Windows
Windows is an operating system (OS)
Windows supports only GUI
-GUI stands for graphical user interface

- To create a file in windows
  Right click
  New
  Text Document

- To create a folder in windows
  Right click
  New
  Folder

- To rename a file or folder in windows
  Right click on file & folder
  Rename

- To delete a file or folder in windows
  Right click on file & folder
  Delete

****

We cannot create multiple files & folders at a time in windows
We can delete multiple files & folders at a time in windows

****

Linux

- Linux is an operating system
- Linux supports both GUI & CLI
- GUI stands for graphical user interface &
- CLI stands for command line interface
- Linux is case sensitive
- Linux is space sensitive

****

Difference between windows & Linux

****

1. Windows supports only GUI &
2. Linux supports both GUI & CLI
3. In windows we cannot create multiple files & folders at a time
4. but In Linux we can Create multiple files & folders at a time

****

What is Linux

****

- Linux is an operating system, with the help of that users can communicate/interact with hardware components i.e computer.
- It was developed/created in 1960s by Linus Torvalds

Features of Linux:

1. Linux is free of cost
2. Linux is open source operating system
3. Linux is multi User operating System.

   3) Linux is multi tasking operating system.

   4) Linux supports both GUI & CLI
      - GUI means graphical user interface
      - CLI means command line interface

   5) Linux is more secure as compared to windows because of permissions to each file & directory

****

Difference between Linux & Windows(OR)Advantages of Linux

****

1. Linux is free of cost but windows is not free of cost. we have to purchase licence for windows operating system
2. Linux is open source operating system but windows is not open source operating system
3. Linux supports both GUI & CLI but windows supports only GUI
4. Linux is more secure as compare to windows

****

Flavors of Linux

****

- RedHat
- Ubuntu
- Centos
- Fedora
- kali
- mint
- Slackware
- open salaries
- Open Suse
- Suse Linux Enterprise server (SLES)

****

Components of Linux

****

1. **1. shell**
2. **2. kernel**

1. shell

- shell is an interface between user & kernel.
- shell receives our command and check whether our command is properly use or not
- If our command is incorrect then shell will show error like command not found
- if everything is proper, then shell converts our command into kernal understandable form and handover to the kernel

Types of shells

1. bourne shell
   Bourne shell is the 1st shell in linux

2. bash shell
   bash means bourne again shell which is advance version of bourne shell.
   bash shell is default shell in linux.

3. cshell

4. tcshell
   tcshell is advance version of cshell

To see the default shell

- echo $SHELL

To see current or running shell

- echo $0

To go go bourne shell

- sh

To go to bash shell

- bash

To go to cshell

- csh

To go to tcshell

- Tcsh

2. Kernel

kernel is an interface between shell & hardware i.e computer

- It is the core component of linux operating system
- It is responsible to execute our commands.

To see kernel version

uname -r

Types of kernel

1. monolytic kernel
2. microlytic kernel
3. hybrid kernel

****

For practice purpose lab.redhat.com Basic commands in linux

****

Basic commands in linux

- To see current date & time
  date

- To see current month calendar
  cal
  cal -1

- To see previous, current, next month calender
  cal -3

- To see a particular year calendar
  cal -y 2023
  y means year

To clear session / terminal / screen

- clear
- control l

To see the history of commands

- history

****

commands create a file in linux

****

1. touch
2. echo
3. cat
4. vi
5. Vim

1).Touch command is used to create files in linux

- touch filename
- touch file1

To create multiple files

- touch file2 file3

To create file1 to file 100

- touch file{1..100}

2)mkdir command is used to create a directory in linux

- mkdir dirname
- mkdir dir1

To create multiple directories

- mkdir dir2 dir3

To create dir11 to dir20

- mkdir dir{11..20}

To see all files & folders

- ls

3)**To delete all files & folders**

- rm -rf *

To delete file

- rm filename

To delete the file forcefully

- rm -f filename

To delete directory

- rm -r directoryname

To delete the directory forcefully

- rm -rf directoryname

To delete all files & folders

- rm -rf *

To delete all files & folders which are starting with a

- rm -rf a*

To delete all files & folders which are starting with ab

- rm -rf ab*

To delete all files & folders which are starting with a & ending with b

- rm -rf a*b

To create a file using the cat command

- cat > filename
- (or)
- cat >> filename

1)> is called a single redirection arrow
2)>> is called as a double redirection arrow

To save a file

- ctrl d

To read a file

- cat filename

To overwrite the data

- cat > filename
- Overwrite data means remove old data and add new data

To upend the data

- cat >> filename
- Upend data means add new data in old data

How to create a file using echo command

- echo "data" > filename
- or
- echo "data" >> filename

Can we read a file using the echo command?

- No, we cannot read a file using echo command
- touch command is used to create only empty file
- We cannot create a non-empty file using the touch command
- cat & echo command is used to create both empty & non empty files
- We can create multiple files at a time using the touch command
- We cannot create multiple files at a time using the cat & echo command
- We cannot read files using the touch & echo command

To see the history of commands

- history

To clear or delete history

- history -c

To recover history

- history -r

Man command is use to see information of a particular command

- man command
- man touch
- man mkdir

****

Head & tail command

****

To see the 1st five lines of the file

- head -5 filename
- cat filename | head -5

To see 1st seven lines of file

- head -7 filename
- cat filename | head -7

To see 1st 10 lines of file

- head -10 filename
- cat filename | head -10

- head filename
- cat filename | head

To see the last five lines of the file

- tail -5 filename

To see the last seven lines of the file

- tail -7 filename

To see the last 10 lines of file

- tail -10 filename
- tail filename

head -5 filename | tail -2

to see line number 4 & 5

****

Create files using vi & vim

****

How to create files using vi & vim

- vi filename
- vim filename

Modes in vi & vim

- Command mode
- Command mode is the default mode.
- To enter command mode, press Esc.

