## Linux

| | | |
|---|---|---|
| 1 | Creating File without data | Single file : `touch <filename>.<extension>`<br>Multiple files : `touch <file1> <file2> <file3>...` |
| 2 | Creating file with minimal data | `echo “Hii ! , This is Data” > <filename2>.<extension>` |
| 3 | Adding extra data to existing file | `echo “Hii ! , This is Extra Data” >> <filename2>.<extension>` |
| 4 | To Replace the existing data : | `echo “Hii ! , This is Replacing Data”> <filename2>.<extension>` |
| 5 | Move file /cut | `mv <filename> <folder-location/directory>` |
| 6 | Copy file | `cp <filename> <location where you need to copy>` |
| 7 | Rename file | `mv <filename> <newfilename>` |
| 8 | Remove file | `rm <filename>` |
| 9 | Find file | `find <directory> -name <name-of-file>` |
| 10 | Locate file (search with keyword without directory) | `i) update db : sudo updatedb && locate -e bench-repo`<br>`ii) locate <keyword/filename/foldername>` |
| 11 | File edit using VI Editor (existing file) | `vi <filename+extension if its having>` |
| 12 | Basic VI Editor Keystrokes : | To Enter into Input mode / to add data/txt : type I |
| | | To Delete backward : backspace |
| | | To add next line : enter |
| | | To Save file : Click ESC , and type wq! Enter (means write quite) |

## Linux Directories - read

| | | |
|---|---|---|
| 1 | To Read File | `cat <filename>` |
| 2 | To Read Top 5 lines | `head -5 <filename>` |
| 3 | To Read bottom 6 lined | `tail -6 <filename>` |
| 4 | To Read File with line number | `cat -n <filename>` |
| 5 | Copy content from old file to new | `Cat <oldfile > > <newfile>` |
| 6 | Add multiple files data into one | `cat <file1> <file2 >... > <newfilename>` |
| 7 | Reverse line reading | `tac <filename>` |
| 8 | Read sufficient in screen size | `more <filename>` |

| | | |
|---|---|---|
| 9 | Read with search | `less <filename> ,next /<enter text to search>` |
| 10 | Word count | `wc <filename>` |

## Linux Directories

| | | |
|---|---|---|
| 1 | Creating Directory/Folder | `mkdir <folder-name>` |
| 2 | Removing Directory | `rmdir <directory/folder name>` |
| 3 | To know present directory | `pwd` |
| 4 | Moving directory / Cut | `mv <old-dir> <new-dir>` |
| 5 | Copy directory / Copy | `Cp -R <dir> <new-dir-location>` |
| 6 | To goto home directory | `cd` |
| 7 | To goto particular directory | `cd </directory/subdir-directory-location>` |
| 8 | To go backward directory | `cd ..` |
| 9 | Create directory within another dir | `mkdir -p <dir1/dir2/dir3> (inside dir3 dir will be created)` |
| 10 | Remove Directory inside directories | `Rmdir -R <directory name>` |
| 11 | To see hidden directories | `dir -a` |
| 12 | To see which user created dir | `dir -l --author` |

## Linux Directories - Listing

| | | |
|---|---|---|
| 1 | Listing directories | `ls` |
| 2 | Long listing | `ll` |
| 3 | Listing with hidden/all | `ls -a` |
| 4 | Long with hidden file | `ll -a` |
| 5 | Listing with human readable(memory in kb,mb,gb) | `ll -lh` |
| 6 | Listing with human readable(memory in MB),G for GB,T for TB | `ls -l --block-size=M` |
| 7 | Listing with index | `ls -li` |
| 8 | Listing with reverse order of name | `ls -llr` |
| 9 | Listing directories along with directories | `ls -R` |

## Linux listing directory explanation

```
-rw-rw-r--. 1 opc opc    0 Sep  5 02:31 file2
```

| Filetype + users permission | Index/links | owner | user | size | date | time | File name |
|---|---|---|---|---|---|---|---|
| - rw-rw-r– | 1 | opc | opc | 0 | Sep 5 | 02:31 | file2 |

|  |  |  |  |
|---|---|---|---|
| File Types ( first letter ) | - : file | d : directory | l: link |
| File Permissions | r → read | w → write | e → execute |
| rw- rw- r– ( - means no permission) | first three for owner | next : group | last : other users |

