DVS Technologies Aws & Devops

Compiled and Scrutinized by

Mr. Shaan Shaik

(Senior DevOps Lead)

Words To The Students

Though we have taken utmost efforts to present you this book error free, but still it may contain some errors or mistakes. Students are encouraged to bring, if there are any mistakes or errors in this document to our notice. So that it may be rectified in the next edition of this document.

"Suppressing your doubts is Hindering your growth".

We urge you to work hard and make use of the facilities we are providing to you, because there is no substitute for hard work. We wish you all the best for your future.

"The grass isn't greener on the other side; the grass is greener where you water it."

You and your suggestions are valuable to us; Help us to serve you better. In case of any suggestions, grievance, or complaints, please feel free to write us your suggestions, grievance and feedback on the following

Dvs.training@gmail.com

DVS Technologies Aws & Devops

Linux Administration

Contents

- How to create a group
- Removing users

9. Filesystem Management

- Working with Filesystem
- Standard partition
- Creating Filesystem using Standard partition
- Logical Volume Management
- Creating the filesystem in LVM
- Filesystem size increase
- Removing File system

10. Software Management

- RPM – Redhat Package Manager
- YUM – Yellow Dog Update Manager

11. Network Configuration & Trouble Shooting

- The basic requirements for Networking
- Changing the hostname
- Assigning /Changing the IP Address

12. Booting Procedure and Kernel Parameter

13. Configuring password less login in Linux

DVS Technologies Aws & Devops

Introduction to Linux

Linux is a Unix clone written from scratch by Linus Torvalds with assistance from a loosely-knit team of hackers across the Net. From the start, it was Linus' goal to have a free system that was completely compliant with the original UNIX. The Linux operating system is written in the C programming language. Unix is a multitasking, multi-user computer operating system originally developed in 1969 by a group of AT&T employees at Bell Labs. Linux and Unix strive to be POSIX compliant. 64% of the world's servers run some variant of Unix or Linux. The Android phone and the Kindle run Linux. Linux is available (also via internet) in different distributions (Suse, Fedora, Debian, Centos, Redhat etc.).It is also worth to note that modern Linux not only runs on workstations, mid- and high-end servers, but also on "gadgets" like PDA's, mobiles, a shipload of embedded applications and even on experimental wristwatches. This makes Linux the only operating system in the world covering such a wide range of hardware.

Linux is an ideal operating system for power-users and programmers, because it has been and is being developed by such people. Everything a good programmer can wish for is available: compilers, libraries, development and debugging tools. These packages come with every standard Linux distribution. The C-compiler is included for free – as opposed to many UNIX distributions demanding licensing fees for this tool. All the documentation and manuals are there, and examples are often included to help you get started in no time. It feels like UNIX and switching between UNIX and Linux is a natural thing.

Linux Pros

- Linux is free
- Linux is portable to any hardware platform
- Linux was made to keep on running
- Linux is secure and versatile
- Linux is scalable
- The Linux OS and most Linux applications have very short debug-times

Linux Cons

- There are far too many different distributions like RedHat for servers and SuSE for workstations
- Linux is not very user friendly and confusing for beginners

Differences between Linux & Unix

	Linux	Unix
Introduction	**Linux** is an open source, free to use operating system widely used for computer hardware and software, game development, tablet PCS, mainframes etc.	**Unix** is an operating system commonly used in internet servers, workstations and PCs by Solaris, Intel, HP etc.
Cost	Linux can be freely distributed, downloaded freely, distributed through magazines, Books etc. There are priced versions for Linux also, but they are normally cheaper than Windows	Different flavors of Unix have different cost structures according to vendors
Development and Distribution	Linux is developed by Open Source development i.e. through sharing and collaboration of code and features through forums etc and it is distributed by various vendors.	Unix systems are divided into various other flavors, mostly developed by AT&T as well as various commercial vendors and non-profit organizations
Manufacturer	Linux kernel is developed by the community. Linus Torvalds oversees things.	Three bigest distributions are Solaris (Oracle), AIX (IBM) & HP-UX Hewlett Packard. And Apple Makes OSX, an unix based os..

DVS Technologies Aws & Devops

User	Everyone. From home users to developers and computer enthusiasts alike.	Unix operating systems were developed mainly for mainframes, servers and workstations except OSX, Which is designed for everyone. The Unix environment and the client-server program model were essential elements in the development of the Internet
Usage	Linux can be installed on a wide variety of computer hardware, ranging from mobile phones, tablet computers and video game consoles, to mainframes and supercomputers.	The UNIX operating system is used in internet servers, workstations & PCs. Backbone of the majority of finance infrastructure and many 24x365 high availability solutions.
File system support	Ext2, Ext3, Ext4, Jfs, ReiserFS, Xfs, Btrfs, FAT, FAT32, NTFS	jfs, gpfs, hfs, hfs+, ufs, xfs, zfs format
Text mode interface	BASH (Bourne Again SHell) is the Linux default shell. It can support multiple command interpreters.	Originally the Bourne Shell. Now it's compatible with many others including BASH, Korn & C.
What is it?	Linux is an example of Open Source software development and Free Operating System (OS).	Unix is an operating system that is very popular in universities, companies, big enterprises etc.
GUI	Linux typically provides two GUIs, KDE and Gnome. But there are millions of alternatives such as LXDE, Xfce, Unity, Mate, twm, ect.	Initially Unix was a command based OS, but later a GUI was created called Common Desktop Environment. Most distributions now ship with Gnome.
Price	Free but support is available for a price.	Some free for development use (Solaris) but support is available for a price.

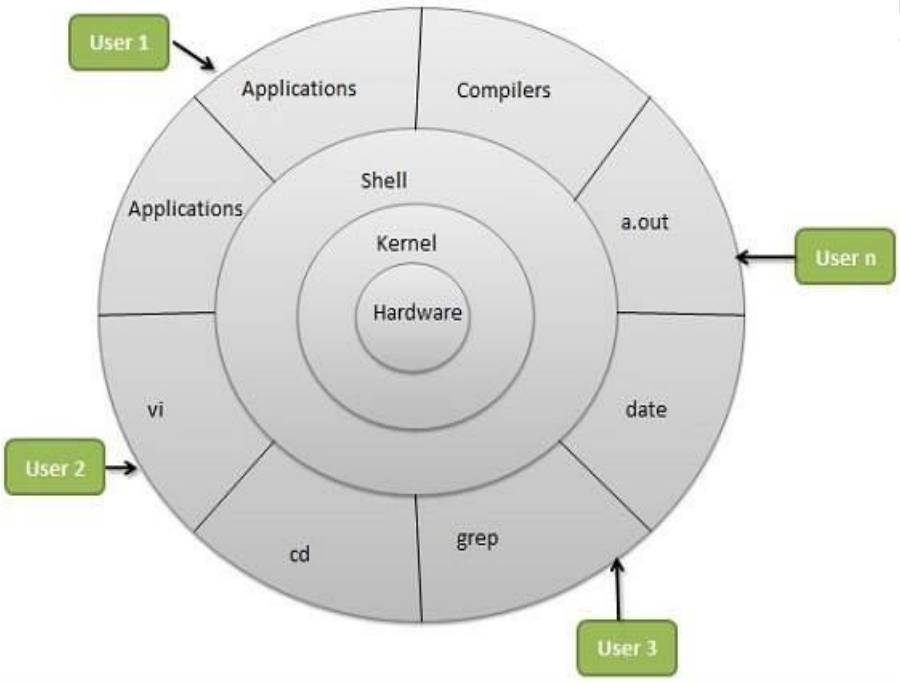
Security	Linux has had about 60-100 viruses listed till date. None of them actively spreading nowadays.	A rough estimate of UNIX viruses is between 85 -120 viruses reported till date.
Processors	Dozens of different kinds.	x86/x64, Sparc, Power, Itanium, PA-RISC, PowerPC and many others.
Examples	Ubuntu, Fedora, Red Hat, Debian, Archlinux, Android etc.	OS X, Solaris, All Linux

Why we are going for Unix Than windows

Please find the below reasons:

- Linux/Unix is more flexible and can be installed on many different types of machines, including main-frame computers, supercomputers and micro-computers.
- Linux/Unix is more stable and does not go down as often as Windows does, therefore requires less administration and maintenance.
- Linux/Unix has greater built-in security and permissions features than Windows.
- Linux/Unix possesses much greater processing power than Windows.
- Linux/Unix is the leader in serving the Web. About 90% of the Internet relies on Linux/Unix operating systems running on Apache, the world's most widely used Web server.
- Software upgrades from Microsoft often require the user to purchase new or more hardware or prerequisite software. That is not the case with Linux/Unix.
- The mostly free or inexpensive open-source operating systems, such as Linux and BSD, with their flexibility and control, are very attractive to (aspiring) computer wizards. Many of the smartest programmers are developing state-of-the-art software free of charge for the fast growing "open-source movement".
- Linux/Unix also inspires novel approaches to software design, such as solving problems by interconnecting simpler tools instead of creating large monolithic application programs.

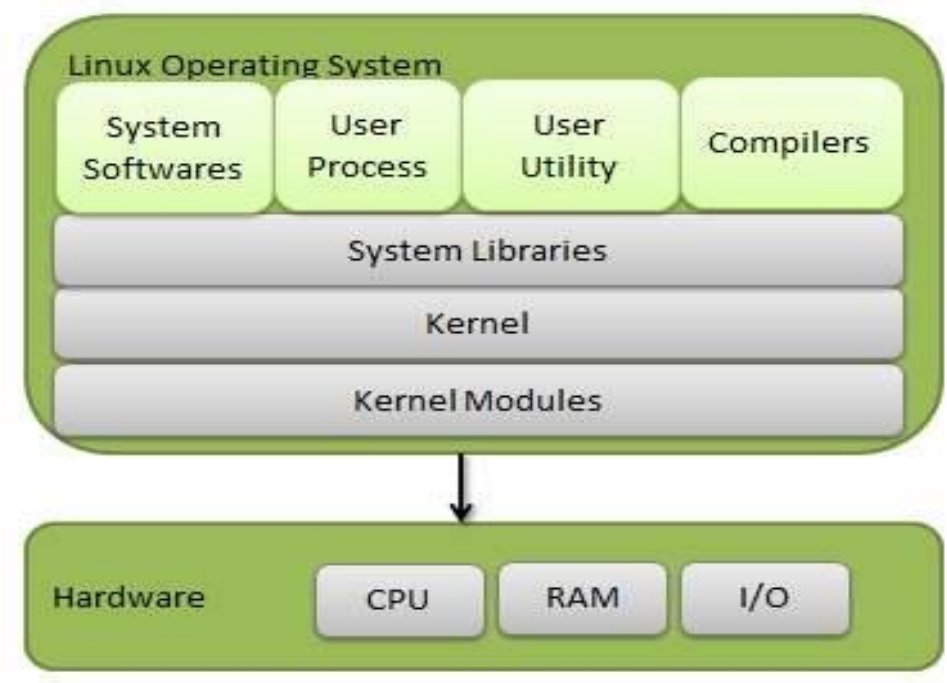
Basic Kernel & Shell Architecture



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

DVS Technologies Aws & Devops



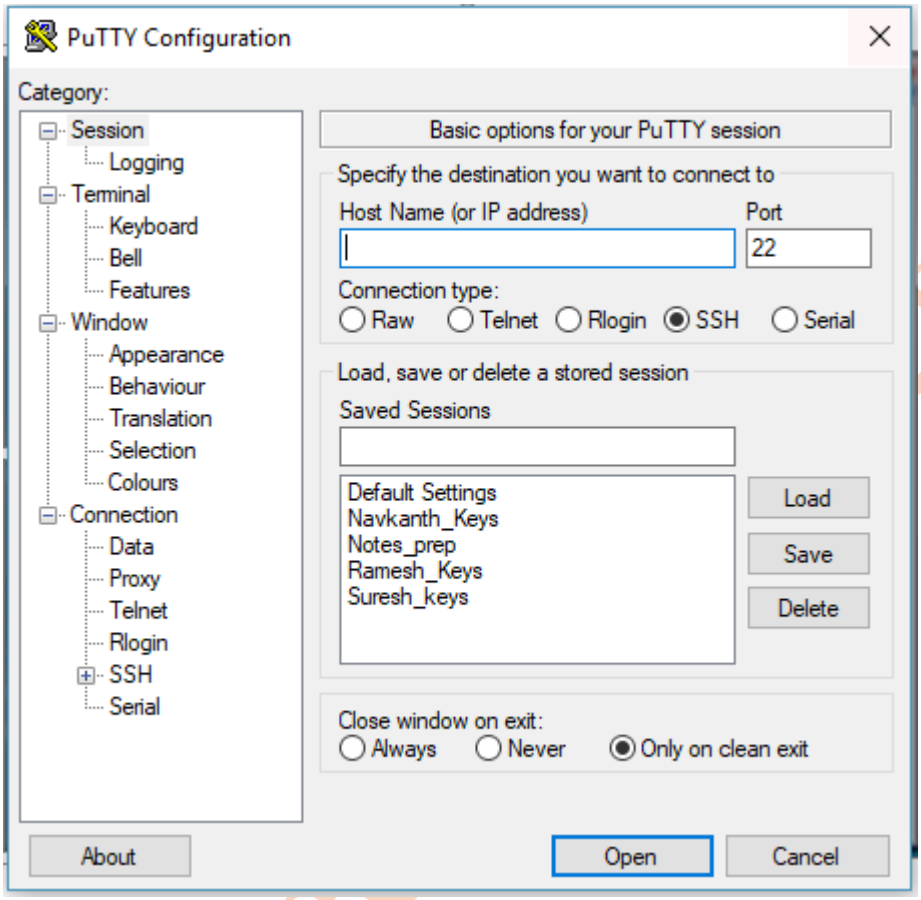
The architecture of a Linux System consists of the following layers

- **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.

Software's used to reach the servers

Download putty software from below site

https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html



- Install using the default settings on your computer.

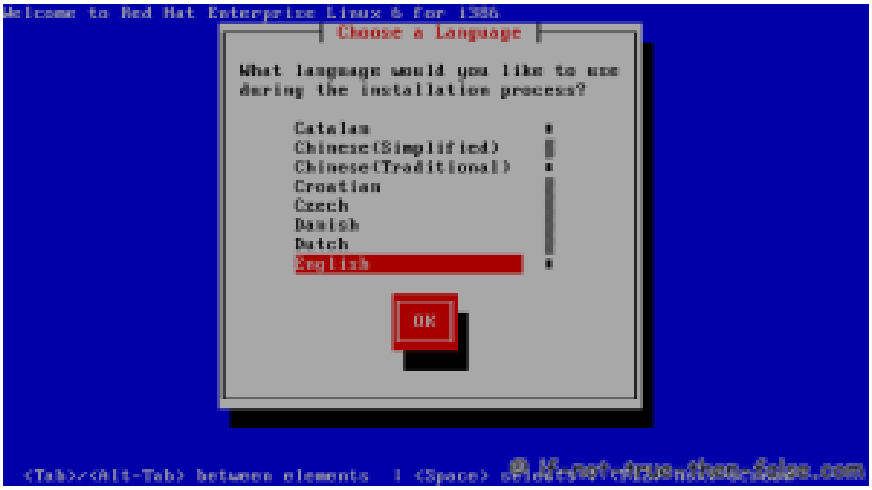
  - Double-click the PuTTY icon.
  - Enter the UNIX/Linux server hostname in the 'Host Name' box, and press the 'Open' button at the bottom of the dialog box.
  - Enter your username and password when prompted.