Insert mode

- To enter insert mode, press i.

To save the file

- :wq!

To quit the file without saving changes

- :q!

To go to the last line

- Shift + g

To go to the first line

- gg

To go to the 5th line:

- :5

To go to the 7th line

- :7

To delete 1 line starting from the cursor

- dd
- (or)
- 1dd

To delete 2 lines starting from the cursor:

- 2dd

To delete 3 lines starting from the cursor

- 3dd

To delete 5 lines starting from the cursor

- 5dd

To delete the 5th line, irrespective of the cursor

- :5d

To delete the 6th and 7th lines, irrespective of the cursor

- :6,7d

To delete all lines in a file from the cursor

- :.,$d

To delete all lines in a file, irrespective of the cursor

- :%d

To set line numbers (temporarily)

- :set nu
- (or)
- :set number

****

Yanking method

****

The process of copying and pasting multiple lines is called the yanking method.

To copy a line from the cursor

- yy

To copy 2 lines from the cursor

- 2yy

To copy 5 lines from the cursor

- 5yy

To paste lines

- p (below the cursor)
- or
- Shift + p (above the cursor)

****

To Search and Replace

****

To search for a particular word

- :/word

To search for a particular letter

- :/letter

To search for a particular number

- :/number

To search for a particular word and replace it

- :%s/word_to_search/word_to_replace

To search for a particular word and replace it globally

- :%s/word_to_search/word_to_replace/g

- % means all
- s means substitute
- g means globally

****

Linux File System Hierarchy

****

Linux File System Hierarchy

- The Linux file system has a tree-like structure.
- It starts with / (forward slash).
- The / (forward slash) is the topmost directory.
- It is the main directory that contains the following important subdirectories:
  bin, sbin, lib, etc, dev, opt, home, usr, tmp, media.

1) bin Directory

- bin stands for binaries.
- Normal user binaries (commands) are present in this directory.

2) sbin Directory

- sbin stands for system binaries.
- Root user binaries (commands) are present in this directory.

Q) What is the difference between bin and sbin?

- Normal user binaries (commands) are present in the bin directory.
- Root user binaries (commands) are present in the sbin directory.

3) etc Directory

- This directory contains all system configuration files.
- All user information is available in the /etc/passwd file.
- All group information is available in the /etc/group file.

4) tmp Directory

- tmp stands for temporary.
- It contains all temporary files created during the current session.
- If any file is required only for the current session, it should be created inside the tmp directory. These files are automatically deleted when the system shuts down.
- It is not recommended to create files that are needed permanently inside the tmp directory.

5) mnt Directory

- mnt stands for mounting.
- The mnt directory is used for temporarily mounting file systems.

6) media Directory

- The media directory is used for permanently mounting file systems.

7) home Directory

- Since Linux is a multi-user operating system, a separate directory is created for each user in the home directory.
- The home directory of a normal user is /home/username.
- For example, the home directory of the user lekharaj would be /home/lekharaj.

8) root Directory

- It is the home directory of the root/admin/superuser.
- The home directory of the root user is /root.

Q) What is the difference between / (forward slash) and the Root Directory?

- / (forward slash) is the topmost directory in the Linux file system, and the root is a subdirectory present in / (forward slash).
- The root directory is the home directory of the root user/admin/superuser.

9) boot Directory

- This directory contains the files required to boot the Linux operating system.

