UNIT – II

**Introduction to Linux OS:** OS basics, Linux GUI: Exploring folders, Installation of binary packages, Built in Package Managers, Introduction to Linux file system, man pages, The first command cat, Command History, Basic Unix Commands: vi editor, Redirection operators, some Unix commands.

**Introduction to Linux OS**

It was created in October 1991 by a University of Helsinki student named Linus Torvalds (Linux stands for Linus’s UNIX). Linux itself is actually just the kernel; it Implements multitasking and multiuser functionality, manages hardware, allocates Memory, and enables applications to run. A very popular open source operating system that runs on a verity of hardware platforms. Linux is widely deployed as a server OS. Linux is multi-tasking, multi user operating system. Although modified by numerous people.

Linux is a UNIX-based operating system originally developed as for Intel-compatible PC's. It is now available for most types of hardware platforms, ranging from PDAs (and according to some reports, a wristwatch) to mainframes. Linux is a "modern operating system", meaning it has such features as virtual memory, memory protection, and preemptive multitasking.

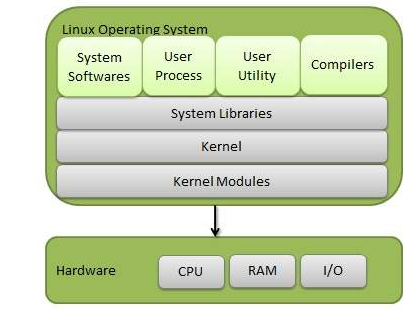
**Why use Linux?**  
**Reasons to Install Linux**  
**Components of Linux System**

Linux Operating System has primarily three components

**Kernel** − Kernel is the core part of Linux. It is responsible for all major activities of this operating system. It consists of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.

**System Library** − System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries implement most of the functionalities of the operating system and do not require kernel module's code access rights.

**System Utility** − System Utility programs are responsible to do specialized, individual level tasks.



**Kernel Mode vs User Mode**

Kernel component code executes in a special privileged mode called **kernel mode** with full access to all resources of the computer. This code represents a single process, executes in single address space and do not require any context switch and hence is very efficient and fast. Kernel runs each processes and provides system services to processes, provides protected access to hardware to processes.

Support code which is not required to run in kernel mode is in System Library. **User programs and other system programs works in** **User Mode** which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low level tasks.

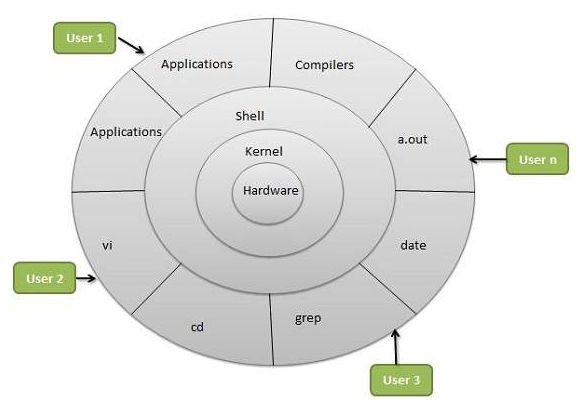
## Basic Features

Following are some of the important features of Linux Operating System.

* **Portable** − Portability means software can works on different types of hardware in same way. Linux kernel and application programs supports their installation on any kind of hardware platform.
* **Open Source** − Linux source code is freely available and it is community based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
* **Multi-User** − Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
* **Multiprogramming** − Linux is a multiprogramming system means multiple applications can run at same time.
* **Hierarchical File System** − Linux provides a standard file structure in which system files/ user files are arranged.
* **Shell** − Linux provides a special interpreter program which can be used to execute commands of the operating system. It can be used to do various types of operations, call application programs. etc.
* **Security** − Linux provides user security using authentication features like password protection/ controlled access to specific files/ encryption of data.

## Architecture

The following illustration shows the architecture of a Linux system −



The architecture of a Linux System consists of the following layers −

* **Hardware layer** − Hardware consists of all peripheral devices (RAM/ HDD/ CPU etc).
* **Kernel** − It is the core component of Operating System, interacts directly with hardware, provides low level services to upper layer components.
* **Shell** − An interface to kernel, hiding complexity of kernel's functions from users. The shell takes commands from the user and executes kernel's functions.
* **Utilities** − Utility programs that provide the user most of the functionalities of an operating systems.

**Advantages of Linux**

**Low cost:**

You don’t need to spend time and money to obtain licenses since Linux and much of its software come with the GNU General Public License. You can start to work immediately without worrying that your software may stop working anytime because the free trial version expires. Additionally, there are large repositories from which you can freely download high quality software for almost any task you can think of.

**Stability:**

Linux doesn’t need to be rebooted periodically to maintain performance levels. It doesn’t freeze up or slow down over time due to memory leaks and such. Continuous up-times of hundreds of days (up to a year or more) are not uncommon.

**Performance:**

Linux provides persistent high performance on workstations and on networks. It can handle unusually large numbers of users simultaneously, and can make old computers sufficiently responsive to be useful again.

**Network friendliness:**

Linux was developed by a group of programmers over the Internet and has therefore strong support for network functionality; client and server systems can be easily set up on any computer running Linux. It can perform tasks such as network backups faster and more reliably than alternative systems.

**Flexibility:**

Linux can be used for high performance server applications, desktop applications, and embedded systems. You can save disk space by only installing the components needed for a particular use. You can restrict the use of specific computers by installing for example only selected office applications instead of the whole suite.

**Compatibility:**

It runs all common Unix software packages and can process all common file formats.

**Choice**:

The large number of Linux distributions gives you a choice. Each distribution is developed and supported by a different organization. You can pick the one you like best; the core functionalities are the same; most software runs on most distributions. Fast and easy installation: Most Linux distributions come with user-friendly installation and setup programs. Popular Linux distributions come with tools that make installation of additional software very user friendly as well.

**Full use of hard disk**:

Linux continues work well even when the hard disk is almost full.

**Multitasking:**

Linux is designed to do many things at the same time; e.g., a large printing job in the background won’t slow down your other work.

**Security:**

Linux is one of the most secure operating systems. “Walls” and flexible file access permission systems prevent access by unwanted visitors or viruses. Linux users have to option to select and safely download software, free of charge, from online repositories containing thousands of high quality packages. No purchase transactions requiring credit card numbers or other sensitive personal information are necessary.

**Open Source:**

If you develop software that requires knowledge or modification of the operating system code, Linux’s source code is at your fingertips. Most Linux applications are Open Source as well.

**Disadvantages of Linux**

1. There’s no standard edition of Linux. Whereas Microsoft offers several different editions of each version of Windows, there are countless variations of Linux. For a new user it can be confusing to work out which is best for you.

2. Linux has patchier support for drivers (the software which coordinates your hardware and your operating system). This means you’ll sometimes find it trickier to get a new device set up.

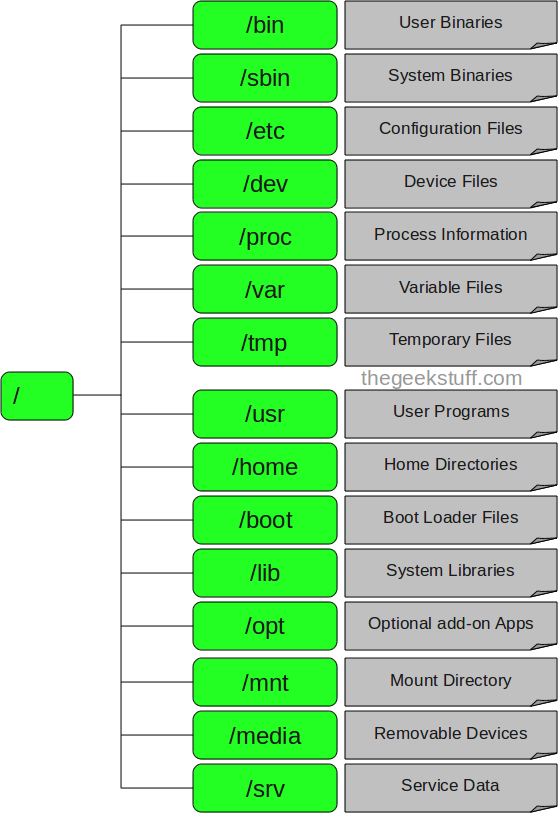
3. Linux is, for new users at least, not as easy to use as Windows. That’s largely because Linux gives you more control, but does mean you’ll have to spend some time getting used to the way it works.

