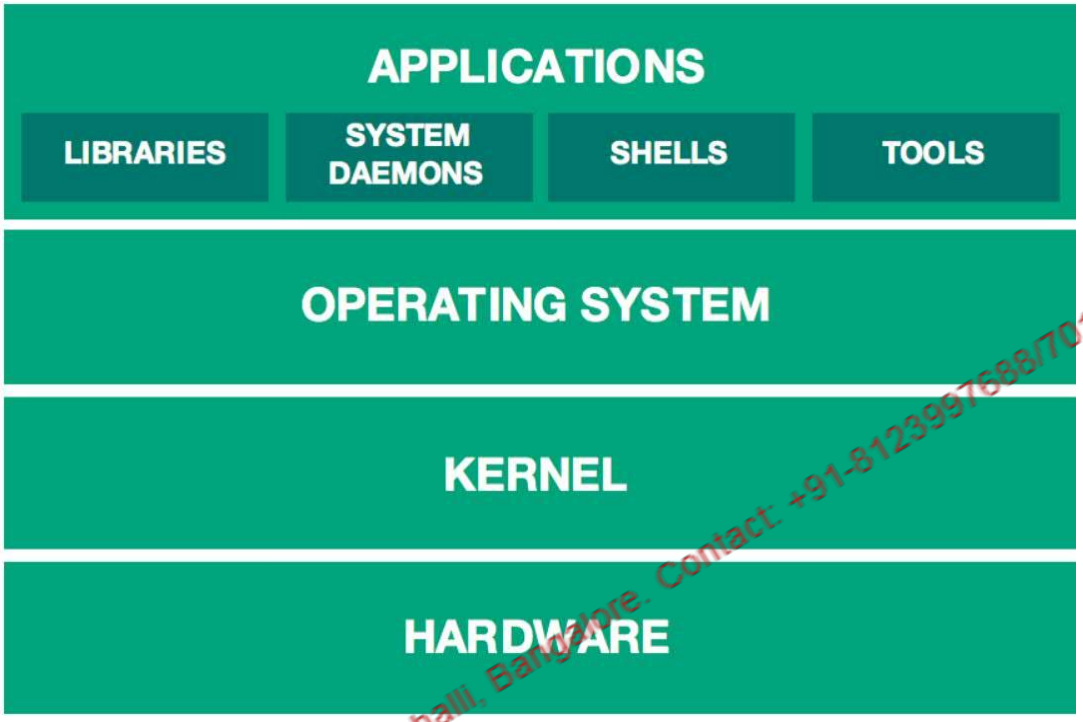
6) Gentoo

7) Mandriva.

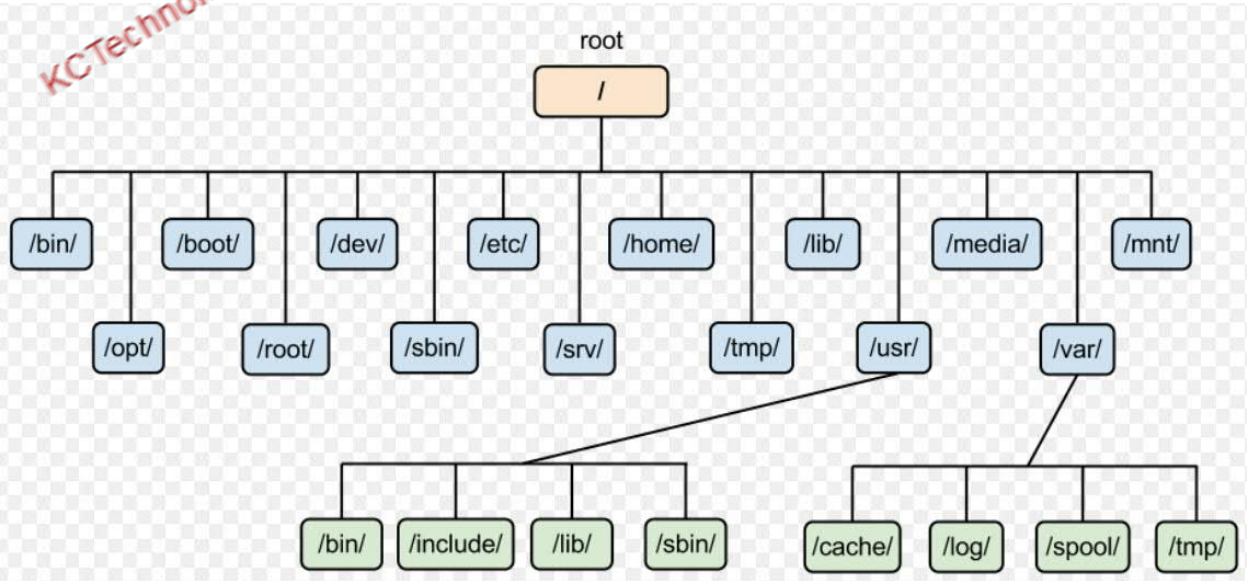
8) Debian.

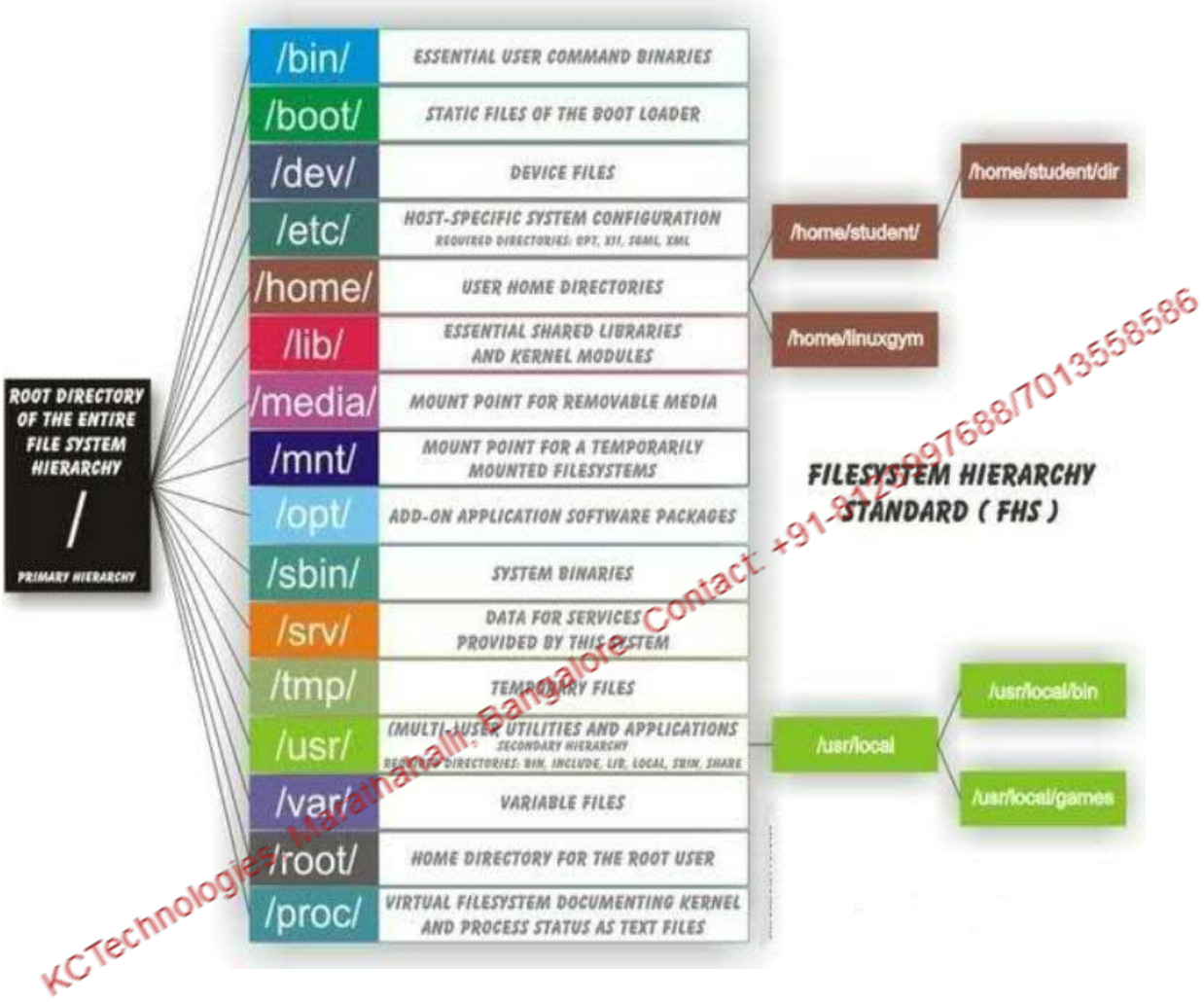
9) Slackware

Linux Architecture:



Linux File Structure:





Explanation of few important directories:

Linux has 2 kind of users: 1) **root** 2) **normal** users. We can have only 1 root user in Linux where we can have any no.of normal users.