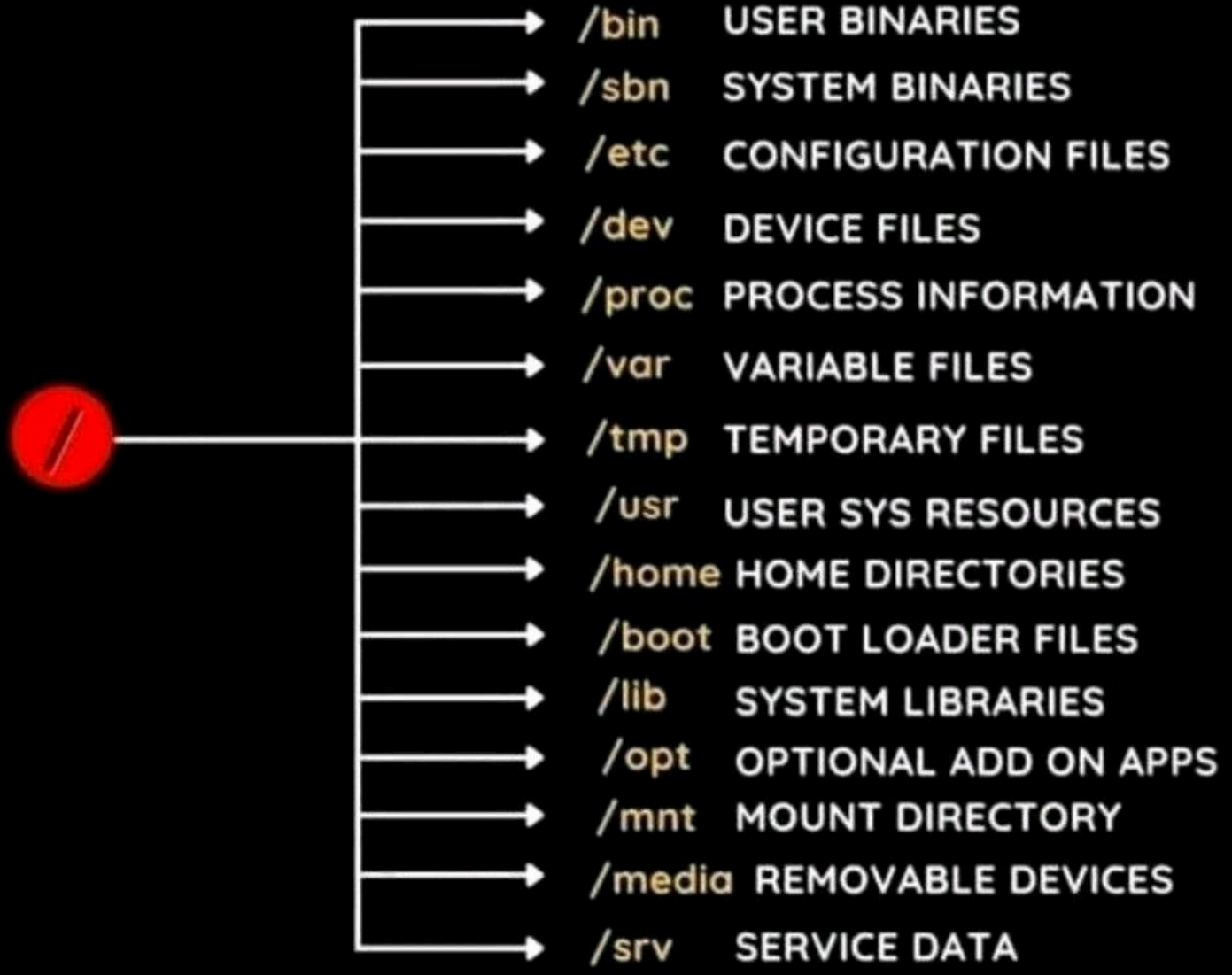
10) var Directory

- var stands for variable data.
- If any data is expected to change frequently, it will be stored inside the var directory.
- Log files are stored inside var.

11) proc Directory

- proc stands for processes.
- To view memory information, go to the file /proc/meminfo.
- To view CPU information, go to the file /proc/cpuinfo.

LINUX
PATH CHEATSHEET



pwd

- To see the present working directory

ls

- To see all files & folders in the present working directory

Absolute path

- An absolute path always starts with a / (forward slash).
- In an absolute path, the current or present working directory does not matter.

Example commands

- cd /
- cd /home
- cd /etc
- cd /root

Absolute Path:

- An absolute path always starts with a / (forward slash).
- In an absolute path, the current or present working directory does not matter.

Relative Path:

- In a relative path, the present working directory matters.

To create dir2 inside /dir1, /dir1 must already exist.

- mkdir /dir1
- mkdir /dir1/dir2

How to create recursive directories:

- mkdir -p /dir1/dir2/dir3
- -p stands for path or parents which allows the creation of any necessary parent directories along the specified path.

options of cd command

To go to the current directory

- cd .

To go to the parent directory

- cd ..

To go 2 steps back

- cd ../..

To go to the users home directory

- cd

To go to the users home directory

- cd ~

To go to the previous working directory

- cd -

****

Hostname in Linux & Windows

****

To see the hostname in Linux & Windows:

- hostname

To change the hostname:

- hostname name

To see the IP address in Windows:

- ipconfig

To see the IP address in Linux:

- ifconfig
- hostname -i
- ip addr
- ip add
- ip ad
- ip a

To see the current logged-in user

- whoami

****

Options of ls command

****

ls

- To see all files and directories in the present working directory.

ls a*

- To see all files and directories starting with 'a' in the present working directory.

ls ab*

- To see all files and directories starting with 'ab' in the present working directory.

ls *a

- To see all files and directories ending with 'a' in the present working directory.

ls a*a

- To see all files and directories starting and ending with 'a' in the present working directory.

ls a*b

- To see all files and directories starting with 'a' and ending with 'b' in the present working directory.

ls -a

- To see all files, including hidden files.

ls -l

- To see a long listing of all files and directories in the present working directory.

ls -i

- To see the inode number of files and directories.

ls -lt

- To see a long listing of all files and directories sorted by timestamp (latest first).

ls -ltr

- To see a long listing of all files and directories sorted by reverse timestamp (oldest first).

Understanding file sizes

1. **1,024 bytes = 1 kilobyte (1 KB)**
2. **1,024 KB = 1 megabyte (1 MB)**
3. **1,024 MB = 1 gigabyte (1 GB)**
4. **1,024 GB = 1 terabyte (1 TB)**

Archiving of Files

- Archiving of files refers to the process of combining multiple files into a single file, often for easier storage or transfer.

tar Command

- The tar command is used to create archive files, optionally with compression.

To create an archive file without compression

- tar -cvf archive_filename file1 file2 file3

To create an archive file with compression

- With gzip compression (using .gz)
- tar -cvzf archive_name.tar.gz file1 file2 file3
- With bzip2 compression (using .bz2) tar -cvjf archive_name.tar.bz2 file1 file2 file3
- With xz compression (using .xz) tar -cvJf archive_name.tar.xz file1 file2 file3

To view the contents of an archive file

- tar -tvf archive_filename

To extract files from an archive file

- tar -xvf archive_filename
- **c** → Create a new archive
- **v** → Verbose (display file names as they are archived or extracted)
- **f** → Specify the archive file name
- **t** → List the contents of the archive
- **x** → Extract files from the archive
- **z** → Use gzip compression
- **j** → Use bzip2 compression
- **J** → Use xz compression

rpm (Red Hat Package Manager)

- RPM stands for Red Hat Package Management.
- RPM is a package management tool that helps us install, update, and remove software packages.
- However, rpm does not resolve dependencies.
- This means that if a package (e.g., xyz) requires other packages (e.g., a, b, c), we must install both xyz and its dependencies manually.

To use rpm:

Install a package:

- rpm -ivh packagename
- i stands for install
- v stands for verbose
- h stands for hash (displays progress)

Erase/Remove a package:

- rpm -ev packagename

Upgrade a package:

- rpm -Uvh packagename
- U stands for upgrade

yum

- yum stands for Yellowdog Updater Modified.
- YUM is also a package management tool, but it resolves dependencies. This means if you want to install a package (e.g., xyz), YUM will automatically handle the installation of its required dependencies (e.g., a, b, c).

