**What is Linux**

* **Linux is Operating system.**
* **OS is system software that directly runs on physical machines.**

Linux Features

* Most widely used operating system for servers.
* Open source Operating system or Free os.
* 32/64 Bit support
* Multiuser, Multi task
* Can Support Multiple Hardware platforms
* Source code os open
* OS Customization is Possible
* X Windows is Gui

Windows VS Linux

|  |  |
| --- | --- |
| Windows | Linux |
| License based | Free |
| Secured | More secured |
| Downtime required | No Downtime required |
| Virus effect | No Virus effect |
| Preferred for desktops | Preferred for servers |
| Not open | Source code is Open |
| No OS Customization | Os Customization |

**Linux Distributions**

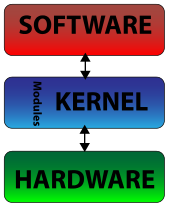
• Distribution is a set of packages that build OS

• Various Distributions are available

* Redhat
* Centos
* Debian
* Mandriva
* Slackware
* SuSE
* Caldera
* Ubuntu etc..
* Fedora

**KERNEL**

* Core component of Operating System
* Interface between Software and Hardware
* Responsible to manage system resources (CPU, Memory Etc..)

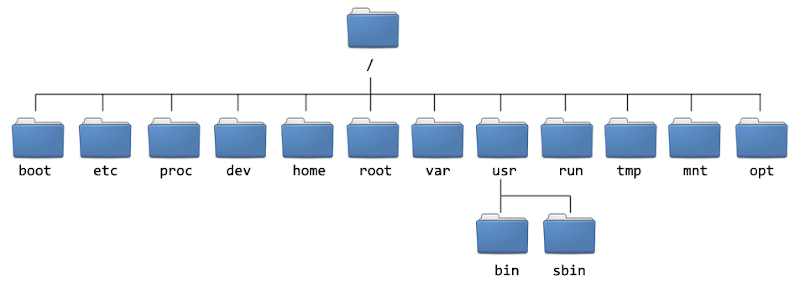


**Shell**

* Shell is a command interpreter that execute commands. Read from the standard i/p device (keyboard)
* Responsible for finding commands and execution
* Several Shells are available, Bash is popular.

**Linux File system**

* Linux has Hierarchical File System Structure
* All directories, files, devices etc reside under “ / “
* Linux OS creates some default directories under /.



**/bin:** This directory holds the commands. Those commands used by all users.

**/sbin:** This directory holds the commands. These commands using by Super user/root/

- It contains commands. Which can be by super user.

**/boot:** It contains programs which are used to boot the system.

**/proc:** It contains the system processors information.

**/usr:**( unix system resources)

- It contains system libraries and Man Pages

**/var:**( Variable )

- It contains system logs (Messages)

**/lib:** (Libraries)

- It contains system libraries which accessed dynamically.

**/etc:** It contains the configuration files

**/opt:** (Optional) Here We can store the third party software.

**/dev:** The directory contains the special device files. The device files are created during installation

**/home:** It can used as Home directory for Normal Users.

**/root:** Home Directory for Root.

**/mnt:** It can be used as Mount point for the removable storages.

Basic Commands

**logname:** It displays logged-in user name

# logname

- O/P:



**Whoami:**

Who executing the commands or operating

**pwd:** It displays current working directory

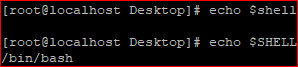
# pwd

- O/P :



**echo $SHELL:**It contains default shell of system. (in shell default shell is bash)

- O/P:



**echo $0** : It contains current shell of system.(we have different shells like cshell, kshell so it will display the current shell name)

- O/P :



I.Q :Diffrence between echo $SHELL and echo $0

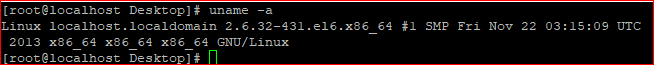
**Clear: To** clear the screen.(It will only clear the screen what your are able to see in your screen, but history will save in top of the screen)

O/P:



**uname -a:** It displays the system information.

O/P :



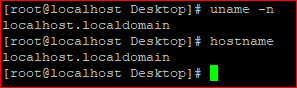
**uname:** It display only "OS" in the system or Machine.