## Linux Users

|  |  |  |
|---|---|---|
| 1 | Current logged in username | `whoami` |
| 2 | All users logged in to system | `who` |
| 3 | To find user id | `id` |
| 4 | To see logged in users along with from where they logged in | `w` |
| 5 | Switch to another user, you need to know password for user | `su <username>` |
| 6 | Switch to another user with his home directory | `su - <username>` |
| 7 | To list users of environment(machine) : cat /etc/passwd<br>`Tecmint :x:1000:1000:tecmint:/home/tecmint:/bin/bash`<br><username> :<encrypted password> : <uid>:<user’s GID>:<Comment>:<home dir >: <users shell> |  |

## Linux Users - Create/Delete/Modify

|  |  |  |
|---|---|---|
| 1 | **Create user** - w/o password login ***as root user*** | `useradd <username>` |
| 2 | Create user - w/o password - as ***normal user with sudo*** | `sudo useradd <username>` |
| 3 | To add password to user ***as root use***r | `passwd <username>` |
| 4 | To add password to user ***as normal user - with - sudo*** | `sudo passwd <username>` |
| 5 | Create User with uid ( select 4 digit ) | `useradd -u <uid> <username>` |
| 6 | Create user with groupid | `useradd -g <group id> <username>` |
| 7 | Create user with comment | `Useradd -c <comment> <username>` |
| 8 | Create user with home directory | `useradd -d <directory> <username>` |
| 9 | Create user with different shell | `useradd -s <bin/sh> <username>` |
| 10 | Create User with all details :<br>`useradd -u 1451 -g 1000 -c ggluser -d /home/ggluser -s /bin/sh ggluser` |  |
| 11 | Create user with expire date | `useradd -e 2022-10-10 <username>` |
| 12 | Delete user | `userdel <username>` |
| 13 | Delete user with home directory | `userdel -r <username>` |
| 14 | Default assigned values for new user to be created | `useradd -D` |
| 15 | Modifying the existing user | `Usermod -c <comment> <username>` |
| 16 | Modify the user home directory with data | `usermod -d <dir> -m <username>` |

| | | |
|---|---|---|
| 17 | Modify the user id and groupid | `usermod -u <uid> -g <gid> <username>` |
| 18 | Modify username | `usermod -l <new name> <old name>` |
| 19 | Modify normal user to root user(sudo group :wheel) | `usermod -aG wheel <username>` |
| 20 | To add group with groupid | `groupadd -gid 1458 <group name>` |
| 21 | To verify that group created | `getent group \| grep <group name>` |

## Linux User File Permissions

| Filetype + users permission | Index / links | owner | user | size | date | time | File name |
|---|---|---|---|---|---|---|---|
| - rw-rw-r– | 1 | opc | opc | 0 | Sep 5 | 02:31 | file2 |

| | | | |
|---|---|---|---|
| File Types ( first letter ) | - : file | d : directory | l: link |
| **File Permissions** | **r → read (r)** | **w → write (w)** | **e → execute (x)** |
| **rw- rw- r– ( - means no permission)** | **first three for owner/admin (a)** | **next : group (g)** | **last : others (o)** |

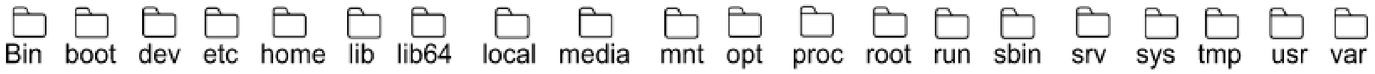
| | | |
|---|---|---|
| 1 | Add - Owner(a) + execution(x) to file | `chmod a+x <filename>` |
| 2 | Add - group(g) + execution(x) to file | `chmod g+x <filename>` |
| 3 | Add - others(o) + execution(x) to file | `chmod o+x <filename>` |
| 4 | Add owner,group,other + exe(x) to file | `chmod o+x,g+x,o+x <filename>` |
| 5 | Add all permission to all | `chmod o+rwx,g+rwx,o+rwx <filename>` |

