Linux 2 Days Training Content

RHEL Linux: An Introduction	**15 Min**
Boot Process and Grub	**45 min**
Installation – Explanation **& Understanding Inverted Root Tree Structure**	**45 min**
YUM Repository Configuration **YUM Server and Client**	**45 min**
Basic Command Line **Creating Viewing and Editing Test Files** **I/O Redirection and Pipelines**	**60 min**
Managing Users and Groups • User and Group Concepts • /etc/shadow, and /etc/passwd concepts • User administration, Group Administration, • Password Aging, Controlling logins	**90 min**
Controlling Access to Files and Linux File System Permissions	**60 min**
Filesystem Administration **Partitioning, File System, Mounting File system, Swap** **Disk Quota**	**60 min**
Monitoring and Managing Linux Processes **At & Cron**	**90 min**
Configuring and Securing OpenSSH **Analyzing and Storing Logs**	**45 min**
Managing Logical Volume Management (LVM) Storage • LVM Concepts • Creating LVM using multiple block devices • Extending Logical Volumes • Advanced LVM Concepts	90 min
Managing Network, • Network configuration • Configuring Link Aggregation, Network Teaming	**120 min**
Network Port Security	**45 min**
Configuring File-Based Storage NFS File Sharing & Mounting	**60 min**
Configure DNS Client Configure DHCP Client Configure NTP Client Enabling IPv6	**65 min**

Day # 1

RHEL Linux: An Introduction

1. Introduction to Linux

What is Linux?

Note: ***Linux is the kernel. GNU is the OS environment***

- **Linux is a Unix-like computer operating system assembled under free model and open source software development and distribution.**
- **Core part of LINUX Operating system is Kernel.**

  The kernel is the essential center of a computer operating system, the core that provides basic services for all other parts of the operating system. A synonym is *nucleus*. A kernel can be contrasted with a shell, the outermost part of an operating system that interacts with user commands. *Kernel* and *shell* are terms used more frequently in Unixoperating systems than in IBM mainframe or Microsoft Windows systems.

- **The main forms of distribution are Linux distributions.** The defining component of Linux is the Linux kernel, an operating system kernel first released on 5 October 1991, by Linus Torvalds. Because it considers Linux to be a variant of the GNU operating system

Andrew S. Tanenbaum (left), author of the MINIX operating system, and Linus Torvalds (right), principal author of the Linux kernel

- **Linux was originally developed as a free operating system for Intel x86-based architecture computers.** It has since been ported to more computer hardware platforms than any other operating system.
- **Linux also runs on embedded systems** (devices where the operating system is typically built into the firmware and highly tailored to the system) such as mobile phones, tablet

computers, network routers, building automation controls etc. Android system is being widely used on mobile devices which is built on the Linux kernel.

- **The development of Linux is one of the most prominent examples of free and open source software collaboration:**

  The underlying source code may be used, modified, and distributed—commercially or non-commercially—by anyone under licenses such as the GNU General Public License. Typically Linux is packaged in a format known as a Linux distribution for desktop and server use.

- **Some popular mainstream Linux distributions include :**

  Debian (and its derivatives such as Ubuntu and Linux Mint),

  Fedora (and its derivatives such as the commercial RHEL **Red Hat Enterprise Linux** and its open equivalent **CentOS**),

  OpenSUSE (and its commercial derivative **SUSE Linux Enterprise Server**), and Arch Linux. Linux distributions include the Linux kernel,

  - supporting utilities and libraries and usually a large amount of application software to fulfill the distribution's intended use.

- A distribution oriented toward desktop use will typically include the windowing systems X11 and Wayland and an accompanying **desktop environment** such as **GNOME** or the **KDE** Software Compilation.

What is Open Source Operating System?

- **Open source refers to a program or software in which the source code** (the form of the program when a programmer writes a program in a particular programming language) **is available to the general public for use and/or modification from its original design free of charge.**

- ***We can customize Linux Operating system as per requirement and it is possible because of source code availability.***

The Rationale behind Open Source Software

We can create our own OS as per requirement:

OpenFiller: **Openfiler** is an operating system that provides file-based network-attached storage (NAS) and block-based storage area network (SAN) such as iSCSI.

Open Firewall: SOPHOS
https://www.sophos.com/en/products/free-tools/sophos-utm-home-edition.aspx

Financial Gain: The GNU General Public License is a free, the GNU General Public License model gives us a freedom to share and change all versions of the program.

Performance: We can tune up the performance of the OS by controlling processes running.

Security: NO OS vendor claims 100% security for the product, even then Linux is a superior OS as compare to any other OS. It allows us to control and configure over MAC and DAC. We can change standard algorithms to harden security. Since source code is available, we can modify security. For example: we can control, unnecessary ports should not be opened, Password hashing, services configurations, Mandatory Access Control (MAC) such as SELinux, FileSystem Encryption, Trusted Platform Module, User Space Hardening, Kenel Hardening etc.