- O/P :



**uname -n or hostname:**  It display hostname of system

O/P:



**uname -i:** To Display the machine architecture.

O/P:



**who -b:** It display last booted time.

O/P:



**uptime:** It display the system running time.

- O/P :



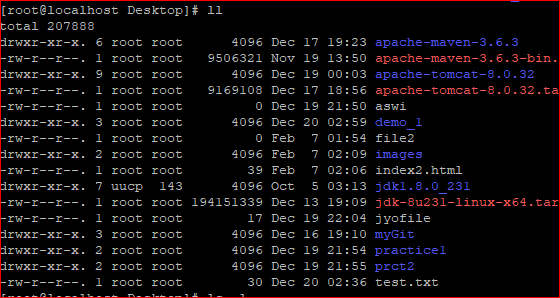
**ls:** List of files and directories in current directory

O/P:



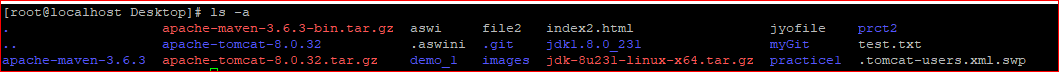
**llor ls -l:** Long List of files and directories in current directory

O/P:



**ls -a:** It will display the all files in pwd ( Including hidden files)

O/P:

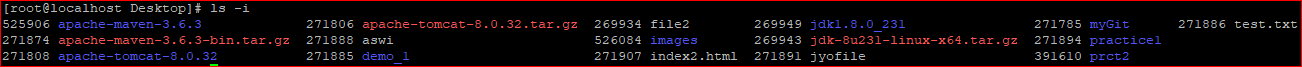


**ls -r :** Display the files in reverse order

O/P:



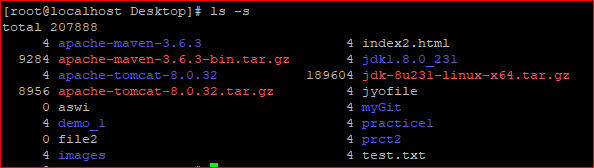
**ls -i** : It will display the inode numbers of files or dirs.

O/P:

- inode is an entry in inode table. It contains the information like meta data about regular file &dirs

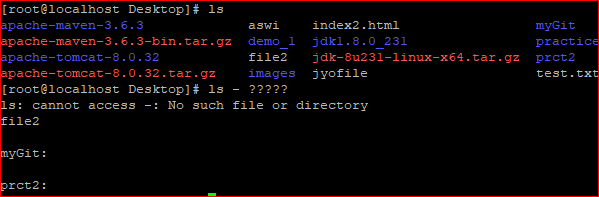
ls -s : To display the files sizes

O/P:



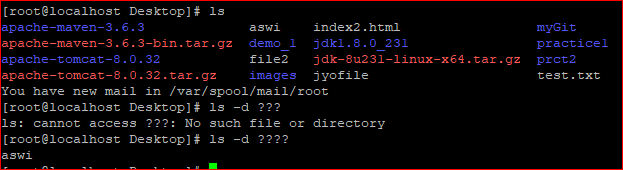
ls - ????: Display the file name only four characters in name

O/P:



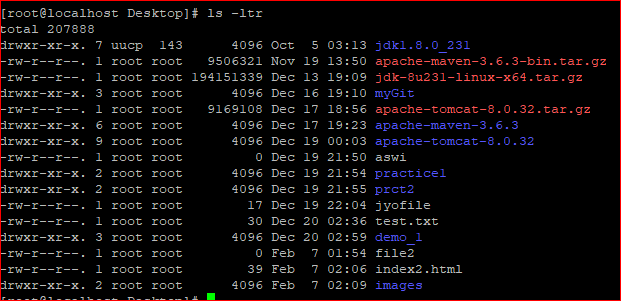
ls -d ?????: Display the dir name only four characters in name

O/P:



ls –ltr: IT Will display the list of files in long format along with time and reverse format.

O/P:



**Working with files**

In linux we have different categories of files

i. Regular files

ii. Directories

* **File type:** regular file, directory, pipe etc.
* **Permissions to that file:** read, write, execute
* **Size of file**
* **Time stamp**

# stat<filename>

**You can display the inode data on a file or directory by using stat command**

**Cat (**concatenate)

By using cat we can create files. We can append the data to existing files and we can list the content of the file.