DVS Technologies Aws & Devops

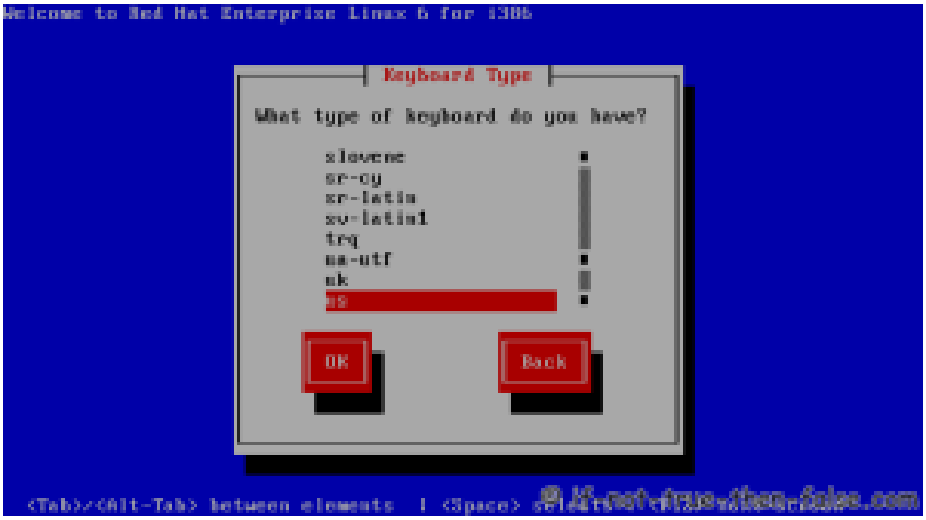
RHEL 6 Installation & Configuration

1. Select Install or upgrade an existing system option on Grub Menu

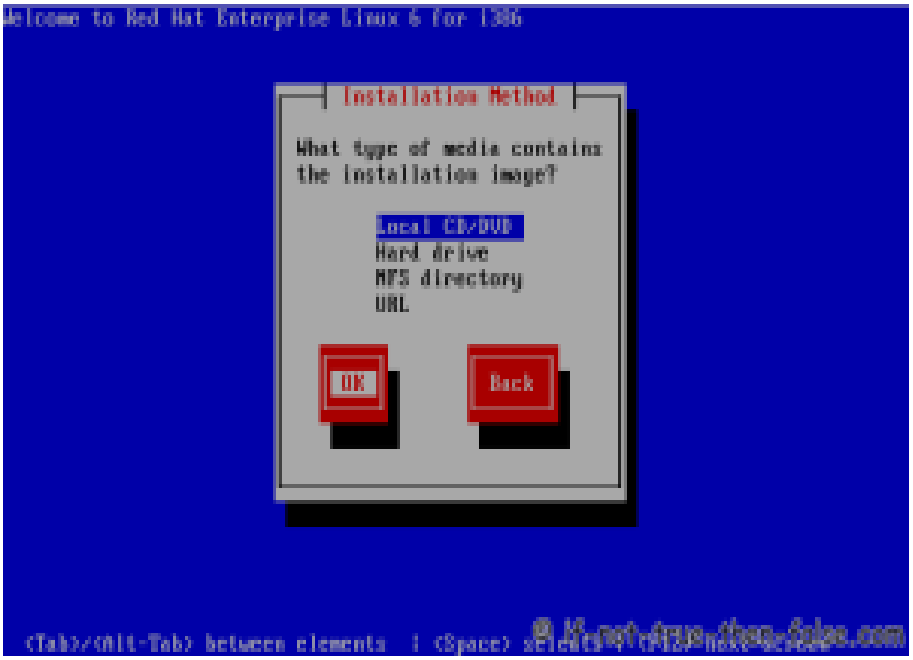
2. Choose a language



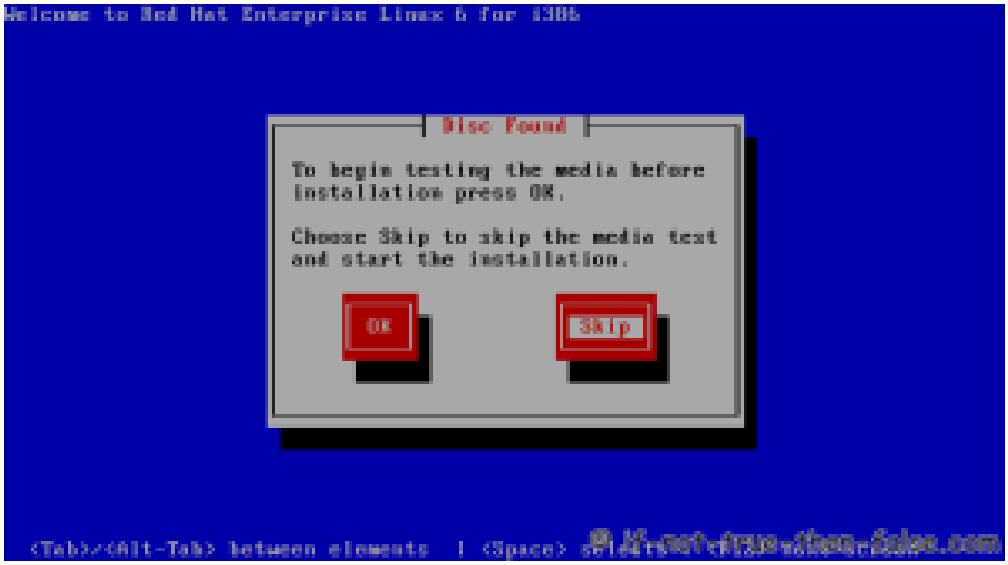
3. Choose a keyboard type



4. Choose a installation media

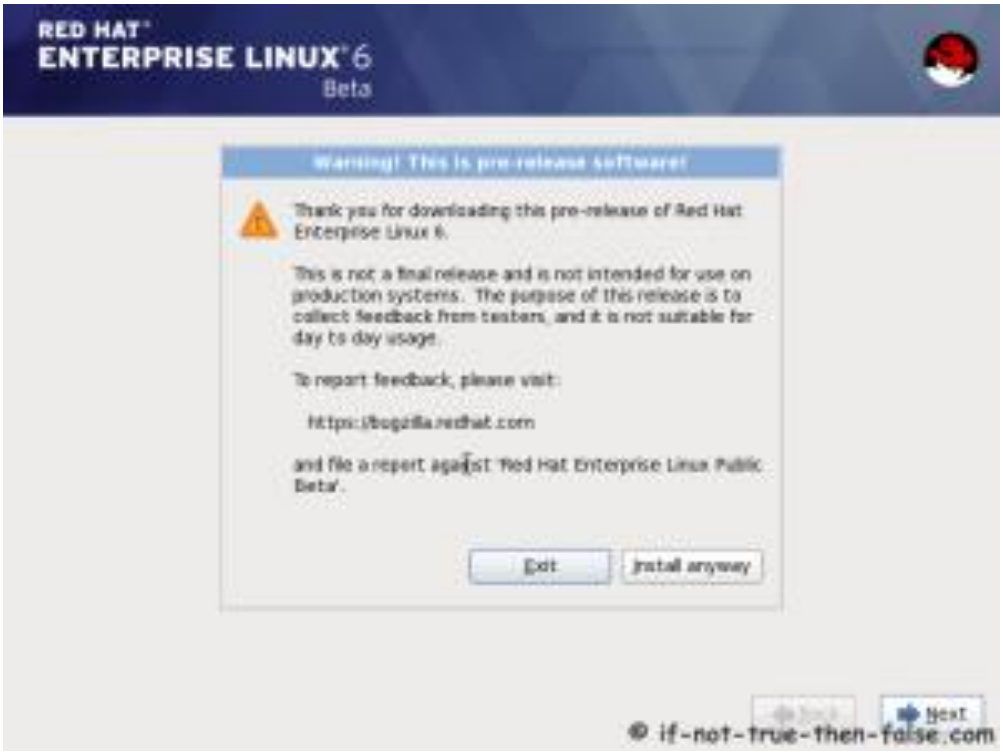


5. Skip DVD media test (or select media test, if you want to test installation media before)

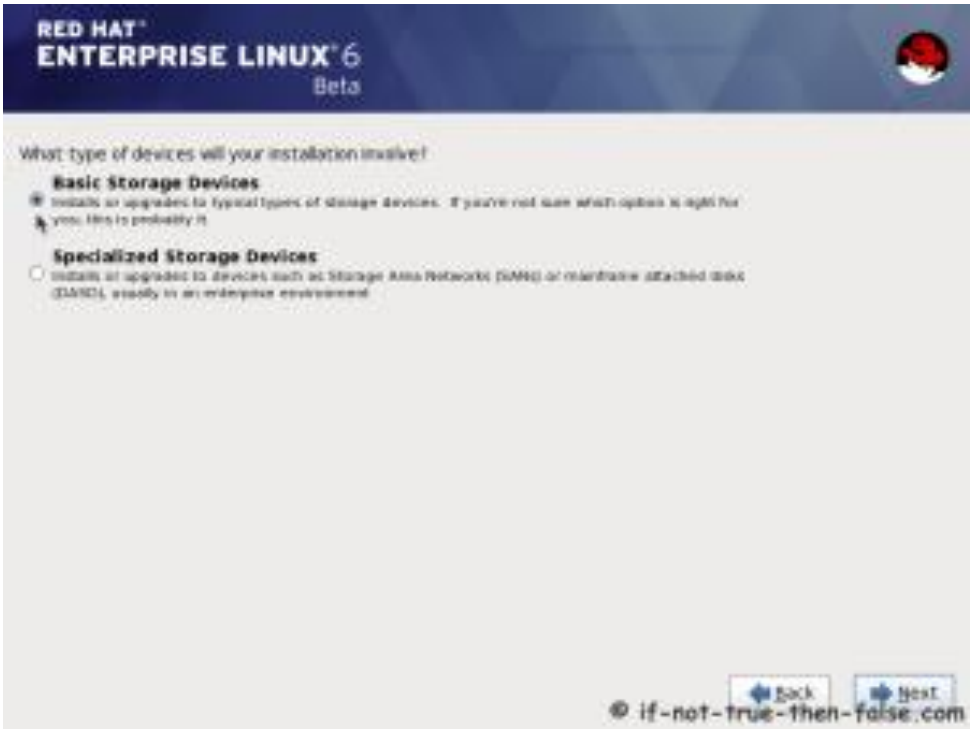


6. Red Hat 6 graphical installer starts, select next

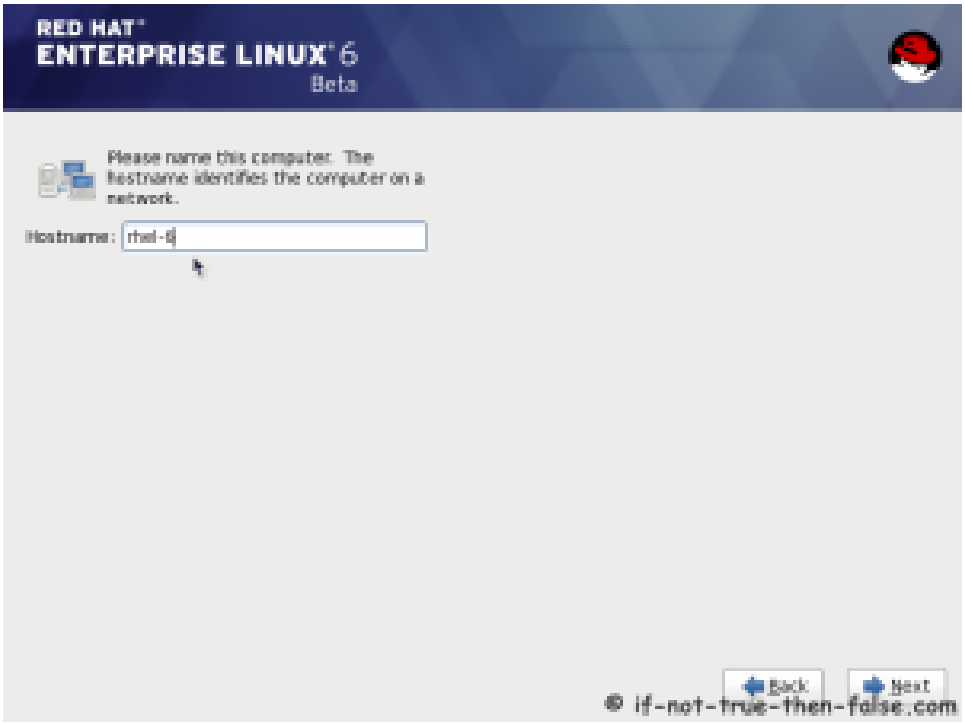
7. Accept Pre-Release Installation



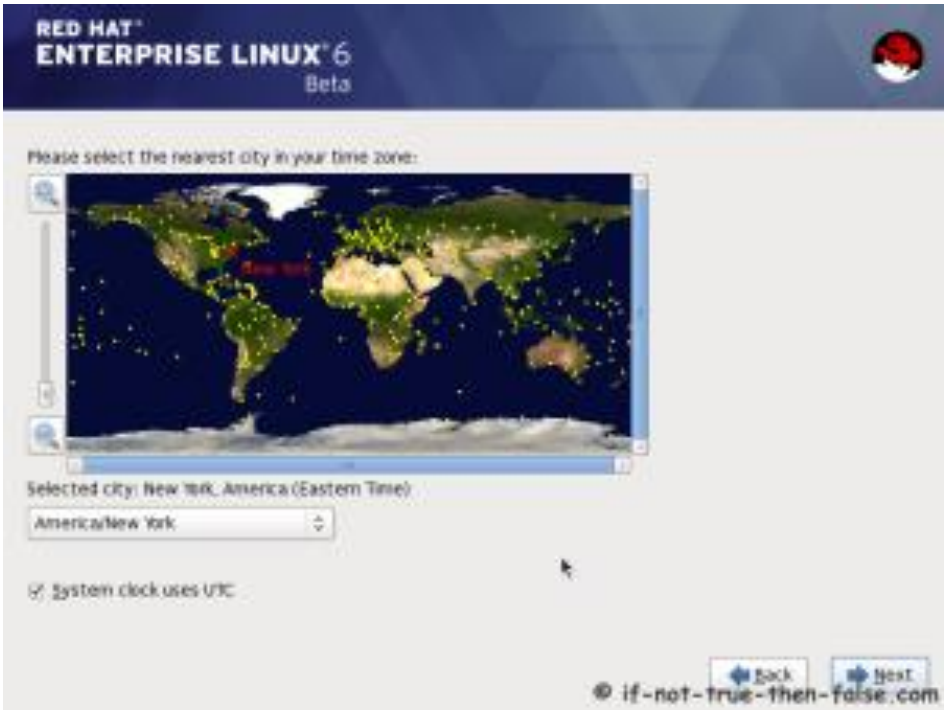
8. Select storage devices



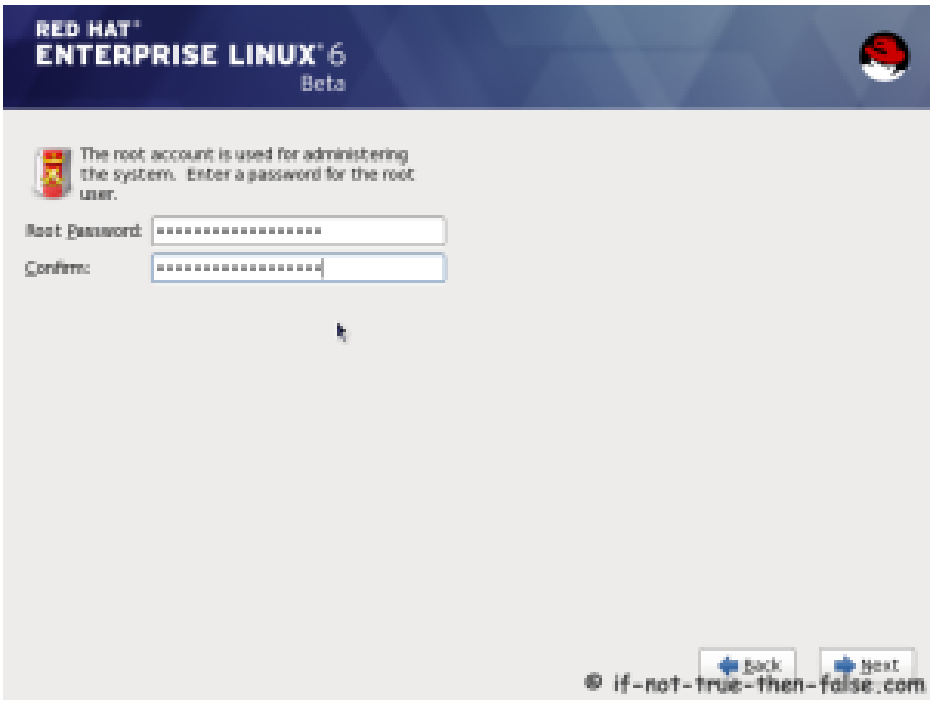
9. Insert computer name



10. Select time zone



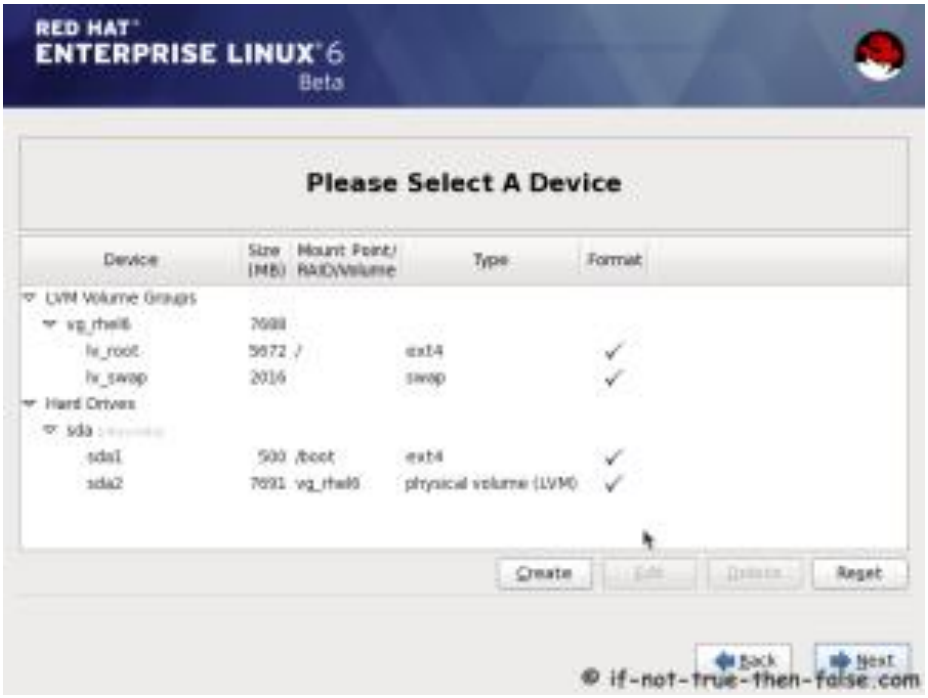
11. Enter a password for *root* user



12. Select type of installation Read every options info carefully. And select encrypting if eeded and option to review and modify partition layout.

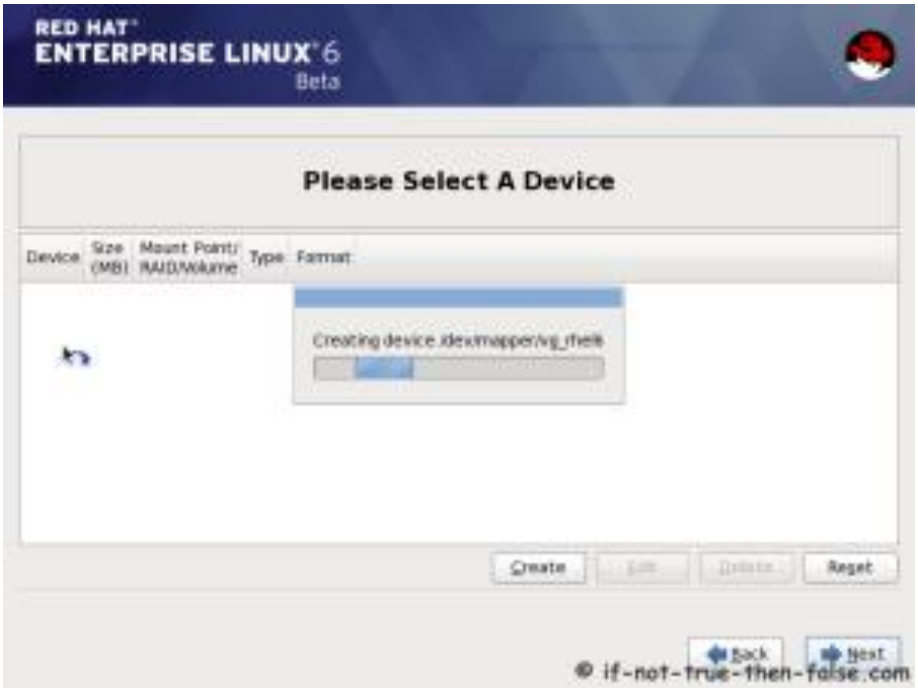
13. Review partition layout

Modify if needed. Default setup with ext4 and LVM looks good for desktop machine.



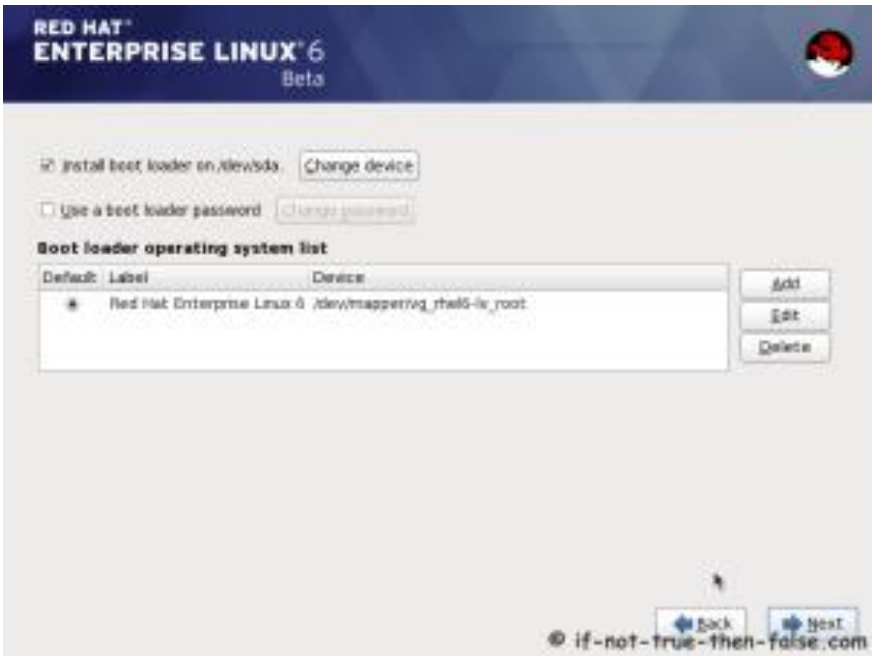
14. Accept write changes to disc

15. Writing changes (creating partitions) to disc

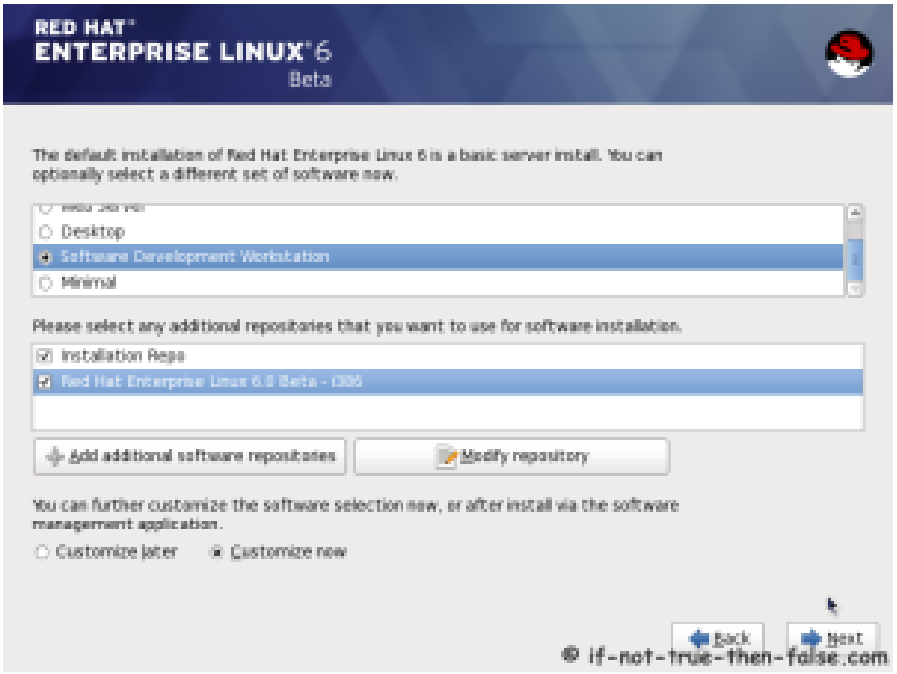


16. Configure boot loader options

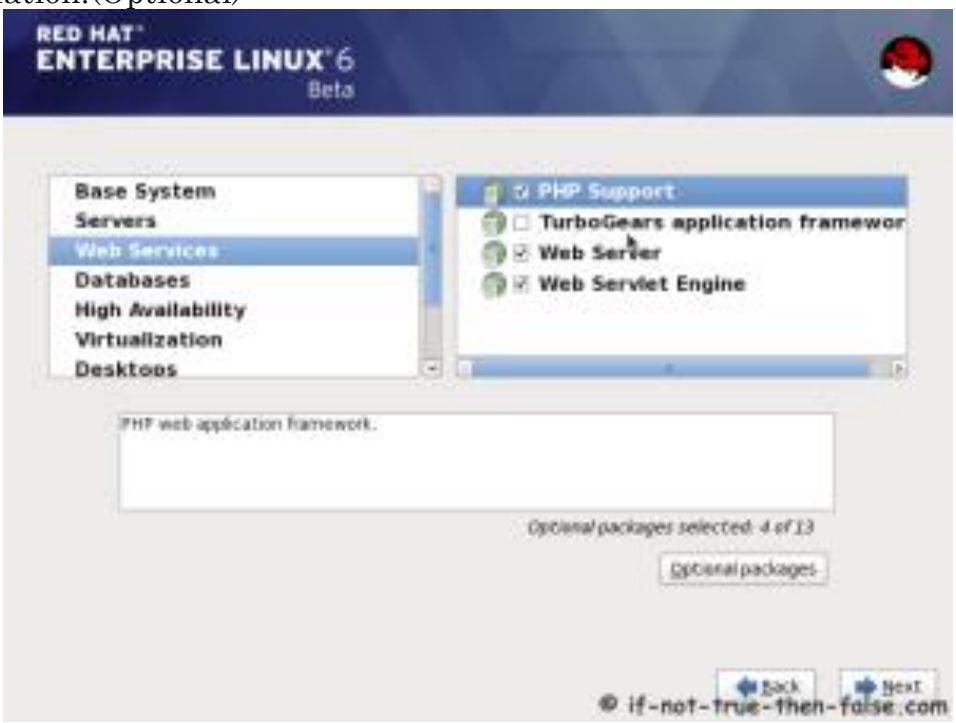
Select device to install bootloader and check/create boot loader operating system list.