4. Because Linux is neither as popular as Windows, nor a commercial product, support works in a different way. You may have to look harder to find the answer to a problem and, while Linux supporters are more likely to offer help, it may not always match your own level of technical understanding.

5. Many of the programs you are used to in Windows will only run in Linux through a complicated emulator. These programs aren’t guaranteed to work perfectly, and in some cases may be noticeably slower.  
6. While Linux can be suitable for an individual user, its small market share means it’s much harder to introduce in a corporate setting. With most office workers already familiar with Windows and Microsoft programs, there’ll likely be a notable time cost in converting staff to using a Linux system.

7. While there are perfectly passable alternatives to many popular Windows programs (such as the various Office components), some high-end applications such as Photoshop don’t have as close equivalents in Linux.

**🡪Linux GUI: Exploring folders**

Let us review the Linux file system structures and understand the meaning of individual high-leveldirectories.  


### 1. / – Root

* Every single file and directory starts from the root directory.
* Only root user has write privilege under this directory.
* Please note that /root is root user’s home directory, which is not same as /.

### 2. /bin – User Binaries

* Contains binary executables.
* Common linux commands you need to use in single-user modes are located under this directory.
* Commands used by all the users of the system are located here.
* For example: ps, ls, ping, grep, cp.

### 3. /sbin – System Binaries

* Just like /bin, /sbin also contains binary executables.
* But, the linux commands located under this directory are used typically by system aministrator, for system maintenance purpose.
* For example: iptables(iptables(**Iptables** is used to set up, maintain, and inspect the tables of IPv4 packet filter rules in the Linux kernel), reboot, fdisk, ifconfig, swapon(swapon is used to specify devices on which paging and swapping are to take place.)

### 4. /etc – Configuration Files

* Contains configuration files required by all programs.
* This also contains startup and shutdown shell scripts used to start/stop individual programs.
* For example: /etc/resolv.conf, /etc/logrotate.conf

### 5. /dev – Device Files

* Contains device files.
* These include terminal devices, usb, or any device attached to the system.
* For example: /dev/tty1, /dev/usbmon0

### 6. /proc – Process Information

* Contains information about system process.
* This is a pseudo filesystem contains information about running process. For example: /proc/{pid} directory contains information about the process with that particular pid.
* This is a virtual filesystem with text information about system resources. For example: /proc/uptime

### 7. /var – Variable Files

* var stands for variable files.
* Content of the files that are expected to grow can be found under this directory.
* This includes — system log files (/var/log); packages and database files (/var/lib); emails (/var/mail); print queues (/var/spool); lock files (/var/lock); temp files needed across reboots (/var/tmp);

### 8. /tmp – Temporary Files

* Directory that contains temporary files created by system and users.
* Files under this directory are deleted when system is rebooted.

### 9. /usr – User Programs

* Contains binaries, libraries, documentation, and source-code for second level programs.
* /usr/bin contains binary files for user programs. If you can’t find a user binary under /bin, look under /usr/bin. For example: at, awk, cc, less, scp
* /usr/sbin contains binary files for system administrators. If you can’t find a system binary under /sbin, look under /usr/sbin. For example: atd, cron, sshd, useradd, userdel
* /usr/lib contains libraries for /usr/bin and /usr/sbin
* /usr/local contains users programs that you install from source. For example, when you install apache from source, it goes under /usr/local/apache2

### 10. /home – Home Directories

* Home directories for all users to store their personal files.
* For example: /home/john, /home/nikita

### 11. /boot – Boot Loader Files

* Contains boot loader related files.
* Kernel initrd, vmlinux, grub files are located under /boot
* For example: initrd.img-2.6.32-24-generic, vmlinuz-2.6.32-24-generic

### 12. /lib – System Libraries

* Contains library files that supports the binaries located under /bin and /sbin
* Library filenames are either ld\* or lib\*.so.\*
* For example: ld-2.11.1.so, libncurses.so.5.7

### 13. /opt – Optional add-on Applications

* opt stands for optional.
* Contains add-on applications from individual vendors.
* add-on applications should be installed under either /opt/ or /opt/ sub-directory.

### 14. /mnt – Mount Directory

* Temporary mount directory where sysadmins can mount filesystems.

### 15. /media – Removable Media Devices

* Temporary mount directory for removable devices.
* For examples, /media/cdrom for CD-ROM; /media/floppy for floppy drives; /media/cdrecorder for CD writer

### 16. /srv – Service Data

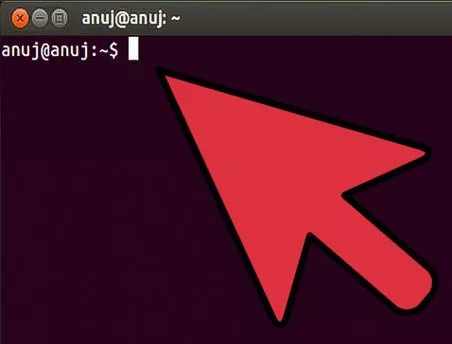
* srv stands for service.
* Contains server specific services related data.
* For example, /srv/cvs contains CVS related data.

**🡪Installation of binary packages:**

**Steps**

****

1. If the bin file is an installer/self-extracting archive, first download the thing and keep it somewhere safe just to avoid having to download it again.

****

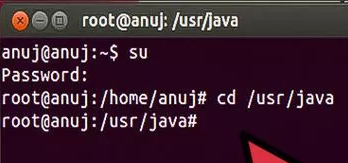
2. Enter the terminal.

****

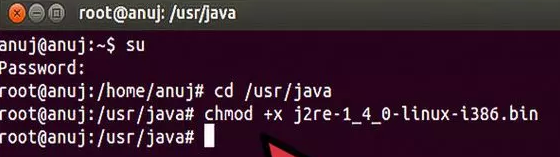
3. Enter Root mode, like so : su - (the hyphen is required) and provide the root password.



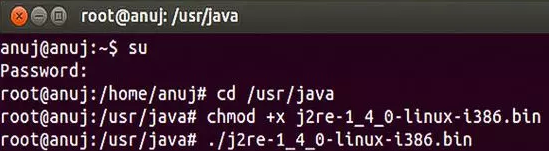
4. If needed copy the bin file to its final output folder - packages like the Java Runtime Environment require this. Read the online instructions first...