Create a filename with devops

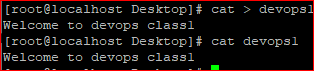
# cat > devops1 (File created)

Welcome to DevOps

# Ctrl D (Save the file)

Now View the content of the file

# cat devops1 or cat < devops1



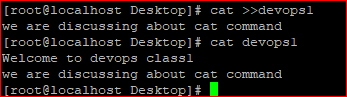
Now Append the data to existing file

# cat >> devops1

We are discussing about cat command

# Ctrl d

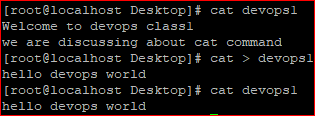
O/P:



**Note**: The data to be added at the end of the file

**NOTE** :if file is already existed cat command override it

FOR EX:



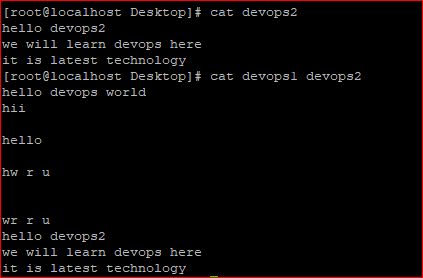
To Display the data through line number

# cat -n devops1



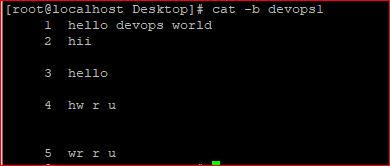
Shows the content of the files at a time

# cat file1 file2



To skip the empty lines in file

# cat -b myfile



NOTE: IF YOU WANT TO KNOW MORE COMMANDS ABOUT CAT RUN THE BELOW COMMAND

# man cat

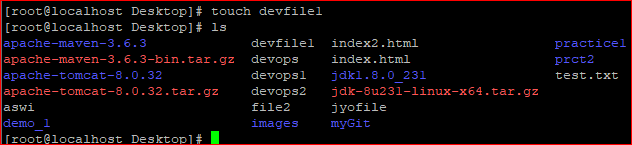
**Same like for lscommand also:**

**#man ls**

**Touch**

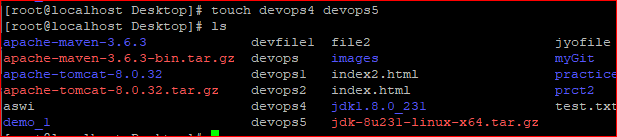
Touch command is used to create empty files or create one or more files at time

# touch devfile1



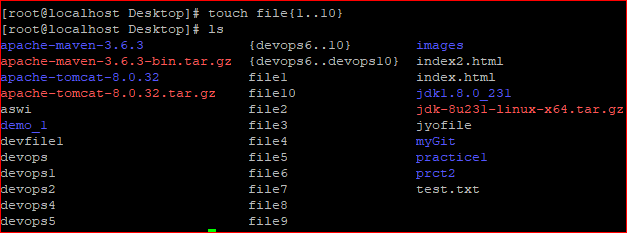
or

# touch myfile1 myfile2



To create files in the range

# touch {1..10} or touch {a..z}



**Note:**

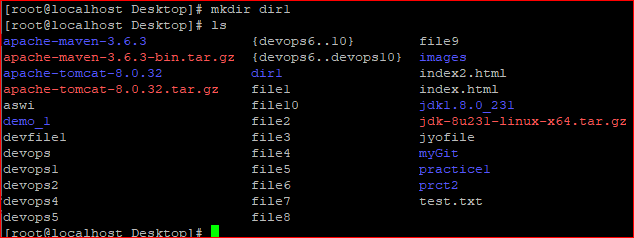
If file is created already then it is not override the existing file. It only changes the time stamp of the existing file.

I.Q: DIFFRENCE BETWEEN CAT AND TOUCH COMMAND?

Day-2

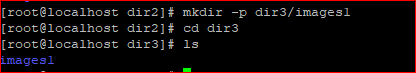
**Creating directory**

# mkdir dir1



command to create parent directory and sub directory at a time

# mkdir -p dir3/images1



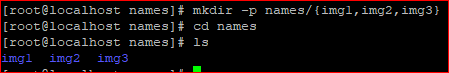
To create multiple directories at a time

# mkdir img3 img4



If you want to create several directories under parent directory

# mkdir –p names/ {img1,img2,img3} (with out space)



**Copy:**

cp stands for copy. *cp* is a Linux shell command to *copy* files and directories.