The rationale for this movement is that a larger group of programmers not concerned with proprietary ownership or financial gain will produce a more useful and bug-free product for everyone to use. The concept relies on peer review to find and eliminate bugs in the program code, a process that commercially developed and packaged programs do not employ.

2. Introducing different flavors of Linux Operating System

(**RHEL, Centos, Fedora, SUSE, Ubuntu** etc.)

- **Debian** GNU/Linux is a distribution that emphasizes free software. It supports many hardware platforms. Debian and distributions based on it use the .deb package format and the dpkgpackage manager and its frontends.

- **Ubuntu** is a distribution based on Debian, designed to have regular releases, a consistent user experience and commercial support on both desktop and server.

- **Fedora** is a community supported distribution based on Red Hat. It aims to provide the latest software while maintaining a completely Free Software system.

- **Red Hat Linux** and **SUSE Linux** **were the original major distributions that used the RPM file format, which is today used in several package management systems. Both of these later divided into commercial and community-supported distributions.** ***Red Hat Linux divided into a community-supported distribution sponsored by Red Hat called Fedora, and a commercially supported distribution called Red Hat Enterprise Linux (RHEL).***

Red Hat Linux was one of the first and most popular Linux distributions. This was largely because, while a paid-for supported version was available, a freely downloadable version is also available. Since the only difference between the paid-for option and the free option was support, a great number of people chose to use the free version.

Red Hat made the decision to split its Red Hat Linux product into two: **Red Hat Enterprise Linux** for customers who were willing to pay for it, and Fedora that was made available free of charge but gets updates for every release for approximately 13 months.

Fedora has its own beta cycle and has some issues fixed by contributors who include Red Hat staff. However, its quick and non conservative release cycle means it might not be suitable for some users. Fedora is somewhat a test-bed for RedHat, allowing them to beta test their new features before they get included in Red Hat Enterprise Linux. Since the release of Fedora, Red Hat has no longer made binary versions of its commercial product available free-of-charge.

Red Hat Enterprise Linux	
 Red Hat Enterprise Linux 6's default GNOME 2 desktop	
Company /developer	Red Hat, Inc.
OS family	Unix-like (based on Red Hat Linux/Fedora)
Working state	Current
Source model	Free and open source software(with exceptions)[1]

Initial release	March 31, 2003 (RHLdiscontinued) & started as RHEL on March 26, 2002[clarification needed]
Latest stable release	6.4, 5.10 / February 21, 2013; 7 months ago, September 30, 2013; 24 days ago
Marketing target	Commercial market (including formainframes, servers,supercomputers)
Availablelanguage(s)	Multilingual
Update method	Yum / PackageKit
Package manager	RPM
Supported platforms	x86, x86-64; Power Architecture;S/390; z/Architecture[2]
Kernel type	Monolithic (Linux)
Userland	GNU
Default user interface	GNOME
License	Various free software licenses, plus proprietary binary blobs.[1]
Preceded by	Red Hat Linux
Official website	www.redhat.com/rhel

CentOS ("Community Enterprise Operating System") is a Linux distribution that attempts to provide a free enterprise class computing platform which has 100% binary compatibility with its upstream source, Red Hat Enterprise Linux (RHEL). As of version 6.4, it officially supports

the x86 architecture with Physical Address Extension and the x86-64 architecture, while a beta is expected to be available for PowerPC.

Linux Principles

i. Everything is a file. (Including hardware)
ii. Small, single-purpose programs.
iii. Ability to chain programs together to perform complex tasks.
iv. Avoid captive user interfaces.
v. Configuration data stored in text.

- **Everything is a File** : -

UNIX systems have many powerful utilities designed to create and manipulate files. The UNIX security model is based around the security of files. By treating everything as a file, a consistency emerges. You can secure access to hardware in the same way as you secure access to a document.

- **Small, single-purpose programs** : -

UNIX provides many small utilities that perform one task very well. When new functionality is required, the general philosophy is to create a separate program – rather than to extend an existing utility with new features.

- **Ability to chain programs together to perform complex tasks** :-

A core design feature of UNIX is that the output of one program can be the input for another. This gives the user the flexibility to combine many small programs together to perform a larger, more complex task.

- **Avoid captive user interfaces** :-

Interactive commands are rare in UNIX. Most commands expect their options and arguments to be typed on the command line when the command is launched. The command completes normally, possibly producing output, or generates an error message and quits. Interactivity is reserved for programs where it makes sense, for example, text editors (of course, there are non-interactive text editors too.)