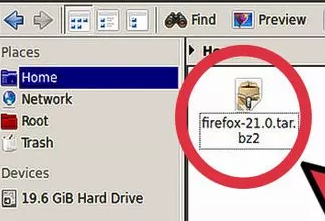
5.Change directories (folders) to the one containing the bin file, like so : cd /topmost/folder, for example cd /usr/share



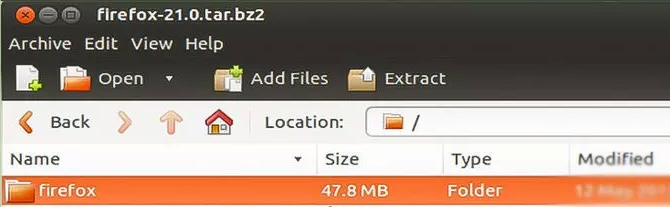
6.Give the bin file execute permissions: chmod +x thefile.bin



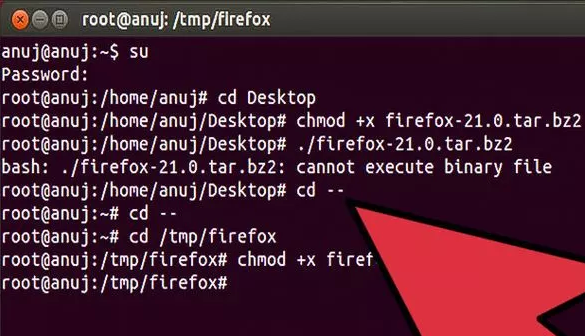
7.Execute it: ./thefile.bin - the dot-slash has to be there



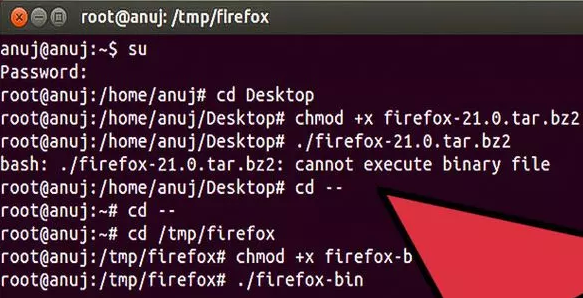
8. If the bin file is the program itself, chances are the file is compressed, untar/unzip in the destination folder, Firefox comes like that.



9.Copy the archive and unpack it in the output folder, that should produce a folder.



10.Enter the folder, locate the program - it is a bin file, give it execute permissions if need be (see step 6).



11.Make a starter for convenience, right-click on the desktop, select the option you need and follow the lead - an icon should appear**.**

### [Or]

### Terminal

First, open the Terminal, then mark the file as executable with thechmod command.

chmod +x file-name.run

Now you can execute the file in the terminal.

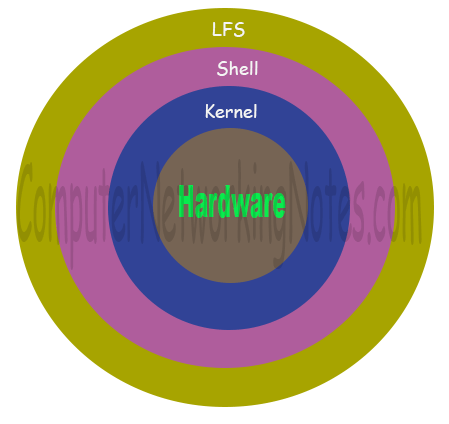
./file-name.run

If an error message including a problem such as 'permission denied' appears, use sudo to run it as root (admin). Be careful, sudo allows you to make critical changes to your system. Many software installs will requiresudo.

sudo ./file-name.run

🡪 **Introduction to Linux file system**

A Linux system is basically divided in three major components: File System (LFS), Shell and Kernel. Kernel is the core program which manages system hardware devices. Shell provides user interface to run the commands. File system organizes the data in systematic way. Collectively LFS, Shell and kernel provides a way to interact with system and an environment to run commands and manage data.



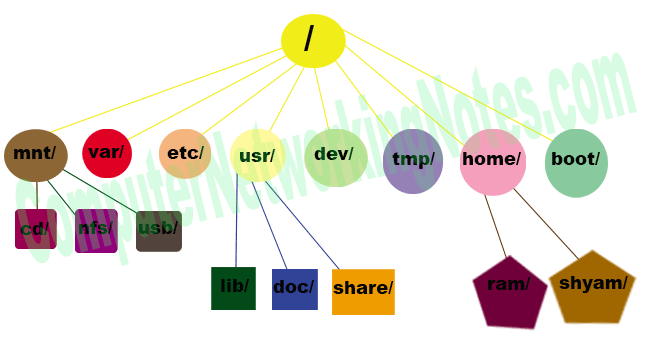
Let’s understand these components in more details one by one.

## Linux File System (LFS)

Linux accesses every object as file. Files are systematically organized in directories. Linux starts file system with root directory (**/**). All files and directories are created and managed under the root directory. Since root directory stands on the top in file system, it has no parent directory. Besides root directory, every directory in Linux has a parent directory. Linux allows us to create as many files and directories as we want. We can create files under the existing directories or may create new directories.

### System Directories

System directories contain files, software, applications and scripts which are required to run and maintain the Linux. System directories are automatically created during the installation. Following figure illustrates some common system directories with their location in LFS.



**Common System directories**

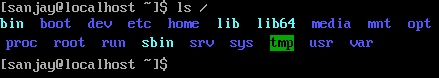
|  |  |
| --- | --- |
| **Directory** | **Description** |
| / | First directory in Linux File System. It is also known as root directory or main directory.  All files and directories are created and managed under this directory. |
| /home | Default directory for user data. Whenever we add a new user, Linux automatically creates a home directory matching with his username in this directory. Whenever user login, Linux starts his login session from home directory. |
| /root | This is the home directory for root user. Root user is the super user in Linux. For security reason Linux creates a separate home directory for root user. Root user account is also being created during the installation automatically. |
| /bin | This directory contains standard commands files. Commands stored in this directory are available for all users and usually do not require any special permission to run. |
| /sbin | This directory contains system administration commands files. Commands stored in this directory are available only for root user and usually requires special privilege to run. |
| /usr | This directory contains user application software files, third party software and scripts, document files and libraries for programming languages. |
| /var | This directory stores variable data files such as printing jobs, mail box etc. |
| /etc | This directory contains system configuration files. |
| /boot | This directory contains Linux boot loader file. |
| /mnt | This directory is used to mount remote file system and temporary devices such as CD, DVD and USB. |
| /dev | This directory contains device files. Usually files in this directory are dynamically generated and should be never edited. |
| /tmp | This directory provides temporary location for applications. |

###### **Exercise -1**

**List all directories from root directory.**

###### **Solution**

**ls / command will list all directories from root directory.**

****

###### **Key points**

1. A Linux file name may have any characters or letters.

2. Maximum length for file name is 256 characters.

3. File name can use space, underscore, minus, period and comma.

4. File name cannot use question marks, asterisks and slashes.

5. File extension is not compulsory. We can create files with file extension or without file extension.

6. To create a hidden file, start its name with dot.

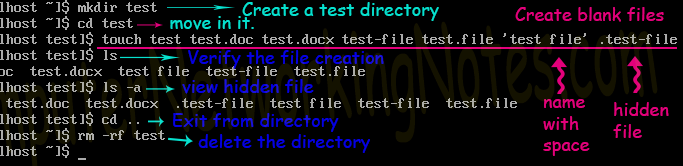
7. If file name has spaces, it need to be quoted before we can access it on command prompt. On desktop we can use it without quotes.

###### **Exercise -2**

Create a test directory and move in it. Create blank files with following names:-

test, test.doc, test.docx, test-file, test\_file, test.file, test file (‘test file’ at command prompt), .test-file (hidden file) Verify the file creation. Verify the hidden file. Exit from directory and delete the directory.

###### **Solution**



1.Create new directory from **mkdir** command.

2. Use **cd** command to move in directory.

3. Use **touch** command to create files. touch command creates blank file. It accept multiple files name. File name would be separated by a space. Shell use space to parse arguments on command line.

4. We can create multiple files with single command by separating file names with space. If a file name has space in it, we have to quote it.