/ → is the root and is the starting point in Linux.

→ Indicates root user.

$ → Indicates normal user.

. → Indicates current location.

~ → Indicates user home.

Linux has many pre-defined directories under **/** called: /etc, /bin, /usr, /dev,

/lib, /tmp, /sbin, /home, /root

Note: Linux file system is case sensitive. (Linux and linux both are different).

But, windows will not allow to create same file/directory with same name with different case. Ex (Windows and windows are same).

Let's discuss about few important points of Linux:

In Linux, Hidden files start with . (dot) extension.

Following are the directories in the RedHat Operating System.

In Linux directory structure starts with / and / is called root directory.

bin : It has the commands and binary files. User can access all the commands present in it.

Ex: mkdir, ls, cd, ps….

sbin : Just like /bin it contains the commands and files, but only root user can access.

Ex: ifconfig, reboot, shutdown and swapon …

dev : It is the location of special or device files.

Ex: USB or any device attached to the server.

etc : It is the location configuration files.

lib : It is the location which contains all helpful system libraries. In simple terms, these are helpful files which are used by an application or a command process for their proper execution.

proc : It contains the process information.

tmp : It contains all the temporary files.

usr : It contains the binaries and libraries. WAS, IHS or DB2 etc related software will be install in this directory.

var : It contains variable files. This includes system log files(/var/log), emails (/var/mail) and temp files needed across reboots (/var/tmp)

home : It contains home directories for all users to store their personal files. Ex: /home/krishna or /home/Chaitanya

For every new user, a directory will be created under **home** directory.

root: It is the home for root user.

Contents:

touch

cat

Copy a file

cp

Rename a file

mv

Listing files and directories

ls

Changing file permissions

chmod

Removing a file

rm

Directory related commands

mkdir

rmdir

cd

3) **Essential Unix Commands**

passwd

File related commands

wc

sort

cut

grep

fgrep

Viewing files

head

tail

4) **Process in Unix**

What is running right now?

Background processes

Killing a process

History of UNIX:

UNIX is a CUI (Command User Interface) operating system which was first developed in the 1960s.

Operating System: An operating system can be defined as the software that controls the H/W resources of the computer and provides an environment under which programs can run.

UNIX is almost 45 years old OS. Before development of UNIX OS at AT & T Bell labs, s/w team lead by Ken Thomson, Dennis Ritchie and Rudd Canday worked on MULTICS project (Multi Information Computing System). Initially, MULTICS was developed for only two users. Based on the same concept in 1969, UNICS (Uniplexed Information Computing System) OS was developed for 100's of users. In 1973 named as UNIX. It is open source OS.

Linux almost had same Unix Like feature for e.g.

- Like UNIX and Linux are written is C.
- Like Unix and Linux are Multi-user/Multitasking OS.
- Like Unix and Linux runs on different hardware platform (Portable)

Flavours of UNIX:

- Aix by IBM
- Mac OS by apple
- RedHat Linus by red hat s/w.
- Solaris by Sun Solaris

Multitasking Capability

Performing tasks simultaneously rather than sequentially. A multi-tasking operating system allows more than one program to be running at a time.

Communication:

Communication between different terminals.

Security

UNIX provides 3 levels of security to protect data.

1) Assigning passwords and login names to individual users.

2) At file level.

3) File encryption utility.

Portability:

It can be ported (Transfer from one system to another) to almost any computer system.

The First Faltering Steps:

When you try to access your system, UNIX will display a prompt that looks something like this:

Login:

Password:

$who am i

It displays current user name, terminal number, date and time at which you logged in.

$who

Krishna tty3a Jan 16 01:25

Chaitanya tty6c May 22 15:10

Hari tty3b June 18 10:19

It displays login name, terminal number/serial port, date & time when logged in. Note that this shown only for users who are currently logged in.

$whoami: It displays working user name.

$w

$pwd(print working directory): It displays the present working directory.

$logname: It prints user's login name

$date: It displays system date and time (current date and time)

$clock: Determine processor time.

$cal

$cal 9 2003

It will display calendar of September 2003.

$cal 2010

It will display calendar of entire year 2010.

UNIX File System:

A file is the basic structure used to store information on the UNIX system. All utilities, applications, data in UNIX is stored as files. Even a directory is treated as a file which contains several other files. An UNIX file system resembles an upside-down tree. File system begins with a directory called **root.** The root directory is denoted as slash (/).

Creating files:

$touch sample

This creates an empty file called sample. **The size of the file would be zero bytes, since touch does not allow you to store anything in a file.** Then does touch serve any purpose? Yes, to create several empty files quickly.

$touch sample1 sample2 sample3 sample4

But what if you want to store a few lines in a file. Just type the command

$cat > sample1

******

******

******

Then press, **Ctrl + d** to exit.

To append data to the existing file.

$cat >> sample1

Then press, **Ctrl + d** to exit.

To view the contents of an existing file.

$cat filename

cat: Concatenate files and print on the standard output.

cat krishna.txt

cat krishna.txt | tr a-z A-Z

cat -n hello.sh

tac: Does the same as cat, but displays the contents in reverse order

#tac krishna.txt

#cat /dev/null > krishna.txt : It will clear/null the contents of krishna.txt file.

#ls | tee teeoutput.txt: It will display the output in console as well into file also.

#ls | tee -a teeoutput.txt: It will append to the file.

#ls | tee -a teeoutput1.txt teeoutput2.txt teeoutput3.txt: It will append to the file.

Copy a file:

Syntax: cp sourceFile targetFfile

$cp sample1 sample2

This will copy contents of sample1 into a sample2. If sample2 already exist, then it overwrites.

$cp –i sample1 sample2 → If sample2 already existed then it asks the confirmation.

e.g. cp Krish.txt Krish_cp.txt

cp Krish.txt /root/Testing/TestDir1/Krishna

Moving and renaming a file:

If you want to rename the file test to sample, then we would say:

$mv test sample

mv command also has the power to rename directories.

$mv olddir newdir

Note: Moving a file implies removing it from its current location and copying it at a new location.

$mv file1 file2 newdir

mv : Rename SOURCE to DEST, or move SOURCE to DIRECTORY/file.

Syntax : mv <<File/ Directory Actual Name>> << File/ Directory New Name>>

Listing files and directories

$ls –l

$ls -> Show contents of working directory

ls file1 -> list file1, if it exists in working directory

$ls dir1 -> show contents of the directory dir1

$ls -a -> shows all your files, including hidden ones.

$ls -al -> gives detailed listing of contents.

$ls *.doc -> shows all files with suffix ".doc"

$ls -lt -> Lists files based on last modified timestamp (last modified/created files display first on the screen).

$ls –ltr -> Time of last modification will come last. Here **r** represents reverse.

ls: Lists the contents of your current working directory.

Some options with ls are:

-l list in long format

-a all files including those which begin with dot

-F It is used to classify the directories, executables. It will append indicator (one of */=>@|) to entries

-i list the inode number in the first column

-R recursively list all directories and subdirectories

-r Display in reverse order.

-t sorts files by access time.

-d shows the names of directories and not their contents.

-h Indicates human readable. This mentions file sizes in kilobytes, megabytes, or

Gigabytes, instead of just bytes, which is the default setting. Use this option with the

–l option only.

-S Sorts files by file size. This option is useful only when used together with the

option –l

#ls –l

1 2 3 4 5 6 7

1) Permissions 2) Number of links 3) Owner 4) Group Owner 5) Size of the File in Bytes 6) Date and Time of created 7) File name/Dir Name

#ls –F

Here / indicates the directories, * indicates the executable files, @ indicates soft link.