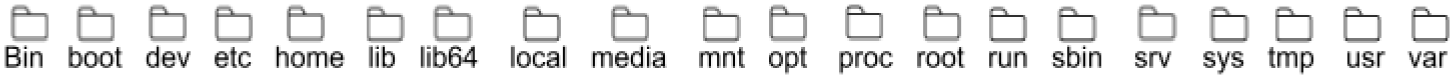
| | | | |
|---|---|---|---|
| **File Permissions** | **r → read (4)** | **w → write (2)** | **e → execute (1)** |
| **rw- rw- r--(-means no permission)** | **rw- →owner : r+w+e=4+2+0=6** | **rw- →group : r+w+e=4+2+0=6** | **r- - →others : r+w+e=4+0+0=4** |

| | | |
|---|---|---|
| 1 | Add Permissions a→rwx, g→rw, o→r ⇒ **rwxrw-r--** | `chmod 764 <filename>` |
| 2 | Add Permissions a→rwx, g→rw, o→r ⇒ **rwxr--r--** | `chmod 744 <filename>` |
| 3 | change **--xr--r-** ⇒ **rwxr-xr--** | `chmod 754 <filename>` |

## Linux Filesystem Hierarchy Standard (FHS)

All the directories in the Linux system come under the root(/) directory which is represented by a forward slash (/).





|  | **Directory type** | **Types of files stored** |
|---|---|---|
| 1 | Binary directories | Contains binary or compiled source code files, eg, `/bin, /sbin, etc.` |
| 2 | Configuration directories | Contains configuration files of the system, eg, `/etc, /boot`. |
| 3 | Data directories | Stores data files, eg, `/home, /root,/media, /mnt, /tmp` etc. |
| 4 | Memory directories | files which don't take up actual harddisk space, eg, `/dev, /proc, /sys.` |
| 5 | Usr (Unix System Resources) | Contains sharable, read only data, eg, `/usr/bin, /usr/lib`, etc. |
| 6 | var (variable directory) | Contains larger size data, eg, `/var/log, /var/cache`, etc. |
| 7 | Non-standard directories | Dir. which do not come under FHS, eg, `lost+found`, `/run`, `/cdrom` etc. |

## Linux - VI Editor

VI=VIsual Editor , it's text editor in UNIX/Linux environment. It has two modes, ***(i) Command Mode :*** based on commands you can edit text , eg: pressing **dd** → will delete the entire one line. ***(ii) Insert Mode :*** whenever you enter vi (vi <filename> , by default its command mode, by pressing I you can shift into Insert mode, here you can type , copy, paste text.

|  |  |  |
|---|---|---|
| 1 | Enter into Insert mode | `I` |
| 2 | Enter back to command mode | `esc` |
| 3 | Move down line | `j` |
| 4 | Move up line | `k` |
| 5 | Move left | `h` |
| 6 | Move Right | `l` |
| 7 | Go to last line | `G` |
| 8 | Last Position | ` `` ` |
| 9 | Delete current character at cursor | `x` |
| 10 | Delete character before cursor | `X` |
| 11 | Replace current character | `r` |
| 12 | Delete current line | `dd` |
| 13 | Delete the current line from current character | `D` |
| 14 | Delete current line to end of the file | `dG` |
| 15 | undo | `u` |
| 16 | Repeat last command | `.` |
| 17 | Copy a line | `yy` |
| 18 | Paste after current line | `p` |

|  |  |  |
|---|---|---|
| 19 | Paste before current line | P |
| 20 | Join two lines | j |
| 21 | Repeat the current line | yyp |
| 22 | Move to next word | w |
| 23 | Move to back word | b |
| 24 | Delete one word | dw |
| 25 | Copy one word | yw |
| 26 | Search | / |
| 27 | Replace all (s: string, g:globally) | :<startLine,endLine> s/<oldString>/<newString>/g → :1,$ s/one/two/ |

## Linux - Archiving

File archiving is used when one or more files need to be transmitted or stored as efficiently as possible.
**Archiving:** Combines multiple files into one, which eliminates the overhead in individual files and makes the files easier to transmit.
**Compression:** Makes the files smaller by removing redundant information. Its two types : lossless, lossy

> **Lossless:** No info is removed from the file. Compressing a file and decompressing= identical to the original.