5. To create a hidden file start its name with dot (**.**).

6. Use **ls** command to list the files of current directory.

7. Use **ls** command with **–a** option to list hidden files.

8. Use **cd ..** command to return in parent directory.

9. Use **rm –rf** command to delete the directory with all files.

## Shell

Shell is a command interpreter. It take commands from user, execute them and display the results. Shell supports I/O ( Input / Output) redirection which means it can read commands from non-standard sources such as script files. As well as it can also redirect output to any supportive device (such as printer) or data server.

Several Shells are available in Linux such as Kom, TCSH, Z shell, Bash etc. Although several shells are available, only one shell is set to default in RedHat Linux. **Bash (*Bourne Again shell*)**shell is the default shell in Red-Hat Linux.

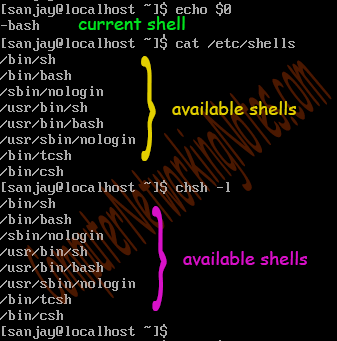
|  |  |
| --- | --- |
| **Command** | **Description** |
| cat /etc/shells | To list the all available shells |
| chsh –l | To list the all available shells |
| echo $0 | To view the current shell name |
| exec /bin/csh | To change current shell temporary. At next login default shell will be restored. |
| chsh –s /bin/csh | To change current shell permanently. Change will be applied at next login.  Default shell will be used in current session. |

###### **Exercise -3**

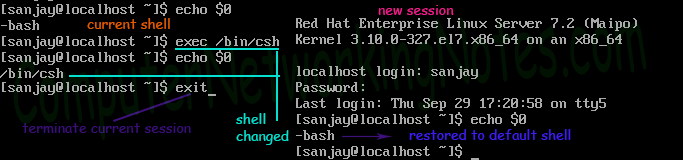
View the current shell and list all available shells. Change current shell temporary for this login session and verify the change. Logout from current session and login again and verify that default shell get restored. Change shell again but this time change it permanently.

###### **Solution**

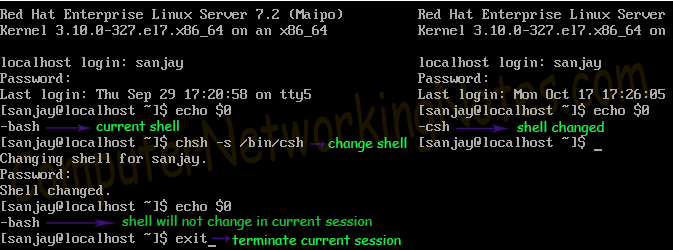
To view the current shell use **echo $0** command. To list all available shells either use **cat/etc/shells** or use **chsh –l** command.



To change current shell temporary use **exec */bin/csh*** command. This command will change current shell to ***csh***. This change will be temporary. It will be lost with current session. To verify this terminate the current session with exit command. Login again and view the current shell. It should be restored with default shell.

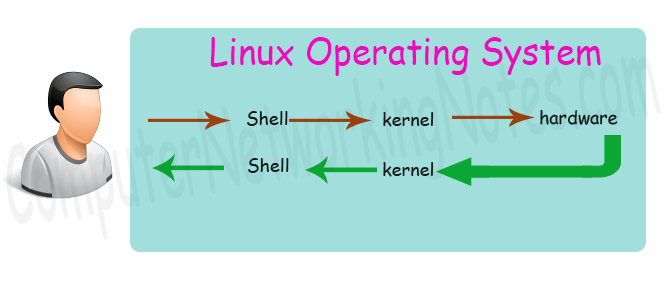


To change the current shell permanently use **chsh *–s /bin/csh*** command. Change will take place at next login.



**Kernel**

Kernel is the core application in Linux operating system. It communicates directly with system devices such as memory, CPU, CDROM, Hard disk etc.



When a user wants to access any device, he types appropriate command at command prompt. Shell interpreters the command and hands over the instruction to kernel. Kernel communicates with device and process the user requests.

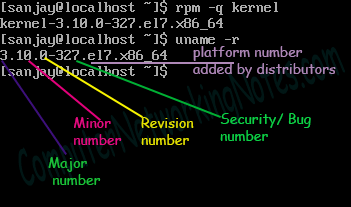
Kernel name provides information about its version. Kernel version number is built from four segments : major, minor, revision, and security / bug fix.

* **Major number :-** This number reflects major changes in kernel.
* **Minor number :-** This number reflects a minor revision in kernel.
* **Revision number :-** This number reflects that new supporting features are added in kernel.
* **Security/ Bug number :-** This number reflects security or bug fix in kernel.

Kernel development is an ongoing process. Development versions will first available as release candidates. Release Candidates (***rc***) are intended for developers. RC allows developers to test new features of kernel before final version is released. Release candidate have an indicative keyword (rc) in name for example kernel-2.6.22-rc3.

Kernel is an open source project. Distributors are allowed to make changes in kernel. If distributer makes any change in kernel, he will add a patch number in the end of the name. Besides patch number, distributors may add platform number.

* To view which kernel package is install, we can use **rpm –q kernel** command.
* To view only the kernel version number we can use **uname –r** command.



**🡪Built in Package Managers**

### Package Management

In few words, package management is a method of installing and maintaining (which includes updating and probably removing as well) software on the system. In the early days of Linux, programs were only distributed as source code, along with the required man pages, the necessary configuration files, and more. Nowadays, most Linux distributors use by default pre-built programs or sets of programs called packages, which are presented to users ready for installation on that distribution. However, one of the wonders of Linux is still the possibility to obtain source code of a program to be studied, improved, and compiled.

**How package management systems work**

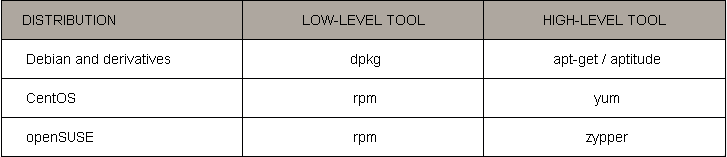
If a certain package requires a certain resource such as a shared library, or another package, it is said to have a dependency. All modern package management systems provide some method of dependency resolution to ensure that when a package is installed, all of its dependencies are installed as well.

**Packaging Systems**

Almost all the software that is installed on a modern Linux system will be found on the Internet. It can either be provided by the distribution vendor through central repositories (which can contain several thousands of packages, each of which has been specifically built, tested, and maintained for the distribution) or be available in source code that can be downloaded and installed manually. Because different distribution families use different packaging systems (Debian: \*.deb / CentOS: \*.rpm / openSUSE: \*.rpm built specially for openSUSE)

**High and low-level package tools**

In order to perform the task of package management effectively, you need to be aware that you will have two types of available utilities: low-level tools (which handle in the backend the actual installation, upgrade, and removal of package files), and high-level tools (which are in charge of ensuring that the tasks of dependency resolution and metadata searching -”data about the data”- are performed).

****

**1. dpkg**

It is a low-level package manager for **Debian-based systems.** It can install, remove, provide information about and build \*.deb packages but it can’t automatically download and install their corresponding dependencies.

**Dpkg Command**

dpkg is the main package management program in Debian and Debian based System. It is used to install, build, remove, and manage packages. Aptitude is the primary front-end to dpkg.

**1. Install a Package**

For installing an “.deb” package, use the command with “-i” option. For example, to install an “.deb” package called “flashpluginnonfree\_2.8.2+squeeze1\_i386.deb” use the following command.

****

**2. List all the installed Packages**

To view and list all the installed packages, use the “-l” option along with the command.



**3. Remove a Package**

To remove the “.deb” package, we must specify the package name “flashpluginnonfree“, not the original name “flashplugin-nonfree\_3.2\_i386.deb“. The “-r” option is used to remove/uninstall a package.