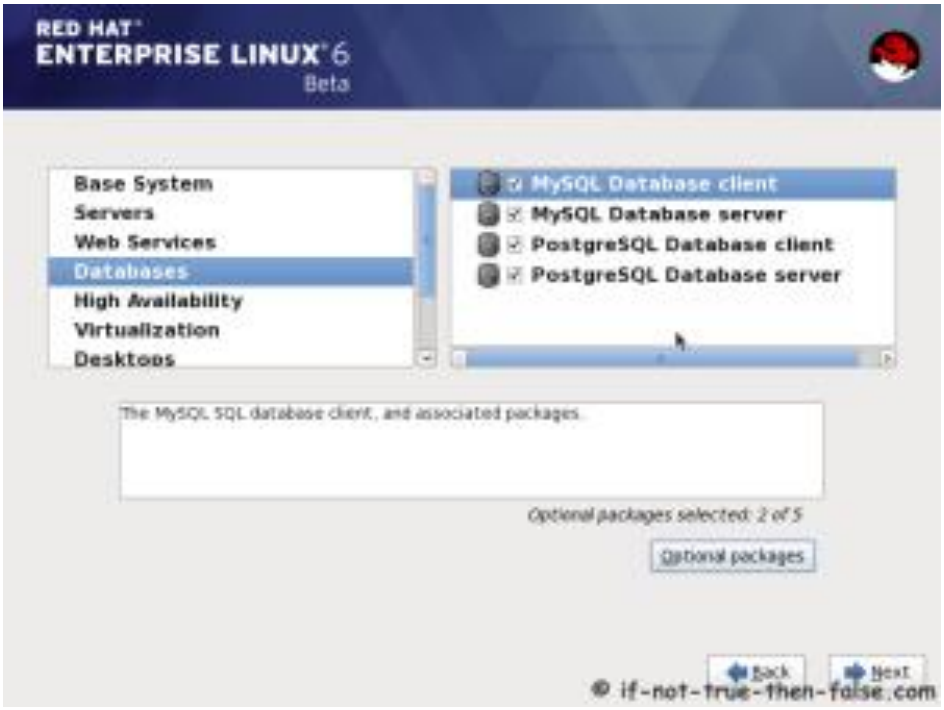
17. Select software's to install and enable repositories. This case we select Software Development Workstation and enable Red Hat Enterprise Linux 6.0 Beta Repository and select Customize now.



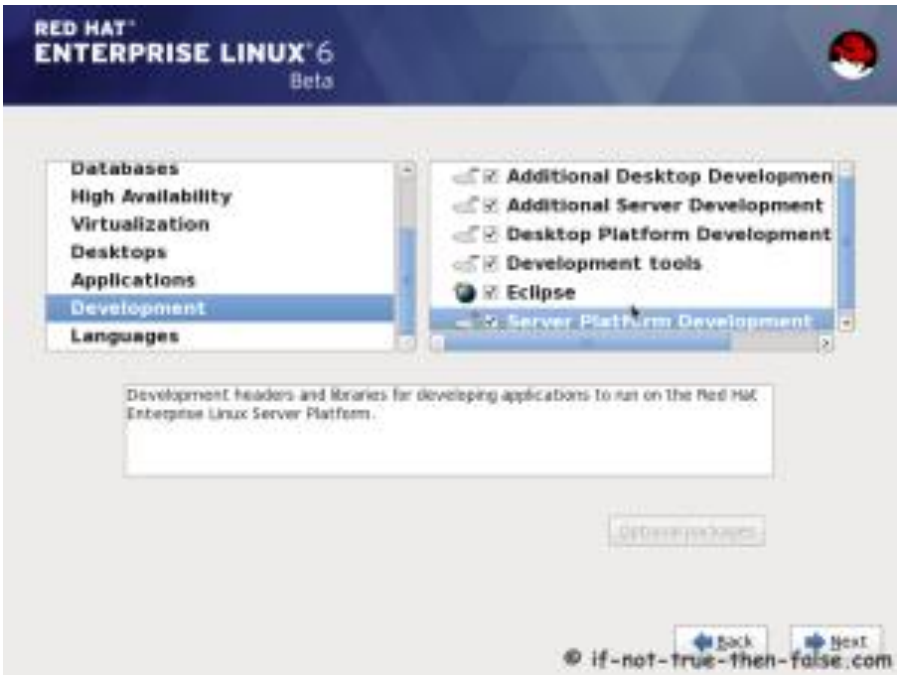
18. Customize package selection, Select PHP and Web Server to installation.(Optional)



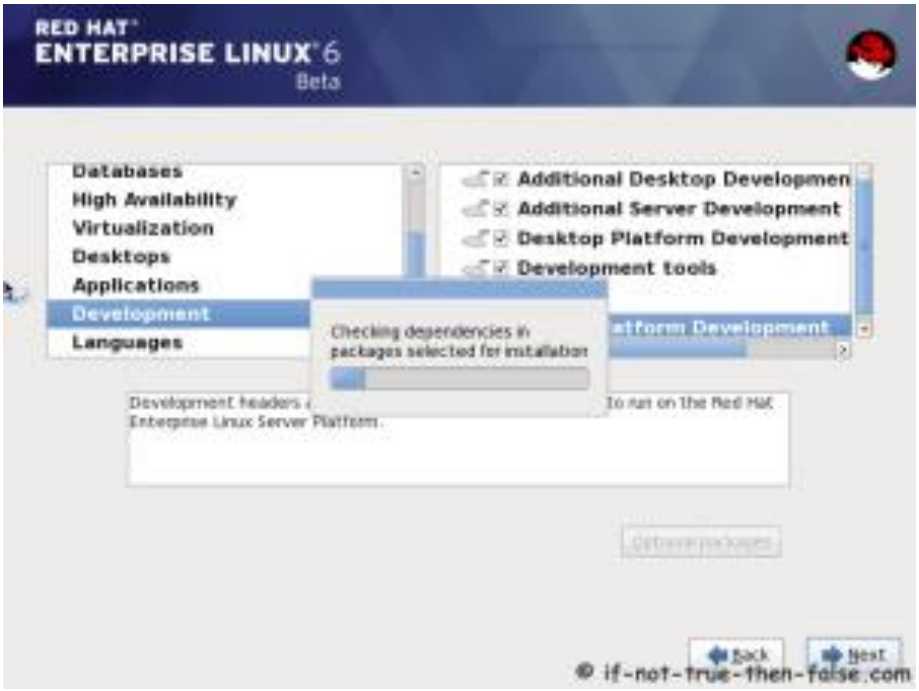
Select MySQL and PostgreSQL Databases. **(optional)**



Select set of Development tools like Eclipse IDE.



19. Checking dependencies for installation



20. Starting installation process

21. Installing packages

22. Installation is complete. Click reboot computer and remove installation media.

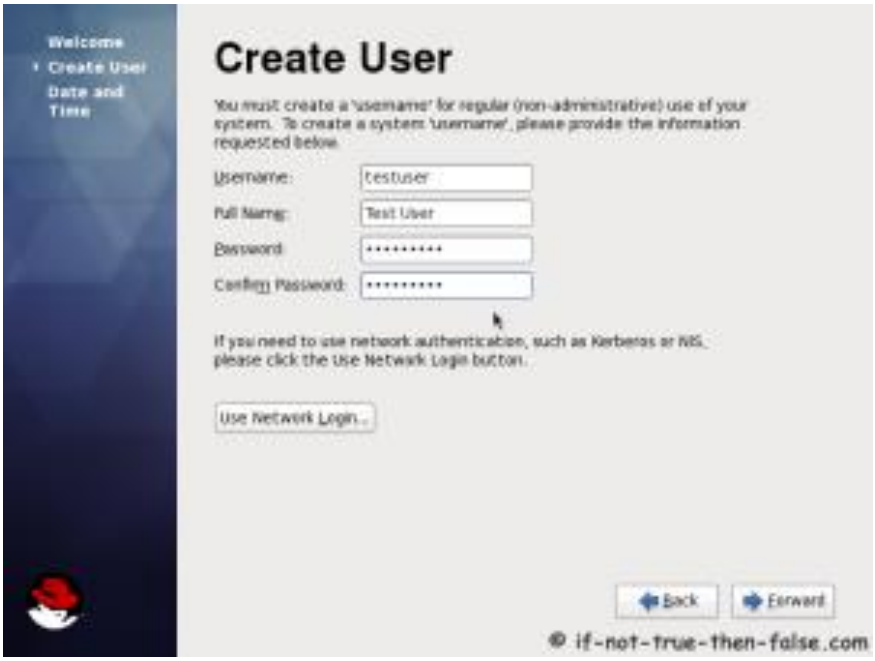
Red Hat 6 RHEL Finishing Installation

23. Selecting RHEL 6 from grub

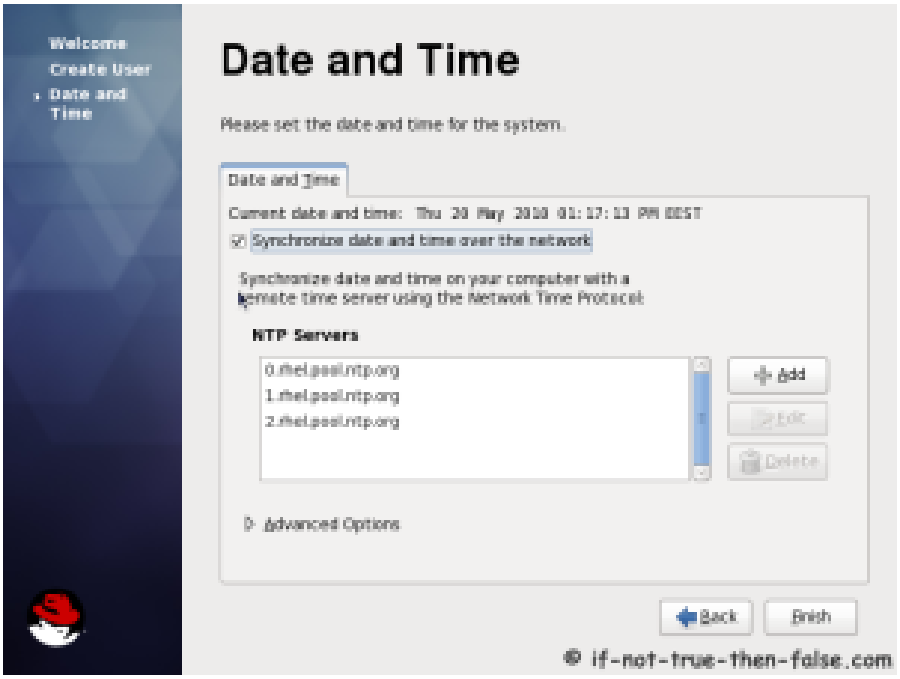
24. Booting Red Hat 6

25. Red Hat 6 Welcome screen

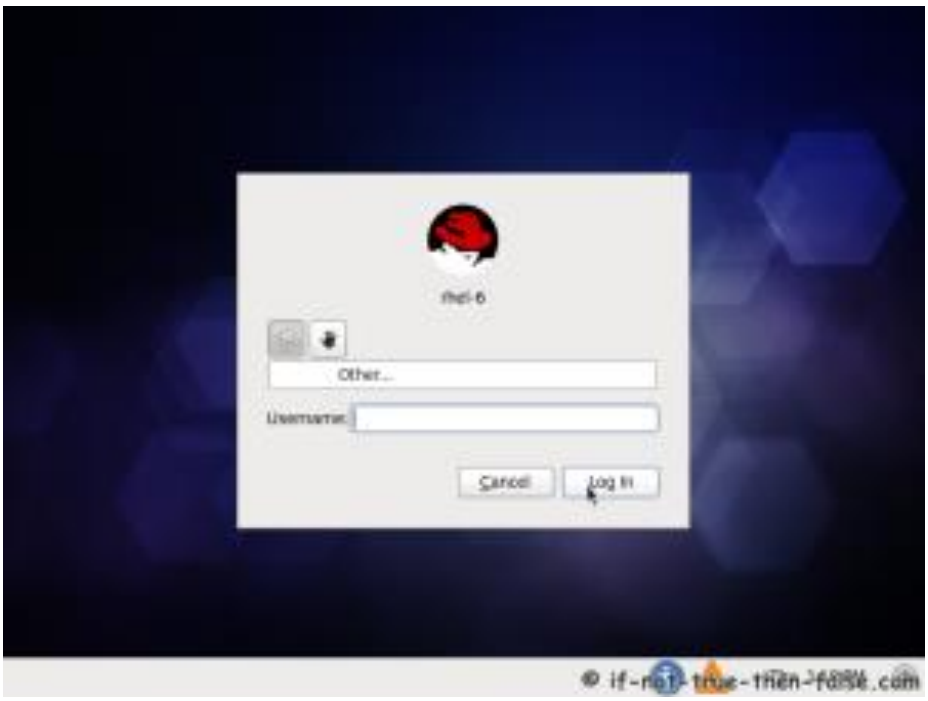
26. Create normal user



27. Setup date and time and keep up-to-date with NTP



28. Login Red Hat 6 Gnome Desktop



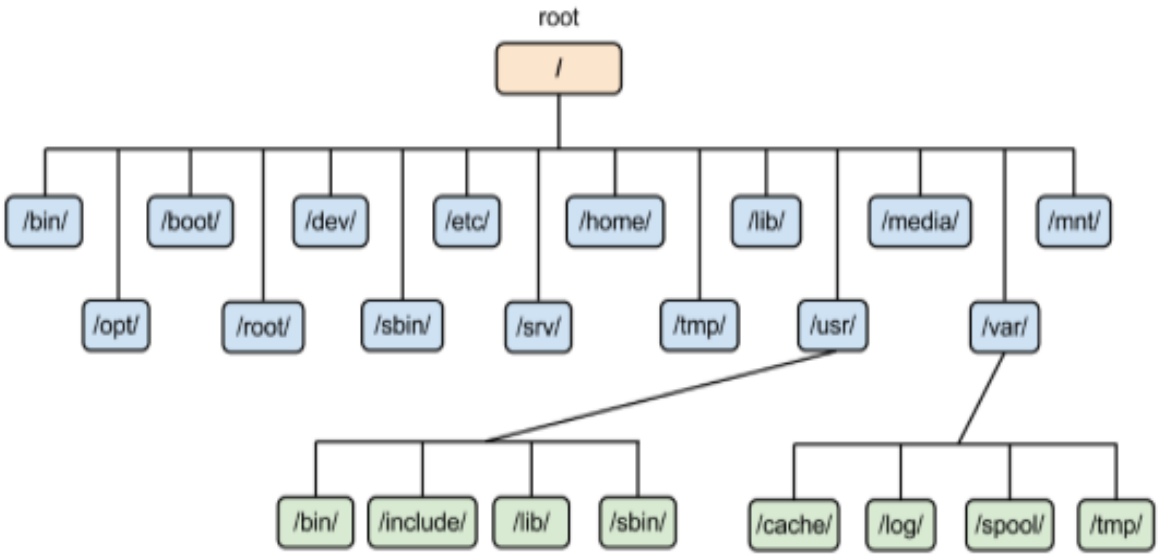
29. Red Hat (RHEL) 6 Gnome Desktop, empty and default look

File system Hierarchy

Definition

The Linux File Hierarchy Structure or the Filesystem Hierarchy Standard (FHS) defines the directory structure and directory contents in Unix-like operating systems.It is maintained by the Linux Foundation.

- In the FHS, all files and directories appear under the root directory /, even if they are stored on different
  Physical or virtual devices.
- Some of these directories only exist on a particular system if certain subsystems, such as the X Window
  System, are installed.
- Most of these directories exist in all UNIX operating systems and are generally used in much the same way;
  However, the descriptions here are those used specifically for the FHS, and are not considered authoritative for platforms other than Linux.



File system hierarchy of Linux

1. **/ (Root)** : Primary hierarchy root and root directory of the entire file system hierarchy.

    - Every single file and directory starts from the root directory
    - Only root user has the right to write under this directory
    - /root is root user's home directory, which is not same as /

2. **/bin** : Essential command binaries that need to be available in single user mode;
    for all users, e.g., cat, ls, cp.
    - 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 e.g. ps, ls, ping, grep,

- Following commands are the part of this directory.

```
cat       Utility to concatenate files to standard output
chgrp     Utility to change file group ownership
chmod     Utility to change file access permissions
chown     Utility to change file owner and group
cp        Utility to copy files and directories
date      Utility to print or set the system data and time
dd        Utility to convert and copy a file
df        Utility to report filesystem disk space usage
dmesg     Utility to print or control the kernel message buffer
echo      Utility to display a line of text
false     Utility to do nothing, unsuccessfully
hostname  Utility to show or set the system's host name
kill      Utility to send signals to processes
ln        Utility to make links between files
login     Utility to begin a session on the system
ls        Utility to list directory contents
mkdir     Utility to make directories
mknod     Utility to make block or character special files
more      Utility to page through text
mount     Utility to mount a filesystem
mv        Utility to move/rename files
ps        Utility to report process status
pwd       Utility to print name of current working directory
rm        Utility to remove files or directories
rmdir     Utility to remove empty directories
```

3. /boot/:

   This directory contains everything required for the boot process except for configuration files not needed at boot time (the most notable of those being those that belong to the GRUB boot-loader) and the map installer. Thus, the /boot directory stores data that is used before the kernel begins executing user-mode programs.

4. /dev/:

   It is the location of special or device files. It is a very interesting directory that highlights one important aspect of the Linux filesystem - everything is a file or a directory. The /dev/ directory contains file system entries which represent devices that are attached to the system. These files are essential for the system to function properly.

5. /etc :

   This is the nerve center of Linux system, it contains all system related configuration files in here or in its sub-directories. A "configuration file" is defined as a local file used to control the operation of a program; it must be static and cannot be an executable binary. For this reason, it's a good idea to backup this directory regularly. It will definitely save a lot of re-configuration later if one re-installs or lose current installation. Normally, no binaries should be or are located here.

6. /home:

   Linux is a multi-user environment so each user is also assigned a specific directory that is accessible only to them and the system administrator. These are the user home directories, which can be found under '/home/$USER' (~/). It is your playground: everything is at your command, you can write files, delete them, install programs, etc.... Your home directory contains your personal configuration files, the so-called dot files (their name is preceded by a dot). Personal configuration files are usually 'hidden', if you want to see them, you either have to turn on the appropriate option in your file manager or run ls with the -a switch. If there is a conflict between personal and system wide configuration files, the settings in the personal file will prevail.