> **Lossy** **:** Info might be removed from the file. uncompressing a file = slightly different from the

original.
When an archive is decompressed, and one or more files are extracted, this is called **un-archiving**.

|  |  |  |  |
|---|---|---|---|
| 1 | zip | Archive=combines files and keeps original files as it is |  |
| 2 | gzip | Compress files along archive**(only files)** by replacing original files | Lempel-Ziv data compression |
| 3 | tar | To compress a directory |  |
| 3 | bzip2 | Compress files+archive by replacing files, gen2 of gz(good compr.) | Burrows-Wheeler block sorting |

**zip:** files+folders - combining - no compress

|  |  |  |  |
|---|---|---|---|
| 1 | Zip file/folder | zip <zip file name> <folder/file needs to zip> | zip one.zip /home/user/dir |
| 2 | Unzip file/folder | Unzip <zip file name> | unzip <zipfile.zip> |
| 3 | Remove file from zip | Zip -d <zipfile.zip> <files/folders to zip> | zip -d hii.zip hii2 |
| 4 | Add file zipped file | zip -u <zipfile.zip> <files/folders to zip> | zip -u hii.zip hii2-add |
| 5 | Delete original after zip | zip -m <zipfile.zip> <files/folders to zip> | Zip -m one.zip /home/user/one/ |
| 6 | Recursive zip(inner folder) | zip -r <zipfile.zip> <files/folders to zip> | zip -r one.zip /home/user/one |
| 7 | Exclude file | zip -x <zipfile.zip> <files/folders to exclude> | zip -m one.zip one.txt |
| 8 | Verbose (info as response) | Zip -v <zipfile.zip> <files/folders to zip> | zip -v one.zip one.txt |

**gzip (GNU (GNU'S Not Unix) zip) :** files - compression at low size

|   |   |   |   |
|---|---|---|---|
| 1 | compress file | gzip | `gzip one.zip` |
| 2 | Uncompress files | gunzip <file><br>gunzip -d <File> | `gunzip one.zip.gz`<br>`gzip -d one.zip.gz` |
| 3 | To see compress ratio | gzip -l <file> | `gzip -l onetwo.zip` |

**bzip2 (bunzip)** : compress a file like gzip command but takes a little more time but compresses better.

|   |   |   |   |
|---|---|---|---|
| 1 | compress file | bzip2 <files> | `bzip all.zip` |
| 2 | Uncompress files | bzip2 -d <files> | `bzip -x all.zip.bz2` |

**Tar(Tap Archive):** combine+compress → files and folder → again you can add .gz and bz2 compression

|   |   |   |   |   |
|---|---|---|---|---|
| c : create | v : display output | f : mentioned destination dir. to store | z : compress with gz | j : compress with bzip2 |

|   |   |   |   |
|---|---|---|---|
| 1 | compress file/folder | tar -cvf <filename.tar> <files/folders> | `tar -cvf all.tar *` |
| 2 | compress file/folder **gz** | tar -cvfz <filename.tar> <files/folders> | `tar -cvfz all.tar *` |
| 3 | compress file/folder **bzip2** | tar -cvfj <filename.tar> <files/folders> | `tar -cvfj all.tar *` |
| 4 | To list the files in | tar -tvf <filename.tar> | `tar -tvf all.tar` |
| 5 | Extract the tar | tar -xvf <filename.tar> | `tar -xvf all.tar` |

## Linux I/O Redirection

Redirection can be defined as changing the way from where commands read input to where commands send output. You can redirect input and output of a command. Using characters like > , < , >> , |

**standard input (stdin < ) :** numbered as stdin (0). The bash shell takes input from stdin. By default keyboard as i/p.