To use yum:

To see all the packages:

- yum list

To see all installed packages:

yum list installed

To check if a specific package is installed:

yum list pkgname

To install a package:

- yum install pkgname
- or
- yum install pkgname -y

- The -y flag auto-confirms the installation.

To update a package:

- yum update pkgname

To update all packages:

- yum update

To remove a package:

- yum remove pkgname
- or
- yum remove pkgname -y

Package location in YUM:

- /etc/yum.repos.d/

YUM configuration/master file:

- /etc/yum.conf

Homework

Try installing and removing the following packages:

- httpd
- tree
- git

****

Cron/Crontab

****

Cron is a Unix-based utility used to schedule jobs (commands or scripts) to run automatically at specified times.
These tasks can be one-time or repetitive.

Cron Fields

The cron schedule is divided into five fields, which define the timing for executing a job:

1. Minutes (0-59)
2. Hours (0-23)
3. Date (1-31)
4. Month (1-12)
5. Day (0-6)

The days of the week are represented as follows:

- Sun → 0
- Mon → 1
- Tue → 2
- Wed → 3
- Thu → 4
- Fri → 5
- Sat → 6

To create or edit a cron job

- crontab -e

To list scheduled cron jobs

- crontab -l

To remove scheduled cron jobs

- crontab -r

This runs every minute.

- * * * * *

This runs every minute

- */1 */1 */1 */1 */1

This runs every 2 minutes.

- */2 * * * *

This runs every 10 minutes.

- */10 * * * *

This runs daily at 11:05 AM.

- 05 11 * * *

This runs at 11:05 AM on the 1st of every month.

- 05 11 01 * *

Configuration file of crontab

- /etc/crontab

****

Hard Link & Soft Link of files

****

Hard Link

- A hard link is just another name for the same exact file.
- We can create a hard link of a file using the command:

  ln originalfilename hardlinkfilename

- The original file and the hard link both have the same inode number, size, and timestamp.
- If we add or delete any data in the original file, the same changes will be reflected in the hard link file and vice versa.
- If we delete the original file, there will be no effect on the hard link.
- Hard links can only be created for files, not directories.

Soft Link

- A soft link (also called a symlink or symbolic link) is just a pointer to the original file and is used for shortcut purposes.
- We can create a soft link to a file using the command:

ln -s originalfilename softlinkfilename

- The original file and the soft link have different inode numbers, sizes, and timestamps.
- If we add or delete any data in the original file, the same changes will be reflected in the soft link file and vice versa.
- If we delete the original file, the soft link becomes useless.
- Soft links can be created for both files and directories.

Difference Between Hard Link and Soft Link

- A hard link is used for backup purposes, while a soft link is used for shortcut purposes.
- We can create a hard link of a file using the command:
- ln originalfilename hardlinkfilename
- We can create a soft link of a file using the command:
- ln -s originalfilename softlinkfilename

- The original file and its hard link have the same inode number, timestamp, and size.
- The original file and its soft link have different inode numbers, timestamps, and sizes.

- Hard links can only be created for files, not directories.
- Soft links can be created for both files and directories.

- If we delete the original file, there is no effect on the hard link.
- If we delete the original file, the soft link becomes useless.

****

cp command (copy command) & mv(move command)

****

cp command

- cp stands for "copy" and is used to copy files and directories from one location to another.

1) To Copy from File1 to File2 (File to File):

- cp source_file destination_file

For example:

cp file1 file2

- This will copy the contents of file1 to file2.
- If file2 doesn't exist, it will be created.
- If file2 already exists, its contents will be overwritten with file1's content.

2) To Copy File to Directory:

cp file1 file2 dir1

- Both file1 and file2 will be copied into dir1.
- We can specify multiple files, but the last argument must be a directory.
- The directory dir1 must exist.

3) To Copy All Files of One Directory to Another Directory:

cp dir1/* dir2

- All files from dir1 will be copied to dir2.
- dir2 must already exist.

4) To Copy a Whole Directory to Another Directory:

- To copy a directory, the -r (recursive) option must be used:

cp -r dir1 dir2

- This will copy the entire dir1 to dir2.
- If dir2 exists, dir1 will be copied into it.
- If dir2 does not exist, it will be created, and dir1 will be copied into dir2.

Note:

- **Always use -r when copying directories, otherwise, it will result in an error.**

5) To Copy Multiple Directories into a Single Directory:

cp -r dir1 dir2 dir3 dir4 dir5

- dir1, dir2, dir3, and dir4 will be copied into dir5.
- dir5 must already exist.

Moving and Renaming of Directories:

- The mv command is used to move and rename files and directories.
- Both operations can be performed with a single command: mv.

Renaming Files:

This will rename oldname to newname.

- mv oldname newname

For example:

- mv file1.txt file2.txt
- file1.txt will be renamed to file2.txt.

Renaming Directories:

- mv dir1 dir2
- This will rename the directory dir1 to dir2.

Moving Files to a Directory:

- mv file1 file2 dir1
- This will move both file1 and file2 to the directory dir1.

Moving All Files from One Directory to Another Directory:

- mv dir1/* dir2

- This will move all the files from dir1 to dir2.
- After executing this command, dir1 will be empty, but dir1 itself will not be deleted unless explicitly removed.

Moving a Directory to Another Directory:

- mv dir1 dir2
- If dir2 exists, dir1 will be moved inside dir2.
- If dir2 does not exist, dir1 will be renamed to dir2.

cp -r dir1 dir2

- This will copy the entire dir1 directory into dir2.
- If dir2 already exists, it will create a dir1 subdirectory inside dir2, and the content of dir1 will be placed inside it.

cp -r dir1/* dir2

- This will copy the contents (files and directories) of dir1 into dir2, but it will not copy dir1 itself.

cp -r dir1/a* dir2

- This will copy all files and directories in dir1 whose names start with the letter a (or any other character matching a* pattern) into dir2.

cp -r dir1/*b dir2

- This will copy all files and directories from dir1 whose names end with the letter b into dir2.

