File Related Commands

1. Dd

The dd command stands for “**data duplicator**” and used for copying and converting data according to the operands specified. It is very powerful low level utility of Linux which can do much more like;

- Backup and restore the entire hard disk or partition.
  Backup of MBR (Master Boot Record)
- It can copy and convert magnetic tape format, convert between ASCII and EBCDIC formats, **swap** bytes and can also convert lower case to upper case.
- It can also be used by Linux kernel make files to make boot images.

2. DU

The Linux “**du**” (**Disk Usage**) is a standard **Unix/Linux** command, used to check the information of disk usage of files and directories on a machine. The **du** command has many parameter options that can be used to get the results in many formats. The **du** command also displays the files and directory sizes in a recursively manner

du estimates and displays the disk space used by files.

-a, **--all**	write counts for all files, not justdirectories.
-c	total count

3. df

The ‘**df**‘ command stand for “**disk filesystem**“, it is used to get full summary of available and used disk space usage of file system on Linux system.

Example:

- Displays information of dummy file systems along with all the file system disk usage and their memory utilization:

```
# df -a
```

- Display sizes in Human Readable formats:

```
# df -h
```

- To see the information of only device /home file system in human readable format:

```
# df -hT /home
```

- To display all file system information and usage in 1024-byte blocks:

```
# df -k
```

- To display information of all file system usage in MB (Mega Byte):

```
# df -m
```

- To display information of all file system statistics in GB (Gigabyte):

```
# df -h
```

- To check the file system type of your system:

```
# df -T
```

4. fdisk

fdisk stands (for “**fixed disk** or **format disk**“) is an most commonly used command-line based disk manipulation utility for a **Linux/Unix** systems. With the help of fdisk command you can view, create, resize, delete, change, copy and move partitions on a hard drive using its own user friendly text based menu driven interface.

fdisk is a partition table manipulator for Linux.

Tip: You must have root access for this command to work.

```
fdisk -l  : List the partition information of the computer you're logged into

fdisk -v : Print version information, and exit.

fdisk -h : Print help and then exit.
```

5. free

The *free* command provides information about unused and used *memory* and *swap space* on any computer running Linux or another Unix-like operating system.

6. reboot

it can be used to restart or **reboot linux**.

7. poweroff

This is roughly equivalent to pressing the **power** button on a typical desktop computer. If you are logged in as root, issuing the reboot **command** will immediately initiate a reboot sequence. The system will **shut** down and then commence a warm boot.

Compressing and Uncompressing

gzip, gunzip

gzip, gunzip are used to compress or expand files.

- **gzip** reduces the size of the named files. Whenever possible, each file is replaced by one with the extension **.gz**, while keeping the same ownership modes, access and modification times.
- **gunzip** can currently decompress files created by **gzip**

  Eg: gzip f1.sh   gunzip f1.sh.gz

Example:

- Compress the file file1.txt

  # gzip file1.txt

- Compress the multiple file file1.txt , f1.sh , f3.c

  # gzip file1.txt f1.sh f3.c

- UnCompress the file file1.txt

  # gunzip file1.txt   or   #gzip -d file1.txt

- UnCompress the multiple file file1.txt , f1.sh , f3.c

  # gunzip file1.txt f1.sh f3.c or   #gzip -d file1.txt f1.sh f3.c

- Compress the files in the particular folder

  # gzip -r xml

- UnCompress the files in the particular folder

  # gunzip -r xml   or   #gunzip -dr xml

zip, unzip

- Zip is a compression and file packaging utility for Linux and Unix and unzip will decompress the file

Example:

- The command given below creates a file "shell.zip" which contains a copy of the files named f1.sh ,f2.sh , f3.sh located in the current directory.

  zip shell f1.sh f2.sh f3.sh

- The command given below will compress current directory and also all subdirectories:

  zip -r new1 xml lss

- To extract the files you can use the unzip command given below. This will extract all files from backupfile.zip file to the current directory.

  #unzip backupfile.zip

- To list all the files inside the .zip file you can try the "-l" option given below:

  #unzip -l filename.zip   (shows the length of each file)   &

  # unzip -v filename.zip (shows both compressed and uncompressed size of each file in the archive along with the percentage of compression archieved.)

tar

The **tar** program is used to create, maintain, modify, and extract files that are archived in the **tar** format.