**standard output (stdout** > **) :**numbered as stdout (1). The bash shell sends output to stdout. Output goes to display.

**standard error (stderr) :** numbered as stderr (2). The bash shell sends an error message to stderr - goes to display

|   |   |   |
|---|---|---|
| 1 | > | `echo "hii" > hii.txt` |
| 2 | < | `run.sh < new-commands.txt` |
| 3 | >> | Network troubleshooting utility. |

## Linux - Networking Commands

|   |   |   |
|---|---|---|
| 1 | ifconfig | Display and manipulate route and network interfaces. |
| 2 | ip | It is a replacement of the ifconfig command. |
| 3 | traceroute | Network troubleshooting utility. |
| 4 | tracepath | Similar to traceroute but doesn't require root privileges. |
| 5 | ping | To check connectivity between two nodes. |
| 6 | netstat | Display connection information. |
| 7 | ss | It is a replacement for netstat. |

| | | |
|---|---|---|
| 8 | dig | Query DNS related information. |
| 9 | nslookup | Find DNS related query. |
| 10 | route | Shows and manipulates IP routing table. |
| 11 | host | Performs DNS lookups. |
| 12 | arp | View or add contents of the kernel's ARP table. |
| 13 | iwconfig | Used to configure wireless network interface. |
| 14 | hostname | To identify a network name. |
| 15 | curl or wget | To download a file from the internet. |
| 16 | mtr | Combines ping and tracepath into a single command. |
| 17 | whois | Will tell you about the website's whois. |
| 18 | ifplugstatus | Tells whether a cable is plugged in or not. |

## Linux - man

The "man" is a short term for a manual page. In unix-like operating systems such as linux, man is an interface to view the system's reference manual.

| | | |
|---|---|---|
| 1 | `man <command>` | Eg: man passwd → to view manual related to man command |
| 2 | `whereis <command>` | To see where is manual related to than command |

## Linux - System Administration Commands

| | | |
|---|---|---|
| 1 | `uptime` | Shows how long system is on and running |
| 2 | `users` | Shows users - current logged in |
| 3 | `free -m/g` | shows storage details in m:mb , g:gb |
| 4 | `last` | Last users activity |
| 5 | `history` | Command history |
| 6 | `ping` | Checking the ping |
| 7 | `ifconfig` | Network information |
| 8 | `traceroute <domain>` | Tracing the domain server path |
| 9 | `nslookup` | Finding the server ip |
| 10 | `wget` | To download via link |
| 11 | `ps` | List the process of current user |
| 12 | `top` | Processor activity of entire system including root activities |
| 13 | `last` | Last user activity |
| 14 | `kill` | To kill the process |

|  |  |  |
|---|---|---|
| 15 | `jobs` | To see running jobs , ex : starting sleep job : sleep 1000 & , & id run in bg |

## Linux - processes

To view currently running processes on the system with the ps command. It mostly return PID(Process ID), TTY(type of terminal ), TIME(process cpu use time),CMD(command used)

|  |  |  |
|---|---|---|
| 1 | `ps -ef` | List currently running process in full format |
| 2 | `ps -ax` | List current running process |
| 3 | `ps -u <username>` | Listing the processes related to specific user |
| 4 | `ps -C <command>` | Listing the process related to specific command |
| 5 | `ps -p <pid>` | Listing the process with pid |
| 6 | `ps -ppid <ppid>` | Listing with parent process id related process |
| 7 | `pstree` | Shows processes in hierarchy |
| 8 | `ps -L` | List all threads for a particular process |

**Linux Terminating :** There are four ways to kill or terminate a process. commands allow you to run the system uninterruptedly after terminating a process without rebooting the system. commands can be internal or external.

|  |  |  |
|---|---|---|
| 1 | `kill` | To terminate a process is a `kill` command. You need to know the PID of the process |
| 2 | `killall` | It kills all the processes with the specified name in the system. |
| 3 | `pkill` | The pkill command uses the name of the process either by typing full name or partial name. |
| 4 | `xkill` | Command xkill is used to kill a process on X server without passing process name or PID |