- **Configuration data stored in text** : -

Text is a universal interface, and many UNIX utilities exist to manipulate text. Storing configuration in text allows an administrator to move a configuration from one machine to another easily. There are several revision control applications that enable an administrator to

track which change was made on a particular day, and provide the ability to roll back a system configuration to a particular date and time.

Extra Package for Enterprise Linux (EPEL)

EPEL (Extra Packages for Enterprise Linux) is a volunteer-based community effort from the Fedora project to create a repository of high-quality add-on packages that complement the Fedora-based Red Hat Enterprise Linux (RHEL) and its compatible spinoffs, such as CentOS and Scientific Linux. EPEL provide lots of packages for CentOS / RHEL, It is not part of RedHat or CentOS but is designed to work with these major distributions. Please note that EPEL only provides free and open source software unencumbered by patents or any legal issues. In short you will not find mp3, dvd and music / media player under EPEL. However, you will find many programs related to networking, monitoring, sys admin, programming and so on.

LINUX Boot Process and GRUB:

How Linux OS boots up?
How does it work?

What is Boot Process?

In computing, booting (also known as booting up) is the initial set of operations that a computer system performs after electrical power to the CPU is switched on or when the computer is reset. The process begins when a computer is turned on for the first time, is re-energized after being turned off, when it is reset or when the operator invokes a LOAD function from the console, and ends when the computer is ready to perform its normal operations.

a. **Stages of a typical Linux boot process**
   - **BIOS**
   - **MBR & its components**
   - **GRUB (Grand Unified Bootloader)**
     - **Function and about its configuration file (/boot/grub/grub.conf)**
   - **Kernel**
     - **Kernel monitoring and configuration**
     - **How to see Kernel Version, & Linux Version?**
   - **init**
   - **Runlevel**

b. **GRUB password**

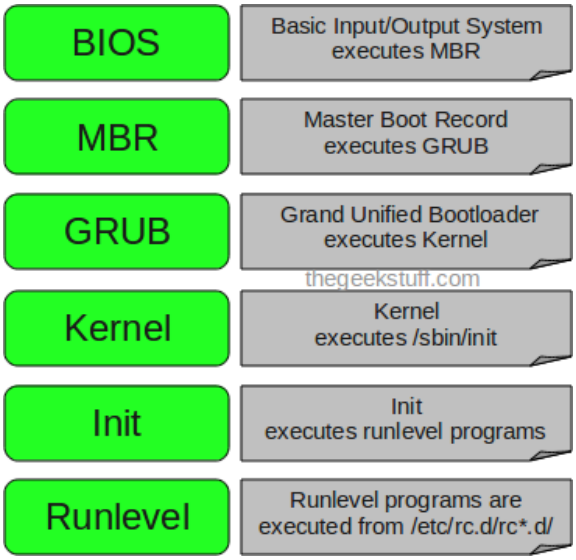
c. **Changing Runlevels**

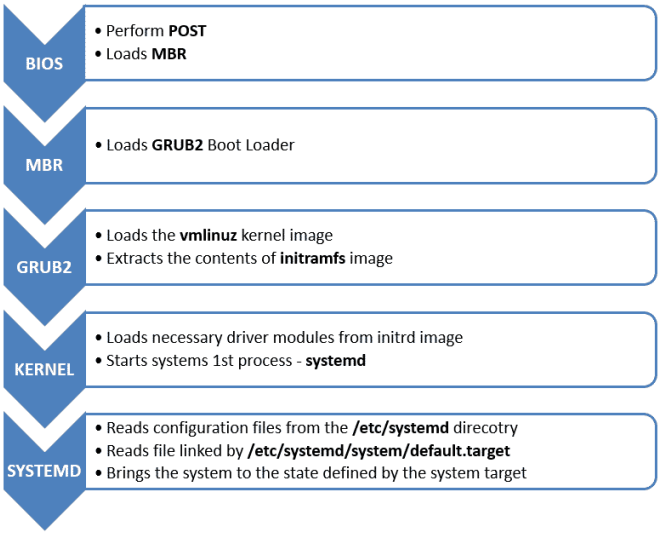
Stages of Linux Boot Process

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.