****

user management

****

user management

- user management means managing users.

Managing users involves:

- To add a user
- To delete a user
- To set a password for a user

Types of users

1. root user/admin/superuser

- The uid & gid of the root user are both 0.

2. system users/system accounts

- The uid & gid of system users are in the range 1 to 999.

3. normal users

- The range of uid & gid of normal users is from 1000 to 60,000.
- When we add a user, a directory for that user will be created in the /home directory with the same name as the user.

For example:-

- when we use the command useradd lekharaj, a directory named lekharaj will be created in /home, so the absolute path of the directory will be /home/lekharaj.
- This directory is called the home directory of the lekharaj user.

What is a system user/system account?

- When we install any package, a user is created with the same name as the package, called a system user.

Examples of system users include:

- mysql, httpd, nfs, sshd, ftp.

How to create a user

- useradd username
- We cannot create multiple users at a time using the useradd command.

How to check whether a user is present or not

- id username

How to see the current logged-in user

- whoami

To set the password for a user

- passwd username

To set the password for the currently logged-in user

- passwd

****

Options of echo

****

To set password for a user

- echo "password" | passwd --stdin username

To see the default shell in linux

- echo $SHELL

To see the current or current shell in linux

- echo $0

To create a file

- echo "data" > filename

To see all files and directories in the current directory

- echo *

To see the current month's calendar

- echo $(cal)

To see the current date and time

- echo $(date)

****

How to delete a user

****

userdel username:

- This command deletes the user, but the user's home directory will remain intact.

userdel -r username:

- This command deletes both the user and their home directory.

- When we add a user, a directory for that user is created in the /home directory with the same name as the user.

For example:-

- when we use the command useradd lekharaj, a directory called lekharaj is created in /home.
- Therefore, the absolute path of the directory is /home/lekharaj, and this is referred to as the home directory of the lekharaj user.

When a user is added, an entry for that user is created in the following files:

/etc/passwd

- (This file contains user profile information.)

/etc/shadow

- (This file contains user password information.)

Fields of /etc/passwd file

1. **Username**
2. **Password, denoted by "x"**
3. **UID of the user**
4. **GID of the user**
5. **Common name of the user**
6. **Home directory of the user**
7. **Default shell of the user**

Can we add multiple users at a time using the useradd command in Linux?

- No, we cannot add multiple users simultaneously using the useradd command.

How to add multiple users at a time in Linux:

a) Create a file with any name (e.g., file1).
b) Enter the user details in that file (file1) in the same format as /etc/passwd.
c) Use the command newusers filename (e.g., newusers file1) to add the users.

Fields of /etc/shadow

1) Username

2) !! Or Password of user in encrypted form

3) Last password change

- The number of days since January 1, 1970 when the password was last changed.

4) Minimum password age (m=0)

- The minimum number of days that must pass before the user is allowed to change their password again.

5) Maximum password age (M=99,999)

- The maximum number of days the password is valid.
- After this period, the user must change the password.

6) Password warning days (W=7)

- The number of days before password expiration that the user will be warned (e.g., 7 days).

7) Account expiration date (E)

- The date on which the user's account will be disabled.

8) Inactive period

- The number of days after the password has expired during which the user can still log in (account will be disabled after this period).

/etc/default/useradd

- (To change home dir, shell, expiration date of account, inactive days of upcoming users)
- The /etc/default/useradd file is used to set default settings for new user accounts, such as home directory, shell, password expiration, and inactive days.

/etc/login.defs

- (To change M, m, W, and to see the password encryption algorithm)
- The /etc/login.defs file contains login configuration options like password aging, encryption algorithms, and more.

chage command

- chage command is used to see or change the password policy of existing users.

To see the user's password policy

- **chage -l username**

To change the password policy of an existing user

- **chage username**

- **chage -m 1 username**
- **chage -M 30 username**
- **chage -W 7 username**

The chage command is used to display or modify the password expiration details for an existing user.

- **m** sets the minimum number of days between password changes.
- **M** sets the maximum number of days the password is valid.
- **W** sets the warning period before password expiration.

Types of algorithms used to encrypt user passwords

- MD5 (starts with $1)
- SHA256 (starts with $5)
- SHA512 (starts with $6)

How to change the encryption algorithm used for user passwords

- authconfig --passalgo=md5 --update
- authconfig --passalgo=sha256 --update
- authconfig --passalgo=sha512 --update

****

How to create a user

****

How to create a user

- useradd username

How to create a user without a home directory

- useradd -M username

How to create a user named sumit with a home directory in /etc

- useradd -d /etc/sumit sumit

How to create a user with a custom UID

- useradd -u uid username

How to change the UID of an existing user

- usermod -u uid username

How to add a user without an interactive shell

- useradd -s /sbin/nologin username

How to remove the shell of an existing user

- usermod -s /sbin/nologin username

****

How to create a group

****

How to create a group

- groupadd groupname

How to delete a group

- groupdel groupname

How to create a user and add them to groups

- useradd -G groupname username

How to add a user to a secondary group

- usermod -G groupname username

How to add a user to multiple groups

- usermod -G group1,group2 username

How to see which groups a user belongs to

- groups username

How to create a user without a primary group

- useradd -N username

How to add an existing user to a secondary/supplementary group using the gpasswd command

- gpasswd -M username groupname

How to add multiple users to a secondary group using the gpasswd command:

- gpasswd -M username1,username2 groupname

How to remove a user from a group using the gpasswd command:

- gpasswd -d username groupname

To switch user

- su - username
- su means switch user

What is the importance of the sudoers file?

- If we need to allow a normal user to execute commands as the root user, we just need to add that user to the sudoers file.