7. /initrd:

   initrd provides the capability to load a RAM disk by the boot loader. This RAM disk can then be mounted as the root file system and programs can be run from it. Afterwards, a new root file system can be mounted from a different device. The previous root (from initrd) is then moved to a directory and can be subsequently unmounted. initrd is mainly designed to allow system startup to occur in two phases, where the kernel comes up with a minimum set of compiled-in drivers, and where additional modules are loaded from initrd.

8. /lib:

   The /lib directory contains kernel modules and those shared library images (the C programming code library) needed to boot the system and run the commands in the root filesystem, ie. by binaries in /bin and /sbin. Libraries are readily identifiable through their filename extension of *.so. Windows equivalent to a shared library would be a DLL (dynamically linked library) file. They are essential for basic system functionality. Kernel modules (drivers) are in the subdirectory /lib/modules/'kernel-version'. To ensure proper module compilation you should ensure that /lib/modules/'kernel-version'/kernel/build points to /usr/src/'kernel-version' or ensure that the Makefile knows where the kernel source itself are located.

9. /mnt/:

   The /mnt/ directory is for temporarily mounted file systems, such as CD-ROMs and 3.5 diskettes.

10. /opt/:

    The /opt/ directory provides storage for large, static application software packages. A package placing files in the /opt/ directory creates a directory

bearing the same name as the package. This directory, in turn, holds files that otherwise would be scattered throughout the file system, giving the system administrator an easy way to determine the role of each file within a particular package.

11. **/proc :**

The /proc/ directory contains special files that either extract information from or send information to the kernel.

12. **/sbin/:**

The /sbin/ directory stores executables used by the root user. The executables in /sbin/ are only used at boot time and perform system recovery operations. Of this directory, the FHS says:

/sbin contains binaries essential for booting, restoring, recovering, and/or repairing the system in addition to the binaries in /bin. Programs executed after /usr/ is known to be mounted (when there are no problems) are generally placed into

/usr/sbin. Locally-installed system administration programs should be placed into/usr/local/sbin

13. **/usr/ :**

The /usr/ directory is for files that can be shared across multiple machines. The /usr/ directory is often on its own partition and is mounted read-only.

Working with directories

In this session, we have covered an overview of the most common commands to work with directories : pwd, cd, ls, mkdir, rmdir. These commands are available on any Linux (or Unix) system.

How to create a directory

The mkdir command in UNIX allows users to create directories or folders as they are referred to in some OS's.The mkdir command can create multiple directories at once and also set permissions when creating the directory. To create a directory in UNIX or Linux using the mkdir command pass the name of directory to the mkdir command.

Syntax: mkdir <directoryname>

```
[root@Linux test]# mkdir mydirectory
[root@Linux test]# ls -ld mydirectory/
drwxr-xr-x 2 root root 4096 Sep 30 17:13 mydirectory/
[root@Linux test]#
```

How to create multiple directories

To create multiple directories in UNIX or Linux using the mkdir command pass the names of directories to be created to the mkdir command. The names of directories should be separated by spaces.

Syntax: mkdir <directoryname1> <directoryname2> <directoryname3>

```
[root@Linux test]# mkdir dir1 dir2 dir3
[root@Linux test]# ls -ld dir1 dir2 dir3
drwxr-xr-x 2 root root 4096 Sep 30 17:14 dir1
drwxr-xr-x 2 root root 4096 Sep 30 17:14 dir2
drwxr-xr-x 2 root root 4096 Sep 30 17:14 dir3
[root@Linux test]#
```

How to create parent directories

To create parent directories using the mkdir command pass the -p option. Suppose directory path /foo/bar/baz is to be created even if /foo/bar directory is not existing then you can use the below.

```
[root@Linux test]# ls -ld /foo/bar/baz
ls: cannot access /foo/bar/baz: No such file or directory
[root@Linux test]# mkdir /foo
[root@Linux test]# mkdir /foo/bar/baz
mkdir: cannot create directory '/foo/bar/baz': No such file or directory
[root@Linux test]# mkdir -p /foo/bar/baz
[root@Linux test]# ls -ld /foo/bar/baz/
drwxr-xr-x 2 root root 4096 Sep 30 17:16 /foo/bar/baz/
```

How to remove a directory

To remove an empty directory (folder), you can use the "rmdir" command.

Syntax:

rmdir directoryname

So if you had an empty directory named "mydirectory", you can remove it with:

```
[root@Linux test]# ls -ld mydirectory/
drwxr-xr-x 2 root root 4096 Sep 30 17:13 mydirectory/
[root@Linux test]# rmdir mydirectory
[root@Linux test]# ls -ld mydirectory/
ls: cannot access mydirectory/: No such file or directory
[root@Linux test]#
```

But, if the directory had files in it, you would not be able to remove it with "rmdir". If you tried removing a directory with files, you would get an error like this:

```
[root@Linux test]# ls -l mydir/
total 0
-rwx------ 1 root root 0 Sep 30 16:47 file1
-rwx------ 1 root root 0 Sep 30 16:47 file2
[root@Linux test]# rmdir mydir
rmdir: failed to remove `mydir': Directory not empty
[root@Linux test]#
```

Instead, you should use a different command

So, if we had a directory with subdirectories and files in it, we would use this syntax:

```
[root@Linux test]# rm -r mydir
rm: descend into directory `mydir'? y
rm: remove regular empty file `mydir/file1'? y
rm: remove regular empty file `mydir/file2'? y
rm: remove directory `mydir'? y
[root@Linux test]# ls -ld mydir
ls: cannot access mydir: No such file or directory
[root@Linux test]#
```

To remove a single file, you can use the "rm" command with no additional options:

Syntax: rm filename

- When using the "rm" command, you should not always use the "-f" (force) option. By using that option, you won't get any warnings or prompts when removing something.
- If you get a "permission denied" error or any other similar error, then you are probably using a non-root user. You should either use the root user or use sudo to run the commands. So just append "sudo" to each command and enter your sudo password when prompt:

  Syntax: sudo rm -r directoryname

Warning- once you delete something with rm, it's gone. Linux assumes you are smart and you know what you are doing.

Permissions and rights

As Linux is a multi-user operating system which can be accessed by many users simultaneously. Hence this raises security concerns as an unsolicited or malign user can corrupt, change or remove crucial data. For effective security, Linux divides authorization into 2 levels.

1. Ownership
2. Permission

How to change the owner ship

The chown command changes ownership of files and directories in a Linux filesystem.

To change the owner of a file/directory

Syntax: chown <ownername> <filename>

```
[root@Linux ~]# ls -l file.txt
-rw-r--r-- 1 root root 0 Sep 30 15:39 file.txt
[root@Linux ~]# chown ec2-user file.txt
[root@Linux ~]# ls -l file.txt
-rw-r--r-- 1 ec2-user root 0 Sep 30 15:39 file.txt
[root@Linux ~]#
```

To change the group of a file

"chown" command is also used to change group name of a file, like "chgrp" command.

Syntax: chown <:groupname> <filename>

```
[root@Linux ~]# ls -l file.txt
-rw-r--r-- 1 ec2-user root 0 Sep 30 15:39 file.txt
[root@Linux ~]# chown :ec2-user file.txt
[root@Linux ~]# ls -l file.txt
-rw-r--r-- 1 ec2-user ec2-user 0 Sep 30 15:39 file.txt
[root@Linux ~]#
```

To recursively change ownership of directories and their contents

Syntax: chown –R root <filename>

```
[root@Linux ~]# mkdir test
[root@Linux ~]# ls -ld test
drwxr-xr-x 2 root root 4096 Sep 30 15:56 test
[root@Linux ~]# cd test
[root@Linux test]# touch file1 file2 file3
[root@Linux test]# ls -l *
-rw-r--r-- 1 root root 0 Sep 30 15:57 file1
-rw-r--r-- 1 root root 0 Sep 30 15:57 file2
-rw-r--r-- 1 root root 0 Sep 30 15:57 file3
[root@Linux test]# cd ..
[root@Linux ~]# ls -ld test
drwxr-xr-x 2 root root 4096 Sep 30 15:57 test
[root@Linux ~]# chown -R ec2-user:ec2-user test
[root@Linux ~]# ls -l test
total 0
-rw-r--r-- 1 ec2-user ec2-user 0 Sep 30 15:57 file1
-rw-r--r-- 1 ec2-user ec2-user 0 Sep 30 15:57 file2
-rw-r--r-- 1 ec2-user ec2-user 0 Sep 30 15:57 file3
[root@Linux ~]#
```

DVS Technologies Aws & Devops

How to change the permission

Unix-like systems, including the Linux systems that run on the inode platform, have an incredibly robust access control system that allows systems administrators to effectively permit multiple users access to a single system without giving every user access to every file on the file system. The chmod command is the best and easiest way to modify these files permissions.

All file system objects on Unix-like systems have three main types of permissions: **read**, **write**, and **execute access**. Permissions are based upon three possible classes: the **user**, the **usergroup**, and **all system users.**

To view the file permissions of a set of files, use:

Syntax : ls -lha

In the first column of the output, there are 10 characters that represent the permission bits.
To view permissions for a file we use the long listing option for the command ls.

Syntax: ls –l <path of file>

```
[root@Linux test]# ls -l
total 4
-rw-r--r-- 1 ec2-user ec2-user    0 Sep 30 15:57 file1
-rw-r--r-- 1 ec2-user ec2-user    0 Sep 30 15:57 file2
-rw-r--r-- 1 ec2-user ec2-user    0 Sep 30 15:57 file3
drwxr-xr-x 2 root     root     4096 Sep 30 16:01 mydir
[root@Linux test]#
```

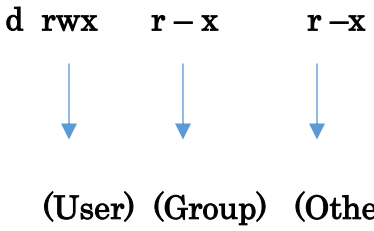
The first character represents the type whether it is a file or directory or symbolic link file. In The remaining nine bits first three represent the permissions for the user, rest 3 bits represents permissions for group, and other three for all system users. Each stands for:

- r: Read → 4
- w: Write → 2
- x: Execute → 1

Read will have a value of 4, Write will have 2 & execute will have 1

Examples

column lets divide the permissions



Setting 442 permissions for the file test.txt. Here first 4 value if for User, the second 4 represents for Group & final 2 is for all other machine users.

```
[root@Linux test]# ls -l test.txt
-rwxrwxrwx 1 root root 0 Sep 30 16:45 test.txt
[root@Linux test]# chmod 442 test.txt
[root@Linux test]# ls -l test.txt
-r--r---w- 1 root root 0 Sep 30 16:45 test.txt
[root@Linux test]#
```

How to change the permissions & owner ships recursively

To change file access permissions you need to use the chmod command. It has -R or –recursive option that change files and directories recursively.

Syntax : chmod –R <Directory name>

Example chmod –R 755 sample1.txt

```
[root@Linux mydir]# pwd
/root/test/mydir
[root@Linux mydir]# ls -l
total 0
-rw-r--r-- 1 root root 0 Sep 30 16:47 file1
-rw-r--r-- 1 root root 0 Sep 30 16:47 file2
[root@Linux mydir]# cd ..
[root@Linux test]# ls -ld mydir
drwxr-xr-x 2 root root 4096 Sep 30 16:47 mydir
[root@Linux test]# chmod -R 700 mydir
[root@Linux test]# ls -l mydir/
total 0
-rwx------ 1 root root 0 Sep 30 16:47 file1
-rwx------ 1 root root 0 Sep 30 16:47 file2
[root@Linux test]# ls -ld mydir/
drwx------ 2 root root 4096 Sep 30 16:47 mydir/
[root@Linux test]#
```

DVS Technologies Aws & Devops

Unix/Linux Command Reference

File Commands

ls – directory listing
ls -al – formatted listing with hidden files
cd *dir* - change directory to *dir*
cd – change to home
pwd – show current directory
mkdir *dir* – create a directory *dir*
rm *file* – delete *file*
rm -r *dir* – delete directory *dir*
rm -f *file* – force remove *file*
rm -rf *dir* – force remove directory *dir* *
cp *file1 file2* – copy *file1* to *file2*
cp -r *dir1 dir2* – copy *dir1* to *dir2*; create *dir2* if it doesn't exist
mv *file1 file2* – rename or move *file1* to *file2* if *file2* is an existing directory, moves *file1* into directory *file2*
ln -s *file link* – create symbolic link *link* to *file*
touch *file* – create or update *file*
cat > *file* – places standard input into *file*
more *file* – output the contents of *file*
head *file* – output the first 10 lines of *file*
tail *file* – output the last 10 lines of *file*
tail -f *file* – output the contents of *file* as it grows, starting with the last 10 lines

Process Management

ps – display your currently active processes
top – display all running processes
kill *pid* – kill process id *pid*
killall *proc* – kill all processes named *proc* *

System Info

date – show the current date and time
cal – show this month's calendar
uptime – show current uptime
w – display who is online
whoami – who you are logged in as
finger *user* – display information about *user*
uname -a – show kernel information
cat /proc/cpuinfo – cpu information
cat /proc/meminfo – memory information
man *command* – show the manual for *command*
df – show disk usage
du – show directory space usage
free – show memory and swap usage
whereis *app* – show possible locations of *app*
which *app* – show which *app* will be run by default

Compression

tar cf *file.tar files* – create a tar named *file.tar* containing *files*
tar xf *file.tar* – extract the files from *file.tar*
tar czf *file.tar.gz files* – create a tar with Gzip compression
tar xzf *file.tar.gz* – extract a tar using Gzip
tar cjf *file.tar.bz2* – create a tar with Bzip2 compression
tar xjf *file.tar.bz2* – extract a tar using Bzip2
gzip *file* – compresses *file* and renames it to *file.gz*
gzip -d *file.gz* – decompresses *file.gz* back to *file*

bg – lists stopped or background jobs; resume a stopped job in the background
fg – brings the most recent job to foreground
fg *n* – brings job *n* to the foreground

File Permissions

chmod *octal file* – change the permissions of *file* to *octal*, which can be found separately for user, group, and world by adding:

- 4 – read (r)
- 2 – write (w)
- 1 – execute (x)

Examples:
chmod 777 – read, write, execute for all
chmod 755 – rwx for owner, rx for group and world
For more options, see **man chmod**.

SSH

ssh *user@host* – connect to *host* as *user*
ssh -p *port user@host* – connect to *host* on port *port* as *user*
ssh-copy-id *user@host* – add your key to *host* for *user* to enable a keyed or passwordless login

Searching

grep *pattern files* – search for *pattern* in *files*
grep -r *pattern dir* – search recursively for *pattern* in *dir*
command* | grep *pattern – search for *pattern* in the output of *command*
locate *file* – find all instances of *file*

Network

ping *host* – ping *host* and output results
whois *domain* – get whois information for *domain*
dig *domain* – get DNS information for *domain*
dig -x *host* – reverse lookup *host*
wget *file* – download *file*
wget -c *file* – continue a stopped download

Installation

Install from source:
./configure
make
make install
dpkg -i *pkg.deb* – install a package (Debian)
rpm -Uvh *pkg.rpm* – install a package (RPM)

Shortcuts

Ctrl+C – halts the current command
Ctrl+Z – stops the current command, resume with **fg** in the foreground or **bg** in the background
Ctrl+D – log out of current session, similar to **exit**
Ctrl+W – erases one word in the current line
Ctrl+U – erases the whole line
Ctrl+R – type to bring up a recent command
!! - repeats the last command
exit – log out of current session

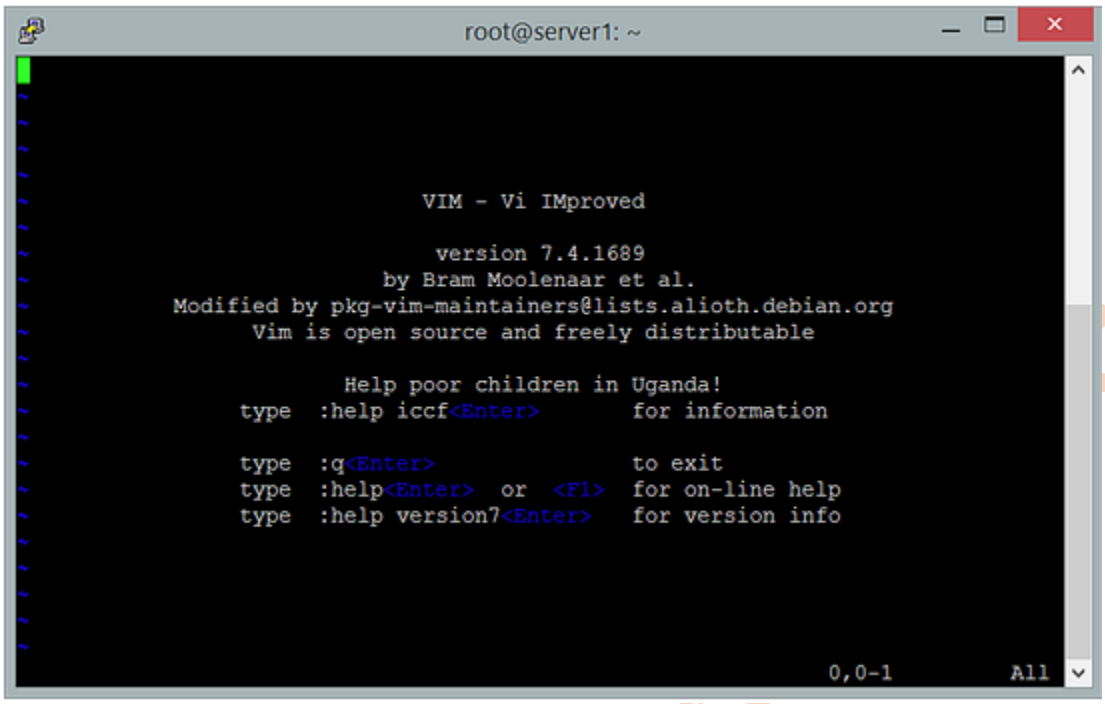
* use with extreme caution.

Vim Editor

Vim is a powerful text editor used in CLI (command line interface). Linux uses a lot of configuration files, you'll often need to edit them and vim is a great tool to do so. Alternatives to vim are the command line editors such as vi, nano and joe.

VI: Visual display editor.

VIM: Visual display editor improved.



It has 3 modes:

1. Command Mode
2. Insert Mode (edit mode)
3. Extended Command Mode

Note: When you open the vim editor, it will be in the command mode by default.

In the command mode, the cursor's can be used as: h/l/k/j to move cursor left/right/up/down.

Inset Mode

i	To begin insert mode at the cursor position
I	To insert at the beginning of line
a	To append to the next word's letter
A	To Append at the end of the line

o	To insert a new line below the cursor position
O	To insert a new line above the cursor position

Command Mode

gg	To go to the beginning of the page
Shift+g	To go to end of the page
w	To move the cursor forward, word by word
b	To move the cursor backward, word by word
nw	To move the cursor forward to n words (5W)
nb	To move the cursor backward to n words (5B)
u	To undo last change (word)
Ctrl+u	To undo the previous changes (entire line)
Ctrl+r	To redo the changes
yy	To copy a line
nyy	To copy n lines (5yy or 4yy)
p	To paste line below the cursor position
P	To paste line above the cursor position
dw	To delete the word letter by letter (like Backspace)
x	To delete the world letter by letter (like DEL Key)
dd	To delete entire line
ndd	To delete n no. of lines from cursor position(5dd)
/	To search a word in the file

Extended Mode (Colon Mode)

Extended Mode is used for save and quit or save without quit using "Esc" Key with ":" symbol.

Esc+:w	To Save the changes
Esc+:q	To quit (Without saving)
Esc+:wq	To save and quit
Esc+:w!	To save forcefully
Esc+wq!	To save and quit forcefully
Esc+:x	To save and quit
Esc+:X	To give password to the file and remove password
Esc+:20(n)	To go to line no 20 or n
Esc+: se nu	To set the line numbers to the file
Esc+:se nonu	To Remove the set line numbers

To open multiple files in vim editor:

#vim –o file1 file2

To switch between files use, Ctrl +w.

Monitoring

Top command

- When you need to see the running processes on your Linux in real time, you have top as your tool for that.
- 'top' also displays other information besides the running processes, like free memory both physical and swap.