You can also use ‘p‘ option in place of ‘r’ which will remove the package along with configuration file. The ‘r‘ option will only remove the package and not configuration files.



**4. View the Content of a Package**

To view the content of a particular package, use the “-c” option as shown. The command will display the contents of a “.deb” package in long-list format.



**5. Check a Package is installed or not**

Using “-s” option with package name, will display whether an deb package installed or not.



**6. Check the location of Packages installed**

To list location of files to be installed to your system from package-name.



**7. Display dpkg Version**

The “–version” argument will display dpkg version information.



**8. Get all the Help about dpkg**

The “–help” option will display a list of available options of dpkg command.



**2. apt-get**

It is a high-level package manager for Debian and derivatives, and provides a simple way to retrieve and install packages, including dependency resolution, from multiple sources using the command line. Unlike dpkg, apt-get does not work directly with \*.deb files, but with the package proper name.

**What is apt-get**

The apt-get utility is a powerful and free package management command line program, that is used to work with Ubuntu’s APT (Advanced Packaging Tool) library to perform installation of new software packages, removing existing software packages, upgrading of existing software packages and even used to upgrading the entire operating system.

**What is apt-cache?**

The apt-cache command line tool is used for searching apt software package cache.

**1. How Do I List All Available Packages?**

To list all the available packages, type the following command.



**2. How Do I Find Out Package Name and Description of Software?**

****

**3. How Do I Check Package Information?**

****

**4. How to Update System Packages**

****

**5. How to Upgrade Software Packages**

****

**6. How Do I Install or Upgrade Specific Packages?**

****

**7. How I can Install Multiple Packages?**

****

**8. How to Install Several Packages using Wildcard**

****

**9. How Do I Remove Packages Without Configuration**



**10. How Do I Completely Remove Packages**

****

**11. How I Can Clean Up Disk Space**

****

**3. rpm**

It is the package management system used by Linux Standard Base (LSB)-compliant distributions for low-level handling of packages. Just like dpkg, it can query, install, verify, upgrade, and remove packages, and is more frequently used by Fedora-based distributions, such as RHEL and CentOS.

**Some Facts about RPM (RedHat Package Manager)**

* RPM is free and released under GPL (General Public License).
* RPM keeps the information of all the installed packages under /var/lib/rpm database.
* RPM is the only way to install packages under Linux systems, if you’ve installed packages using source code, then rpm won’t manage it.
* RPM deals with .rpm files, which contains the actual information about the packages such as: what it is, from where it comes, dependencies info, version info etc.

**There are five basic modes for RPM command**

**Install :** It is used to install any RPM package.

**Remove :** It is used to erase, remove or un-install any RPM package.

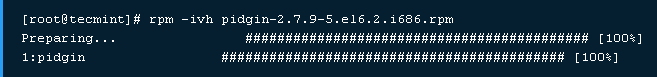
**Upgrade :** It is used to update the existing RPM package.

**Verify :** It is used to query about different RPM packages.

**Query :** It is used for the verification of any RPM package.

**1. How to Install an RPM Package**

For installing an rpm software package, use the following command with -i option. For example, to install an rpm package called pidgin-2.7.9-5.el6.2.i686.rpm.



**RPM command and options**

-i : install a package

-v : verbose for a nicer display

-h: print hash marks as the package archive is unpacked.

**2. How to check an Installed RPM Package**

****

**3. How to List all files of an installed RPM package**

To view all the files of an installed rpm packages, use the -ql (query list) with rpm command.



**4. How to List Recently Installed RPM Packages**

Use the following rpm command with -qa (query all) option, will list all the recently installed rpm packages.



**5. How to Upgrade a RPM Package**

If we want to upgrade any RPM package “–U” (upgrade) option will be used.



**6. How to Remove a RPM Package**

To un-install an RPM package, for example we use the package name nx, not the original package name nx-3.5.0-2.el6.centos.i686.rpm. The -e (erase) option is used to remove package.



**4.yum**

It adds the functionality of automatic updates and package management with dependency management to RPM-based systems. As a high-level tool, like apt-get or aptitude, yum works with repositories.

**What is YUM?**

YUM (Yellowdog Updater Modified) is an open source command-line as well as graphical based package management tool for RPM (RedHat Package Manager) based Linux systems. It allows users and system administrator to easily install, update, remove or search software packages on a systems.

**1. Install a Package with YUM**

To install a package called Firefox 14, just run the below command it will automatically find and install all required dependencies for Firefox.



**2. Removing a Package with YUM**

****

**3. Updating a Package using YUM**

****

**4. Update System using Yum**

****

**5. List all available Group Packages**

****

**6. View History of Yum**

****

**5.Zypper**

Zypper is the native command-line interface of the ZYpp package manager to install, remove, update and query software packages of local or remote (networked) media.

**1. Install a Package using zypper**

To install a package using zypper command, use the following syntax:

zypper install <package Name>

**Example: zipper install MozillaFirefox**

**2. Update a Package**

**The following will upgrade only MozillaFirefox package.**

zypper update MozillaFirefox

**The following will upgrade all the packages on your system to the latest version.**

zypper update

**You can also view all the available updates using the list-updates option as shown below.**

Zipper list-updates

**3. System Wide Full Distribution Upgrade**

Zypper dup

**4. Delete a Package**

zypper remove MozillaFirefox

**5. Lock a Specific Package**

zypper al ypbind

Note: al (stands for add lock)

**6. Remove Lock from a Package**

zypper rl ypbind

**🡪** **Man pages**

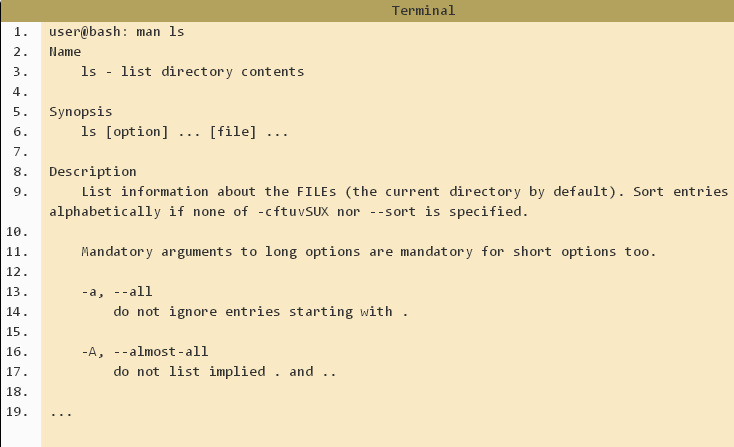
**Introduction:**

The Linux command line offers a wealth of power and opportunity. If your memory is like mine then you find it hard to remember a large number of details. Fortunately for us there is an easy to use resource that can inform us about all the great things we can do on the command line. That's what we're going to learn about in this section. I know you're keen and eager to get stuck into doing stuff, and we'll get started on that in the next section, I promise, first we need to learn how to use Manual pages however.

**So what are they exactly:**

The manual pages are a set of pages that explain every command available on your system including what they do, the specifics of how you run them and what command line arguments they accept. Some of them are a little hard to get your head around but they are fairly consistent in their structure so once you get the hang of it it's not too bad. You invoke the manual pages with the following command:

**man <command to look up>**

****

**Line 3** tells us the actual command followed by a simple one line description of it's function.

**Lines 6** is what's called the synopsis. This is really just a quick overview of how the command should be run. Square brackets ( [ ] ) indicate that something is optional. (option on this line refers to the command line options listed below the description)

**Line 9** presents us with a more detailed description of the command.

**Line 11 onwards**Below the description will always be a list of all the command line options that are avaialable for the command.