LAB: Access partition table of LINUX, check partition type (MBR or GPT)

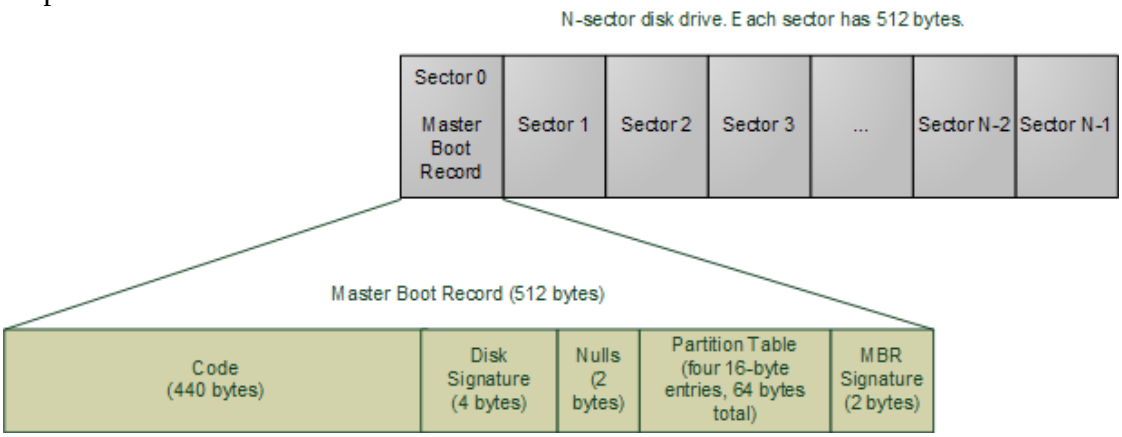
Q. How to see partition table in Linux

fdisk –l

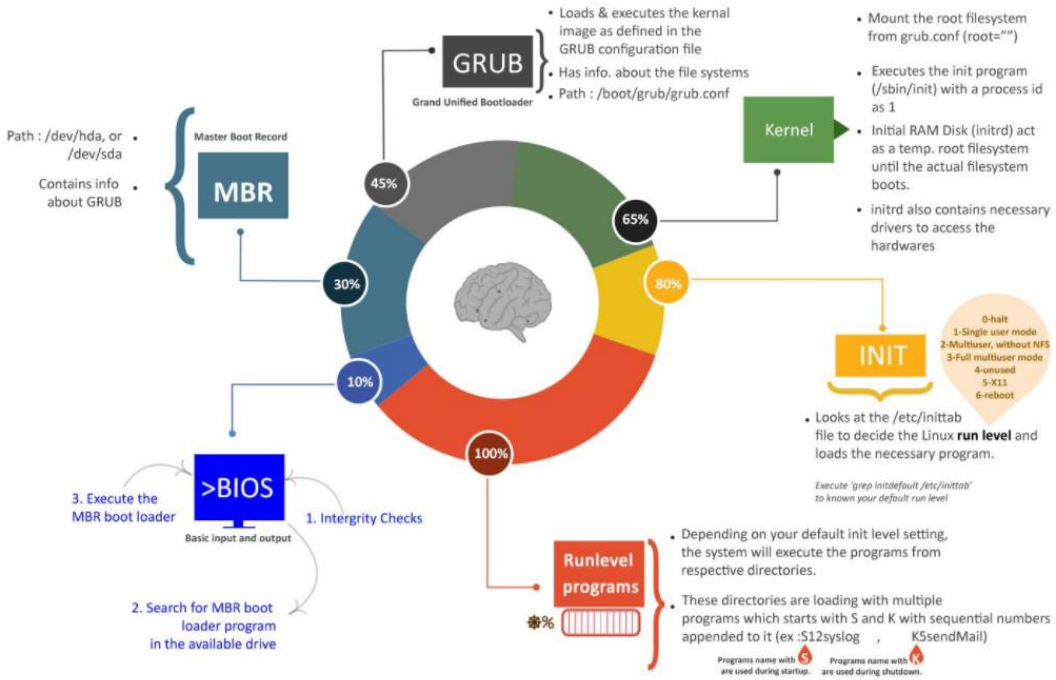
Q Check the partition is MBR or GPT

parted -l

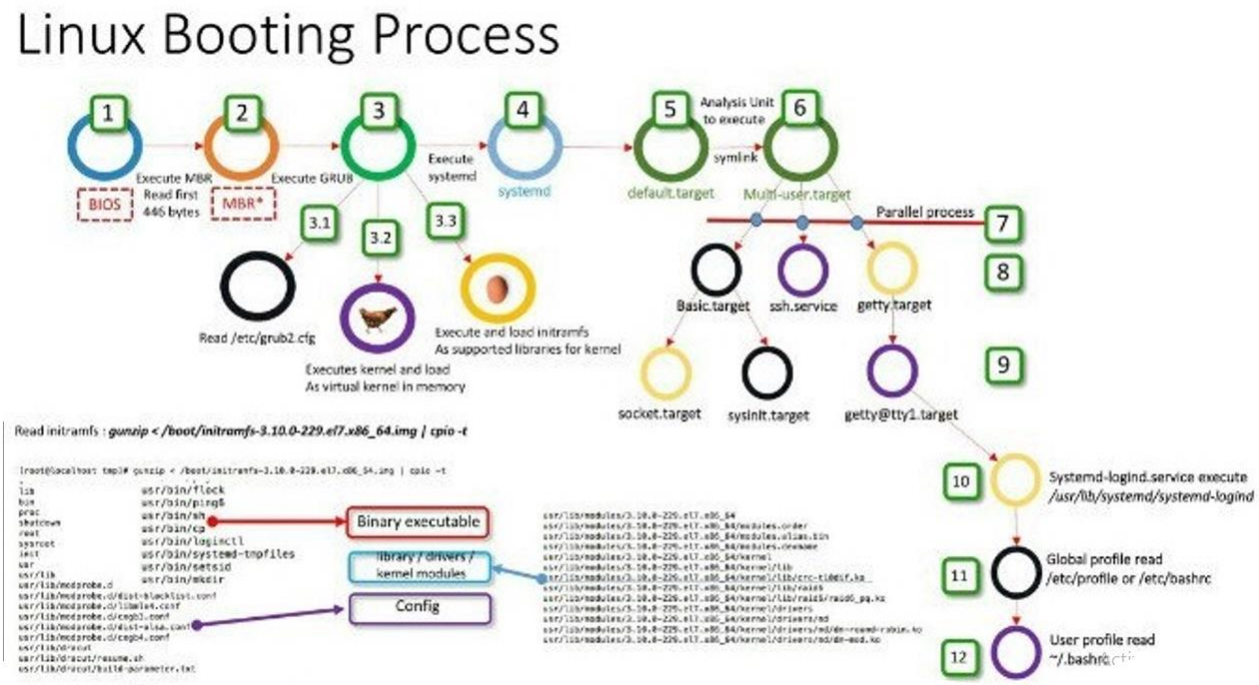
- It contains information about GRUB (or LILO in old systems).
- So, in simple terms MBR loads and executes the GRUB boot loader.



3. GRUB



4.



https://www.certdepot.net/rhel7-get-started-grub2/

- 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 of CentOS.
-

```
#boot=/dev/sda
default=0
timeout=5
splashimage=(hd0,0)/boot/grub/splash.xpm.gz
hiddenmenu
title CentOS (2.6.18-194.el5PAE)
        root (hd0,0)
        kernel /boot/vmlinuz-2.6.18-194.el5PAE ro root=LABEL=/
        initrd /boot/initrd-2.6.18-194.el5PAE.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.

- Important RHEL7: The default boot loader in **Redhat** enterprise Linux **7** is grub2(GRand Unified Bootloader2). The "grub2" supports almost all the operating systems. The main configuration file for grub2 is /boot/grub2/**grub**.**cfg**.
- What is initrd: `initrd` (*initial ramdisk*) is a scheme for loading a temporary root file system into memory, which may be used as part of the Linux startup process.

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**

- Looks at the /etc/inittab file to decide the Linux run level.
- 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
- Init identifies the default initlevel from /etc/inittab and uses that to load all appropriate program.
- Execute 'grep initdefault /etc/inittab' on your system to identify the default run level
- 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.
- Typically you would set the default run level to either 3 or 5.

Important:

This **systemctl** command is available because the Developers behind **CentOS** 7 replaced upstart with **systemd** as the default init system. It means now **systemd**becomes default system management daemon which is different from the old default init system