Edit the sudoers file using

- vim /etc/sudoers

Add the following entry for the user sagar

- sagar ALL=(ALL) ALL

Now, to switch to the user

- sagar
- su - sagar

We can run commands as root using sudo
for example:

- sudo yum install -y java
- sudo useradd user1
- sudo userdel -r user1
- sudo means superuser do

```
root user --> normal user --> password --> no

normal user --> root user --> password  --> yes

normal user1 --> normal user2 --> password  --> yes
```

The meaning of

- **sudo su - root**
- **&**
- **sudo su -**
- is the same.

To add a group to the sudoers file, use the command below:

- **%groupname ALL=(ALL) ALL**
-
- **%rockstar ALL=(ALL) ALL**

****

Permissions

****

Fields of ls -l command

- **rw-r--r-- 1 root root 6 May 18 04:19 file2**

1. Type of the file

- **-** means a regular file
- **D** means a directory
- **L**. means soft link file

2. Permissions

- The first 3 permissions are for the owner of the file or directory.
- The next 3 permissions are for group members.
- The last 3 permissions are for others.

3. Hard link count

- Generally, the hard link count is 1 for a file and 2 for a directory.

4. Owner name of the file/directory

5. Group name of the file/directory

6. Size of the file in bytes

- 1024 bytes = 1 kilobyte (1 KB)
- 1024 KB = 1 megabyte (1 MB)
- 1024 MB = 1 gigabyte (1 GB)
- 1024 GB = 1 terabyte (1 TB)

- To see the size of a file or directory in MB, use the command ls -lh
- The -h flag stands for human-readable format.

7. Creation date or modification time

8. Filename / Directory name

Types of Permissions

1. r permission r = 4
2. w permission w = 2
3. x permission x = 1
4. - (no permission) - = 0

The possible permission combinations are as follows:

- rwx = 7
- rw- = 6
- r-x = 5
- r-- = 4
- -wx = 3
- -w- = 2
- --x = 1
- --- = 0

Meaning of r, w, x permissions for a file:

- r (read):

The user can read the contents of the file.

- w (write):

The user can modify the contents of the file.

- x (execute):

The user can execute the file as a program or script.

To change the permissions of a file/directory:

- chmod permissions filename/dirname

To change the ownername of a file/directory:

- chown ownername dir/filename
- chown lekharaj /file1

To change the groupname of a file/directory:

- chgrp groupname dir/filename
- chgrp rockstar /file1

To change both the owner and group in a single command:

- chown ownername:groupname dir/filename
- chown sumit:linux /file1

Meaning of r,w,x permissions for a directory

- x permission

The user can access (cd into) the directory.

- r permission

The user can list the contents of the directory (using ls).

- w permission

The user can create, delete, or rename files and subdirectories within that directory.

- Default permission for file is 644
- Default permission for dir is 755
- Maximum permission for file is 666
- Maximum permission for dir is 777

****

ACL (Access Control List)

****

ACL (Access Control List)

- ACL stands for Access Control List, which is used to apply specific permissions to particular users or groups for a specific file or directory.

To apply ACL to a user

- setfacl -m u:username:permissions dirname

To apply ACL to a group

- setfacl -m g:groupname:permissions filename

To check whether ACL is applied

- getfacl filename

To remove ACL from a specific user

- setfacl -x u:username filename

To remove ACL from all users

- setfacl -b filename

****

Scripting

****

scripting / shell scripting / bash scripting

- A script file is a simple text file that contains a sequence of commands.
- A sequence of commands written into a file is called a script or scripting.
- We give executable permission to that file.

Importance of shebang:

- By using shebang, we can specify the interpreter that is responsible for executing the script.

#! /bin/bash

- This means the script should be executed by bash.

#! /bin/sh

- This means the script should be executed by the Bourne shell.

1. Create a file with the extension .sh
(The .sh extension is not mandatory.)
2. Define the shebang and write commands to execute in this file.
3. Give executable permission to the file:
(chmod 700 filename)
4. Define the shebang and write the absolute path of the file.

1. Script to create multiple users

```
for n in user1 user2 user3
do
useradd $n
id $n
done
```

2. Script to delete multiple users

```
for n in user1 user2 user3
do
userdel -r $n
id $n
done
```

3. Script to create multiple users and set passwords for those users

```
for n in user1 user2 user3
do
useradd $n
id $n
echo "Android@50" | passwd --stdin $n
done
```

****

operators

****

operators

1. arithmetic operators

+ - * /

2. relational operators

- greater than -gt
- less than -lt
- equal to -eq
- not equal to -ne
- greater than or equal to -ge
- less than or equal to -le

3. logical operators

- AND -a
- OR -o
- NOT !

4. assignment =

Examples

1)Script for addition of 2 numbers

2)Script for

```
a=20
b=10
c=expr $a + $b
echo "Addition is $c”
```

subtraction of 2 numbers

```
a=20
b=10
c=expr $a - $b
echo "Subtraction is $c"
```

3)Script for multiplication of 2 numbers

```
a=20
b=10
c=expr $a \* $b
echo "Multiplication is $c"
```

4)Script for division of 2 numbers

```
a=20
b=10
c=expr $a / $b
echo "Division is $c"
```

5)Script for addition, subtraction, multiplication, and division of 2 numbers

```
a=20
b=10

c=expr $a + $b
d=expr $a - $b
e=expr $a \* $b
f=expr $a / $b

echo "Addition is $c"
echo "Subtraction is $d"
echo "Multiplication is $e"
echo "Division is $f"
```

Ways to define the values of variables

There are 2 ways to define the values of variables:

1) Inside the script file
2) By providing the absolute path of the file

Examples:-

1)Script for addition of 2 numbers

```
read -p "Enter the 1st number: " a
read -p "Enter the 2nd number: " b

c= expr $a + $b
echo "The addition is $c"
```

2)Script for subtraction of 2 numbers

```
read -p "Enter the 1st number: " a
read -p "Enter the 2nd number: " b

c= expr $a - $b
echo "The subtraction is $c"
```