DVS Technologies Aws & Devops

Monitoring all process using Top command

To monitor all processes in the system use the following command
#top :

```
[root@python ~]# top
top - 16:42:37 up 1 min,  1 user,  load average: 0.00, 0.00, 0.00
Tasks:  80 total,   1 running,  53 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   1011176k total,   234936k used,   776240k free,     9304k buffers
Swap:        0k total,        0k used,        0k free,   168752k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 2278 root      20   0  6520  100    0 S  0.3  0.0   0:00.02 rngd
 2653 root      20   0 15356 2236 1956 R  0.3  0.2   0:00.01 top
    1 root      20   0 19692 2628 2300 S  0.0  0.3   0:01.10 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.00 kthreadd
    3 root      20   0     0    0    0 I  0.0  0.0   0:00.00 kworker/0:0
    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   0:00.01 kworker/u30:0
    6 root       0 -20     0    0    0 I  0.0  0.0   0:00.00 mm_percpu_wq
    7 root      20   0     0    0    0 S  0.0  0.0   0:00.01 ksoftirqd/0
    8 root      20   0     0    0    0 I  0.0  0.0   0:00.01 rcu_sched
    9 root      20   0     0    0    0 I  0.0  0.0   0:00.00 rcu_bh
```

First Line of top command

```
top - 16:43:01 up 2 min,  1 user,  load average: 0.00, 0.00, 0.00
```

- "016:43:01" is the current time; "up 2 mins" shows how long the system has been up for; "1 user" how many users are logged in; "load average: 0.00, 0.00, 0.00" the load average of the system (1minute, 5 minutes, 15 minutes).

Second Line of top command

```
Tasks:  80 total,   1 running,  53 sleeping,   0 stopped,   0 zombie
```

Shows the number of processes and their current state.

Third Line of top command

```
Cpu(s): 10.2%us,  0.4%sy,  0.0%ni, 88.2%id,  0.3%wa,  0.0%hi,  0.0%si,  0.9%st
```

Shows CPU utilization details. “10.2%us” user processes are using, “0.4%sy” system processes are using, 88.2% of system idle cpu.

Fourth and fifth Lines of top command

```
Mem:   1011176k total,   243932k used,   767244k free,    11912k buffers
Swap:        0k total,        0k used,        0k free,   172112k cached
```

“1011176k total” is total memory in the system; “243932K used” is the part of the RAM that currently contains information; “767244k free” is the part of RAM that contains no information; “11912K buffers and 172112k cached” is the buffered and cached data for IO.

By default, top starts by showing the following task's property:

Field	Description
PID	Process ID
USER	Effective User ID
PR	Dynamic priority
NI	Nice value, also known as base priority
VIRT	Virtual Size of the task. This includes the size of process's executable binary, the data area and all the loaded shared libraries.
RES	The size of RAM currently consumed by the task. Swapped out portion of the task is not included.
SHR	Some memory areas could be shared between two or more task, this field reflects that shared areas. The example of shared area are shared library and SysV shared memory.
S	Task status
%CPU	The percentage of CPU time dedicated to run the task since the last top's screen update.
%MEM	The percentage of RAM currently consumed by the task.
TIME+	The total CPU time the task has been used since it started. "+" sign means it is displayed with hundredth of a second granularity. By default, TIME/TIME+ doesn't account the CPU time used by the task's dead children.
Command	Showing program names.

Interacting with TOP:

Now that we are able to understand the output from TOP lets learn how to change the way the output is displayed.

Just press the following key while running top and the output will be sorted in real time.

- M – Sort by memory usage
- P – Sort by CPU usage
- T – Sort by cumulative time
- z – Color display
- k – Kill a process
- q – quit
- r – to renice a process
- h – help

Ping Command

Ping (Packet Internet Groper) command is used to check the network connectivity between host and server/host. Ping is generally measured in millisecond every modern operating system has this ping pre-installed.

ICMP (Internet Control Message Protocol) to send an ICMP echo message to the specified host if that host is available then it sends ICMP reply message.

TTL (Time To Live) It is maximum hop a packet can travel before getting discarded. A value 0 will restrict packet to same host.

Now, try to ping with hostname both server and client:

Syntax: # ping <Servername/Ipaddress>

#ping –c2 google.com
#ping –c2 anyclient

```
[root@python ~]# ping -c2 google.com
PING google.com (172.217.0.14) 56(84) bytes of data.
64 bytes from ord38s04-in-f14.1e100.net (172.217.0.14): icmp_seq=1 ttl=44 time=18.2 ms
64 bytes from ord38s04-in-f14.1e100.net (172.217.0.14): icmp_seq=2 ttl=44 time=18.3 ms

--- google.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 18.290/18.309/18.328/0.019 ms
```

```
[root@python ~]# ping -c2 gmail.com
PING gmail.com (216.58.192.197) 56(84) bytes of data.
64 bytes from ord30s25-in-f197.1e100.net (216.58.192.197): icmp_seq=1 ttl=44 time=18.6 ms
64 bytes from ord30s25-in-f5.1e100.net (216.58.192.197): icmp_seq=2 ttl=44 time=18.6 ms

--- gmail.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 18.602/18.606/18.611/0.136 ms
```

Note: Here "c" represents no.of hops you want as output. In our case we gave 2, hence we are getting hops and then its getting exited.

Vmstat command

vmstat (virtual memory statistics) is a valuable monitoring utility, which also provides information about block IO and CPU activity in addition to memory.

Vmstat basics

vmstat provides a number of values and will typically be called using two numerical parameters.

Example: vmstat 1 5

1 -> the values will be re-mesured and reported every second

5 -> the values will be reported five times and then the program will stop

The first line of the report will contain the average values since the last time the computer was rebooted. All other lines in the report will represent their respective current values. Vmstat does not need any special user rights. It can run as a normal user.

```
[root@python ~]# vmstat 2 5
procs -----------memory---------- ---swap-- -----io---- --system-- -----cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 0  0      0 753248  14728 174996    0    0    84     6   28   45  5  0 94  0  0
 0  0      0 753248  14728 174996    0    0     0     0   11   20  0  0 100  0  0
 0  0      0 753248  14728 174996    0    0     0     0    9   17  0  0 100  0  0
 0  0      0 753248  14728 174996    0    0     0     0    9   15  0  0 100  0  0
 0  0      0 753248  14728 174996    0    0     0     0   11   19  0  0 100  0  0
[root@python ~]#
```

DVS Technologies Aws & Devops

User administration

User management is an important part of Linux system administration. We need to perform different operations such as adding, deleting, managing the accounts which are part of the servers.

Type of User in Linux OS

- **Super User:** This account is nothing but "root" account who is holding super power in doing anything as part of the machine. Super user has all the privileges in Linux operating system. So, Super user does all the administration tasks such as stop or starts any service, grant or revokes permissions, open ports especially less than 1024 ports, user management and much more.

- **System User:** System users are created by system such as bin, games, ftp, name, mail, daemon, apache etc. These types of user are different service user and required for running different services. System user cannot login to the system because by default their login shell is **nologin**.

- **Regular User:** These users are created by super user. Regular user can login to the system but has access limitations. Regular user cannot do administration tasks. If super user provides permission, regular user can do permitted administration tasks. A regular user may be an ftp user, a samba user or a mail user.

"/etc/passwd" is the file which is holding all the information about the system users in the machine.

```
[root@Linux ~]# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
rpc:x:32:32:Rpcbind Daemon:/var/lib/rpcbind:/sbin/nologin
ntp:x:38:38::/etc/ntp:/sbin/nologin
saslauth:x:499:76:"Saslauthd user":/var/empty/saslauth:/sbin/nologin
mailnull:x:47:47::/var/spool/mqueue:/sbin/nologin
smmsp:x:51:51::/var/spool/mqueue:/sbin/nologin
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
ec2-user:x:500:500:EC2 Default User:/home/ec2-user:/bin/bash
demo:x:501:501::/home/demo:/bin/bash
[root@Linux ~]#
```

How to Add/Create User in Linux (useradd)

Inorder to create a user in linux we have to use a command called “useradd”. It accepts many arguments some of them are below.

```
Options:
  -b, --base-dir BASE_DIR       base directory for the home directory of the
                                new account
  -c, --comment COMMENT         GECOS field of the new account
  -d, --home-dir HOME_DIR       home directory of the new account
  -D, --defaults                print or change default useradd configuration
  -e, --expiredate EXPIRE_DATE  expiration date of the new account
  -f, --inactive INACTIVE       password inactivity period of the new account
  -g, --gid GROUP               name or ID of the primary group of the new
                                account
  -G, --groups GROUPS           list of supplementary groups of the new
                                account
  -h, --help                    display this help message and exit
  -k, --skel SKEL_DIR           use this alternative skeleton directory
  -K, --key KEY=VALUE           override /etc/login.defs defaults
  -l, --no-log-init             do not add the user to the lastlog and
                                faillog databases
  -m, --create-home             create the user's home directory
  -M, --no-create-home          do not create the user's home directory
  -N, --no-user-group           do not create a group with the same name as
                                the user
  -o, --non-unique              allow to create users with duplicate
                                (non-unique) UID
  -p, --password PASSWORD       encrypted password of the new account
  -r, --system                  create a system account
  -s, --shell SHELL             login shell of the new account
  -u, --uid UID                 user ID of the new account
  -U, --user-group              create a group with the same name as the user
  -Z, --selinux-user SEUSER     use a specific SEUSER for the SELinux user mapping
```

So based up on the requirement you can start using these options, for suppose if you want to use the shell you need to use " **-s**" option, similary if you want to write the comments then you can use "**-c**". Please check the below execution of create a user in the machine.

```
[root@Linux ~]# grep dvsmaterial /etc/passwd
[root@Linux ~]# useradd -s /bin/bash -c "Hi Team this how our material looks like" \
> -d /home/dvsmaterial -u 1000 -g ec2-user dvsmaterial
[root@Linux ~]# grep dvsmaterial /etc/passwd
dvsmaterial:x:1000:500:Hi Team this how our material looks like:/home/dvsmaterial:/bin/bash
[root@Linux ~]#
```

Explanation:

Here if you are observing before our "**useradd**" command execution we don't have our user "**dvsmaterial**" but once we created the user we could see the entries in the file /etc/passwd. What happens when we executed "useradd" command .

- It adds a new entry in both **/etc/passwd** file and **/etc/shadow** file.
- It also adds a new entry in **/etc/group** file and **/etc/gshadow** file.
- A home directory is created in **/home** directory for the new user.
- Permissions and ownership are also set to home directory by this command.

Now lets try to understand the output, here each and every coloumn in the /etc/passwd file has its own identity. They are as follows.

```
[root@Linux ~]# grep dvsmaterial /etc/passwd
dvsmaterial:x:1000:500:Hi Team this how our material looks like:/home/dvsmaterial:/bin/bash
[root@Linux ~]#
```

Field No	Field Name	Field Value	Explanation
1	Username	**dvsmaterial**	This is the username that is provided while creating a user. It is used to login into the system and it should be between 1to 32 characters as well as unique.
2	Password	x	An 'x' character indicates that encrypted password is stored in /etc/shadow file. If we put '*' inplace of x, the user cannot login. If we keep second field blank, the user can login without password.

3	User ID	1000	Every user must have a User Identification Number (UID). Generally, UID = 0 is reserved for the super user that is root. UID 1 to 99 are reserved for other predefined user accounts. Further UID 100 to 999 is reserved for the system users. Regular user's UID starts from 1000. As **dvsmaterial** is a regular user here, The UID is showing 1000. If UID of any regular user is changed to 0, the user is considered as a super user in your system.
4	Group ID	500	This is the primary Group Identification Number (GID) of this user. Whenever a user is created a primary group is also created for that user. Every user has its own primary group but it can also have supplementary groups.
5	Comments or User Info	Hi Team this how our material looks like	This is an optional field and only used for informational purpose. Usually it contains the full name of the user or any user comment can be put. This field is filled by *finger* command.
6	Home Directory	/home/ **dvsmaterial**	This is the absolute path of the user's home directory. If this field value is not present, the '/' root directory becomes the home directory of the user.
7	User Shell	/bin/bash	This is the absolute path of the user command shell.

User attribute modifications (usermod)

If you want to change the user related permissions in the Linux then you can use "usermod" option. For suppose I want to change the comments of user **"dvsmaterial" then it will looks like below.**

```
[root@Linux ~]# grep dvsmaterial /etc/passwd
dvsmaterial:x:1000:500:Hi Team this how our material looks like:/home/dvsmaterial:/bin/bash
[root@Linux ~]# usermod -c "Hi Everyone this is Linux" dvsmaterial
[root@Linux ~]# grep dvsmaterial /etc/passwd
dvsmaterial:x:1000:500:Hi Everyone this is Linux:/home/dvsmaterial:/bin/bash
[root@Linux ~]#
```

Explanation

If you observe properly previous comment for the user **dvsmaterial** user is been changed to "Hi Everyone this is Linux". Hence we can use "usermod" command to change the attributes of a user on the fly. We have different options to use for changing the user attributes they are like below.

```
Options:
  -c, --comment COMMENT         new value of the GECOS field
  -d, --home HOME_DIR           new home directory for the user account
  -e, --expiredate EXPIRE_DATE  set account expiration date to EXPIRE_DATE
  -f, --inactive INACTIVE       set password inactive after expiration
                                to INACTIVE
  -g, --gid GROUP               force use GROUP as new primary group
  -G, --groups GROUPS           new list of supplementary GROUPS
  -a, --append                  append the user to the supplemental GROUPS
                                mentioned by the -G option without removing
                                him/her from other groups
  -h, --help                    display this help message and exit
  -l, --login NEW_LOGIN         new value of the login name
  -L, --lock                    lock the user account
  -m, --move-home               move contents of the home directory to the
                                new location (use only with -d)
  -o, --non-unique              allow using duplicate (non-unique) UID
  -p, --password PASSWORD       use encrypted password for the new password
  -s, --shell SHELL             new login shell for the user account
  -u, --uid UID                 new UID for the user account
  -U, --unlock                  unlock the user account
  -Z, --selinux-user            new SELinux user mapping for the user account
```

Checking the user status & modification (chage)

Users some time face issues with the access, in such cases we need to check the below things for his/her account.

If you want to check the details of an account then you can use chage option like below.

```
[root@Linux ~]# chage -l dvsmaterial
Last password change                                    : Oct 02, 2018
Password expires                                        : never
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 0
Maximum number of days between password change          : 99999
Number of days of warning before password expires       : 7
[root@Linux ~]#
```

If you observe above output we can see that its convyeing different information about the user like when its going to expire, minimum no.of days between password change, max no.of days and etc ., If you wan to change these options for any user based up on the requirement then you can use the below options.

```
Options:
  -d, --lastday LAST_DAY        set date of last password change to LAST_DAY
  -E, --expiredate EXPIRE_DATE  set account expiration date to EXPIRE_DATE
  -h, --help                    display this help message and exit
  -I, --inactive INACTIVE       set password inactive after expiration
                                to INACTIVE
  -l, --list                    show account aging information
  -m, --mindays MIN_DAYS        set minimum number of days before password
                                change to MIN_DAYS
  -M, --maxdays MAX_DAYS        set maximim number of days before password
                                change to MAX_DAYS
  -W, --warndays WARN_DAYS      set expiration warning days to WARN_DAYS
```

Now I want to change the warning days from 7 to 10 then I will be using the option "-W" then my output will be like below.

```
[root@Linux ~]# chage -W 10 dvsmaterial
[root@Linux ~]# chage -l dvsmaterial
Last password change                                    : Oct 02, 2018
Password expires                                        : never
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 0
Maximum number of days between password change          : 99999
Number of days of warning before password expires       : 10
[root@Linux ~]#
```

So finally I have changed the no.of warning days before to 10 from 7 days. Like this what ever the attribute you want to change you can simply specify the option and execute the command.

User & Group Management

Till now we have gone through the different things about user administration but apart from this sometimes there will be a requirement that the user should be part and parcel of different groups, hence he should be having access for all those groups.

If you want to check the groups details of a user to which he belongs then you can use the below.

```
[root@Linux ~]# id -a dvsmaterial
uid=1000(dvsmaterial) gid=500(ec2-user) groups=500(ec2-user)
[root@Linux ~]#
```

From the above we can convey that user **dvsmaterial** is having a primary group as "ec2-user". If you want to see the entire groups which are part of the machine then we have a file called "/etc/group" which holds information about the groups.

```
[root@Linux ~]# cat /etc/group
root:x:0:
bin:x:1:bin,daemon
daemon:x:2:bin,daemon
sys:x:3:bin,adm
adm:x:4:adm,daemon
tty:x:5:
disk:x:6:
lp:x:7:daemon
mem:x:8:
kmem:x:9:
wheel:x:10:ec2-user
mail:x:12:mail
uucp:x:14:
man:x:15:
games:x:20:
gopher:x:30:
video:x:39:
dip:x:40:
ftp:x:50:
lock:x:54:
```

How to create a group

If you want to create a group then you can simply execute the below.

```
[root@Linux ~]# grep mygroup /etc/group
[root@Linux ~]# groupadd mygroup
[root@Linux ~]# grep mygroup /etc/group
mygroup:x:502:
[root@Linux ~]#
```

Now we are having a group called "mygroup". Now I would like to add my user "**dvsmaterial**" to this group. Then I can do the below.

```
[root@Linux ~]# id -a dvsmaterial
uid=1000(dvsmaterial) gid=500(ec2-user) groups=500(ec2-user)
[root@Linux ~]# gpasswd -a dvsmaterial mygroup
Adding user dvsmaterial to group mygroup
[root@Linux ~]# id -a dvsmaterial
uid=1000(dvsmaterial) gid=500(ec2-user) groups=500(ec2-user),502(mygroup)
[root@Linux ~]#
```

From the above output we can observe that myuser "**dvsmaterial**" is part and parcel of two groups "ec2-user" & "mygroup". In this way you can add a single user to multiple groups. Here "gpasswd" helps me to do the changes for my user. Similarly if you want to delete the user from the group, then you can execute the below.

```
[root@Linux ~]# id -a dvsmaterial
uid=1000(dvsmaterial) gid=500(ec2-user) groups=500(ec2-user),502(mygroup)
[root@Linux ~]# gpasswd -d dvsmaterial mygroup
Removing user dvsmaterial from group mygroup
[root@Linux ~]# id -a dvsmaterial
uid=1000(dvsmaterial) gid=500(ec2-user) groups=500(ec2-user)
[root@Linux ~]#
```

From the above we can observe that the user **dvsmaterial** is now having only one group called ec2-user.

Removing users

If you want to remove the users then you can use "userdel" command. Make sure that you are using "-r" option for removing the content associated with that user during the execution otherwise his home directory will not get removed.

```
[root@Linux ~]# id -a dvsmaterial
uid=1000(dvsmaterial) gid=500(ec2-user) groups=500(ec2-user)
[root@Linux ~]# ls -ld /home/dvsmaterial
drwx------ 2 dvsmaterial ec2-user 4096 Oct  2 06:26 /home/dvsmaterial
[root@Linux ~]# userdel -r dvsmaterial
[root@Linux ~]# ls -ld /home/dvsmaterial
ls: cannot access /home/dvsmaterial: No such file or directory
[root@Linux ~]# id -a dvsmaterial
id: dvsmaterial: no such user
[root@Linux ~]#
```

If you observe above we could see that soon after we removed the user **dvsmaterial** his home directory got deleted. This is how we work with our user administartion in Linux.

There are few important configuration files related to the user administration they are below.

/var/log/wtmp, /etc/passwd, /etc/group, /etc/shadow, /etc/login.defs

Filesystem Management

Linux plays a crucial role in full filing the file system management. Before entering deep lets understand the basic difference between a **folder** and a **filesystem**.

Folder:

```
[root@Linux /]# ls -ld /test
drwxr-xr-x 2 root root 4096 Oct  2 15:49 /test
[root@Linux /]#
```

If you are observing above output here /test is the directory which is residing under /. Which means the size of test folder depends on the "**/**" **filesystem**. In simple max data which you can store under /test directory is max size of **"/" filesystem**. Let's understand bit more on this.

```
[root@Linux /]# df -hT /
Filesystem     Type  Size  Used Avail Use% Mounted on
/dev/xvda1     ext4  7.8G  1.6G  6.1G  21% /
[root@Linux /]#
```

From the above output we can convey that the size of **"/"** partition is 7.8G in simple our "**test**" directory max it can save a data of size 7.8G. Here your "test" is a directory which is residing in the **filesystem** called **"/". In** simple words a **filesystem** is nothing but a drive in your machine like C,D,E,F (in windows). Where directories are nothing but the folders inside these drives.