#ls b* : List files starting with b.

#ls *b : List files ending with b.

#ls *b* : Files just having letter b.

#ls ? : List single lettered file

#ls ?? : List double lettered file

#ls -l | grep ^d: It will display only the subdirectories from the current directory.

 # ls -ld */ : It will display only the subdirectories from the current directory along with soft link directories also.

File types in a long list

Symbol Meaning

- Regular file

d Directory

L Soft Link

C Special file

S Socket

B Block device

clear : Clears the terminal screen.

exit (OR) Ctrl+d : Terminate the process, returning returnCode to the system as the exit status. If returnCode isn't specified then it defaults to 0.

Changing file permissions:

chmod: chmod is the command to change file or directory permissions.

Permissions **weight**

r- read 4

w- write 2

x- execute 1

Ex: $chmod 700 filename

u for user or owner

g for group

o for others

chmod : Change file access permissions.

Syntax : chmod << options >> <<access permissions >><< filename/directory >>

#mv Krish.txt Krishna.txt

#mv Krishna.txt /root/Testing/TestDir1

777 ----> Change permission of the file, make it to executable by all (Owner, Group and Others).

400 ----> Change the permission of the file – only Owner can read.

#chmod –R 777 Testing

Here R represents recursively.

Following example shows another way of giving the permissions.

 # clear

 # pwd

 # ls -l

 # chmod u-rwx Krish.txt

 # ls -l

 # chmod og-rwx Krish.txt

 # ls -l

 # chmod uog+rwx Krish.txt

 # ls –l

 # chmod uog-rwx Krish.txt

 # ls –l

 # chmod o+wx,u-x,g=r-x Krish.txt

 # ls -l

+ represents adds the designated permission(s) to a file or directory.

- represents removes the designated permission(s) from a file or directory.

= represents Sets the designated permission(s)

umask : User Mask or User file creation MASK : It is used to set the permissions for files/directories newly created on a Linux Machine.

• The default **umask 002** used for normal user. With this mask default directory permission are ***drwxrwxr-x (775)*** and default file permissions are ***-rw-rw--r-- (664)***.

• The default **umask for the root user is 022** result into default directory permissions are ***drwxr-xr-x (755)*** and default file permissions are ***-rw-r--r—(644)***.

• For directories, the **base permissions** are (rwxrwxrwx) 0777 and for files they are

0666 (rw-rw-rw-).

• Umask setup is in **/etc/bashrc** or **/etc/profile** file as follows.

Below sample code from **/etc/bashrc** file.

groupadd: Create a new group.

usermod : Changes user attributes.

usermod –G <<Group Name>> <<User Name>>

#usermod –G linuxadm krishna

#usermod –g <<Group Name>> <<User Name>>: To change primary group.

For locking:

#usermod -e : The date on which the user account will be disabled. The date is specified

in the format YYYY-MM-DD.

#usermod -e 2013-05-23 hari

groupmod : Modify a group.

Syntax: groupmod -n << group name >> << new group name >>

#groupmod -n krishnaadmin linuxadmin

In the above example the groupmod command would change the group "linuxadmin" to

"krishnaadmin".

groupdel: Deletes a new group.

Syntax: groupdel << group name >>

#groupdel linuxadmin

Removing a file:

$rm file1

It removes file1, if file permissions permit.

Remove multiple files:

rm command is used to remove files and directories, even if they are non-empty.

$rm file1 file2 file3

$rm –i filename **Here, i represents interactively**

rmdir : Removes a directory.

Some conditions to use **rmdir** command

- **Must be empty before it is deleted**
- **Should not be the current directory or a directory at a higher level**
- **The directory name should exists.**
- **Should not be HOME directory of the user**

rm: Removes (unlinks) files or directories.

Syntax: rm <<File/Directory>>

rm –rf /Testing ----> Removes directory which has contents.

rm Krish.txt -----> Removes the file.

rm *.txt -----> Removes all the files extension with .txt.

Options can be

- **-i interactively asks for confirming the deletion of files. It is useful in**

avoiding accidental erasure of the file

- **-r option provides a convenient way to erase a directory even if it is not empty**
- **-f option will forcefully remove a file to which we don't have a Write permission.**

What is the difference between rm and rmdir?

Ans) rm removes the files and directories.

rmdir command removes the directories from the file system. The directory must be empty before it can be remove.

Directory related commands:

$mkdir dir1

Make/create directory called dir1 in your working directory.

$mkdir dir1 dir2 dir3 dir4

To create multiple directories.

$mkdir –p dir1/dir2/dir3/dir4

Creates all the parent directories specified in the given path.

Some options with mkdir are.

-m Sets the access mode for the new directory.

-p If the parent directories don't exist, this command creates them.

-v Print a message for each created directory

e.g.

1)# mkdir Test1

 # ls -lt

 # mkdir -m 777 Test2

 # ls -lt

2)# mkdir –p Test1/Test2/Test3 : It will create the parent directories.

 # ls –lt

3)# mkdir –v Krishna

4)#mkdir Krishna{1,2,3,4,5}

Note: Same like for **rmdir Chaitanya{1,2,3,4,5}**.

5)#mkdir -m 777 Chaitanya : It will create the **Chaitanya** folder with 777 permissions.

$rmdir dir1

It removes directory dir1.

Note: Directories must be empty before you remove them.

To recursively remove nested directories, use the rm command with the **-r** option.

$rm -r directoryName

chown : Change file owner and group.

ls –l

chown krish Krish.txt

ls -l

ls –l

chown --from=root krishna devops.txt

ls -l

Changing the owner and group.

 # ls –l

 # chown krishna:staff devops.txt

Here, krisha will be the owner of the filer and staff will be the group.

Changing directory:

$cd dirname

$cd .. → To change into parent directory (To move one level up).

cd : To Change working directory.

Syntax : cd << Directory >>

Owner Group Others

cd <<Path/ Directory>>

e.g. : # mkdir /root/Testing/TestDir1/Krishna

cd /root/Testing/TestDir1/Krishna

More about cd command

cd / takes to home directory

cd << dirname >>

cd << pathname >>

cd ..To move one level up (immediate parent directory)

cd ../.. To move two levels up (so on)

cd ~ It will takes to home directory.

cd It will takes to home directory.

cd - It will takes to previous directory.

Relative path cd local

Absoulte path cd /usr/local

What is the difference between absolute path and relative path?

Ans) An absolute path begins with the root directory /.

Ex: /home/chaitanya/devops/devops.txt

A relative path starts from the current working directory.

Ex: ./devops/devops.txt

Most of the commands in Linux will work from current directory. Ex: ls, cd, mkdir etc.

Essential Unix Commands:

$ useradd → To add user.

$ passwd → Updates a user authentication.

#passwd -u hari

#passwd -d < User Name > : It will deletes the already existing password in /etc/shadow file for specified user, resulting it restricts the login into server.

#passwd -d krishna

File related commands:

$wc filename

This command is used to count the number of lines, words & characters from a file.

Options:

-l → Lists no.of lines.

-w → Lists no.of words.

-m → Lists no.of characters

-c → Lists no.of bytes.

-L → Length of longest line

$wc –l filename

$wc –w filename

$wc –lw filename

$wc –c filename

$wc –l file1 file2 file3

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

Syntax : wc [-c | -m] [-l] [-w] [-L] [File ...]

Options can be

- **-c Display number of bytes**
- **-m Display number of character.**
- **-l Display number of lines.**
- **-w Display number of words.**
- **-L Displays the length of the longest line.**

sort command:

1. Sort command can be used for sorting the contents of a file.
2. It can merge multiple sorted files and store the result in the specified output file.
3. Sort can display unique lines.

$sort myfile1

$sort file1 file2 file3

$sort –o myresult file1 file2 file3 → Here, with –o option write result to myresult instead of standard output

$sort –u –o result file1 file2 file3 → **-**u option is to display **unique** lines

$sort –m file1 file2 → -m Merge file1 content with file2.

sort: Sort allows us to sort data in numeric, alphabetic and reverse order whether it's of column or multiple columns with delimiters. Sort order can be restricted to individual columns and then merged to one single file. Sorting becomes handy in many situations.