**Note: To exit the man pages press 'q' for quit.**

**Searching**

It is possible to do a keyword search on the Manual pages. This can be helpful if you're not quite sure of what command you may want to use but you know what you want to achieve. To be effective with this approach, you may need a few goes. It is not uncommon to find that a particular word exists in many manual pages.

**man -k <search term>**

If you want to search within a manual page this is also possible. To do this, whilst you are in the particular manual page you would like to **search press forward slash '/' followed by the** term you would like to search for and hit 'enter' If the term appears multiple times you may cycle through them by pressing the 'n' button for next.

**🡪The first command cat**

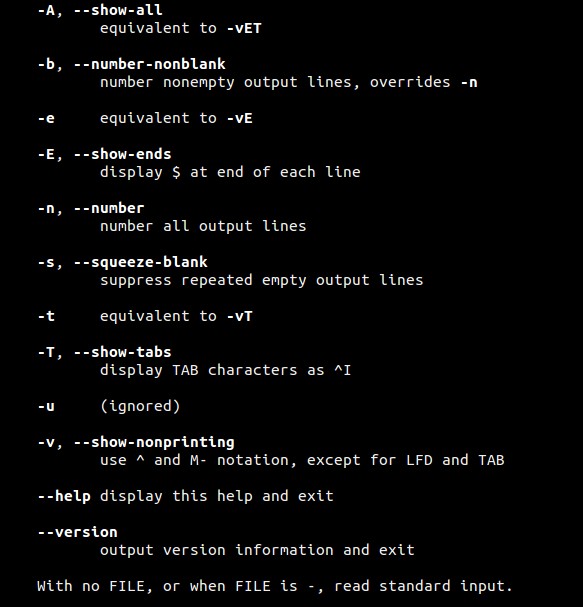
Cat stands for **Concatenate**. Cat is the basic command when we start learning Linux/Unix, as the name suggest it is used to create new file ,concatenate files and display the output of files on the standard output.

In this post we will discuss 16 different examples of cat command which will be useful for the beginners.

**Basic Syntax of cat command :**

# cat <options> <File>

Some of basic options are listed below that can be used in cat command.



##### **Example:1 Create a new file using ‘cat > {file\_name}’**

Let’s suppose i want to create a new file with name ‘linux\_world’. Type the following cat command followed by the text you want in to insert in the file. Make sure you type ‘**Ctrl-d’** at the end to save the file.

[root@linuxtechi ~]# cat > linux\_world

Hi this is my first file in linux.

Linux always rocks

Thanks

[root@linuxtechi ~]#

##### **Example:2 View the Contents of a File.**

To display or view the contents of a file using cat command use the below syntax

# cat {file\_name}

Let’s display the contents of linux\_world file.

[root@linuxtechi ~]# cat linux\_world

Hi this is my first file in linux.

Linux always rocks

Thanks

root@linuxtechi ~]#

##### **Example:3 View the Contents of Multiple Files**

[root@linuxtechi ~]# cat linux\_world linux\_distributions /etc/fstab

Above command will display output of three files on the terminal.

##### **Example:4 Display the output of a file using page wise.**

For example if we have a big file whose contents can’t be display at once on the screen , in that case we can use more and less command with cat to view the contents page wise.

[root@linuxtechi ~]# cat /etc/passwd | more

[root@linuxtechi ~]# cat /etc/passwd | less

##### **Example:5 cat command without filename arguments**

if we don’t specify any arguments in the cat command then it will read the inputs from the keyboard attached to the system. Type some text after entering the cat command.

[root@linuxtechi ~]# cat

Ubuntu Linux Rocks at desktop Level

Now press ‘**Ctrl-d**‘ to inform cat that it has reached end of file (EOF). In this case it will display the line of text twice because it copies std input to std output.

[root@linuxtechi ~]# cat

Ubuntu Linux Rocks at desktop Level

Ubuntu Linux Rocks at desktop Level

[root@linuxtechi ~]#

##### **Example:6 Display the contents of a file with Line Numbers**

[root@linuxtechi ~]# cat -n linux\_world

1 Hi this is my first file in linux.

2 Linux always rocks

3 Thanks

[root@linuxtechi ~]#

In case if your file has blank lines , then above command will also display the number of blank lines as well, so to remove the numbering of blank lines , we can use ‘**-b**‘ option in place of ‘-n’ in the above command.

##### **Example:7 Copy the contents of One file to Another file.**

Using greater than **‘>**‘ symbol in cat command we can copy the contents of one file to another , example is shown below :

[root@linuxtechi ~]# cat linux\_world > linux\_text

[root@linuxtechi ~]#

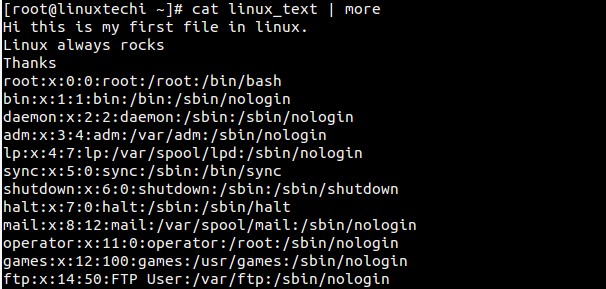
##### **Example:8 Appending the contents of one file to another.**

Using double greater than symbol ‘**>>**‘ in cat command we can append the contents of one file to another. Example is shown below :

[root@linuxtechi ~]# cat /etc/passwd >> linux\_text

[root@linuxtechi ~]#

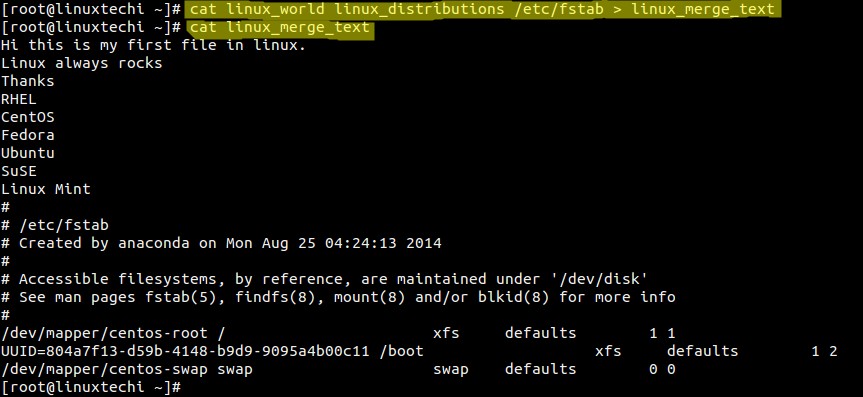
Above Command will append the contents of /etc/passwd file to linux\_text file at the end. Now we can verify the contents of linux\_text file.



##### Example:9 Redirecting the output of multiple files into a Single File.

[root@linuxtechi ~]# cat linux\_world linux\_distributions /etc/fstab > linux\_merge\_text

Above command will merge the output of 3 files into a single file ‘linux\_merge\_text’.



##### **Example:10 Getting input using standard input operator.**

[root@linuxtechi ~]# cat < linux\_distributions

RHEL

CentOS

Fedora

Ubuntu

SuSE

Linux Mint

[root@linuxtechi ~]#

Above cat command is getting input from the file using std input operator ‘<‘

##### **Example:11 Sorting the output of multiple files into a single file**

[root@linuxtechi ~]# cat linux\_text linux\_distributions /etc/passwd | sort > linux\_sort

By default sorting will done on the alphabetic order, if you want the sorting on basis of number then use ‘**-n’**option in the sort command.