**Syntax:**

**cp [OPTION] Source Destination**

**#** cp file1 file2

Here copying the file1 to file2

To copy the directories

# cp -r dir1 dir2

**Move**

- Move command is used to move the files and directories

- It rename a file or folder

syntax:

# mv [*options*] *source* *dest*

# mv file1 file2 - Moving the file

# mv dir1 dir2 - Moving the directories

**rm**

Removing the files and directories

# rm file or directory ( It will ask you permission)

# rm -rf file or directory ( Forcefully delete the files without asking)

# rmdir empty\_directory\_name ( It will delete the empty directory )

**cd**

Change directory from one location to another location

# cd /etc

cd - ( change to previous directory )

File permissions:

When we create a file or directory it is having default permissions. The permissions will applied on three attributes.

i. Owner

ii. Group

iii. Others

Permission modes are three types

1. read (r=4)

ii. write (w=2)

iii. execute (x=1)

The default permission of file "644" and directory "755".

Here "umask" value determine the default permission on a file (or) directory.

Note: The default umask value is "022"

To checking the umask value

# umask

To change permission of file or directory use the command "chmod"

- Only owner (or) root user can change the file permission

- The permissions can be applied is two ways

i. Absolute mode (Numeric values)

ii. Symbolic Mode (Absolute Mode)

Symbolic Absolute

r 4

w 2

x 1

u - owner or user

g - group

o - others

a - all

+ - Add permissions

­­**-** - Remove permissions

= Assign permissions

Examples:

#chmod 644 filename

Set the permissions of **file** to "owner can read and write; group can read only; others can read only".

#chmod 666 file

Set the full permissions for all users

# chmod a=rw filename (Absolute Mode)

Set the full permissions for all users

# chmod u-r file

Removing the write permission owner/user

# chmod -R 777 dir1

Recursively (**-R**) Change the permissions of the directory **myfiles**, and all folders and files it contains, to mode **777**: User can read, write, and execute; group members and other users also can read, execute and write.

**pipe (|) :**

To combine two or more commands with pipe command

Pipe command will take command1 output is take to command2 input

# ls -l | head

**Wc**

wc stands for word count.

# wc [options] filenames

wc -l : Prints the number of lines in a file.

wc -w : prints the number of words in a file.

wc -c : Displays the count of bytes in a file

Example:

# ls -l | wc -l

It will display number of files and folder in present working directory

# wc -l filename

It will display number of lines in a given file (filename)

**Head**

The head command outputs the first part (the head) of a file or files.

**Note:**

head, by default, prints the first 10 lines of each FILE to standard output.

# cat file | head

**Tail**

Tail is a command which prints the last few number of lines (**10 lines by default**) of a certain file, then terminates.

# cat file | tail

Note: It give the dynamic changes at any time. When code is changing the automatically it gives last "10" lines

**Grep:**

Grep command in Unix/Linux is a powerful tool that searches for matching a regular expression against text in a file, multiple files or a stream of input. It searches for the pattern of text that you specify on the command line and prints output for you.

Examples:

# grep email nagi.txt

It will match the email pattern against file

or

# cat nagi.txt | grep email

# grep -i email nagi.txt (i-ignore case/Insensitive)

It will match the email pattern against file (It maybe Capital/Small letters)

# grep -i -n email greptest (i-ignore case)

Display the email lines with their line numbers

# grep -v email greptest

Print the lines excluding the pattern called email

# grep -vi email greptest (Ignore case)

Print the lines excluding the pattern called email with all cases of character

# grep -ino email greptest

Display the just word with line number

# grep –r linux /etc/

Search the pattern recursively using -r option

# grep [0-9] 1.txt

Display the lines, those lines are numbers contains with 0 to 9

**Anchors**

**^ -** Cap- Beginning of the line

**$-** Dollar- End of the line

# ls -l | grep ^-

Display the all files

# ls -l | grep ^- | wc -l

Display all the files with count

# ls -l | grep ^d

Display all the directories

# cat 1.txt | grep spam$

Display the lines end with spam

# cat 1.txt | grep ^Hello

Display the lines start with Hello

**Find**

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

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

**Example**

# find . -name data.txt

It find the data.txt in current location

# find . -i -name data.txt

* I = Ignore case

# find / -name test -type f

Search for file in entire file system based on condition

# find / -name testdata -type d

Search for directory in entire file system

# find . -type f -name "\*.php"

Display all the files end with ".php"