3)Script for multiplication of 2 numbers

```
read -p "Enter the 1st number: " a
read -p "Enter the 2nd number: " b

c= expr $a \* $b
echo "The multiplication is $c"
```

4)Script for division of 2 numbers

```
read -p "Enter the 1st number: " a
read -p "Enter the 2nd number: " b

c= expr $a / $b
echo "The division is $c"
```

5)Script for addition, subtraction, multiplication, and division of 2 numbers

```
read -p "Enter the 1st number: " a
read -p "Enter the 2nd number: " b

c= expr $a + $b
d= expr $a - $b
e= expr $a \* $b
f= expr $a / $b

echo "The addition is $c"
echo "The subtraction is $d"
echo "The multiplication is $e"
echo "The division is $f"
```

Variables

Types of variables
variables are divided into 3 types

1) local variables

local variables are also called as User defined variables
ex: a=10, b=5
value of local variable changes from script to script

2) Environment variables

environmental variables are also called as system defined variables
value of environmental variable is same throughtout the system/script
ex: hostname, ip, default shell,kernerl version,path

3) special variables/ command line arguments

The variables which are passing through command line are called as command line variables

$0 filename
$1 1st argument
$2 2nd argument
$# total number of argument
$@ All command line arguments with space separator
$* All command line arguments as single string

```
cat > /file1

echo "filename: $0"
echo "1st argument: $1"
echo "2nd argument: $2"
echo "3rd argument: $3"
echo "total no of arguments: $#"
echo "All arguments: $@"
echo "All arguments: $*"

/file1 hello all
```

Loop:

we use a loop to do one thing multiple times

1. for loop
2. infinity loop
3. while loop
4. until loop

1) **a. for loop to add multiple users**

```
for n in user1 user2 user3
do
  useradd $n
done
```

1) **b. for loop to delete multiple users**

```
for n in user1 user2 user3
do
  userdel -r $n
done
```

1) **c. for loop to to create multiple dir**

```
for n in /dir1 /dir2 /root/dir3
do
  mkdir $n
done
```

1) **d. for loop to print numbers**

```
for n in 1 2 3 4 5
do
  echo "number is $n"
done
```

1) **e. for loop to create multiple dirs**

```
for n in 1 2 3 4 5
do
  mkdir dir$n
done
```

1) **f. for loop to print numbers**

```
for ((n=1;n<6;n++))
do
  echo "number is $n"
done

for ((n=1;n<6;n++))
do
  echo "Hello all"
done
```

Infinity loop

```
for ((n=2;n>1;n++))
do
  echo "number is $n"
done
```

while loop

while loop exec the command till the condition is true

```
a=1
while [ $a -lt 10 ]
do
echo "$a"
a= expr $a + 1
done
```

until loop

until loop exec the command till the condition is not true

```
a=1
until [ $a -eq 10 ]
do
echo "$a"
a= expr $a + 1
done
```

(OR)

```
a=1
until [ $a -gt 10 ]
do
echo "$a"
a= expr $a + 1
done
```

Difference between while & until loop

- while loop executes the command till the condition is true
  &
- until loop executes the command till the condition is not true

Decision making statements

1. if statement
2. if else statement
3. if elif else statement
4. Nested if statement

1. if statement

syntax for if

```
if [ condition ]
then
statement to be executed when condition is true
Fi
```

Example

```
read -p "enter 1st value:" a
read -p "enter 2nd value:" b

if [ $a -gt $b ]
then
   useradd sumit
fi
```

2. if else statement

syntax for if-else

```
if [ condition ]
then
statement to be executed when condition true
else
statement to be executed when condition not true
fi
```

Example

```
read -p "enter 1st value:" a
read -p "enter 2nd value:" b

if [ $a -eq $b ]
then
   useradd sagar
else
   useradd jagdish
fi
```

Script to find greater number

```
read -p "enter 1st number:" a
read -p "enter 2nd number:" b
if [ $a -gt $b ] ; then
echo "Greater number is $a"
else
echo "Greater number is $b"
fi
```

script to find a smaller number

```
read -p "enter 1st number:" a
read -p "enter 2nd number:" b
if [ $a -lt $b ] ; then
echo "smaller number is $a"
else
echo "smaller number is $b”
fi
```

Script to check whether 2 numbers are the same or not

```
read -p "enter 1st number:" a
read -p "enter 2nd number:" b
if [ $a -lt $b ] ; then
echo "Both numbers are equal"
else
echo "Both numbers are different"
fi
```

Script to print result

if marks are greater than 35 then print Result is Pass
else print Result is Fail

```
read -p "enter your marks:" a
if [ $a -gt 35 ] ; then
echo "Result is Pass"
else
echo "Result is Fail"
fi
```

3. if elif else statement

syntax for if-elif else

```
if [ condition1 ]
then
statement to be executed if condtition1 is true
elif [ condition2 ]
statement to be executed if condtition2 is true
else
statement to be executed if no condition is true
Fi
```

Example:

```
read -p "enter 1st value:" a
read -p "enter 2nd value:" b
if [ $a -gt $b ]
then
   mkdir /dir1
   ls
elif [ $a -lt $b ]
then
   mkdir /dir2
   ls
else
   mkdir /dir3
   ls
fi
```

4. Nested if

syntax for Nested if

```
if [ condition1 ]
then
statement to be executed when condition1 is true
else
if [ condition2 ]
statement to be executed when condition2 is true
else
if [ condition3 ]
statement to be executed when condition3 is true
    fi
  fi
fi
```

Example:

```
read -p "enter 1st value:" a
read -p "enter 2nd value:" b
if [ $a -gt $b ]
then
    mkdir /dir1
    ls
else
if [ $a -lt $b ]
then
    mkdir /dir2
    ls
else
if [ $a -eq $b ]
then
    mkdir /dir3
    ls
   fi
 fi
fi
```

1) What is scripting?
2) What is a shebang?
3) What is the importance of the shebang?
4) What do you know about scripting?
5) What are the types of operators?
6) What are the types of variables?
7) What are the types of statements?
8) What are the types of loops?