#cat names.txt

#sort names.txt

#cat numbers.txt

#sort numbers.txt

#sort -n numbers.txt

#sort -nr numbers.txt

1. Sort command can be used for sorting the contents of a file.
2. It can merge multiple sorted files and store the result in the specified output file.
3. Sort can display unique lines.

$sort myfile1

$sort file1 file2 file3

$sort –o myresult file1 file2 file3 here, with –o option write result to myresult instead of standard output

$sort –u –o result file1 file2 file3 **-**u option is to display **unique** lines

$sort –m file1 file2 -m Merge file1 content with file2.

---

diff: $diff file1 file2 ->It displays line by line difference between file1 and file2.

Finger: $ **finger** ->It displays information about user.

cut Command:

The cut command in UNIX is a command for cutting out the sections from each line of files and writing the result to standard output. It can be used to cut parts of a line by **byte position, character and field**. Basically, the cut command slices a line and extracts the text. It is necessary to specify option with command otherwise it gives error. If more than one file name is provided, then data from each file is **not precedes** by its file name.

Syntax: cut OPTION... [FILE]..

Create a file with below text:

$ cat state.txt

Andhra Pradesh

Arunachal Pradesh

Assam

Bihar

Chhattisgarh

List without ranges

$ cut -b 1,2,3 state.txt

Output:

And

Aru

Ass

Bih

Chh

List with ranges

$ cut -b 1-3,5-7 state.txt

Output:

Andra

Aruach

Assm

Bihr

Chhtti

It uses a special form for selecting bytes from beginning up to the end of the line:

In this, 1- indicate from 1st byte to end byte of a line

$ cut -b 1- state.txt

Output:

Andhra Pradesh

Arunachal Pradesh

Assam

Bihar

Chhattisgarh

In this, -3 indicate from 1st byte to 3rd byte of a line

$ cut -b -3 state.txt

Output:

And

Aru

Ass

Bih

Chh

$ cut -c 2,5,7 state.txt

Output:

nr

rah

sm

ir

hti

$ cut -c 1-7 state.txt

Output:

Andhra

Arunach

Assam

Bihar

Chhatti

$ cut -c 1- state.txt

Output:

Andhra Pradesh

Arunachal Pradesh

Assam

Bihar

Chhattisgarh

$ cut -c -5 state.txt

Output:

Andhr

Aruna

Assam

Bihar

Chhat

Syntax: $cut -d "delimiter" -f (field number) file.txt

$ cut -f 1 state.txt

Output:

Andhra Pradesh

Arunachal Pradesh

Assam

Bihar

Chhattisgarh

$ cut -d " " -f 1 state.txt

Output:

Andhra

Arunachal

Assam

Bihar

Chhattisgarh

$ cut -d " " -f 1-4 state.txt

Output:

Andhra Pradesh

Arunachal Pradesh

Assam

Bihar

Chhattisgarh

$ cat state.txt | cut -d ' ' -f 1 | sort -r

Output:

Chhattisgarh

Bihar

Assam

Arunachal

Andhra

$cut –f 2 file1 → It displays second filed in file1.

$cut –f 2,4 file1 → It displays 2,4th fields in file1.

$cut –f 1-5 file1

It displays 1 to 5th fields in file1.

Let us say, each piece of information is separated by a "," then command would be

$cut –f 1-5 –d"," file2

It displays 1 to 5th fields in file12.

$cut –c 1-3,5-8 abc

c: character by character.

It displays 1-3 characters and 5-8 characters from file **abc.**

grep command:

Globally search a regular expression.

Syntax: grep "word-to-find" {file-name}.

$grep bangalore sample1

grep will locate all lines for the "**bangalore**" pattern and print all (matched) such line(s) on-screen.

Options

-c → it returns only number of matches.

-i → ignores case while searching.

-v → returns lines that do not match the test.

grep : This command is used to search specific string in specified file. By default, grep prints the matching lines.

#grep "I am" chaitanya.txt

#grep -i "chaitanya" chaitanya.txt : Ignore case distinctions in both the PATTERN.

#grep -c -i "krishna" krishna.txt : With –c option it will display how many lines matches the given pattern/string.

find : find command used to search and locate list of files and directories based on conditions you specify for files that match the arguments. Find can

be used in variety of conditions like you can find files by permissions, users, groups, file type, date, size and other possible criteria.

#find ~ -name chaitanya.sh: It will search the chaitanya.sh file under logged in user home
directory.

#find . -name chaitanya.sh: It will search the chaitanya.sh file under current directory.

#find . -iname chaitanya.sh: It will search the chaitanya.sh file under current directory ignoring the case.

find . -type f -perm 777: It will search all the files which have 777 permissions.

find . -perm /a=x: It will search all the files which have execute permissions.

find . -type f -empty: It will search all the empty files.

find . -type f –name ".*": It will search all the hidden files in current directory.

find / -name <<FIleName>> : It will display the all files which we provide the name.

ex.**# find / -name apachectl :** It will display the all the locations, where the file apachectl is available.

#find / -iname apachectl : It will display the all the locations, where the file apachectl is available. Here case insensitive.

#find . -mtime 1 : It will find all the files modified exact 1 day in current directory.

#find / -mtime 1 : It will find all the files modified exact 1 day in all directories.

#find . -mtime -1 : It will find all the files modified less than 1 day

#find . -mtime +1 : It will find all the files modified more than 1 day

#find / -mmin -10 : It will locate/display the files which modified less than 10 minutes
ago.

#find / -perm 600 : It will displays the files and directories which have the 600 permissions.–perm option is used to find files based upon permissions.

#find . –name "*.java" | xargs grep "This" : It will displays java files which have the This word in those files.

 # find . -name "*.java" -exec chmod 700 '{}' \; : It will set the access permissions (700) for all java files.

The argument '{}' inserts each found file into the chmod command line. The \; argument indicates the exec command line has ended.

OR

find . -name "*.java" | xargs chmod 700

#find . -size +1000c -exec ls -l {} \; : It will display the files which size greater than 1000bytes.

#find /Practice/Temp1/opt/logs/ -mmin +30 -exec rm -f {} \; : It will remove the files which are created more than 30 minutes back.

What is the difference between locate and find?

Ans) The find command has several options and is very configurable. There are many ways to reduce the depth and breadth of your search and make it more efficient.

locate uses a previously built database, If database is not updated then locate command will not show the output. to sync the database it is must to execute updatedb command.

Help Commands

man : Displays manual entries online.(To get help of the commands.)

Syantax : man <<Type any command>>

e.g. man man

man who

To search any word in “man” page use “/” and string name.

“n” for checking next search and “N” for previous search.

To quit press “:q”.

info : Gives the information about any command.

Syantax : info <<Type any command>>

e.g.

#info man

#info who

help : It gives a short explanation about how to use the command and a list of available options.

Syantax : <<Type any command>> --help

e.g.

#ls --help

#passwd --help

The Gnome Help Browser is very user friendly as well. You can start it selecting

Applications->Help from the Gnome menu, by clicking the lifeguard icon on your desktop or by entering the command **gnome-help** in a terminal window. The system documentation and man pages are easily browsable with a plain interface.

You may want to try "info:info" in the *Location* address bar, and you will get a browsable info page about the **info** command. Similarly, "man:ls" will present you with the man page for the **ls** command.

By default, the system will run **updatedb** which takes a snapshot of the system files once a day, locate uses this snapshot to quickly report what files are where. However, recent file additions or removals (within 24 hours) are not recorded in the snapshot and are unknown to
locate.

#**updatedb** : Only root user can able to run.

---

scp : Secure copy (remote file copy program)

Source Server : dev.marathahalli.kc.com

Source Server Directory: /AHRefresh/data/lp3data/AHNEW/

Target Server : dev.addr.krpuram.com

Target Server Directory: /AHstagdata/

Ex:1 scp /AHRefresh/data/lp3data/AHNEW/Temp.zip
root@dev.addr.krpuram.com:/AHstagdata/

2. scp -r /AHRefresh/data/lp3data/AHNEW/
root@dev.addr.krpuram.com:/AHstagdata/

Syntax: scp SourceFilePath
DestinationOSUserName@DestinationMachineName:DestinationPath

fgrep Command:

It is almost similar to grep, but by using fgrep you can search for multiple patterns. But it doesn't allow you to use regular expressions.