Working with Filesystem

There are two types of approaches which we use to achieve Filesystem management in Linux. They are as follows

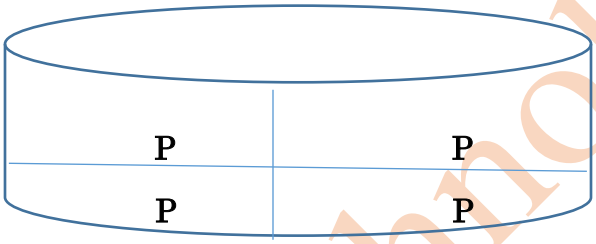
1. Standard Partition
2. Logical Volume Management

Standard partition

This type of partition is same as our windows drive creation. For suppose I have 100GB disk where I need to create 4 partitions each with 25GB then I can create them. Under standard type of partition you will get different terminologies such as

- Primary Partition (P)
- Extended Partition (E)
- Logical Partition (L)

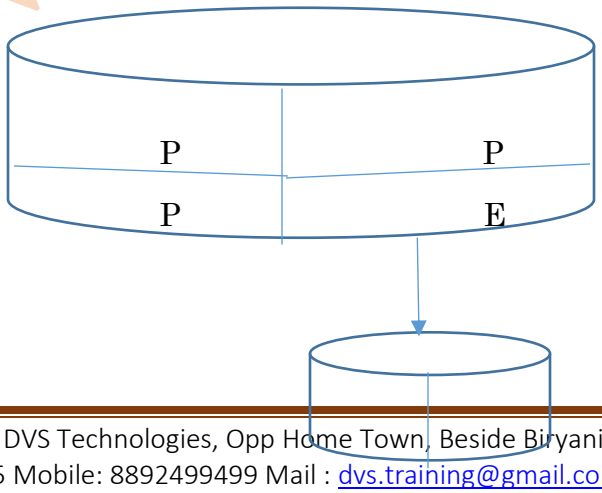
In this type of partition max you can create 4 Primary partitions.



For suppose if I want to create 5 partitions with each 20GB I can't create it using this method. But in order to archive this what I can do it I can opt for 3 Primary partitions, and 1 extended partition. Once you created a extended partition from this partition you can create again "N" no.of logical partitions. Hence we create 3 Primary partitions with 20GB each and extended partition with 40GB from which we will start creating two Logical partitions with each 20GB.

Note: When you are creating an extended partition you need to reboot the machine.

So you picture will look like below.



So here we are goanna have 3 – P * 20 GB & 1 – E * 40 GB out of which we are gonna have

2 – L * 20GB.

Creating Filesystem using Standard partition

Make sure that you are having proper disk exists (free) in the machine. If you want to check the disk information in the linux machine then you can use the below.

You can use "fdisk - l" command or "lsblk" command it will give you the output of disk information .

```
[root@Linux ~]# lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   8G  0 disk
└─xvda1 202:1    0   8G  0 part /
[root@Linux ~]# fdisk -l
WARNING: fdisk GPT support is currently new, and therefore in an

Disk /dev/xvda: 8589 MB, 8589934592 bytes, 16777216 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: gpt


#         Start          End    Size  Type            Name
 1          4096     16777182      8G  Linux filesyste Linux
128          2048         4095      1M  BIOS boot parti BIOS Boot
[root@Linux ~]#
```

From the above output we can see that there is only one disk available in the machine and its size is 8GB. Now lets try to add the disk (EBS) to the machine. In our case we are gonad choose AWS-EBS for our instance. Go to the AWS console and add the disk to the machine. In our example I want to have 4GB filesystem **"/data1"** hence I am gonna add 5GB disk to the machine.

Once I attached the disk now I can able to see the newly attached disk to the machine.

```
[root@Linux ~]# lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   8G  0 disk
└─xvda1 202:1    0   8G  0 part /
xvdf    202:80   0   5G  0 disk
[root@Linux ~]# fdisk -l
WARNING: fdisk GPT support is currently new, and therefore in an experi

Disk /dev/xvda: 8589 MB, 8589934592 bytes, 16777216 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: gpt


#         Start          End    Size  Type            Name
 1         4096     16777182      8G  Linux filesyste Linux
128         2048         4095      1M  BIOS boot parti BIOS Boot Partit

Disk /dev/xvdf: 5368 MB, 5368709120 bytes, 10485760 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

[root@Linux ~]#
```

Here "XVDF" is my new disk with 5G size. Let's create the filesystem.

Explanation

Fdisk is the utility which we use in standard partition for creating the partitions. Here we will have different options such as

n → helps to create a new partition

p → helps to print the partition table

d → helps to delete the partition

w → helps to save the changes and quit.

h → will display all the options associated with fdisk

From the lsblk output make a note of the disk name, in our case disk name is **"/dev/xvdf".** Hence we are using **fdisk /dev/xvdf** option and creating a filesystem of size 4GB.

```
[root@Linux ~]# df -hT
Filesystem     Type      Size  Used Avail Use% Mounted on
devtmpfs       devtmpfs  484M   60K  484M   1% /dev
tmpfs          tmpfs     494M     0  494M   0% /dev/shm
/dev/xvda1     ext4      7.8G  1.6G  6.1G  21% /
[root@Linux ~]# fdisk /dev/xvdf
Welcome to fdisk (util-linux 2.23.2).

Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table
Building a new DOS disklabel with disk identifier 0xf020a349.

Command (m for help): n
Partition type:
   p   primary (0 primary, 0 extended, 4 free)
   e   extended
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-10485759, default 2048):
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-10485759, default 10485759): +4G
Partition 1 of type Linux and of size 4 GiB is set

Command (m for help): p

Disk /dev/xvdf: 5368 MB, 5368709120 bytes, 10485760 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0xf020a349

    Device Boot      Start         End      Blocks   Id  System
/dev/xvdf1            2048     8390655     4194304   83  Linux

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
[root@Linux ~]#
```

Follow the above instructions carefully once you are done with the creation you can check the partition status using below.

```
[root@Linux ~]# lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   8G  0 disk
└─xvda1 202:1    0   8G  0 part /
xvdf    202:80   0   5G  0 disk
└─xvdf1 202:81   0   4G  0 part
[root@Linux ~]#
```

From the above output we can able to see that **/dev/xvdf1** got created and its size is 4GB. Now we have created the partition but this is the raw disk we cannot directly use this in order to make this usable we need to format it. In Linux we have different filesystem types such as ext2, ext3, ext4, xfs. Now lets format the disk with type "ext4". It looks like below.

```
[root@Linux ~]# mkfs.ext4 /dev/xvdf1
mke2fs 1.42.12 (29-Aug-2014)
Creating filesystem with 1048576 4k blocks and 262144 inodes
Filesystem UUID: a364ceb0-883e-4a9c-967c-7cab6dede4df
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

Once you are done with formating now we need to mount disk to a folder in our case the folder name is "/data1". Lets do it

```
[root@Linux ~]# mkdir /data1
[root@Linux ~]# df -hT
Filesystem     Type      Size  Used Avail Use% Mounted on
devtmpfs       devtmpfs  484M   64K  484M   1% /dev
tmpfs          tmpfs     494M     0  494M   0% /dev/shm
/dev/xvda1     ext4      7.8G  1.6G  6.1G  21% /
[root@Linux ~]# mount -t ext4 /dev/xvdf1 /data1
[root@Linux ~]# df -hT
Filesystem     Type      Size  Used Avail Use% Mounted on
devtmpfs       devtmpfs  484M   64K  484M   1% /dev
tmpfs          tmpfs     494M     0  494M   0% /dev/shm
/dev/xvda1     ext4      7.8G  1.6G  6.1G  21% /
/dev/xvdf1     ext4      3.9G  8.0M  3.6G   1% /data1
[root@Linux ~]#
```

If you are observing the above output now we can able to see that "/data1" mount point is visbile and its size is 4GB.

Note: If you restart the machine now you won't able to see the "/data1" mount point as part of the output of "df –hT". Because this is only temporary, but if you want to make it permanent then you need to add the entires to a file called **"/etc/fstab".** It looks like below.

```
[root@Linux ~]# cat /etc/fstab
#
LABEL=/         /           ext4    defaults,noatime  1   1
tmpfs           /dev/shm    tmpfs   defaults          0   0
devpts          /dev/pts    devpts  gid=5,mode=620    0   0
sysfs           /sys        sysfs   defaults          0   0
proc            /proc       proc    defaults          0   0
/dev/xvdf1          /data1  ext4    defaults          0       0
[root@Linux ~]# mount -a
[root@Linux ~]#
```

Drawbacks

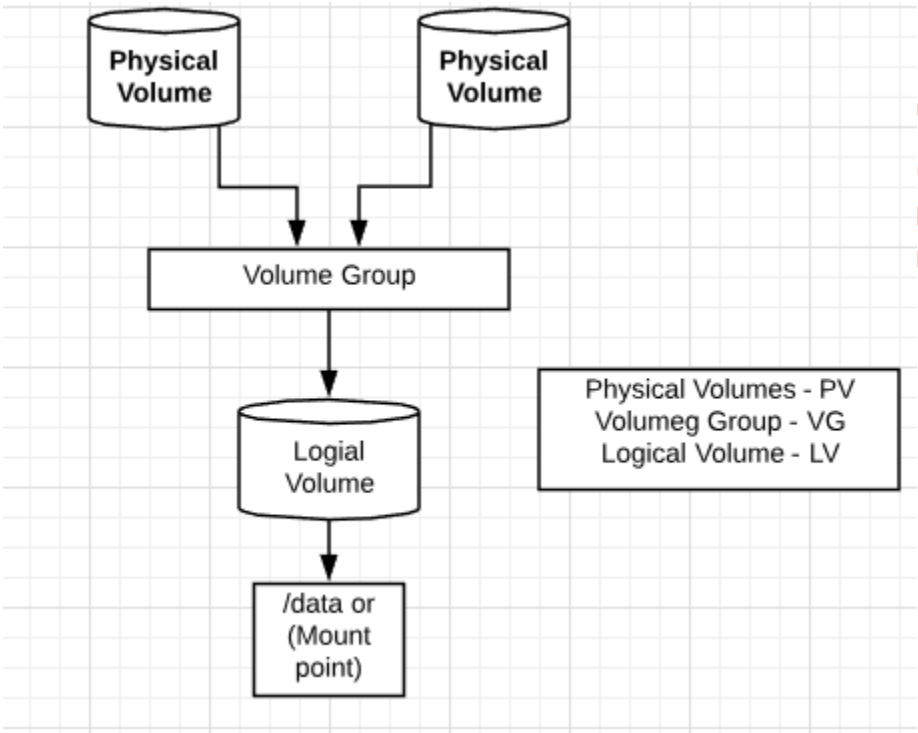
Though we have standard partition why we have one more type of partition is because of below.

- We cannot increase the filesystem size dynamically in standard partition
- We cannot extend the disk size on-fly it requires downtime.

Hence we are opting for one more type of partition which is nothing but your Logical Volume Management.

Logical Volume Management

Using LVM you can overcome all the drawbacks of standard partition. Here again you will come across different terminologies like Physical volume(PV's), Logical Volumes(LV's), Physical extends(PE's), Volume Groups(VG's).



Creating the filesystem in LVM

Make sure that you are having atleast one disk for creating the volume group. Once you have one free disk you can start following the below procedure.

```
[root@Linux ~]# lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   8G  0 disk
└─xvda1 202:1    0   8G  0 part /
xvdf    202:80   0   5G  0 disk
[root@Linux ~]# vgcreate myvg /dev/xvdf
  Volume group "myvg" successfully created
[root@Linux ~]# vgs
  VG   #PV #LV #SN Attr   VSize VFree
  myvg   1   0   0 wz--n- 5.00g 5.00g
[root@Linux ~]# lvcreate -L +4G -n mylv myvg
WARNING: ext4 signature detected on /dev/myvg/mylv at offset 1080
  Wiping ext4 signature on /dev/myvg/mylv.
  Logical volume "mylv" created.
[root@Linux ~]# mkfs.ext4 /dev/mapper/myvg-mylv
mke2fs 1.42.12 (29-Aug-2014)
Creating filesystem with 1048576 4k blocks and 262144 inodes
Filesystem UUID: 386ed842-20e9-44d9-a187-68e686781da9
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

[root@Linux ~]# mkdir /app1
[root@Linux ~]# mount -t ext4 /dev/mapper/myvg-mylv /app1
[root@Linux ~]# df -hT /app1
Filesystem             Type  Size  Used Avail Use% Mounted on
/dev/mapper/myvg-mylv ext4  3.9G  8.0M  3.6G   1% /app1
[root@Linux ~]#
```

Here if you are observing we could see that /app1 mount point got created with 4GB but this time its under LVM. To see more information about the lv,vg,pv then you can issue below commands.

1. lvs
2. vgs
3. pvs
4. lvdisplay –v
5. pvdisplay –v
6. vgdisplay –v

```
[root@Linux ~]# lvs
  LV   VG   Attr       LSize Pool Origin Data%  Meta
  mylv myvg -wi-ao---- 4.00g
[root@Linux ~]# pvs
  PV         VG   Fmt  Attr PSize PFree
  /dev/sdf   myvg lvm2 a--  5.00g 1020.00m
[root@Linux ~]# vgs
  VG   #PV #LV #SN Attr   VSize VFree
  myvg   1   1   0 wz--n- 5.00g 1020.00m
[root@Linux ~]#
```

Filesystem size increase

Now lets try to increase the size of our filesystem in our case it is /app1

```
[root@Linux ~]# df -hT /app1
Filesystem           Type  Size  Used Avail Use% Mounted on
/dev/mapper/myvg-mylv ext4  3.9G  8.0M  3.6G   1% /app1
[root@Linux ~]# vgs
  VG   #PV #LV #SN Attr   VSize VFree
  myvg   1   1   0 wz--n- 5.00g 1020.00m
[root@Linux ~]# lvextend -L +512M /dev/mapper/myvg-mylv -r
  Size of logical volume myvg/mylv changed from 4.00 GiB (1024 extents) t
  Logical volume myvg/mylv successfully resized.
resize2fs 1.42.12 (29-Aug-2014)
Filesystem at /dev/mapper/myvg-mylv is mounted on /app1; on-line resizing
old_desc_blocks = 1, new_desc_blocks = 1
The filesystem on /dev/mapper/myvg-mylv is now 1179648 (4k) blocks long.

[root@Linux ~]# df -hT /app1
Filesystem           Type  Size  Used Avail Use% Mounted on
/dev/mapper/myvg-mylv ext4  4.4G  8.0M  4.1G   1% /app1
[root@Linux ~]#
```

If you are observing the above commands I can able to see that /app1 mount point size got increased from 3.9G to 4.4GB dynamically with out any downtime. But here you should make sure that the volume group (myvg) should have free space in order to increase the size. In our case I can able to see that 1020M is free hence I increased the size of 512M to my filesystem. If in case your volume group is not having free space simply add the new disk to the machine & add the disk to the volumegroup. Once you found space in the volume group you can increase the size of your filesystem.

Note: Make sure that you are giving the entires in to "/etc/fstab" in order to make it permanent like above standard partition.

Removing File system

In this you need to follow the steps from down to top. PFB steps.

```
[root@Linux ~]# umount /app1/
[root@Linux ~]# lvremove /dev/mapper/myvg-mylv
Do you really want to remove active logical volume myvg/mylv? [y/n]: y
  Logical volume "mylv" successfully removed
[root@Linux ~]# vgremove myvg
  Volume group "myvg" successfully removed
[root@Linux ~]# pvremove /dev/xvdf
  Labels on physical volume "/dev/xvdf" successfully wiped.
[root@Linux ~]# df -hT
Filesystem     Type      Size  Used Avail Use% Mounted on
devtmpfs       devtmpfs  484M   60K  484M   1% /dev
tmpfs          tmpfs     494M     0  494M   0% /dev/shm
/dev/xvda1     ext4      7.8G  1.6G  6.1G  21% /
[root@Linux ~]#
```

In this way we will work with our Filesystem management in our Linux environment.

Software Management

Unlike windows linux plays an important role in installing and configuring the software's. There are two approaches via which we can install a software/package in linux they are

1. Redhat package manager (rpm)
2. Yellow dog update manager (yum)

RPM – Redhat Package Manager

RPM is a package managing system (collection of tools to manage software packages). RPM is a powerful software management tool for installing, uninstalling, verifying, querying and updating software packages. RPM is a straight forward program to perform the above software management tasks.

Features:

- RPM can verify software packages
- RPM can be served as a powerful search engine to search for software's
- Components, software's etc can be upgraded using RPM without having to reinstall them
- Installing, reinstalling can be done with ease using RPM
- During updates RPM handles configuration files carefully, so that the customization is not lost

To check all the installed packages in the system,

syntax

#rpm –qa (where q stands for query, and a stands for all)

```
[root@Linux ~]# rpm -qa|more
libmount-2.23.2-33.28.amzn1.x86_64
kbd-misc-1.15-11.4.amzn1.noarch
python27-kitchen-1.1.1-5.6.amzn1.noarch
nss-softokn-freebl-3.28.3-8.41.amzn1.x86_64
rubygem20-rdoc-4.2.2-1.43.amzn1.noarch
zlib-1.2.8-7.18.amzn1.x86_64
python27-docutils-0.11-1.15.amzn1.noarch
readline-6.2-9.14.amzn1.x86_64
python27-iniparse-0.3.1-2.1.9.amzn1.noarch
libcap-ng-0.7.5-4.15.amzn1.x86_64
python27-requests-1.2.3-5.10.amzn1.noarch
lua-5.1.4-4.1.9.amzn1.x86_64
python27-pystache-0.5.3-2.8.amzn1.noarch
file-libs-5.30-11.34.amzn1.x86_64
logrotate-3.7.8-26.14.amzn1.x86_64
libffi-3.0.13-16.5.amzn1.x86_64
lsof-4.82-5.11.amzn1.x86_64
grep-2.20-3.18.amzn1.x86_64
hwdata-0.233-14.1.19.amzn1.noarch
```

Note: - The output of above command will be very lengthier.

To check whether a package is installed or not out of the list of installed package,

Syntax

#rpm –qa <package name> or
#rpm –q < package name>
#rpm –qa python

```
[root@Linux ~]# rpm -qa python*
python27-kitchen-1.1.1-5.6.amzn1.noarch
python27-docutils-0.11-1.15.amzn1.noarch
python27-iniparse-0.3.1-2.1.9.amzn1.noarch
python27-requests-1.2.3-5.10.amzn1.noarch
python27-pystache-0.5.3-2.8.amzn1.noarch
python27-imaging-1.1.6-19.9.amzn1.x86_64
python27-urllib3-1.8.2-1.5.amzn1.noarch
python27-daemon-1.5.2-1.5.amzn1.noarch
python27-libs-2.7.14-1.123.amzn1.x86_64
python27-urlgrabber-3.10-8.16.amzn1.noarch
python27-crypto-2.6.1-1.15.amzn1.x86_64
python27-jinja2-2.7.2-2.15.amzn1.noarch
python27-pygpgme-0.3-9.12.amzn1.x86_64
python27-colorama-0.2.5-1.7.amzn1.noarch
```

One more method of checking the installed package, when you are not sure about the package name, like whether it starts with capital letter and full name etc.

#rpm –qa | grep –i < package name>

```
[root@Linux ~]# rpm -qa |grep -i python
python27-kitchen-1.1.1-5.6.amzn1.noarch
python27-docutils-0.11-1.15.amzn1.noarch
python27-iniparse-0.3.1-2.1.9.amzn1.noarch
python27-requests-1.2.3-5.10.amzn1.noarch
python27-pystache-0.5.3-2.8.amzn1.noarch
python27-imaging-1.1.6-19.9.amzn1.x86_64
python27-urllib3-1.8.2-1.5.amzn1.noarch
python27-daemon-1.5.2-1.5.amzn1.noarch
python27-libs-2.7.14-1.123.amzn1.x86_64
python27-urlgrabber-3.10-8.16.amzn1.noarch
python27-crypto-2.6.1-1.15.amzn1.x86_64
```

To install any package in linux you can use the below

rpm –ivh <package name>

To delete the package you can use

rpm –e <package name>

To update the package you can use

rpm –Uvh <package name>

To see the details about a package then

rpm –qi <package name>

To install any package with out dependencies then

rpm –ivh <package name> --nodeps

Explanation

Everything looks fine right, you can perform all the operations what ever you want on software but why should we opt for YUM. The main problem with RPM based installation is it can't handle the package dependencies, which mean during your installation time if it failed to retrieve the dependicies then it wont allow you to install it.

You have to install the dependencies then only it will allow you to install the software as per your requirement. For example in the below I am trying to install a package called "finger". But if you observe properly, it's getting failed as its unable figure it out the package dependencies.