# find /tmp -type f -empty

Display the all empty files under /tmp directory

**Search by permissions**

# find . -type d -perm 777

Display all directories having the 777 permissions

# find . -type d ! -perm 777

Find directories Without 777 Permissions

**To search by size**

# find / -size +10M

Display the files and directories more than 10MB

# find / -size +50M -size -100M

To find all the files which are greater than 50MB and less than 100MB

# find . -cmin -60

To find all the files which are changed in last 1 hour.

# find . -mtime -15

Days wise

# find / -type f -name "find.txt" -exec rm -f {} \;

To find a single file called find.txt and remove it.

# find / -type f -name "nagi.txt" -print -exec rm -f {} \;

Print the deleted file

**Vi (Visual Editor)**

Vi is a text editor used to create or modify the files.

Vi has three modes

i. command mode

ii. insert mode

iii. Colon mode

**command mode**

we can execute commands to navigate curser and perform deletion, copy, and paste etc operations

**Insert Mode**

Here we can perform insert operations

**Colon Mode (:)**

To perform save, quit, search etc operations

**Note:** default mode is Command mode

**Moving within a file**

|  |  |
| --- | --- |
| **Keystroke** | **Use** |
| K | Move cursor up |
| J | Move cursor down |
| H | Move cursor left |
| L | Move cursor right |

**Saving**

* x or w = save
* q = quit
* q! = quit forcefully
* wq = save and quit
* wq! = save and quit forcefully
* s = search
* 1 = Means from first line
* $ = Means to last line
* g= globally ( If pattern is occurrence multiple times it will change all occurrence)
* se nu = set numbers

g/patter1/s//pattern2/g To Replace string word for Global level

|  |  |
| --- | --- |
| **Keystrokes** | **Action** |
| i | Insert at cursor **(goes into insert mode)** |
| A | Write after cursor **(goes into insert mode)** |
| a | Write at the end of line **(goes into insert mode)** |
| ESC | Terminate insert mode |
| u | Undo last change |
| O | Open a new line **(goes into insert mode)** |
| ndd 3dd | Delete line Delete 3 lines. |
| D | Delete contents of line after the cursor |
| dw 4dw | Delete word Delete 4 words |
| Cw | Change word |
| X | Delete character at the cursor |
| R | Replace character |
| R | Overwrite characters from cursor onward |
| G | Go to end of file |
| gg | Beginning of the file |
| $ | End of the curser present line |

**Day-6**

**User management**

User management includes everything from creating a user to deleting a user on your system. And also modify the users, locking users etc.

# useradd demo

To Create new demo user

# passwd demo

Adding password to demo user

# userdel demo

Deleted user demo

# cat /etc/passwd

To check the all the available users in file system

# cat /etc/shadow

To see the all the available user's password information in file system

# usermod -u (newuserid) username

To modify the userid for particular user

# usermod -l (newusername) (oldusername)

changing the user name

**Package Management**

Package is nothing but collections of files and directories grouped in defined structure or format.

Packages on different Platforms

In Windows- >.exe - It's a software (vlc.exe)

In Linux - .rpm - package (vlc.rpm)

Here in Linux we have two utilities to install packages in Linux Machine

1. rpm - RedHat Package Manager

2. yum - Yellow dog updater Modifier

q - query for packages

a - all packages

l - Display files list particular a packages

i - install

u - update

e - remove

h - Display the hashes # while installing the package

v - Verbose

--force - To install packages forcefully

-- nodeps - It doesn't check for dependencies before installing packages

Examples:

# rpm -qa

It shows the which packages are installed in the system

# rpm -qa | grep tree

Query for particular package

# rpm -i tree-1.5.3-3.el6.x86\_64.rpm

To install a tree package

# rpm -e tree

To uninstall/remove tree package

# rpm -iv tree-1.5.3-3.el6.x86\_64.rpm

It shows the tree packages installation process which progress bar(#)