## Linux - processes kill

**To kill a process use command, extend with different signal numbers, command :** `kill -SIGNAL PID`

**Process signal names and numbers:** process kill methods mentioned below,to see all signals run : **`kill -l`**

| **Signal Names mostly used** | **Signal Number** | **Signal Use** |
|---|---|---|
| SIGNULL | 0 | NULL, check access to PID |
| SIGHUP | 1 | Hangup |
| SIGINT | 2 | Interrupt |
| SIGQUIT | 3 | Quit |
| SIGKILL | 9 | Kill |
| SIGTERM | 15 | Terminate |
| SIGSTOP | 24 | Stop |
| SIGTSTP | 25 | Stop/pause the process |
| SIGCONT | 26 | Continue a stopped process |

|  |  |  |  |
|---|---|---|---|
| 1 | `kill -SIGNAL PID` | `Kill -9 1002` | Command to kill process |

|  |  |  |  |
|---|---|---|---|
| 2 | `kill -SIGNAL PID PID` | `Kill -9 1002 1003` | Command to kill multi process |

**Regular/Real time we mostly check real time activities and memory usage using `top` command,**

```
top - 03:43:03 up 1 day,  2:00,  3 users,  load average: 0.00, 0.00, 0.00
Tasks:  94 total,   1 running,  56 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.0 hi,  0.0 si,
0.3 st
KiB Mem :  1005836 total,   701520 free,    94628 used,   209688 buff/cache
KiB Swap:        0 total,        0 free,        0 used.   767308 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND

    1 root      20   0  123484   5392   3852 S  0.0  0.5   0:03.75 systemd
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd
    4 root       0 -20       0      0      0 I  0.0  0.0   0:00.00
kworker/0:0H
    5 root      20   0       0      0      0 I  0.0  0.0   0:00.39
kworker/u30:0
```

## Linux - filters

Linux Filter commands accept input data from stdin (standard input) and produce output on stdout (standard output). It transforms plain-text data into a meaningful way and can be used with **pipes** to perform higher operations.

|  |  |  |
|---|---|---|
| 1 | `grep` | global regular expression print → filters the content of a file which makes our search easy.<br>E.g : cat /etc/passwd \| grep root or ls -ltr \| grep videos |
| 2 | `sort` | sort <fileName> , eg : sort /etc/passwd |

## Linux - wget(web get)

The wget is a free non-interactive file downloader command. Non-interactive means it can work in the background when the user is not logged in. It supports HTTP, HTTPs ,FTP protocol. Not avail install:`sudo apt-get install wget`