In Unix-based computer operating systems, **init** (short for **initialization**) is the first**process** started during booting of the computer system. **Init** is a daemon **process**that continues running until the system is shut down.

On systems based on **SysVinit**, init is the first process that is executed once the Linux kernel loads. The default init program used by the kernel is /sbin/init provided by systemd-sysvcompat (by default on new installs, see systemd) or **sysvinit**[AUR]. ... inittab is the startup configuration file for init located in /etc .

System startup: The **systemd** process is the first process ID (PID 1) to run on **RHEL 7** system. It initializes the system and launches all the services that were once started by the traditional init process. Managing system services: For **RHEL 7**, the systemctl command replaces service and chkconfig.

Reference: https://access.redhat.com/articles/754933

SysVinit V/s systemd runlevels

Here is a comparison between SysVinit runlevels V/s systemd targets.

Sysvinit Runlevel	Systemd Target	Function
0	runlevel0.target, poweroff.target	System halt/shutdown
1, s, single	runlevel1.target, rescue.target	Single-user mode
2, 4	runlevel2.target, runlevel4.target, multi-user.target	User-defined/Site-specific runlevels. By default, identical to 3.
3	runlevel3.target, multi-user.target	Multi-user, non-graphical mode, text console only
5	runlevel5.target, graphical.target	Multi-user, graphical mode
6	runlevel6.target, reboot.target	Reboot
emergency	emergency.target	Emergency mode

LAB: Changing Runlevel

Changing runlevels with systemd

The runlevel target can be changed by using the systemctl isolate command :

```
# systemctl isolate multi-user.target
```

Changing the default runlevel

The default runlevel can be changed by using the set-default option.