##### **Example:12 Insert $ at end of each line using -E option**

[root@linuxtechi ~]# cat -E linux\_world

Hi this is my first file in linux.$

Linux always rocks$

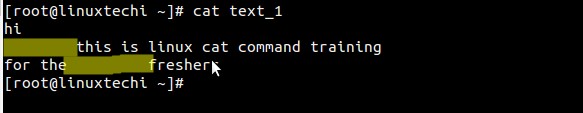
Thanks$

[root@linuxtechi ~]#

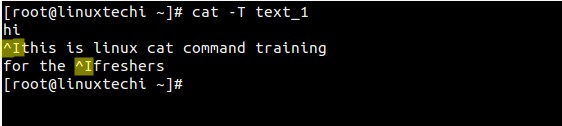
Above command will insert ‘$’ at the end of each line in the output.

##### **Example:13 Show the tab space in the file as ‘^I’ using -T option.**

Let’s create a file with some tab spaces.

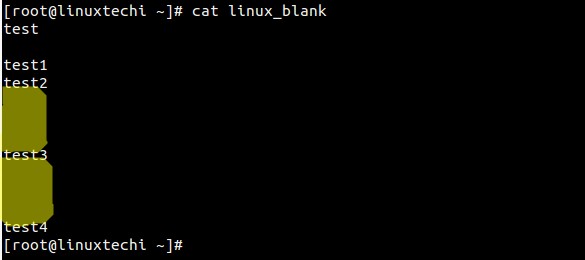


Now display these tab spaces as ^I



##### **Example:14 Squeeze blank repeated lines using -s option**

Let’s take am example of file ‘linux\_blank’ , which consists of multiple repeated blank lines.



Now remove the blank repeated lines in the output using below command.

[root@linuxtechi ~]# cat -s linux\_blank

test

test1

test2

test3

test4

[root@linuxtechi ~]#

##### **Example:15 View the Contents in Reverse Order**

tac is the reverse of cat command. tac will display the output in revers order example is shown below

[root@linuxtechi ~]# tac linux\_world

Thanks

Linux always rocks

Hi this is my first file in linux.

[root@linuxtechi ~]#

##### **Example:16 Display non-printing characters using -v option.**

**-v** option in the cat command is used to show the non-printing characters in the output. This option become useful when we are suspecting the CRLF ending lines, in that case it will show ^M at the end of each line.

[root@linuxtechi tmp]# cat test\_file

hi there

[root@linuxtechi tmp]# cat -v test\_file

hi there^M

[root@linuxtechi tmp]#

**🡪Basic UNIX Commands**

**Vi(Visual) editor:**

There are many ways to edit files in Unix and for me one of the best ways is using screen-oriented text editor **vi**. This editor enables you to edit lines in context with other lines in the file. Now days you would find an improved version of vi editor which is called **VIM**. Here VIM stands for **V**i **IM**proved.

**The vi is generally considered the de facto standard in Unix editors because −**

* It's usually available on all the flavors of UNIX system.
* Its implementations are very similar across the board.
* It requires very few resources.
* It is more user friendly than any other editors like ed or ex.

You can use **vi** editor to edit an existing file or to create a new file from scratch. You can also use this editor to just read a text file.

**Modes of VI:**

**Three different modes:**

1. **Command mode**

Vi editor begins in command mode, where cursor movement (navigation in the file) and editing occur. To enter into command mode from Inserting mode press esc button.

1. **Insert mode**

Used for entering text, this is similar to notepad in Windows. To enter in to inserting mode you can use any of the following.

i or I**=>** present line  
o **=>** one line down the present line  
O **=>** one line above

1. **Last line mode**

The last vi mode is known as vi last line mode. You can only get to last line mode from command mode, and you get into last line mode by pressing the esc, shift + colon key, like this:

:

After pressing this key, you'll see a colon character appear at the beginning of the last line of your vi editor window, and your cursor will be moved to that position. This indicates that vi is ready for you to type in a "last line command".

**Save and quitting a file**:w **=>**save the file

:q **=>** quit the file  
:wq **=>** save and quit  
:w! **=>**force save the file  
:q! **=>** force quit without save  
:wq! **=>**save and quit forcefully

## Inserting Text

|  |  |
| --- | --- |
| i | Insert before cursor |
|  |  |
| I | Insert before line |
| a | Append after cursor |
| A | Append after line |
| o | Open a new line after current line |
| O | Open a new line before current line |
| r | Replace one character |
| R | Replace many characters |

## Motion(**Navigational commands)**

|  |  |
| --- | --- |
| h | Move left |
| j | Move down |
| k | Move up |
| l | Move right |
| w | Move to next word |
| W | Move to next blank delimited word |
| b | Move to the beginning of the word |
| B | Move to the beginning of blank delimited word |
| e | Move to the end of the word |
| E | Move to the end of Blank delimited word |
| ( | Move a sentence back |
| ) | Move a sentence forward |
| { | Move a paragraph back |
| } | Move a paragraph forward |
| 0 | Move to the begining of the line |
| $ | Move to the end of the line |
| 1G | Move to the first line of the file |
| G | Move to the last line of the file |
| nG | Move to nth line of the file |
| :n | Move to nth line of the file |
| fc | Move forward to c |
| Fc | Move back to c |
| H | Move to top of screen |
| M | Move to middle of screen |
| L | Move to botton of screen |
| % | Move to associated ( ), { }, [ ] |

## Deleting Text

|  |
| --- |
| Almost all deletion commands are performed by typing d followed by a motion. For example, dw deletes a word. A few other deletes are: |

|  |  |
| --- | --- |
| x | Delete character to the right of cursor |
| X | Delete character to the left of cursor |
| D | Delete to the end of the line |
| dd | Delete current line |
| :d | Delete current line |

## Changing text

|  |
| --- |
| The change command is a deletion command that leaves the editor in insert mode. It is performed by typing c followed by a motion. For example cw changes a word. A few other change commands are: |

|  |  |
| --- | --- |
| C | Change to the end of the line |
| cc | Change the whole line |

## Putting text

|  |  |
| --- | --- |
| p | Put after the position or after the line |
| P | Put before the position or before the line |

## Search for strings

|  |  |
| --- | --- |
| /*string* | Search forward for *string* |
| ?*string* | Search back for *string* |
| n | Search for next instance of *string* |
| N | Search for previous instance of *string* |

**🡪 Redirection operators**

Once you type the ' cat ' command, use the **Redirection operator** and add a name for a file. Then start typing your line. And finally press Ctrl+D. You will have a file named ' filename ' in the same directory. Suppose you want to add a single line to an existing file.

**Redirect Descriptor:**

|  |  |
| --- | --- |
| **Descriptor** | **Description** |
| 0 (STDIN) | Standard Input |
| 1 (STDOUT) | Standard Output |
| 2 (STDERR) | Standard Error Output |
| 3-9 (UNDEFINED) | These handles are defined individually by the application and are specific to each tool. |
| > | Redirect standard output |
| 2> | Redirect standard error |
| 2>&1 | Redirect standard error to standard output |
| < | Redirect standard input form |
| >> | Append to standard output |
| & | Run as background process |
| | | Reads the output from one command and writes it to the input of another command. Also known as a pipe. |

**Example:**

For example, the output of the cat command, which is commonly used to read files, could be redirected when it reads a file named file1 from the display screen, where it would otherwise appear by default, to a file named file2 with the following:

**cat file1 > file2**

If file2 already exists, the output redirection operator will overwrite its contents. If it does not yet exist, the output redirection operator will create it.

**Input Redirection**

Instead of waiting the input from the normal standard input stdin, the command can accept the input from a file. The standard input is redirected using the **‘<‘ operator.**

[root@centos6 ~ ]# Cat hello.txt

[root@centos6 ~ ]# tr `a-z` `A-Z` < hello.txt Note: tr for translate

**Output Redirection**

Instead of being displayed on the screen, the output can be directed to a file, either in overwrite mode > or in **append mode >>**

[root@centos6 ~ ]# ls

[root@centos6 ~ ]# ls > ls.out

**Redirecting Standard Error**

If an error occurred in the execution of a command, the error is printed on the screen. Instead, it can be redirected to file.