1. Write a script to create multiple users.
2. Write a script to delete multiple users.
3. Write a script to create multiple users and set passwords for those users.
4. Write a script to add two numbers.
5. Write a script to subtract two numbers.
6. Write a script to multiply two numbers.
7. Write a script to divide two numbers.
8. Write a script to find the greater number.
9. Write a script to find the smaller number.
10. Write a script to check whether two numbers are the same or not.
11. Write a script to print numbers from 1 to 100.
12. Write a script to print odd numbers.
13. Write a script to print even numbers.

****

wc command

****

wc command

wc means word count

We can use the wc command to count the number of lines, words, and characters present in the given file.

wc filename

- no_of_lines no_of_words no_of_characters filename

We can use the following options with the wc Command:

To print only the number of lines.

- -l

To print only the number of words.

- -w

To print only the number of characters.

- -c
- -lw

To print only the number of lines and words.

- -lc

To print only the number of lines and characters.

- -wc

To print only the number of words and characters.

- -L

To print the number of characters present in the longest line.

- wc -l filename
- wc -w filename
- wc -c filename

- wc -lw filename
- wc -lc filename
- wc -wc filename

- wc -lwc filename
- wc filename

grep

- grep stands for globally search a regular expression and print it
- (OR)
- global regular expression print.
- We can use the grep command to search a particular pattern in a single or multiple files.
- pattern means data or word.

Note:
(Use Red Hat AMI for grep command)

1. To Search pattern in a Single File
   - grep pattern filename

2. To Search pattern in multiple Files
   - grep pattern filename1 filename2

3. To Display Line Numbers before Results
   - grep -n pattern filename

4. To Display the Number of Occurrences
   - grep -c pattern filename
   - OR
   - grep pattern filename | wc -l

To Search Data by ignoring Case
   - grep -i pattern filename

5. To Print except matched Lines (remaining Lines)
   - grep -v pattern filename

6. To Search for the exact word in the File
   - grep -w pattern filename

7. To display before, after and surrounding lines
   - (including Search Results)

We have to use -A, -B, -C options

- grep -A 1 pattern filename
- grep -B 1 pattern filename
- grep -C 1 pattern filename

- A means after
- B means before
- C means surrounding (after & before)

8. To Print only matched Patterns instead of Total Line
   - grep -o pattern filename
   - wget command
   - wget command is used to download files from the internet
   - wget url

WinSCP Tool

- WinSCP is a file transfer tool that is used to share files between a Windows machine and a Linux machine, and vice versa.
- WinSCP supports cross-platform file transfers between Windows and Linux systems.

Find Command

Find a particular file by name
- find / -name filename

To find all files with size less than 5KB
- find / -size -5k

To find all files with size more than 10MB
- find / -size +10M

Find all files and directories which were modified more than 30 days ago
- find / -mtime +30

Find all files and directories in /root which were modified less than 30 days ago
- find /root -mtime -30 -name '*.txt'

Find all files only which were modified more than 30 days ago
- find / -type f -mtime +30

Find all directories only which were modified more than 30 days ago

- find / -type d -mtime +30

Find all files and directories which were modified less than 30 days ago

- find / -mtime -30

To delete all files which are older than 30 days

- find / -type f -mtime +30 -exec rm {} \;

Difference Between ls -l and ls -ld

- ls -l dirname

Shows a long listing of all files and folders present inside dir

- ls -ld dirname

Shows a long listing only for the directory dirname

How to See the Size of Files and Directories

For directories:

- ls -ld dirname

For files

- ls -l filename

How to See CPU Info

- cat /proc/cpuinfo
- lscpu

How to See Memory Info

- cat /proc/meminfo
- free -mh (Human-readable format)

How to See Storage/Volume/Disk Size

- df -h
- lsblk
- fdisk -l

How to See the Total Number of Disks/Partitions/Storage/Volumes Available

- df -h
- lsblk
- fdisk -l

command to see previous command status
echo $?

hardware related commands

- df -h
- free -mh
- lscpu
- lsblk
- mount
- umount

Network related commands

- nslookup
- hostname
- hostname -i, ip addr, ip add, ip a, ifconfig
- netstat -tulnp
- nmap ipaddress
- telnet ipaddress

Explain any 5 commands

- chattr command
- echo $?
- nslookup
- newusers command
- authconfig command
- chkconfig command
- grep command
- find command
- gpasswd command
- Netstat command

How to Check the Creation Time of a Machine:

1. CloudTrail Method:
   - Go to the CloudTrail service.
   - Click on Event History.
   - Select Resource Name.
   - Enter the Instance ID.

   OR

2. Instance Storage Method:
   - Select the instance and go to the Storage tab.
   - There is an option for Attachment Time of the default volume, which is the actual creation time of the machine.

Common Protocols and Port Numbers

- SSH (Secure Shell)
- Port 22

RDP (Remote Desktop Protocol)
- Port 3389

MySQL
- Port 3306

HTTP (Hypertext Transfer Protocol)
- Port 80

HTTPS(Hypertext Transfer Protocol Secure)
- Port 443

NFS (Network File System)
- Port 2049

DNS (Domain Name System)
- Port 53

DHCP (Dynamic Host Configuration Protocol)
- Port 67, 68

SCP (Secure Copy)
Port 22

FTP (File Transfer Protocol)

- Ports 20, 21

SFTP (Secure File Transfer Protocol)

- Port 22

Rsync

- Port 873

how to see system running time
uptime

what are the states of the processes?

- running state (R)
- stopped (T)
- sleeping (S)

how to see running processes?

- ps

how to see all processes?

- ps -e
- ps -ef
- ps -aux

how to see the PID of a process

- top
- ps -ef

how to find the PID of a particular process?

- pidof processname
- (OR)
- pgrep processname
- (OR)
- ps -aux | grep processname