```
[root@Linux ~]# rpm -ivh finger-server-0.17-40.e16.i686.rpm
warning: finger-server-0.17-40.e16.i686.rpm: Header V3 RSA/SHA1 Signature,
error: Failed dependencies:
        libc.so.6 is needed by finger-server-0.17-40.e16.i686
        libc.so.6(GLIBC_2.0) is needed by finger-server-0.17-40.e16.i686
        libc.so.6(GLIBC_2.1.3) is needed by finger-server-0.17-40.e16.i686
        libc.so.6(GLIBC_2.3.4) is needed by finger-server-0.17-40.e16.i686
        libc.so.6(GLIBC_2.4) is needed by finger-server-0.17-40.e16.i686
        xinetd is needed by finger-server-0.17-40.e16.i686
```

In order to overcome this issue, we are opting for one more utility which is nothing but YUM.

YUM – Yellow Dog Update Manager

- The Yellow dog Updater Modified (YUM) is a package management application for computers running Linux operating systems.

- Yum is a standard method of managing the installation and removal of software. Several graphical applications exist to allow users to easily add and remove packages; however, many are simply friendly interfaces with yum running underneath. These programs present the user with a list of available software and pass the user's selection on for processing. It is yum that actually downloads the packages and installs them in the background.

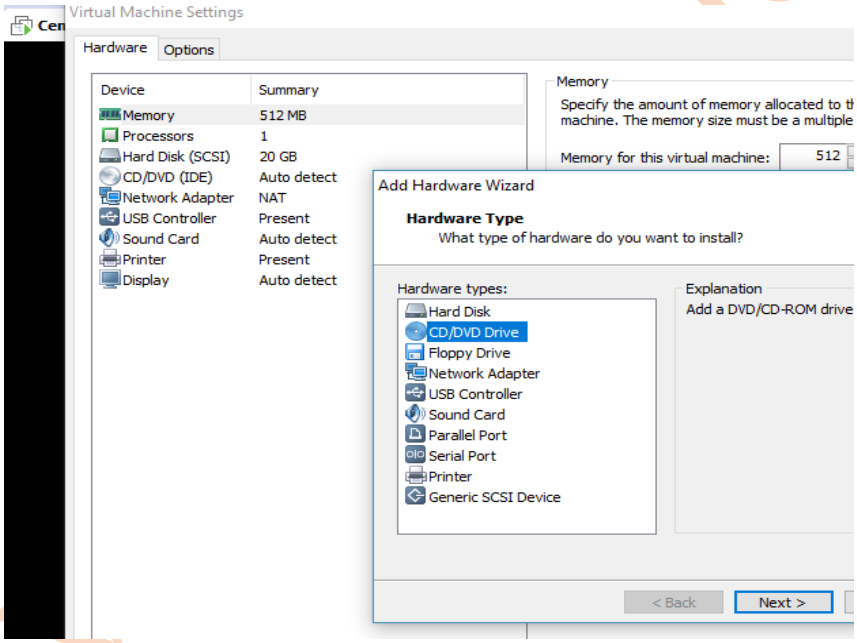
- Packages are downloaded from collections called repositories, which may be online, on a network, and/or on installation media. If one package due to be installed relies on another being present, this dependency can usually be resolved without the user needing to know the details. For example, a game being installed may depend on specific software to play its music. The problem of solving such dependencies can be handled by yum because it knows about all the other packages that are available in the repository.

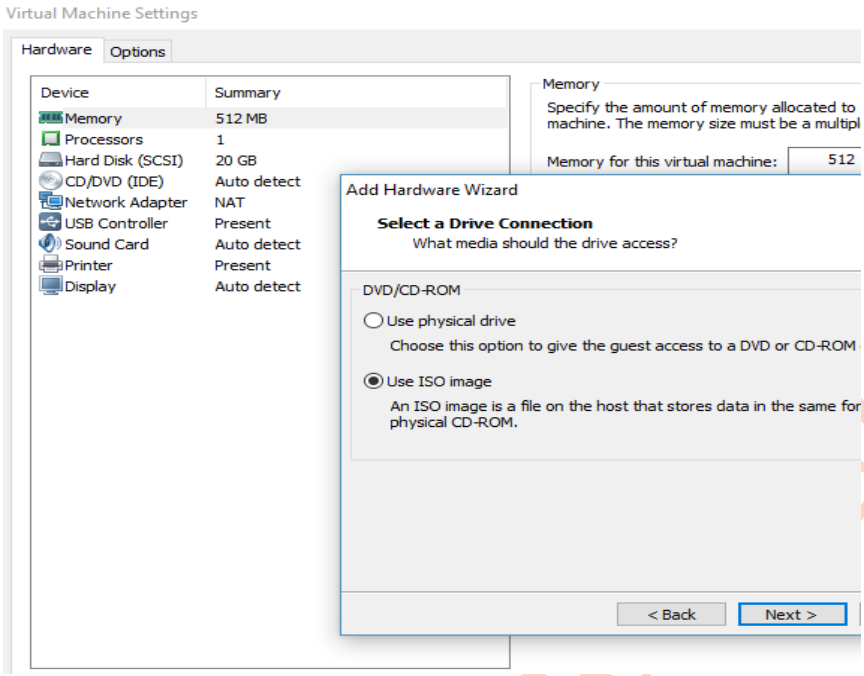
- Yum will work only from Centos 5 / Red hat 5 and latest versions of fedora. For Old releases like RHEL 4 you need to use up2date command to update your rpm based packages.

- Yum uses a configuration file at /etc/yum.conf and configuration directory as /etc/yum.repos.d

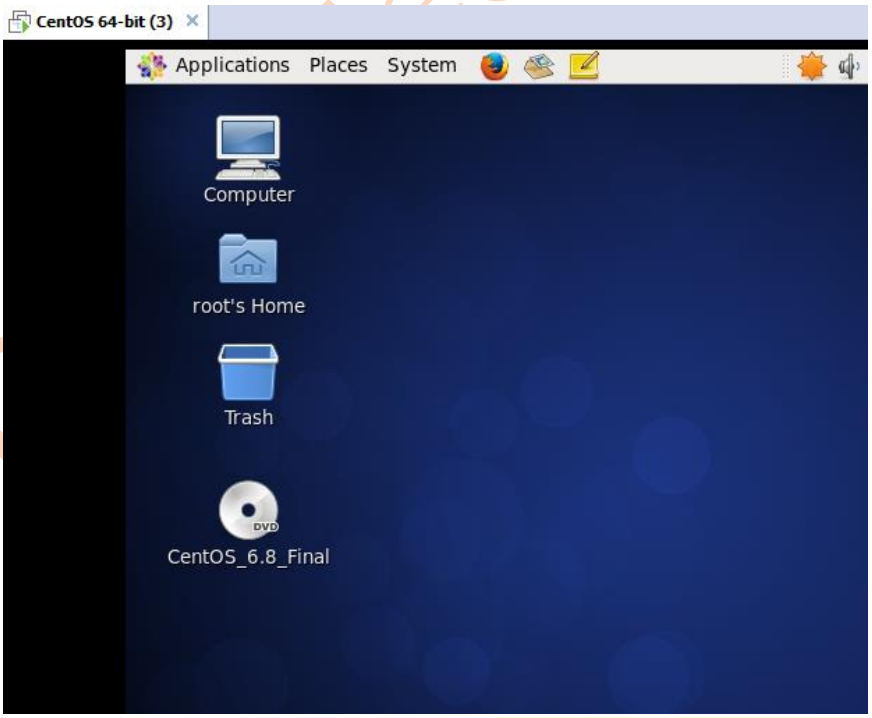
In order to achieve this, we are going to use our VMWare virtual machine. Our goal is very simple we need to install the package "httpd". Let's start our journey!!!

1. Lets add the iso image to the server in our VMWare like below.





Now you can select the ISO image in our case its centos 6.8. Once you added the image you should be able to see it like below in your VmWare



Now once you are done with above go and login to your machine, you could able to see a new mount point under your /media like below.

```
[root@Linux ~]# df -hT
Filesystem     Type      Size  Used Avail Use% Mounted on
/dev/sda2      ext4       18G  2.6G   14G  16% /
tmpfs          tmpfs     238M   80K  238M   1% /dev/shm
/dev/sda1      ext4      283M   34M  234M  13% /boot
/dev/sr0       iso9660   3.7G  3.7G     0 100% /media/CentOS_6.8_Final
[root@Linux ~]#
```

Now lets start creating our repository.

```
[root@Linux ~]# cd /etc/yum.repos.d/
[root@Linux yum.repos.d]# vim mysoftwares.repo
[mysoftwares]
name=mysoftwares
baseurl=file:///media/CentOS_6.8_Final
gpgcheck=0
```

Once you done with above now do the below things

```
[root@Linux yum.repos.d]# ls -l mysoftwares.repo
-rw-r--r--. 1 root root 81 Oct  7 01:16 mysoftwares.repo
[root@Linux yum.repos.d]# yum clean all
Loaded plugins: fastestmirror, refresh-packagekit, security
Cleaning repos: base extras mysoftwares updates
Cleaning up Everything
[root@Linux yum.repos.d]# yum repolist
Loaded plugins: fastestmirror, refresh-packagekit, security
Determining fastest mirrors
 * base: mirrors.fibergrid.in
 * extras: mirrors.fibergrid.in
 * updates: mirrors.fibergrid.in
base
base/primary_db
extras
extras/primary_db
mysoftwares
mysoftwares/primary_db
updates
updates/primary_db
repo id                                                repo name
base                                                   CentOS-6 - Base
extras                                                 CentOS-6 - Extras
mysoftwares                                            mysoftwares
updates                                                CentOS-6 - Updates
repolist: 13,584
[root@Linux yum.repos.d]#
```

From the above output I can convey that our repository “mysoftwares” got create.

Working with YUM Utility

Listing Packages

yum list all

```
[root@Linux yum.repos.d]# yum list all|more
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
 * base: mirrors.fibergrid.in
 * extras: mirrors.fibergrid.in
 * updates: mirrors.fibergrid.in
Installed Packages
ConsoleKit.x86_64                                0.4.1-6.el6
ConsoleKit-libs.x86_64                           0.4.1-6.el6
ConsoleKit-x11.x86_64                            0.4.1-6.el6
DeviceKit-power.x86_64                           014-3.el6
GConf2.x86_64                                    2.28.0-6.el6
GConf2-gtk.x86_64                                2.28.0-6.el6
MAKEDEV.x86_64                                   3.24-6.el6
ModemManager.x86_64                              0.4.0-5.git20100628.el6
NetworkManager.x86_64                            1:0.8.1-107.el6
NetworkManager-glib.x86_64                       1:0.8.1-107.el6
NetworkManager-gnome.x86_64                      1:0.8.1-107.el6
```

Viewing Installed Packages

yum list installed

```
[root@Linux yum.repos.d]# yum list installed|more
Loaded plugins: fastestmirror, refresh-packagekit, security
Installed Packages
ConsoleKit.x86_64        0.4.1-6.el6      @anaconda-CentOS-201605220104.x86_64/6.8
ConsoleKit-libs.x86_64   0.4.1-6.el6      @anaconda-CentOS-201605220104.x86_64/6.8
ConsoleKit-x11.x86_64    0.4.1-6.el6      @anaconda-CentOS-201605220104.x86_64/6.8
DeviceKit-power.x86_64   014-3.el6        @anaconda-CentOS-201605220104.x86_64/6.8
GConf2.x86_64            2.28.0-6.el6     @anaconda-CentOS-201605220104.x86_64/6.8
GConf2-gtk.x86_64        2.28.0-6.el6     @anaconda-CentOS-201605220104.x86_64/6.8
MAKEDEV.x86_64           3.24-6.el6       @anaconda-CentOS-201605220104.x86_64/6.8
```

To install any package

yum install <package name>

```
[root@Linux yum.repos.d]# yum install httpd
Loaded plugins: fastestmirror, refresh-packagekit, security
Setting up Install Process
Loading mirror speeds from cached hostfile
 * base: mirrors.fibergrid.in
 * extras: mirrors.fibergrid.in
 * updates: mirrors.fibergrid.in
Resolving Dependencies
--> Running transaction check
---> Package httpd.x86_64 0:2.2.15-53.el6.centos will be updated
---> Package httpd.x86_64 0:2.2.15-69.el6.centos will be an update
--> Processing Dependency: httpd-tools = 2.2.15-69.el6.centos for package: http
--> Running transaction check
---> Package httpd-tools.x86_64 0:2.2.15-53.el6.centos will be updated
---> Package httpd-tools.x86_64 0:2.2.15-69.el6.centos will be an update
```

To delete any package

yum remove <package name>

```
[root@Linux yum.repos.d]# yum remove httpd
Loaded plugins: fastestmirror, refresh-packagekit, security
Setting up Remove Process
Resolving Dependencies
--> Running transaction check
---> Package httpd.x86_64 0:2.2.15-53.el6.centos will be erased
--> Processing Dependency: httpd >= 2.2.0 for package: gnome-user-share-2.28.2-3.el
--> Running transaction check
---> Package gnome-user-share.x86_64 0:2.28.2-3.el6 will be erased
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                 Arch                 Version
================================================================================
Removing:
 httpd                   x86_64               2.2.15-53.el6.centos
Removing for dependencies:
```

To update any package

yum update <package name>

```
[root@Linux yum.repos.d]# yum update httpd
Loaded plugins: fastestmirror, refresh-packagekit, security
Setting up Update Process
Loading mirror speeds from cached hostfile
```

To check the information of a package

yum info <pakcagename>

```
[root@Linux yum.repos.d]# yum info httpd
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
Installed Packages
Name        : httpd
Arch        : x86_64
Version     : 2.2.15
Release     : 53.el6.centos
Size        : 3.0 M
Repo        : installed
From repo   : anaconda-CentOS-201605220104.x86_64
Summary     : Apache HTTP Server
URL         : http://httpd.apache.org/
License     : ASL 2.0
Description : The Apache HTTP Server is a powerful, efficient, and extensible
            : web server.
```

Network Configuration & Trouble Shooting

Networking

It is a connection between two or more machines to communicate with each other.

The basic requirements for Networking are

1. NIC (Network Interface Controller or Card)
2. Media
3. Topology
4. Protocol
5. IP Addresses

1. NIC (Network Interface Controller or Card)

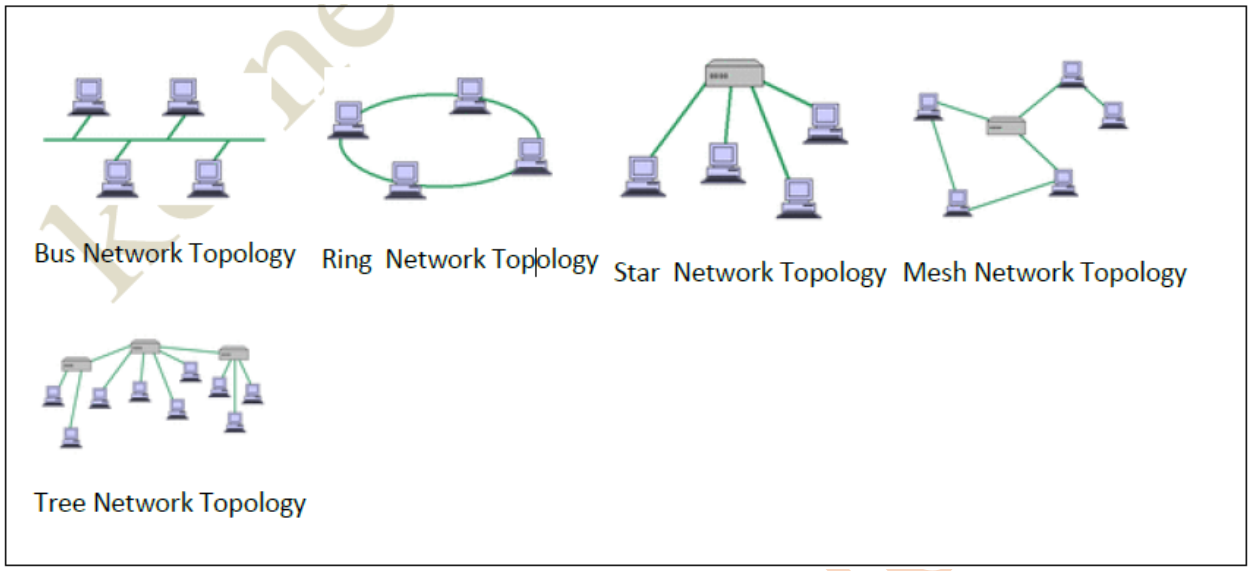
A network interface controller (also known as a network interface card, network adapter, LAN adapter and by similar terms) is a computer hardware component that connects a computer to a computer network. Each NIC will be having a unique MAC addresses (Media Access Control address) to avoid conflicts between same NIC adapters. In Linux these NIC adapter is represented by the word "eth". Example if there are two Ethernet adapters in the system then it will be denoted as eth0, eth1, etc.

2. Media

Media is the medium via which two different computer's NIC card will be connected. The best example for media is Cable. Example RJ 45, CAT 5 etc.

3. Topology

Topology is the scheme or design in which the computers in the network will be connected to each other. Example for topology is Bus, Ring, star, mesh, tree topologies. The following pictures explain it better.



4. **Protocol**

A network protocol defines rules and conventions for communication between network devices. Protocols for computer networking all generally use packet switching techniques to send and receive messages in the form of packets.

Network protocols include mechanisms for devices to identify and make connections with each other, as well as formatting rules that specify how data is packaged into messages sent and received. Some protocols also support message acknowledgement and data compression designed for reliable and/or high-performance network communication. Hundreds of different computer network protocols have been developed each designed for specific purposes and environments.

Example for Protocols are TCP/IP (Transmission Control Protocol), UDP (User Datagram Protocol), HTTP. The most widely and regularly used protocols for transferring data are TCP and UDP. Let's analyze some basic differences between TCP/IP and UDP.

TCP/IP	UDP
Transmission Control Protocol	User Datagram Protocol
It is connection Oriented	Connectionless
Reliable	Non-Reliable
TCP Acknowledgement will be sent/received	No Acknowledgement for UDP
Slow Communication	Faster Communication
Protocol Number for TCP is **6**	Protocol Number for UDP is **17**
HTTP, FTP, SMTP uses TCP	DNS, DHCP uses UDP

5. IP ADDRESS

An IP address can be thought of as being similar to a phone number. Just as every person who communicates with a telephone is using a phone with a unique phone number, every computer that is on the Internet has a unique IP address. Not only on internet but within an organization every computer is assigned an IP address so that they can communicate with each other. Basically IP addressing is very deep concept. To understand the concept of IP address we need to understand some important aspect of IP Address which is

- IP Address Classes
- Subnet mask
- Gateway

The above concepts in IP Addressing are very important to understand networking clearly.

IP Address Classes

The IP addresses are further broken down into classes. These classes are A, B, C, D, E and their possible ranges can be seen in Figure below.

Class	Start	End	Default subnet mask	CIDR
Class A	0.0.0.0	127.255.255.255	255.0.0.0	/8
Class B	128.0.0.0	191.255.255.255	255.255.0.0	/16
Class C	192.0.0.0	223.255.255.255	255.255.255.0	/24
Class D (multicast)	224.0.0.0	239.255.255.255		
Class E (reserved)	240.0.0.0	255.255.255.255		

*CIDR - Classless Inter-Domain Routing
* 127.0.0.0 to 127.255.255.255 is reserved for loopback address

Loopback

A special IP number (127.0.0.1), that is designated for the software loopback interface of a machine. 127.0.0.0 Through 127.255.255.255 is also reserved for loopback and is used for internal testing on local machines.

Multicast

Multicasting allows a single message to be sent to a group of recipients. Emailing, teleconferencing, are examples of multicasting. It uses the network infrastructure and standards to send messages.

Subnet Mask

A subnet mask allows users to identify which part of an IP address is reserved for the network and which part is available for host use. By looking at the IP address alone, especially now with classless inter-domain routing, users cannot tell which part of the address is which. Adding the subnet mask or netmask gives users all the information needed to calculate network and host portions of the address with ease. In summary, knowing the subnet mask can allow users to easily calculate whether IP addresses are on the same subnet or not.
A commonly used netmask is a 24-bit netmask as seen below.

Netmask:	255.	255.	255.	0
Binary:	11111111	11111111	11111111	00000000
Netmask length	8	16	24	--

Gateway

A gateway is a network point that provides entrance into another network. On the Internet, a node or stopping point can be either a gateway node or a host (end-point) node. Both the computers of Internet users and the computers that serve pages to users are host nodes. The computers that control traffic within your company's network or at your local Internet service provider (ISP) are gateway nodes.

For example let's say our network is 192.168. Something and we want to send a file to other computer on 10.10.network, so we need a gateway to communicate between two computers of different networks.

Some Important configuration files/directories of network configurations

#/etc/sysconfig/network-scripts is the directory which keeps the configuration of network devices connected to the system.