$fgrep "string1
> string 2
> string 3" filename

groups : Displays group membership.

userdel : Removes a user account.

Syntax: userdel << user name >>

Examples:

#userdel Krishna

#userdel -r krishna ---> It will delete/remove the krishna account and also the krishna home directory, in this case (/home/krishna).

finger : Finger displays the users login name, real name, terminal name, shell and write status, idle time, login time, office location and office phone number.

#finger

#finger hari

#finger root

Below example shows from Linux machine

id : It is one more command which will show the user details such as his primary group and his secondary group. (Displays the system identifications of a specified user.)

Examples:

#id krish

#id chai

#id ram

#id root

#id

lid: Display user's groups or group's users. Only root user can execute.

chage : It is used to see user related **"threshold details"** such as user disable time etc.

The chage program requires a shadow password file to be available.

#chage -l

#chage chai

#chage -E never krishna : It will make user 's(krishna) account will not expire.

Example1) [root@centos01 ~]# **chage -l demouser**
Last password change : Dec 25, 2017
Password expires : never
Password inactive : never
Account expires : never
Minimum number of days between password change : 0
Maximum number of days between password change : 90
Number of days of warning before password expires : 7

Example2) [root@dev01 ~]# **chage -E 2019-05-12 demouser**
or
[root@dev01 ~]# **chage -E $(date -d +180days +%Y-%m-%d) demouser**
[root@dev01 ~]# chage -l demouser
Last password change : never
Password expires : never
Password inactive : never
Account expires : May 12, 2019
Minimum number of days between password change : 0
Maximum number of days between password change : 99999
Number of days of warning before password expires : 7
[root@dev01 ~]#

su : Switches the user between users and root. (Changes the user ID associated with a
session.)
#su Chaitanya
#whoami
#exit
#whoami

users: Displays a compact list of the users currently logged on the system.

e.g.

#users

ln : Make links between files.

Two types of links:

1) Hard Links

2) Symbolic Links or Sym link or Soft link

The inode contains a list of all the blocks in which a file is stored, the owner information for that file, permissions, and all other attributes that are set for the file. In a sense, you could say that a file really *is* the inode, and names are attached to these inodes to make it easier for humans to work with them.

Use the **debugfs** command start to display the contents of the inode that you want to examine. For example, in this case you would type **stat <InodeNumber>**.

Use the **quit** command to close the **debugfs** interface.

Hard Links:

In Linux all directories and files have inodes. Creating hard link is nothing but add a new name to the inode. To do this we can use the **ln** command.

There is no difference between hard link and the original file.

Both files have the same inode number

Both files have the same size/contents.

Both files have the same access permissions.

#ln Krish.txt Krish_ln_test.txt

If you change one file contents it will update another file also.

#ln Krish.txt /root/Testing/Krishna/Krish_ln_test.txt

Another example for Hard Links

Symbolic links:

The symbolic link and the original file have different inodes.

The size of the symbolic link is significantly different from the size of the real file.

The size of the symbolic link is the number of bytes in the name of the file it refers to, because no other information is available in the symbolic link.

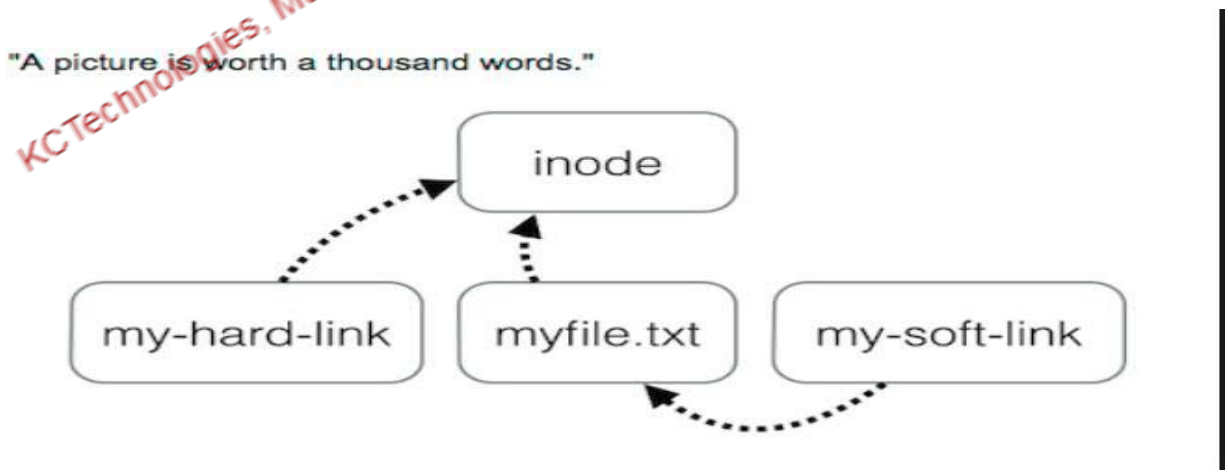
The file type of the symbolic link is set to 1, which indicates that it is a symbolic link.

What is the difference between Hard Link and Symbolic Link?

A symbolic link is like a shortcut. It points to the original file and helps you find it easily. It breaks if you remove the original file.

A hard ink is like a copy of the original file that is synchronized continuously. There is no difference between the original file and the hard link; they both refer to the same blocks.

We can create symbolic link for both files and directories, but we cannot create hard links for directories, can create only for files.



unlink filename: to remove the link between files.

Viewing files:

Text Editor Commands

vi or vim: Text editor. vim we have to install externally.

nano: Another Test editor.

Among all of these, vi and vim are mostly used.

nano : This is also one of the editor like vi/vim.

Nano is a modeless editor so you can start typing immediately to insert text.

nano chaitanya.txt

Ctrl + O : For saving

Ctrl + X : Quit

#id -un :

file: Determine file type.

So far, we have used the cat command to view the contents of a file. However, if the file is large in size then the matter would naturally scroll off the screen. To overcome scroll off the screen **head** and **tail** commands help in viewing lines at the beginning or end of the file.

head: Head prints the first N number of data lines of the given input. By default, it prints first 10 lines of each given file.

Syntax: head –n filename

$head -20 file1 → it displays first 20 lines from file1

head : Print the first lines of each FILE to standard output.

Syntax: head [OPTION]... [FILE]...

This command is very useful when we want to see some lines in big file.

Options are

1) –n: output the last N lines, instead of the last 10.

e.g. 1) **#head krishna.txt** : It will display the first 10 rows for specified file.

Note: By default, it will display 10 rows if don't provide any options for head command.

2) **#head -20 krishna.txt** : It will display the first 20 rows for specified file.

(OR)

#head -n 20 krishna.txt

Note: If you specify the more lines than the available in file, it will display the max lines in the file. See in the above example we specified the 20 lines in the command but only 15 lines there in the krishna.txt file.

tail: Tail prints the last N number of lines from given input. By default, it prints last 10 lines of each given file.

Syntax: tail -5 filename

Command: tail -5 file1 → it displays last 5 lines from file1.

tail : Print the last lines of each FILE to standard output.

Syntax: tail [OPTION]... [FILE]...

Options are

1) –f : output appended data as the file grows.

2) –n : output the last N lines, instead of the last 10.

e.g.

#tail krishna.txt ---> It will display the last 10 rows.

#tail -f krishna.txt : It will display last 10 lines of the growth of a file. The # prompt does not return even after the work is over. With this option we have to abort the process to exit to the shell. Use the interrupt key applicable on your machine. (Ctrl+c or q)

Note: By default, it will display 10 rows if don't provide any options for tail command.

#tail -n 3 krishna.txt : It will display the last 3 rows for specified file.

(OR)

#tail -3 krishna.txt

more : It is a filter for paging through text one screenful at a time (stop the display on each screen)

Syntax : more << file name >>

#more krishna.txt

less : less is the same as more except you can scroll back and forward.

Syntax : less << file name >>

#less krishna.txt

Following are options with less for navigation.

- CTRL+F – forward one window
- CTRL+B – backward one window
- CTRL+D – forward half window
- CTRL+U – backward half window
- G – go to the end of file
- g – go to the start of file
- q or ZZ – exit the less pager

env : Display environment, set environment for process. If you call **env** with no arguments, it displays the environment.

#env

How to set the class path?