Example :

- Create a tar file that hold multiple folder in it.

```
#tar -cvf archive.tar file1 file2
```

Create archive **archive.tar** containing files **file1** and **file2**. Here,

c - tells **tar** you will be creating an archive;

f - tells **tar** that the next option (here it's **archive.tar**) will be the name of the archive it creates.

file1 and **file2**, the final arguments, are the files to be archived.

- Execute the tar file that is created.

```
#tar -tvf archive.tar
```

List the files in the archive **archive.tar** verbosely.

t- tells **tar** to list the contents of an archive;

v tells **tar** to operate verbosely;

f indicates that the next argument will be the name of the archive file to operate on.

- **`tar -xf archive.tar`**

Extract the files from archive **archive.tar**.

x tells **tar** to extract files from an archive;

f tells **tar** that the next argument will be the name of the archive to operate on.

- **`tar -xzvf archive.tar.gz`**

Extract the files from **gzip**ped archive **archive.tar.gz** verbosely. Here,

z tells **tar** that the archive will be compressed with **gzip**.

Dealing with files

- The **file** command is used to determine a file's type.

```
file *
```

- **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.

Examples

Basic Find Commands for Finding Files with Names

1. Find Files Using Name in Current Directory

Find all the files whose name is **tecmint.txt** in a current working directory.

```
# find . -name tecmint.txt
```

2. Find Directories Using Name

Find all directories whose name is **Tecmint** in **current** directory.

```
# find . -type d -name Tecmint
```

3. Find PHP Files Using Name

Find all **php** files whose name is **tecmint.php** in a current working directory.

```
# find . -type f -name tecmint.php
```

4. Find all files with particular extension

# find . -name “*.c” -print   or   #   find . -type f -name "*.php"

5.. **Find all files with starting with A-Z capital letters**

# find . -name “[A-Z]*” -print

Find Files Based on their Permissions

1. Find Files With 777 Permissions

Find all the files whose permissions are **777**.

```
# find . -type f -perm 0777 -print
```

2. Find Files Without 777 Permissions

Find all the files without permission **777**.

```
# find . -type f ! -perm 777
```

3. Find Read Only Files

Find all **Read Only** files.

```
# find . -perm /u=r
```

4. Find Executable Files

Find all **Executable** files.

```
# find . -perm /a=x
```

5. Find all Empty Files

To find all empty files under certain path.

```
# find /tmp -type f -empty
```

6. Find all Empty Directories

To file all empty directories under certain path.

```
# find /tmp -type d -empty
```

7. File all Hidden Files

To find all hidden files, use below command.

```
# find /tmp -type f -name ".*"
```

Find Files and Directories Based on Date and Time

(mtime – list of files that have been modified & atime – list of files that have been accessed)

1. Find Last 50 Days Modified Files

To find all the files which are modified **50** days back.

```
# find / -mtime 50
```

2. Find Last 50 Days Accessed Files

To find all the files which are accessed **50** days back.

```
# find / -atime 50
```

3. Find Last 50-100 Days Modified Files

To find all the files which are modified more than **50** days back and less than **100** days.

```
# find / -mtime +50 –mtime -100
```

note : +365 means greater than 365 days , -365 means less than 365 days . For specifying exactly 365,use 365.

- The **locate** command finds files by name.

  Example :

  ```
  # locate f1.sh
              /home/faculty/f1.sh.gz
              /home/faculty/.local/share/Trash/files/f1.sh
              /home/faculty/.local/share/Trash/info/f1.sh.trashinfo
  ```

- **whereis :Locates the binary, source, and manual page files for a command.**

  Example:

  ```
  #whereis firefox
      firefox: /usr/bin/firefox/etc/firefox/usr/lib/firefox/usr/bin/X11/firefox
      /usr/share/man/man1/firefox.1.gz
  ```

- **which : Locate the executable file associated with a given command.**

  Example:

  ```
  #which gcc
  /usr/bin/gcc
  ```