**Yum**

Yum will install rpm packages along with their dependencies. To achieve this yum maintains packages repository

yum configuration directory /etc/yum.repo.d (Repositories list)

# yum update

To keep your system up-to-date with all security and binary package updates, run the following command. It will install all latest patches and security updates to your system

# yum install tree

To install package called tree

# yum install firefox

To install firefox

# yum install -y tree

It will install the tree package without asking permission

# yum remove tree

It will remove the package called tree

# yum remove -y tree

It will uninstall the tree package without asking permission

# yum update tree

Let’s say you have outdated version of**tree** package and you want to update it to the latest stable version

# yum info tree

To show the all information about Zsh

**Service management**

service is the command to manage services in linux

Syntax: service (servicename) (action means Status, start, stop, restart)

Status: To verify service status

start: To start service

stop: To stop the service

restart: To restart the services

Examples:

# service httpd status -checking the status of httpd service

# service httpd start - start the httpd service

# service httpd stop - stop the httpd service

# service httpd restart - restart the httpd service

**crontab**

cron is the utility to schedule the jobs which can run at specific intervals of time

# service crond status

To check the crond service

# crontab -l

It display the scheduled jobs (By default shows the root jobs)

# crontab -l username

It display the scheduled jobs for specific user

# crontab -e

To create jobs and edit jobs

# crontab -r

To remove the jobs

Note: crond all related files located under /etc/cron.d

# cat /etc/crontab

configuration file of cron

**Example of job definition**:

\* .---------------- minute (0 - 59)

\* | .------------- hour (0 - 23)

\* | | .---------- day of month (1 - 31)

\* | | | .------- month (1 - 12) OR jan,feb,mar,apr ...

\* | | | | .---- day of week (0 - 6) (Sunday=1 or 7) OR sun,mon,tue,wed,thu,fri,sat

\* | | | | |

Examples:

# \* \* \* \* \* cp/etc/passwd /root/Desktop/Linux/passwd.bkp

For every minute it will take the backup of passwd file

# 0 5,17 \* \* \* sh /opt/logs.sh

It will run the job twice a day

# \*/10 \* \* \* \* /scripts/monitor.sh

For every ten minutes it will run the job

# 0 17 \* \* sun,fri /scripts/script.sh

It will run job on selected days at 5 PM

# 0 \*/4 \* \* \* /opt/myjobs/backup.sh

It will run the job for every four hours

# @yearly /opt/myjobs/backup.sh

It will execute the job on the first minute of every year

# @daily /opt/myjobs/backup.sh - daily

# @hourly /opt/myjobs/backup.sh

Note: The location of schedule cron job files are located under /var/spool/cron

**SSH**(Secure shell)

ssh is utility to connect the remote machine

The ssh command provides a secure encrypted connection between two hosts over an insecure network.

The configuration file /etc/ssh/sshd\_config

The ssh port number is "22". With help of port number we can identify the service

# ssh <ip\_remote\_Machine>

To connect remote Machine

# ssh username@<ip\_remote\_Machine>

To connect remote machine with specific user

# service sshd status

Checking the sshd status

**Trust Relation**

Without give credentials (user name and password) login the remote machines

SSH trust between two servers so that the two servers share the same SSH keys and can log into each other.

To establish trust relation between two servers follow the below steps

Step 1: Generate the ssh key

# ssh-keygen -t rsa

Note: Two types of keys are available one is rsa other hand is dsa. rsa keys more secure because of encrypt mode. Dsa keys are less secure so any one can hack it.