Here we are setting the Java class path.

export PATH=$PATH:*/usr/java14/jre/bin*

echo $PATH

Process in UNIX:

Process is kind of program or task carried out by your PC.

"An instance of running command is called **process** and the number printed by shell is called **process-id (PID)**, this PID can be used to refer specific running process."

What is running right now:

$ps →To see currently running process at your terminal.

$ps –a → To see processes of all the users.

$ps username → To see the running processes of specific user.

#ps -f : Full listing of the user's currently running processes

#ps -ef: Full listing of all processes, except kernel processes

#ps -A (OR) ps -Af : All processes, including kernel processes.

Here f means full listing. You can observer output without f option and with f option.

 # ps ux :

(OR) It display all processes owned by a user.

[chaitanya@kctechdev01 root]$ ps U chaitanya : specific user.

 # ps aux : It will display every process on the system.

hostname: Show or set the systems host name.

hostname localhost: Change hostname.

#vi /etc/sysconfig/network

Restart the server as follows.

#service network restart

How to find the IP address?

Ans) We can find the IP address as follows.

ifconfig: The "ifconfig" (interface configurator) command is used to setup network interfaces and allow the user to view information about the configured network interfaces.

#ifconfig

 # hostname -i

Background processes:

To run command in background, you end it with an &.

Command: cp file1 file2 &

Killing a process:

Kill command is used to terminate the process or kill the process.

Syntax: kill pid

Here, pid is the process id.

#kill –l : It will display all signal names. These are found in /usr/include/linux/signal.sh

#kill -9 <<Process ID>> : It will kill the specified process.

#kill -3 <<JavaProcess ID>> : It will generate the core dump for Java processes.

#kill all -u <<username>> : It will kill all the processes under specified user.

jobs: Lists the jobs that you are running in the background and in the foreground.

How to find Linux Flavours and Version

krishna@kctechdev01:/> cat /etc/SuSE-release (Production Server)

krishna@ kctechdev01:~> cat /etc/issue

[root@phlox /]# cat /etc/redhat-release (CVS Server)

(OR)

 # lsb_release -a

#cat /etc/*release

uname : Print system information.

Syntax: uname [OPTION]...

DESCRIPTION

Print certain system information. With no OPTION, same as -s.

-a, --all

print all information, in the following order, except omit -p and -i if unknown:

-s, --kernel-name

print the kernel name

-n, --nodename

print the network node hostname

-r, --kernel-release

print the kernel release

-v, --kernel-version

print the kernel version

-m, --machine

print the machine hardware name

-p, --processor

print the processor type or "unknown"

-i, --hardware-platform

print the hardware platform or "unknown"

-o, --operating-system

print the operating system

--help display this help and exit

--version

output version information and exit

Below command also give the kernel version.

#cat /proc/version

To find Physical RAM(Linux)

#dmesg | grep mem

(OR)

#cat /proc/meminfo

To find Physical RAM(AIX)

Sed:

sed -n '/2018-12-12,/2018-12-12/p' /filepath → To search files with date range.

Replacing or substituting string

 # sed 's/DevOps/Java/' sedfile.txt

Here the 's' represents the substitution operation.

'/' is delimeter.

'DevOps' is search pattern.

'Java' is the substitute string.

Note: By default, the sed command replaces the first occurrence of the pattern in each line and it won't replace the second, third...occurrence in the line.

Replacing the nth occurrence of a pattern in a line.

Use the /1, /2 etc flags to replace the first, second occurrence of a pattern in a line. The below

command replaces the second occurrence of the word "unix" with "linux" in a line.

 # sed 's/DevOps/Java/1' sedfile.txt : It will replace the first occurrence.

 # sed 's/DevOps/Java/2' sedfile.txt: It will replace the second occurrence.

Replacing all the occurrence of the pattern in a line.

 # sed 's/DevOps/Java/g' sedfile.txt

'g' (global replacement is used to make the changes globally.

#sed -n "8p" chaitanya_friends.txt : It will display the particular line number.

#sed -n "8, 12p" chaitanya _friends.txt : It will display the particular line numbers.

To save the changes back to the file use the -i option:

http://www.folkstalk.com/2012/01/sed-command-in-unix-examples.html

expr : The expr command reads the Expression parameter, evaluates it, and writes the

result to standard output.

Syntax : expr << Argument 1 >> << Operator >> << Argument 2 >>

Examples :

#expr 2 + 2

#expr 2 - 2

#expr 2 * 2

#expr 2 / 2

#expr 2 = 2

#expr 2 = 3

#expr 2 \< 2

#expr 2 \< 3

#expr 2 \> 2

#expr 2 \> 1

#expr 2 \! 1

#expr 2 \! 2

#expr 2 != 2

#expr Krishna : Krishna---> Partial match and returns the number of characters matched.

#expr Krishna : Krishna

#expr Krishna : Krish

#expr Krishna : '.*' ---> Regular expression to match any number of characters

#expr Krishna: '....\(...\)' ---> To print the matched characters instead of number of matching positions.

df : Report file system disk space usage.

Syntax: df [OPTION]... [FILE]...

OPTIONS

Show information about the file system on which each FILE resides, or all file systems by default.

Mandatory arguments to long options are mandatory for short options too.

-a, --all

include dummy file systems

-B, --block-size=SIZE use SIZE-byte blocks

-h, --human-readable

print sizes in human readable format (e.g., 1K 234M 2G)

-H, --si

likewise, but use powers of 1000 not 1024

-i, --inodes

list inode information instead of block usage

-k like --block-size=1K

-l, --local

limit listing to local file systems

--no-sync

do not invoke sync before getting usage info (default)

-P, --portability

use the POSIX output format

--sync invoke sync before getting usage info

-t, --type=TYPE

limit listing to file systems of type TYPE

-T, --print-type

print file system type

-x, --exclude-type=TYPE

limit listing to file systems not of type TYPE

-v (ignored)

--help display this help and exit

--version

output version information and exit

SIZE may be (or may be an integer optionally followed by) one of following:

KB 1000, K

1024, MB 1000*1000, M 1024*1024, and so on for G, T, P, E, Z, Y.

#df –h

#df -a

du: Estimate file space usage.

Syntax: du [OPTION]... [FILE]...

du [OPTION]... --files0-from=F

DESCRIPTION

Summarize disk usage of each FILE, recursively for directories.

Mandatory arguments to long options are mandatory for short options too.

-a, --all

write counts for all files, not just directories

--apparent-size

print apparent sizes, rather than disk usage; although the apparent size is

usually smaller, it may be larger due to holes in (âsparseâ) files,
internal fragmentation, indirect blocks, and the like
-B, --block-size=SIZE use SIZE-byte blocks
-b, --bytes
equivalent to â--apparent-size --block-size=1â
-c, --total
produce a grand total
-D, --dereference-args
dereference FILEs that are symbolic links
--files0-from=F
summarize disk usage of the NUL-terminated file names specified in file F
-H like --si, but also evokes a warning; will soon change to be equivalent to
–
dereference-args (-D)
-h, --human-readable
print sizes in human readable format (e.g., 1K 234M 2G)
--si like -h, but use powers of 1000 not 1024
-k like --block-size=1K
-l, --count-links
count sizes many times if hard linked
-m like --block-size=1M
-L, --dereference
dereference all symbolic links
-P, --no-dereference
donât follow any symbolic links (this is the default)
-0, --null
end each output line with 0 byte rather than newline
-S, --separate-dirs
do not include size of subdirectories

-s, --summarize

display only a total for each argument

-x, --one-file-system

skip directories on different file systems

-X FILE, --exclude-from=FILE

Exclude files that match any pattern in FILE.

--exclude=PATTERN Exclude files that match PATTERN.

--max-depth=N

print the total for a directory (or file, with --all) only if it is N or fewer levels below the command line argument; --max-depth=0 is the same as --summarize

--time show time of the last modification of any file in the directory, or any of its

subdirectories

--time=WORD

show time as WORD instead of modification time: atime, access, use, ctime or

status

--time-style=STYLE show times using style STYLE:

full-iso, long-iso, iso, +FORMAT FORMAT is interpreted like âdateâ

--help display this help and exit

--version

output version information and exit.

#du –h

Note: If no path is mentioned, default is current directory, where the action will be

performed.

last: Show listing of last logged in users.