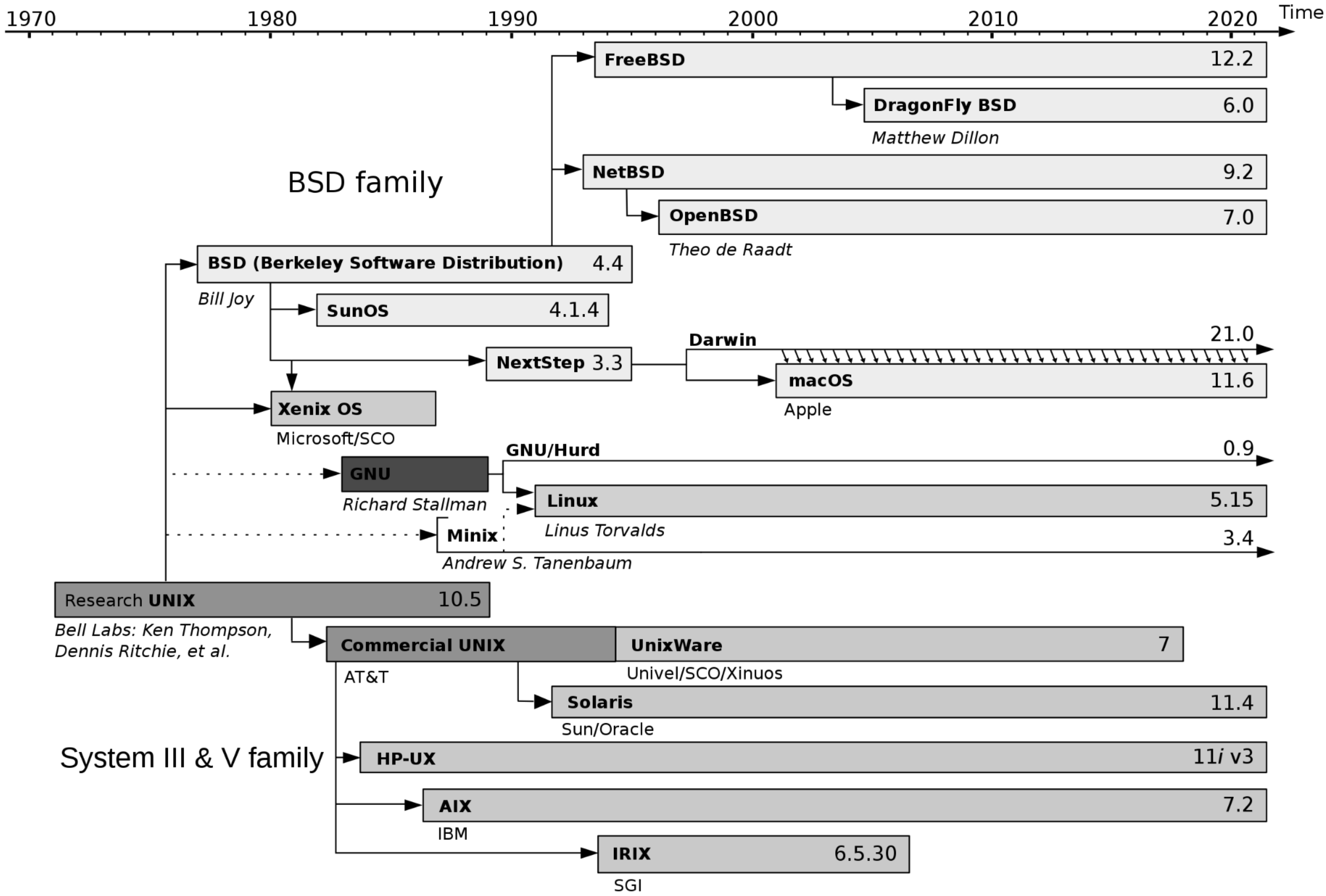
|  |  |  |
|---|---|---|
| 1 | `wget <URL>` | download file |
| 2 | `wget -O <fileName> <URL>` | Downloading file with altering local name to save |
| 3 | `Wget --limit-rate=<150k/speed> <URL>` | To limit the speed |
| 4 | `wget -i <fileName>` | Save links in text file and use that command |
| 5 | `wget -b <URL>` | Download in background |
| 6 | `wget -c <URL>` | Complete the remaining downloaded file s |
| 7 | `wget --tries=<Number> <URL>` | Setting retry times |

## Linux - Distributions

Linux Developed by many developers , associations and companies , all of them shared and developed parts of linux. And linux open source modifying source companies releasing their distribution.

1) **RPM-based :** Fedora, RedHat Linux Enterprise, CentOS, OpenSUSE, package: rpm → yum
2) **Debian-based :** Ubuntu, Kali linux, parrot os , package : dpkg → apt-get or apt
3) **Arch Linux-based :** Arch linux, Garuda Linux , package: pacman → pacman

## Linux - History



In 1969, a team of developers of Bell Labs started a project to make a common software for all the computers and named it 'Unix'. It was simple and elegant, used 'C' language instead of assembly language and its code was recyclable.. As it was recyclable, a part of its code now commonly called 'kernel' was used to develop the os.

In 1991, Linus Torvalds was a student at the university of Helsinki, Finland. Free version of Unix started writing its own code. Later this project became the Linux kernel.released as GNU/Linux in 1992. It is an open source so many companies made their modifications and released their distributions. You can see distributions above.