```
[root@Linux ~]# ls /etc/sysconfig/network-scripts/
ifcfg-eth0   ifdown-eth   ifdown-isdn   ifdown-sit      ifup-bnep  ifup-ipv6
ifcfg-lo     ifdown-ib    ifdown-post   ifdown-tunnel   ifup-eth   ifup-isdn
ifdown       ifdown-ippp  ifdown-ppp    ifup            ifup-ib    ifup-plip
ifdown-bnep  ifdown-ipv6  ifdown-routes  ifup-aliases    ifup-ippp  ifup-plusb
[root@Linux ~]#
```

#/etc/sysconfig/network is a file which keeps the information about the hostname assigned to the system. If you want to change the hostname permanently, you need to change the hostname in this file.

```
[root@Linux ~]# cat /etc/sysconfig/network
NETWORKING=yes
HOSTNAME=Linux.example.com
[root@Linux ~]#
```

#/etc/hosts a file which is responsible for resolving hostname into IP locally, in other word it acts as local DNS if DNS server is not accessible.

```
[root@Linux ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 lo
::1         localhost localhost.localdomain localhost6 lo
192.168.137.128         Linux.example.com
[root@Linux ~]#
```

#/etc/resolv.conf is a file which keeps the address of DNS server to which the clients will be accessing to resolve IP to hostname and hostname to IP.

```
[root@Linux ~]# cat /etc/resolv.conf
# Generated by NetworkManager
domain localdomain
search localdomain
nameserver 192.168.137.2
```

Checking the system ipaddress

```
# ifconfig -a
```

```
[root@Linux ~]# ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:0C:29:45:0E:14
          inet addr:192.168.137.128  Bcast:192.168.137.255  Mask:2
          inet6 addr: fe80::20c:29ff:fe45:e14/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:873 errors:0 dropped:0 overruns:0 frame:0
          TX packets:478 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:80232 (78.3 KiB)  TX bytes:69794 (68.1 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:16 errors:0 dropped:0 overruns:0 frame:0
          TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:960 (960.0 b)  TX bytes:960 (960.0 b)

pan0      Link encap:Ethernet  HWaddr 36:BF:9B:1D:F5:2D
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

To chech the ip of a particular interface

#ifconfig < adapter name >
#ifconfig eth0

```
[root@Linux ~]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:0C:29:45:0E:14
          inet addr:192.168.137.128  Bcast:192.168.137.255  Mask:2
          inet6 addr: fe80::20c:29ff:fe45:e14/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:910 errors:0 dropped:0 overruns:0 frame:0
          TX packets:496 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:83148 (81.1 KiB)  TX bytes:72908 (71.1 KiB)
```

To check the hostname of the system.
#hostname

```
[root@Linux ~]# hostname
Linux
[root@Linux ~]#
```

To check ip of the host

#hostname –i

```
[root@Linux ~]# hostname -i
192.168.137.128
[root@Linux ~]#
```

To check whether DNS is resolving or not

#host < ip address >

#host <hostname>

Same with "nslookup" command

#nslookup < ip address >

#nslookup < hostname >

The most common command used to check DNS function is "dig"

#dig <hostname>

#dig –x <ip address>

Checking network connectivity using ping command

#ping < ip address >

To limit the pinging for specific number of counts

#ping –c <counts> <ip address>

Changing the hostname

- Check the current hostname with hostname command
- The syntax for changing the hostname is

#hostname <new name>

```
[root@Linux ~]# hostname Linux.example.com
[root@Linux ~]# hostname
Linux.example.com
[root@Linux ~]#
```

Note:

The above change is temporary and will be last only till you are logged in, if you want to change it permanently edit the **/etc/sysconfig/network** file and then logout and login to confirm the change. Once done with changes it should look like below.

```
[root@Linux ~]# cat /etc/sysconfig/network
NETWORKING=yes
HOSTNAME=Linux.example.com
[root@Linux ~]#
```

Assigning /Changing the IP Address

If you want to change/assign an ipaddress to a Linux machine there are two types of ipaddress one is DHCP, other one is Static. Here if you are option for DHCP the ipaddress will be dynamic which means your ip will get changed once you restart the server. But if you are opting for Static your ipaddress is dedicated.

Dhcp Ipaddress allocation

```
[root@Linux ~]# cd /etc/sysconfig/network-scripts/
[root@Linux network-scripts]# cat ifcfg-eth0
DEVICE="eth0"
BOOTPROTO="dhcp"
HWADDR="00:0C:29:45:0E:14"
IPV6INIT="yes"
NM_CONTROLLED="yes"
ONBOOT="yes"
TYPE="Ethernet"
UUID="8a45aed1-dcc1-4d8f-acac-71d1254f55ff"
[root@Linux network-scripts]#
```

Make sure that you are providing the details as above, here your HWADDR, UUID will be different for different network interfaces.

Static Ipaddress Allocation

Now lets change the ipaddress of our machine, in our case current ipaddress is “192.168.137.128” lets change it to “192.168.137.130”.

Before:

```
[root@Linux network-scripts]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:0C:29:45:0E:14
          inet addr:192.168.137.128  Bcast:192.168.137.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe45:e14/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2017 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1016 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:175613 (171.4 KiB)  TX bytes:145380 (141.9 KiB)

[root@Linux network-scripts]#
```

After Change

```
[root@Linux network-scripts]# cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE="eth0"
BOOTPROTO="static"
HWADDR="00:0C:29:45:0E:14"
IPV6INIT="yes"
NM_CONTROLLED="yes"
ONBOOT="yes"
IPADDR=192.168.137.130
NETMASK=255.255.255.0
GATEWAY=192.168.137.2
DNS1=192.168.137.2
TYPE="Ethernet"
UUID="8a45aed1-dcc1-4d8f-acac-71d1254f55ff"
[root@Linux network-scripts]#
```

Make sure that you are adding the above things as per your requirement. Once you are done with the above changes please do service restart using below command.

```
# service network restart
```

Once done with the restart you can able to see the new ipaddress getting reflected for our interface eth0 like below.

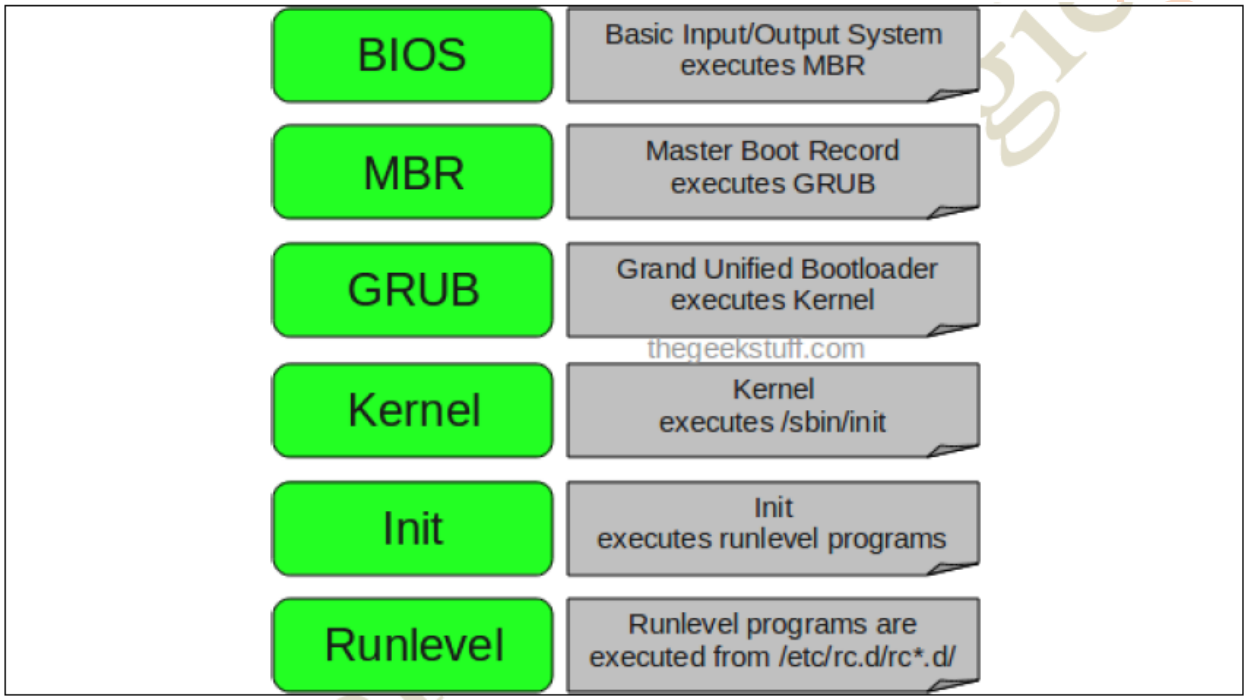
```
[root@Linux ~]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:0C:29:45:0E:14
          inet addr:192.168.137.130  Bcast:192.168.137.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe45:e14/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2553 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1335 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:225319 (220.0 KiB)  TX bytes:198973 (194.3 KiB)

[root@Linux ~]#
```

DVS Technologies Aws & Devops

Booting Procedure and Kernel Parameter

Press the power button on your system, and after few moments you see the Linux login prompt. Have you ever wondered what happens behind the scenes from the time you press the power button until the Linux login prompt appears?
The following are the 6 high level stages of a typical Linux boot process.



1. BIOS

- BIOS stands for Basic Input/Output System
- Performs some system integrity checks
- Searches, loads, and executes the boot loader program.
- It looks for boot loader in floppy, cd-rom, or hard drive. You can press a key (typically F12 of F2, but it depends on your system) during the BIOS startup to change the boot sequence.
- Once the boot loader program is detected and loaded into the memory, BIOS gives the control to it.
- So, in simple terms BIOS loads and executes the MBR boot loader.

2. MBR

- MBR stands for Master Boot Record.
- It is located in the 1st sector of the bootable disk. Typically /dev/hda, or /dev/sda
- MBR is less than 512 bytes in size. This has three components 1) primary boot loader info in 1st 446 bytes 2) partition table info in next 64 bytes 3) mbr validation check in last 2 bytes.
- It contains information about GRUB (or LILO in old systems).
- So, in simple terms MBR loads and executes the GRUB boot loader.

3. GRUB

- GRUB stands for Grand Unified Bootloader.
- If you have multiple kernel images installed on your system, you can choose which one to be executed.
- GRUB displays a splash screen, waits for few seconds, if you don't enter anything, it loads the default kernel image as specified in the grub configuration file.
- GRUB has the knowledge of the filesystem (the older Linux loader LILO didn't understand filesystem).
- Grub configuration file is /boot/grub/grub.conf (/etc/grub.conf is a link to this). The following is sample grub.conf

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,1)
#          kernel /vmlinuz-version ro root=/dev/mapper/vg_ktadm-rootlv
#          initrd /initrd-[generic-]version.img
#boot=/dev/sda
default=0
timeout=5
splashimage=(hd0,1)/grub/splash.xpm.gz
hiddenmenu
title Red Hat Enterprise Linux (2.6.32-131.0.15.el6.x86_64)
        root (hd0,1)
        kernel /vmlinuz-2.6.32-131.0.15.el6.x86_64 ro root=/dev/mapper/vg_ktad
rootlv rd_LVM_LV=vg_ktadm/rootlv rd_NO_LUKS rd_NO_MD rd_NO_DM LANG=en_US.UTF-8
YSFONT=latarcyrheb-sun16 KEYBOARDTYPE=pc KEYTABLE=us rhgb quiet
        initrd /initramfs-2.6.32-131.0.15.el6.x86_64.img
~
```

As you notice from the above info, it contains kernel and initrd image.
So, in simple terms GRUB just loads and executes Kernel and initrd images.

4. **Kernel**

- Mounts the root file system as specified in the "root=" in grub.conf
- Kernel executes the /sbin/init program

Since init was the 1st program to be executed by Linux Kernel, it has the process id (PID) of 1. Do a 'ps -ef | grep init' and check the pid.

- initrd stands for Initial RAM Disk.
- initrd is used by kernel as temporary root file system until kernel is booted and the real root file system is mounted. It also contains necessary drivers compiled inside, which helps it to access the hard drive partitions, and other hardware.

5. **Init**

1. Looks at the /etc/inittab file to decide the Linux run level.
2. Following are the available run levels
   - 0 – halt
   - 1 – Single user mode
   - 2 – Multiuser, without NFS
   - 3 – Full multiuser mode
   - 4 – unused
   - 5 – X11
   - 6 – reboot
3. Init identifies the default initlevel from /etc/inittab and uses that to load all appropriate program.
4. Execute 'grep initdefault /etc/inittab' on your system to identify the default run level
5. If you want to get into trouble, you can set the default run level to 0 or 6. Since you know what 0 and 6 means, probably you might not do that.
6. Typically you would set the default run level to either 3 or 5.

6. Runlevel programs

- When the Linux system is booting up, you might see various services getting started. For example, it might say "starting sendmail …. OK". Those are the run level programs, executed from the run level directory as defined by your run level.

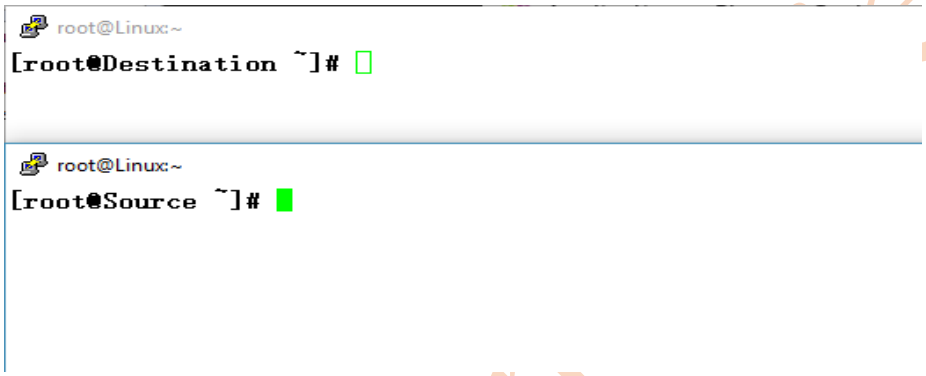
- Depending on your default init level setting, the system will execute the programs from one of the following directories.
  - Run level 0 – /etc/rc.d/rc0.d/
  - Run level 1 – /etc/rc.d/rc1.d/
  - Run level 2 – /etc/rc.d/rc2.d/
  - Run level 3 – /etc/rc.d/rc3.d/
  - Run level 4 – /etc/rc.d/rc4.d/
  - Run level 5 – /etc/rc.d/rc5.d/
  - Run level 6 – /etc/rc.d/rc6.d/

- Please note that there are also symbolic links available for these directory under /etc directly. So, /etc/rc0.d is linked to /etc/rc.d/rc0.d.

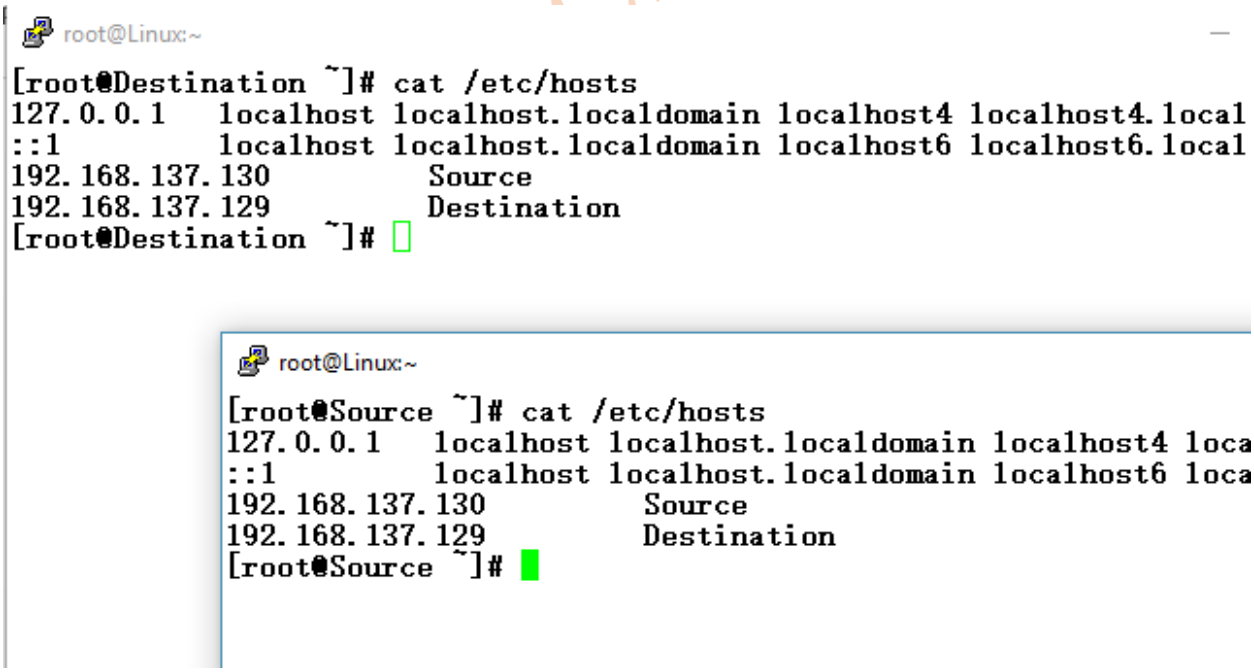
DVS Technologies Aws & Devops

Configuring passwordless login in Linux

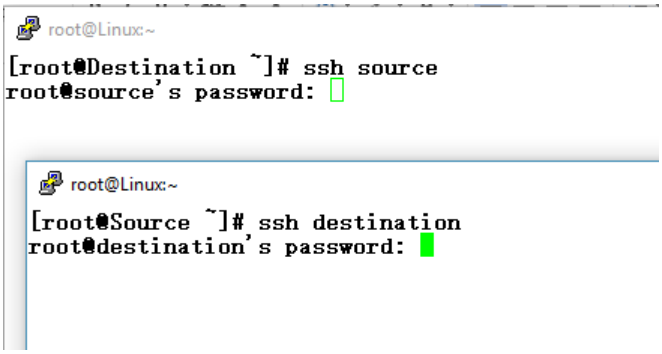
Let's take two machines **source, destination**, my requirement is very simple. I want to access my machine destination from source server. Lets start our work our servers looks like below.



Make sure that you are adding the server's names in the /etc/hosts files of both the servers. Once you are done with adding it looks like below.



Testing the connection



1. If you observe properly in the above both the hosts are recognizing the names but they are asking for the password.

2. Make sure that you are doing the below changes to **"/etc/ssh/sshd_config"** file in both the machines in our case "Source & Destination"

   PermitRootLogin yes
   PasswordAuthentication yes

```
[root@Destination ~]# grep -e "^PermitRootLogin\|^PasswordAuthentication" /etc/ssh/sshd_config
PermitRootLogin yes
PasswordAuthentication yes
[root@Destination ~]#
```

```
[root@Source ~]# grep -e "^PermitRootLogin\|^PasswordAuthentication" /etc/ssh/sshd_config
PermitRootLogin yes
PasswordAuthentication yes
[root@Source ~]#
```

3. Now our requirement is very simple whenever I executed "**ssh destination**" command from the **source** it should not ask me the password.

Perform the below steps in your source server

Generating the keys in source server

```
[root@Source ~]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
34:d3:d6:da:ec:9c:c8:95:31:cd:ad:2b:ad:a2:cc:01 root@Source
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|          . . o .|
|         + o + o .|
|        . + + + .|
|       E S . = . |
|        . .  = o .|
|         . o = o |
|        o ..   o |
|         +. ..   |
+-----------------+
```

Sending our "Source" server public key to the "Destination" server

```
[root@Source ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@destination
root@destination's password:
Now try logging into the machine, with "ssh 'root@destination'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.

[root@Source ~]# ssh destination
Last login: Sun Oct  7 10:45:51 2018 from source
[root@Destination ~]#
```

If you are observing the above output, we could see that soon after we issue the command **"ssh destination"** it's not asking the password. Hence we have full filled our password less login in Linux.