```
# systemctl set-default multi-user.target
```

To get the currently set default, you can use the get-default option.

```
# systemctl get-default
```

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 runlevel 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/

Installation of Linux, Comparison between RHEL 6 and 7, and Understanding Inverted root tree structure

About RHEL 7.x version:

What's Changed in RHEL 7 Administration side:

- System Boot time is optimized to boot faster
- Anaconda Installer completely redesigned
  Anaconda is the installation program used by Fedora, Red Hat Enterprise **Linux** and some other distributions. During installation, a target computer's hardware is identified and configured, and the appropriate file systems for the system's architecture are created.
- Grub boot loader version changed from 0.97 to Grub 2
- No More, SysV Initd system changed to systemd system

- Network Interface names changed from ethx to ensxxx
- Introduced new concept of creating multiple Network profiles to activate based on network you connected (Ex. Home, Office and Other)
- Default Database is changed from MySQL to MariaDB
- No More editing of Network configuration file for assigning IP address and creating Teaming interfaces use nmcli utility
- Ifconfig and route commands are deprecated in RHEL 7, Replaced with ip command
- GNOME Version 2 replaced with GNOME 3 default Desktop
- System User UID range changed from 0-449 to 0-999
- Locate command is changed to mlocate
- Cluster Resource Manager changed from RGManager to Pacemaker and all CMAN features merged into Corosync
- Netstat command replaced with ss command
- NTP Daemon replaced with chronyd faster way to sync time RHEL 6 vs RHEL 7
- Directories /bin, /sbin, /lib and /lib64 moved under /usr directory

Comparison RHEL 6.x vs RHEL 7.x

Feature Name	RHEL 6	RHEL 7
Default File System	Ext4	XFS
Kernel Version	2.6.xx	3.10.xx
Release Name	Santiago	Maipo
Gnome Version	GNOME 2	GNOME 3.8
KDE Version	KDE 4.1	KDE 4.6
Release Date	Wednesday, November 10, 2010	Tuesday, June 10, 2014