#last

#cat /etc/shells : It will give all available shells in your system

#echo $SHELL : It will give you the current shell type.

#chsh: Another way to change shell.

How to change the Shell type?

```
[root@kctech01 ~]# echo $0
bash
[root@kctech01 ~]# sh
sh-3.2# bash
[root@kctech01 ~]# echo $0
bash
[root@kctech01 ~]# tcsh
[root@kctech01 ~]# echo $0
tcsh
[root@kctech01 ~]# csh
[root@kctech01 ~]# echo $0
csh
[root@kctech01 ~]# ksh
# echo $0
ksh
# bash
[root@kctech01 ~]#
```


Note: If you want to come out from shell type 'exit'

netstat : Netstat prints information about the Linux networking subsystem.

Record your session

script : This command records your login session in a typescript (file) in the current directory. All the commands, their output and the error messages are stored in the file (typescript --> ASCII text file) for later viewing.

e.g.

 # script

script

Script started, file is typescript

 # ls

ls

anaconda-ks.cfg isus

Desktop plglogs pluginGSKitUpgradeLog.txt

ihsGSKitUpgradeLog.txt Read_Me_IC2_Default.properties

ihslogs typescript

install.log updilogs

install.log.syslog waslogs

 # pwd

pwd

/root

 # whoami

whoami

root

 # exit

exit

exit

Script done, file is typescript

echo : Display a line of text.

#echo Hi I am Krishna Chaitanya, working in MNC, Bangalore.

Hi I am Krishna Chaitanya, working in MNC, Bangalore.

#echo I am DevOps engineer and Linux admin.

I am Java DevOps engineer and Linux admin. **---> here it will not keep the white spaces**

[root@kctechdev01 ~]# echo " I am DevOps engineer and Linux admin "

I am DevOps engineer and Linux admin **---> here it will keep the white spaces**

#echo $PATH

/usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

:/root/bin:/root/bin:/root/Deployment_Ant/apache-ant-1.8.4/bin

[root@kctechdev01 ~]# echo "*"

*

[root@kctechdev01 ~]# echo '*'

*

 # vi ~/.bash_profile

HISTSIZE=500

HISTFILESIZE=500

By default, history is stored in ~/.bash_history file.

If we want to open the **bash_history file** use the following command.

 # vi ~/.bash_history

Eliminate the continuous repeated entry from history using HISTCONTROL

In the following example **date** was typed three times, when you do history, you can see all the 3 continuous occurrences of it. To eliminate duplicates, set HISTCONTROL to ignoredups as shown below.

 # date

Thu Apr 25 06:09:12 EDT 2013

 # date

Thu Apr 25 06:09:14 EDT 2013

 # date

Thu Apr 25 06:09:16 EDT 2013

 # history | tail -10

1 date

2 date

3 date

4 history | tail -10

Note that there are three date commands in history, after executing date 3 times as

shown above

 # export HISTCONTROL=ignoredups

 # date

Thu Apr 25 06:14:22 EDT 2013

 # date

Thu Apr 25 06:14:23 EDT 2013

 # date

Thu Apr 25 06:14:25 EDT 2013

 # history | tail -10

1 date

2 date

3 date

4 history | tail -10

5 export HISTCONTROL=ignoredups

6 date

7 history | tail -10

Note that there is only one date command in the history, even after executing date 3 times as shown above

Erase duplicates across the whole history using HISTCONTROL

The ignoredups shown above removes duplicates only if they are consecutive commands.

To eliminate duplicates across the whole history, set the HISTCONTROL to erasedups as
shown below.

 # export HISTCONTROL=erasedups

history -c : To clear the history.

Ignore specific commands from the history using HISTIGNORE

Sometimes you may not want to clutter your history with basic commands such as pwd and date. Use HISTIGNORE to specify all the commands that you want to ignore from the history. Please note that adding ls to the HISTIGNORE ignores only ls and not ls -l. So, you have to provide the exact command that you would like to ignore from the history.

 # export HISTIGNORE="pwd:date:whoami:ls:"

reboot : To reboot the system.

#reboot

reboot -f : To reboot the system forcibly.

poweroff (OR) halt –p (OR) init 0 (OR) shutdown -h now : To shutdown the system.

Once we execute the **init 0** command we will get the below error. Means server got shutdown.

mail : Sends and receives mail.

#mail myemail@gmail.com

Once you enter the to address It will ask Subject then Press Enter Body of mail then type Ctrl+D it will ask CC address, then enter CC address and press Enter button.

#mail : It is used to read the mails.

#mail -s "Hello" root < krishna.txt

Here Hello is the Subject and contents in krishna.txt is the subject for mail.

#echo "This is for body of mail" | mail -c myccemail@gmail.com -s "Sending mail using mail command" chaitanya@gmail.com -b krishna@gmail.com

Here options are:

-s subject Use subject for the e-mail title

-c address Send copy of the mail to the specified address

-b address Send blind carbon copy to specified address

mutt: This command is used to send a mail along with attachment.

#mutt -s "Airport Photo" -a KCTech.jpg krish@gmail.com -c chai@gmail.com -b kc@gmail.com \ < krishna.txt

mutt always reads a text from standard input that will become the main body of the e-mail, the text before the attachment. If the mail should consist of attachments only, we can either specify /dev/null as the file to read, e.g.

#mutt -s "KCTEch Photo" -a KCTech.jpg chaitanya@gmail.com -c chai@gmail.com -b kc@gmail.com \ < /dev/null

or use an empty line as the mail body:

#echo | mutt -a KCTech.jpg krish@gmail.com -c kc@gmail.com -b chai@gmail.com

wget: The non-interactive network downloader.

Following example will download and store the file with name: **apache-ant-1.8.4-src.tar.gz**

e.g. # wget http://www.apache.org/dist/ant/source/apache-ant-1.8.4-src.tar.gz

#tree: List contents of directories in a tree-like format.

top: The top program provides a dynamic real-time view of a running system. It can display system summary information as well as a list of tasks currently being managed by the Linux kernel. The types of system summary information shown and the types, order and size of information dis-played for tasks are all user configurable and that configuration can be made persistent across restarts.

 # top

Press (**Shift+O**) to Sort field via field letter, for example press `**a**` letter to sort process with PID (**Process ID**).

Press `**z**` option in running top command will display running process in color which may help you to identified running process easily.

#top -u krishna : It will display specific user process details.

Press `**c**` option in running top command, it will display absolute path of running process.

By default screen refresh interval is **3.0** seconds, same can be change pressing `**d**` option in running top command and change it as desired as shown below.

You can kill a process after finding **PID** of process by pressing `**k**` option in running top command without exiting from top window as shown below.

Press (**Shift+P**) to sort processes as per **CPU** utilization. See screenshot below.

Press (**Shift+W**) to save the running top command results under **/root/.toprc**.

Top output keep refreshing until you press `**q**`. With below command top command will automatically exit after 10 number of repetition.

 # top -n 10

 # cat /proc/meminfo

#free: To find the amount of free and used RAM memory in the system

#fdisk -l /dev/xvda: It will display all partitions of specific hard disk

rsync: rsync stands for remote sync.

rsync utility is used to synchronize the files and directories from one location to another in an effective way. Backup location could be on local server or on remote server.

Syntax : rsync options source destination

Examples:

#rsync -azvr /home/kctech/scripts/Java/Source/ /home/kctech/scripts/Java/Target/

This command will sync the Source and Target directories.

#rsync -azvr /home/kctech/scripts/Java/Source/ root@kctechdev01.dvp.kc.com:/opt/

This command will sync the file from local server to remote server for specified directory.

 # rsync -azvr root@kctechdev01.dvp.kc.com:/opt/FTPReadXML.java /home/isgproject/ scripts/Java/Target/

This command will sync the file from remote server to local server for specified file.

File Archieve/Extraction Commands

zip : zip, zipcloak, zipnote, zipsplit - package and compress (archive) files.

Syntax : zip -9 -r desiredname.zip zipneedfile

or

zip -rm desiredname.zip zipneedfile

e.g.: zip -9 -r Krishna.zip Krishna

zip -rm Krishna.zip Krishna

Syntax: zip tardest filename

Ex: **zip zipfile.zip myfile.txt**

unzip : Unzip - list, test and extract compressed files in a ZIP archive.