For above command we get the two keys i.public key ii. private key

We can find the keys the home directory of user for example

location: /root/.ssh/ id\_rsa, id\_rsa.pub

Step:2

Copy the id\_rsa.pub key into remote machine the location /root/.ssh/authorized\_keys

Step3:

Restart the sshd service

Step4:

Now check the trust relation

sship\_Machine

**SCP**(Secure copy)

scp command is used to copy the files and directories from one machine to another machine (Either local or remote)

Synatx:

Scp <source\_path> <Destination\_path>

# scp /opt/mydata.txt 192.168.249.144:/root/Desktop/

Copy file from one machine to another machine

# scp filename 192.168.249.160:/root/Desktop/file /opt

Copy the file from remote to local machine

# scp -r /opt/dir1 192.168.249.144:/root/Desktop/

Copy the directory form local to remote machine

**Tar** (tape archive) To perform backups

c- to create backups

v- verbose

f- To specify backup filename

x- Extract

t- To display table of content

Note: Basically all softwares are in tar files

# tar cvf mydata.tar /root/Desktop/linux/data

To create a backup file with name of mydata.tar

# tar tvf mydata.tar

To view the content of the tar file

# tar xvf mydata.tar

To extract the tar file

**System monitoring**

**du**(Disk usage)

# du

To find out the disk usage summary of a current directory tree and each of its sub directories.

# du -h

To show the human readable like bytes, kilobytes, megabytes, gigabytes.

# du -sh

To get the summary of a grand total disk usage size of an directory use the option “-s”.

# du -h /opt

To find out the disk usage summary of opt directory tree and each of its sub directories.

# df -h

To display the space utilization of the system with human readable format

**TOP**

Display system performance (Memory and cpu utilization information)

-d = To specify delay time

-n = To specify count

-o = To sort the o/p by specifying field

-m = Sort the o/p memory usage

-p = To specify process id

Shift o - for shorting utilization data

Note: By default it will refresh every 3 seconds

# top -d5 -n10

For every 5 seconds it will refresh and show the info 10 times quit it

# top -u username

Display the specific user process details

Note:

Press c after fire the "top" command Then it will show absolute path of running process

z - Highlighting the running the process which may help you to identified running process easily.

k - we can kill the process after finding the process id

shift + p - It will sort the cpu utilization

**Process**

Whenever a command is issued in unix/linux, it creates/starts a new process. For example, pwd when issued which is used to list the current directory location the user is in, a process starts.

Process is nothing but entity of application. Every process running in the system will have a process ID. The first process start in the linux machine "init". Which will have the process id "1"

When you start a process (run a command), there are two ways you can run it −

* Foreground Processes
* Background Processes

it reads this information from the virtual files in [/proc in filesystem](https://www.tecmint.com/exploring-proc-file-system-in-linux/)

# ps

It is easy to see your own processes by running the **ps** (process status) command

# ps -f

which provides more information about your process

# ps -fU root (username)

To display a user’s processes by real user ID (RUID) or name, use the -U flag

# ps -p 23

It display the complete information about process id 23

**Kill**

kill command in Linux (located in /bin/kill), is a built-in command which is used to terminate processes manually. kill command sends a signal to a process which terminates the process.

# kill process id (Kill the process)

# kill -9 process id (Forcefully delete the process id including any dependency)

**SED-** Stream editor

It is mainly used for text substitution, find & replace but it can also perform other text manipulations like insertion, deletion, search etc..

Options:

-n = To suppress the entire file out put

-e = To specify the expression

q = quit after reading the lines

p = print the lines

d = delete the lines

i = insert the lines

a = append the lines

c = change the lines

s = search the pattern and to replace it

Eg: s/string1/string2/

It replaces the first occurrence of string1 with place of string2 in all lines

# s/string1/string2/g'

It replaces all the occurrence of string1 with place of string2

i - ignore case

^ - beginning of the lines

$ - End of the line

# sed -n '1,10p' filename

It Just display the lines from 1 to 10

# sed -n '50,$p' filename

It Just display the lines from 50 to last line of file

# sed -n -e '10,20p' -e '40,50p' filename

It will display the line numbers from 10 to 20 and 40 to50

# sed '40 50d' filename

It will delete lines from 40 to 50 in a file

Note: It delete temporarily means skip them

# sed '$=' filename

or

# sed -n '$=' filename

It will count the line numbers

# sed -n '/usr/p' filename

Display the line numbers with particular word

# sed -n '/usr/=' filename

Display the line numbers of word

# sed 's/usr/rsa/g' filename

Replace the word in place of usrrsa will appear

#sed 's/usr//g' anaconda-ks.cfg

It will delete the usrword globally in file

# sed 's/.$//g' anaconda-ks.cfg

It will delete the last character of every end of line.

# sed 's/[^ ]\*$//g' anaconda-ks.cfg

It will delete the last word of every line in file