## Linux - package/software installation

| | | |
|---|---|---|
| 1 | RPM Package Manager: by RHEL<br>OS : RHEL,centos,fedora,amazon linux<br>YUM : Yellowdog Updater, Modified | `yum install <package name>`<br>`E.g : Yum install tree`<br>RHEL : `rpm install <package-name>` |
| 2 | Debian-based : by The Debian Project<br>OS : Ubuntu, Kali linux, parrot os<br>APT : Advanced Packaging Tool | `apt-get install <package name>`<br>`E.g : apt install httpd` |

| | | |
|---|---|---|
| 3 | Arch based :<br>OS : Arch linux, Garuda Linux | `pacman -S <package-name>` |

## Linux - cronjob

To Build Job Daily or Time basic you can user **cron job**

Cron Jobs has 5 values :

| Minutes | Hour | Day of Month | Month | Dow |
|---|---|---|---|---|
| 0-59 | 0-23 | 1-31 | 1-12 | 0-6 |

**Task - 1: If you want to run a job every day at 10:00PM, then write cron job**

Answer : 0 22 * * *

| Minutes | Hour | Day of Month | Month | Dow |
|---|---|---|---|---|
| 0 | 22 | * | * | * |

**Task - 2 : If you want to run a job every day at 11:00 PM From Monday to Friday only, then write cron job**

Answer : 0 23 * * 1-5

| Minutes | Hour | Day of Month | Month | Dow |
|---|---|---|---|---|
| 0 | 23 | * | * | 1-5 |

**Examples of Cron jobs 1:**

**Scheduling a Job For a Specific Time The basic usage of cron is to execute a job in a specific time as shown below. This will execute the Full backup shell script (full-backup) on 10th June 08:30 AM. The time field uses 24 hours format. So, for 8 AM use 8, and for 8 PM use 20.**

```
30 08 10 06 * /home/maverick/full-backup
```

**1. Elements of linux /system architecture :** Kernel, system library,system utility, hardware, shell and user

**2. kernel:** it establishes communication between user application and hardware. It stays in b/w app and hardware. Main Functions are Memory, Storage, Device, Process management. **Kernel types:**

| ***Kernel*** | ***Monolithic*** | ***Micro*** | ***Hybrid*** |
|---|---|---|---|
| work | services operate in kernel space | Less services@kernel, left in users | Combo of monolithic and micro |
| advtgs | Good Performance | Good stabilityn | Good performance+stability |
| disadv. | Dependencies more\ | System calls are more | Dependencies more |
| os | Linux,unix and android | Blackberry, symbian, java os | Windows , MAC |

**3. Shell :** it is a terminal environment ,here we can run our commands, it is an interface between user and kernel.
Types : 1) bourne shell (sh), 2) bash(Bourne-Again Shell) - bash, 3) c shell(csh), 4) korn shell(ksh), 5) z shell(zsh)

**4.shell scripting :** it is like programing language, here we write shell script(commands, conditions and more) in file and execute, it will perform actions and process , eg: *`#1/bin/bash…….yum install httpd`*

**4.LILO (Linux Loader) :** used to load Linux into memory and start the OS, simply its a boot loader. functions such as locating kernel, identifying other supporting programs, loading memory, and starting the kernel

**5.Modes of vi editor :** i) Command mode (press ESC), (ii) Insert mode(press I), (iii) Execute mode(press :)

**6.Swap memory :** Sharing the storage memory to RAM, when the RAM is used full, to handle resources/processes

**7. Daemons :** background linux system process , that starts when os starts and ends at shutdown , no control over it. Examples : httpd, sshd, nfsd

**8. Linux user mode :** (i). CLI(Command Line Interface), (ii) GUI(Graphical User Interface)

**9. File permissions in linux :** (i) r - Read , (ii) w - Write , (iii). e - Execution

**10. Cron :** it is program used to execute tasks at scheduled time, min time is minutes, used in servers , runs 24-7

**11. Anacron :** execute tasks in intervals/days – mintime days – not a demon, used in Desktops and laptops, eg: updates

12. Process ID :

.