Syntax : Unzip foldername.zip

e.g. : unzip Krishna.zip

tar: It is used to archive the directory/file.

Syntax: tar options directory/file

The Options may be follows.

This will keep source file

This will not keep source file

- c – create a new archive
- v – verbosely list files which are processed.
- f – following is the archive file name

#tar cvf /home/isgproject/scripts/Java/test.tar Target/: Creating an uncompressed tar archive using option cvf

#tar cvzf test.tar.gz /home/isgproject/scripts/Java/Target/: Creating a tar gzipped archive using option cvzf

z – filter the archive through gzip

Note: .tgz is same as .tar.gz

#tar xvf test.tar : Extract a *.tar file using option xvf.

Here x means extract files from archive.

#tar xvfz test.tar.gz : Extract a gzipped tar archive (*.tar.gz) using option xvzf.

#tar tvf test.tar : View the tar archive file content without extracting using option tvf

#tar tvf test.tar.gz : View the *.tar.gz file content without extracting using option tvzf

#tar rvf test.tar Source/ : Adding a file or directory to an existing archive using option –r

Note: You cannot add file or directory to a compressed archive. If you try to do so, you will get "tar: Cannot update compressed archives" error as shown below.

tar rvfz test.tar.gz Source/

crontab : Crontab is the program used to install, deinstall or list the tables used to drive the cron daemon in Linux Operating system.

Crontab file path: /var/spool/**cron**/**crontabs**

Crontab format:

Minute Hour Day of Month Month Day of Week Command

(0-59) (0-23) (1-31) (1-12 or Jan-Dec) (0-6 or Sun-Sat) /usr/bin/find

Listing a crontab job

#crontab –l

Editing a crontab job

crontab -e

Removing a crontab job

#crontab –r

Example:

*/1 * * * * /home/krishna/devops/lscmd.sh >> /home/krishna/lscmd.log 2>&1

This will execute lscmd file every one minute.

To restrict any user from using cron job facility, enter their names in /etc/cron.deny and save it

#vim /etc/cron.deny

Now login as one of those users and try to use crontab.

If again want to allow krishna1 user to use cron job facilities just remove krishna1 user from /etc/cron.deny file.

Assuming that we have 100 users in our system, putting all 98 names in /etc/cron.deny file is a time consuming process. Instead of that, we can

create one more file /etc/cron.allow, in which we can assign names of those users who are allowed to use cron jobs. Remove the /etc/cron.deny file and create /etc/cron.allow, still if both files are existing cron.allow file will be having precedence over cron.deny file. Just to avoid confusion it is good to remove cron.deny file

Note: /etc/cron.deny file exists by default, but we need to create /cron.allow file. If your name is not there in cron.allow file then you will not be allowed to use cron jobs, and as mentioned above, if both files are existing cron.allow file will be having precedence over cron.deny file. If neither cron.deny nor cron.allow files exists then only root can use cron jobs.

service : This command is used to find the service is running in the server or not.

Syntax: << command >> << options >>

Options are:

status:

stop

start

restart

status-all

Examples:

 # service sshd status: It will display the status of the specified service.

 # service --status-all : It will display the status of all the services.

 # service sendmail stop : It will stop the specified service.

 # service sendmail start : It will start the specified service.

 # service sendmail restart : It will stop and restart the service.

#service crond status : It will check the status of cron is running or not?

Note: All services are available in /etc/rc.d/init.d directory.

test : This command is used to see if an expression is true, and if it is true it return zero(0), otherwise returns nonzero for false.
Syntax : test << EXPRESSION >> << Options >>

ping: It is a network tool that tests of whether a particular host is up and reachable on the network.
The default behavior of ping is to send a 64 byte ICMP (Internet Control Messaging Protocol) packet to the specified host every second until you cancel the operation with a Ctrl+c. When the ping operation is cancelled or done, ping will report summary statistics such as average pocket loss, number of pockets sent/received, etc.

diff: **$diff file1 file2** ->It displays line by line difference between file1 and file2.

Split: **split –bytes=1G file**: This will split the files into different files with 1GB of memory each.

Q) Generally, each block in UNIX is how many bytes?
Ans: Generally, each block in UNIX is of 1024 bytes.

Q) By default, a new file and a new directory which is being created will have how many links?
Ans: New file contains one link. And a new directory contains two links.

Q) What is a file system?

Ans: The file system is a collection of files which contain related information of the files.

Q) What are the different blocks of a file system? Explain in brief.

Ans: Given below are the main 4 different blocks available on a file system.

File System Block No.	Name of the Block
1st Block	Boot Block
2nd Block	Super Block
3rd Block	Inode Table
4th Block	Data Block

Super Block: This block mainly tells about a state of the file system like how big it is, maximum how many files can be accommodated etc.

Boot Block: This represents the beginning of a file system. It contains bootstrap loader program, which gets executed when we boot the host machine.

Inode Table: As we know all the entities in a UNIX are treated as files. So, the information related to these files are stored in an Inode table.

Data Block: This block contains the actual file contents

Q) What is the alternative command available to echo and what does it do?

Ans: tput is an alternative command to echo.

Using this, we can control the way in which the output is displayed on the screen.

Q) What is IFS?

Ans: IFS stands for Internal Field Separator. And it is one of the system variables. By default, its value is space, tab, and a new line.

It signifies that in a line where one field or word ends and another begins.

Q) Which command needs to be used to know how long the system has been running?

Ans: uptime command needs to be used to know how long the system has been running.

Example: $ uptime

Upon entering the above command at shell prompt i.e. $ uptime, the output should look something like this.

9:21am up 86 day(s), 11:46, 3 users, load average: 2.24, 2.18, 2.16

Execution over Shell Interpreter/Editor

Output:

Q) What are the different communication commands available in Unix/shell?

Ans: Basically, there are 4 different communication commands available in Unix/shell. And they are mail, news, wall & motd.

Q) How to find out the total disk space used by a specific user, say for example username is John?

Ans: The total disk space used by John can be found out as shown below.

du –s/home/John

Q) How to debug the problems encountered in shell script/program?

Ans: Though generally it depends on the type of problem encountered. Given below are some common methods used to debug the problems in the script.

- Debug statements can be inserted in the shell script to output/display the information which helps to identify the problem.
- Using “set -x” we can enable debugging in the script.

Q) How to open a read-only file in Unix/shell?

Ans: Read-only file can be opened as shown below:

vi –R <File Name>

Q) How can the contents of a file inside jar be read without extracting in a shell script?

Ans: The contents of the file inside a jar can be read without extracting in a shell script as shown below.

tar –tvf <File Name>.tar

Q) What is the difference between diff and cmp commands?

Ans: diff – Basically, it tells about the changes which need to be made to make files identical.

cmp – Basically it compares two files byte by byte and displays the very first mismatch.

Q) Explain in brief about awk command with an example.

Ans: awk is a data manipulation utility or command. Hence, it is used for data manipulation.

Syntax: awk options File Name

Example:

Script/Code

awk utility/command assigns variables like this.

$0 -> For whole line (e.g. Hello John)

$1 -> For the first field i.e. Hello

$2 -> For the second field

Execution over Shell Interpreter/Editor

The above script prints all the 5 lines completely.

Output:

Execution over Shell Interpreter/Editor

The above script prints only first word i.e. Hello from each line.

Output:

Q) How will you print the login names of all users on a system?

/etc/shadow file has all the users listed.

awk –F ':' '{print $1} /etc/shadow|uniq -u

Q) How to set an array in Linux?

Syntax in ksh:

Set –A arrayname= (element1 element2 ….. element)

In bash

A=(element1 element2 element3 …. elementn)

Q) How can I set the default rwx permission to all users on every file which is created in the current shell?

We can use:

umask 777

This will set default rwx permission for every file which is created for every user.

Q) How can we find the process name from its process id?

We can use "ps –p ProcessId"

Q) How can I send a mail with a compressed file as an attachment?

zip file1.zip file1|mailx –s "subject" Recipients email id

Email content

EOF

Q) How do we create command aliases in a shell?

alias Aliasname="Command whose alias is to be created".

Q) What are "c" and "b" permission fields of a file?

"c " and "b" permission fields are generally associated with a device file. It specifies whether a file is a special character file or a block special file.

https://www.guru99.com/shell-scripting-interview-questions.html