NFS Version	NFS 4	NFS 4.1. NFS V2 is deprecated in RHEL 7
Samba Version	SMB 3.6	SMB 4.4
Default Database	MySQL	MariaDB
Cluster Resource Manager	Rgmanager	Pacemaker
Network InterfaceGrouping	Bonding can be done as Active-Backup, XOR, IEEE and Load Balancing	Team Driver will support multiple types of Teaming methods called Active-Backup, Load-balancing and Broadcast
KDUMP	Kdump does't support with large RAM Size	RHEL 7 can be supported up to 3TB
Boot Loader	Grub 0.97 /boot/grub/grub.conf	Grub 2 /boot/grub2/grub.cfg
File System Check	e2fsck -Inode check. Block and size check –Directory Structure check -Directory Link Check -reference count check -Group Summary Check	xfs_replair – Inode blockmap checks -Inode allocation map checks -Inode size check -Directory check -Path Name check -Link count check -Freemap check -Super block check
Process ID	Initd Process ID 1	Systemd Process ID 1
Port Security	Iptables by default service port is enabled when service is switched on.	Firewalld instead of iptables. Iptables can also support with RHEL 7, but we can't use both of them at the same time. Firewall will not allow any port until and unless you enabled it.
Boot Time	40 Sec	20 Sec
File System Size	EXT4 16TB with XFS 100TB	XFS 500TB with EXT4 16TB
Processor Architecture	32Bit and 64Bit	Only 64Bit.
Network Configuration Tool	Setup	Nmtui
Host name Config File	/etc/sysconfig/network	/etc/hostname No need to edit hostname file to write permanent hostname simply use hostnamectl command

Interface Name	eth0	ens33xxx
Managing Services	service sshd start service sshd restart chkconfig sshd on	systemctl start sshd.service systemctl restart sshd.service systemctl enable sshd.service
System Logs	/var/log/	/var/log journalctl
Run Levels	runlevel 0 – Power Off runlevel 1 – Single User Mode runlevel 2 – Multi User without Networking runlevel 3 – Multi User CLI runlevel 4 – Not USed runlevel 5 – GUI Mode runlevel 6 – Restart	There is no run levels in RHEL 7. Run levels are called as targets Poweroff.target rescue.target multi-user.target graphical.target reboot.target