[root@centos6 ~ ]# ls fooo.txt

ls: cannot access fooo.txt: No such file or directory

**Piping**  
Piping means to redirect the standard output of a command to become the input to another command.  
Thesyntax:  
command1 | command2

[root@centos6 ~ ]# date

Sun Aug 16 22:06:18 EET 2016

[ root@centos6 ~ ]# date | tr `a-z` `A-Z`

Sun Aug 16 22:06:18 EET 2016

**🡪Some UNIX commands.**

cat --- for creating and displaying short files

chmod --- change permissions

cd --- change directory

cp --- for copying files

date --- display date

echo --- echo argument

ftp --- connect to a remote machine to download or upload files

grep --- search file

head --- display first part of file

ls --- see what files you have

lpr --- standard print command (see also print )

more --- use to read files

mkdir --- create directory

mv --- for moving and renaming files

ncftp --- especially good for downloading files via anonymous ftp.

print --- custom print command (see also lpr )

pwd --- find out what directory you are in

rm --- remove a file

rmdir --- remove directory

rsh --- remote shell

setenv --- set an environment variable

sort --- sort file

tail --- display last part of file

tar --- create an archive, add or extract files

telnet --- log in to another machine

wc --- count characters, words, lines

**Files and Directories**

These commands allow you to create directories and handle files.

|  |  |
| --- | --- |
| **Command** | **Description** |
| cat | Display File Contents |
| cd | Changes Directory to dirname |
| chgrp | change file group |
| chmod | Changing Permissions |
| cp | Copy source file into destination |
| file | Determine file type |
| find | Find files |
| grep | Search files for regular expressions. |
| head | Display first few lines of a file |
| ln | Create softlink on oldname |
| ls | Display information about file type. |
| mkdir | Create a new directory dirname |
| more | Display data in paginated form. |
| mv | Move (Rename) a oldname to newname. |
| pwd | Print current working directory. |
| rm | Remove (Delete) filename |
| rmdir | Delete an existing directory provided it is empty. |
| tail | Prints last few lines in a file. |
| touch | Update access and modification time of a file. |

**Manipulating data**

The contents of files can be compared and altered with the following commands.

|  |  |
| --- | --- |
| **Command** | **Description** |
| awk | Pattern scanning and processing language |
| cmp | Compare the contents of two files |
| comm | Compare sorted data |
| cut | Cut out selected fields of each line of a file |
| diff | Differential file comparator |
| expand | Expand tabs to spaces |
| join | Join files on some common field |
| perl | Data manipulation language |
| sed | Stream text editor |
| sort | Sort file data |
| split | Split file into smaller files |
| tr | Translate characters |
| uniq | Report repeated lines in a file |
| wc | Count words, lines, and characters |
| vi | Opens vi text editor |
| vim | Opens vim text editor |
| fmt | Simple text formatter |
| spell | Check text for spelling error |
| ispell | Check text for spelling error |
| ispell | Check text for spelling error |
| emacs | GNU project Emacs |
| ex, edit | Line editor |
| emacs | GNU project Emacs |
| emacs | GNU project Emacs |

**Compressed Files**

Files may be compressed to save space. Compressed files can be created and examined −

|  |  |
| --- | --- |
| **Command** | **Description** |
| compress | Compress files |
| gunzip | Uncompress gzipped files |
| gzip | GNU alternative compression method |
| uncompress | Uncompress files |
| unzip | List, test and extract compressed files in a ZIP archive |
| zcat | Cat a compressed file |
| zcmp | Compare compressed files |
| zdiff | Compare compressed files |
| zmore | File perusal filter for crt viewing of compressed text |

**Getting Information**

Various Unix manuals and documentation are available on-line. The following Shell commands give information −

|  |  |
| --- | --- |
| **Command** | **Description** |
| apropos | Locate commands by keyword lookup |
| info | Displays command information pages online |
| man | Displays manual pages online |
| whatis | Search the whatis database for complete words. |
| yelp | GNOME help viewer |

**Network Communication**

These following commands are used to send and receive files from a local UNIX hosts to the remote host around the world.

|  |  |
| --- | --- |
| **Command** | **Description** |
| ftp | File transfer program |
| rcp | Remote file copy |
| rlogin | Remote login to a UNIX host |
| rsh | Remote shell |
| tftp | Trivial file transfer program |
| telnet | Make terminal connection to another host |
| ssh | Secure shell terminal or command connection |
| scp | Secure shell remote file copy |
| sftp | secure shell file transfer program |

Some of these commands may be restricted at your computer for security reasons.

**Messages between Users**

The UNIX systems support on-screen messages to other users and world-wide electronic mail −

|  |  |
| --- | --- |
| **Command** | **Description** |
| evolution | GUI mail handling tool on Linux |
| mail | Simple send or read mail program |
| mesg | Permit or deny messages |
| parcel | Send files to another user |
| pine | Vdu-based mail utility |
| talk | Talk to another user |
| write | Write message to another user |

**Programming Utilities**

The following programming tools and languages are available based on what you have installed on your Unix.

|  |  |
| --- | --- |
| **Command** | **Description** |
| dbx | Sun debugger |
| gdb | GNU debugger |
| make | Maintain program groups and compile programs. |
| nm | Print program's name list |
| size | Print program's sizes |
| strip | Remove symbol table and relocation bits |
| cb | C program beautifier |
| cc | ANSI C compiler for Suns SPARC systems |
| ctrace | C program debugger |
| gcc | GNU ANSI C Compiler |
| indent | Indent and format C program source |
| bc | Interactive arithmetic language processor |
| gcl | GNU Common Lisp |
| perl | General purpose language |
| php | Web page embedded language |
| py | Python language interpreter |
| asp | Web page embedded language |
| CC | C++ compiler for Suns SPARC systems |
| g++ | GNU C++ Compiler |
| javac | JAVA compiler |
| appletvieweir | JAVA applet viewer |
| netbeans | Java integrated development environment on Linux |
| sqlplus | Run the Oracle SQL interpreter |
| sqlldr | Run the Oracle SQL data loader |
| mysql | Run the mysql SQL interpreter |

**Misc Commands**

These commands list or alter information about the system −

|  |  |
| --- | --- |
| **Command** | **Description** |
| chfn | Change your finger information |
| chgrp | Change the group ownership of a file |
| chown | Change owner |
| date | Print the date |
| determin | Automatically find terminal type |
| du | Print amount of disk usage |
| echo | Echo arguments to the standard options |
| exit | Quit the system |
| finger | Print information about logged-in users |
| groupadd | Create a user group |
| Show group memberships |  |
| homequota | Show quota and file usage |
| iostat | Report I/O statistics |
| kill | Send a signal to a process |
| last | Show last logins of users |
| logout | log off UNIX |
| lun | List user names or login ID |
| netstat | Show network status |
| passwd | Change user password |
| passwd | Change your login password |
| printenv | Display value of a shell variable |
| ps | Display the status of current processes |
| ps | Print process status statistics |
| quota -v | Display disk usage and limits |
| reset | Reset terminal mode |
| script | Keep script of terminal session |
| script | Save the output of a command or process |
| setenv | Set environment variables |
| stty | Set terminal options |
| time | Time a command |
| top | Display all system processes |
| tset | Set terminal mode |
| tty | Print current terminal name |
| umask | Show the permissions that are given to view files by default |
| uname | Display name of the current system |
| uptime | Get the system up time |
| useradd | Create a user account |
| users | Print names of logged in users |
| vmstat | Report virtual memory statistics |
| w | Show what logged in users are doing |
| who | List logged in users |