UID Information	Normal User UID will start from 500 to 65534 System Users UID will start from 1 to 499	Normal User UID start from 1000 – 65534 System Users UID will start from 1 to 999Because Services are increased compare to RHEL 6
By Pass Root Password Prompt	append 1 or s or init=/bin/bash to Kernel command line	Append rd.break or init=/bin/bash to kernel command line
Rebooting and Poweroff	poweroff – init 0 reboot – init 6	systemctl poweroff systemctl reboot
YUM Commands	yum groupinstall yum groupinfo	yum group install yum group info

Installation: RHEL 6

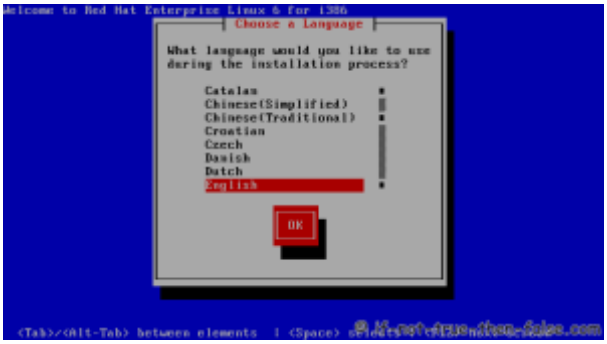
Burn Downloaded Image to DVD and Boot Computer Using Red Hat 6 Installation DVD

Check RHEL image MD5 sum and burn image to DVD with your favourite CD/DVD burner. And boot computer using Red Hat Installation DVD.

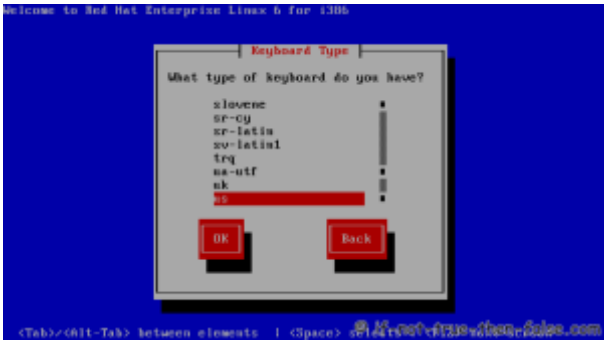
LAB: Red Hat 6 RHEL Installation

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

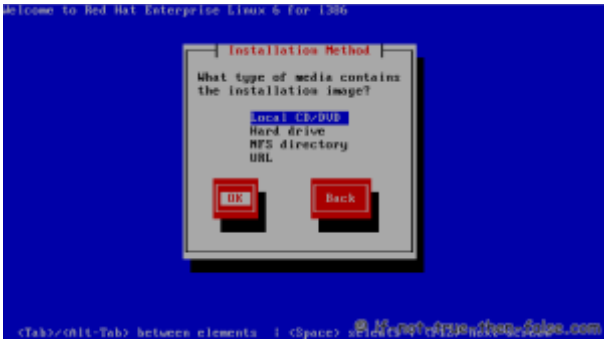
2. Choose a language



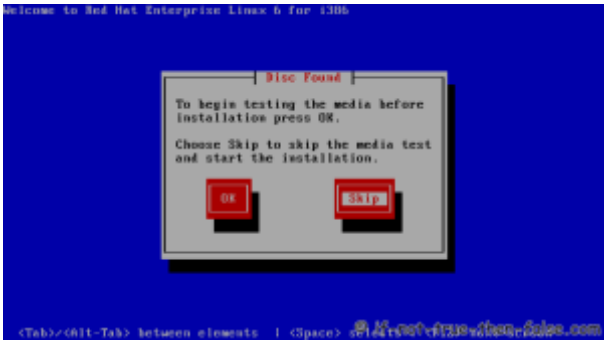
3. Choose a keyboard type



4. Choose a installation media

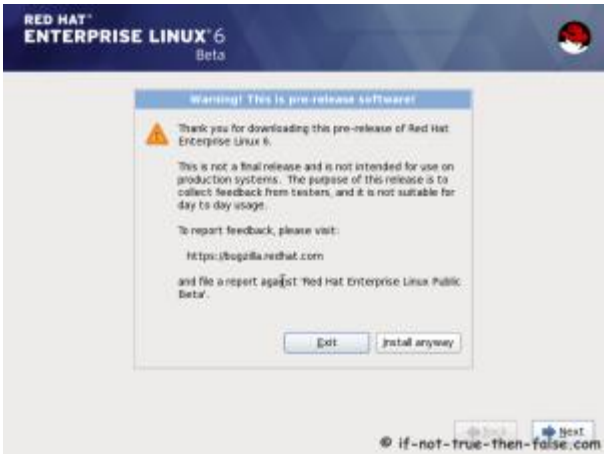


5. Skip DVD media test (or select media test, if you want to test installation media before installation)



6. Red Hat 6 graphical installer starts, select next

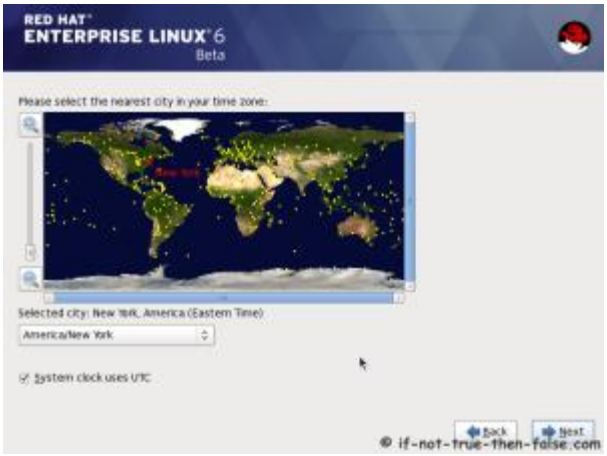
7. Accepct Pre-Release Installation



8. Select storage devices

9. Insert computer name

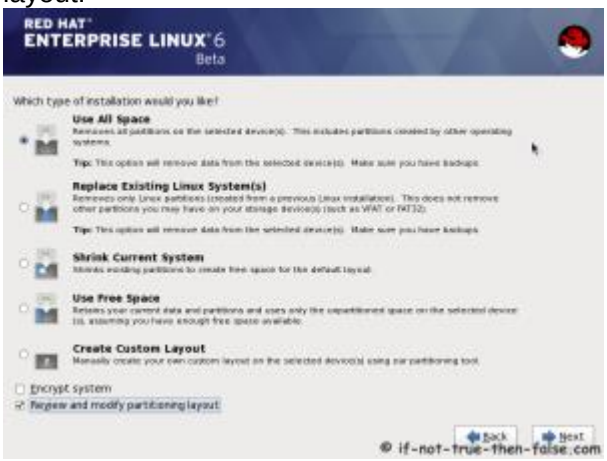
10. Select time zone



11. Enter a password for *root* user

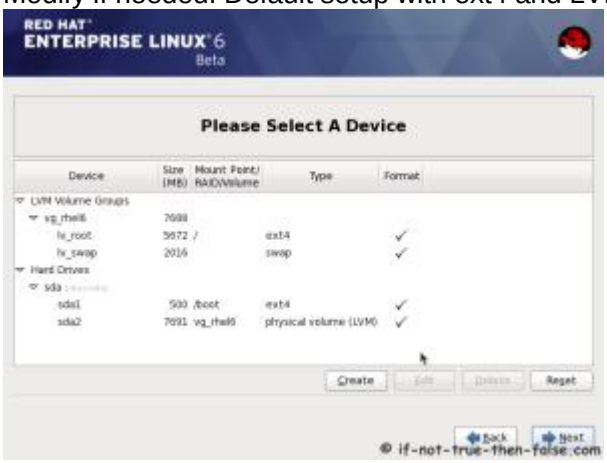
12. Select type of installation

Read every options info carefully. And select encrypting if needed and option to review and modify partition layout.



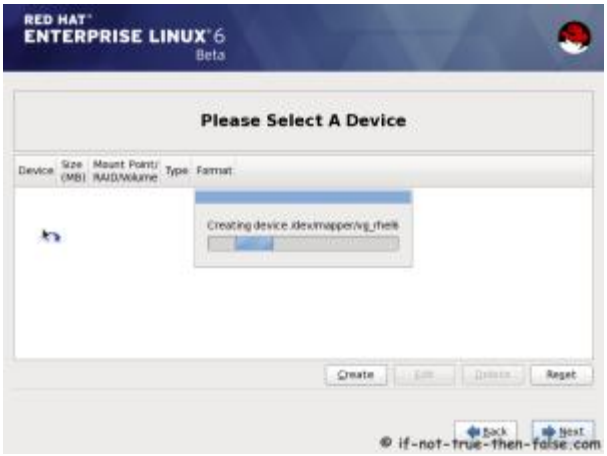
13. Review partition layout

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



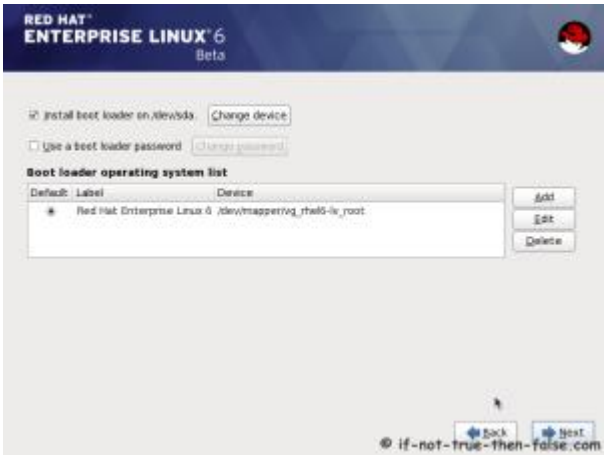
14. Accept write changes to disc

15. Writing changes (creating partitions) to disc



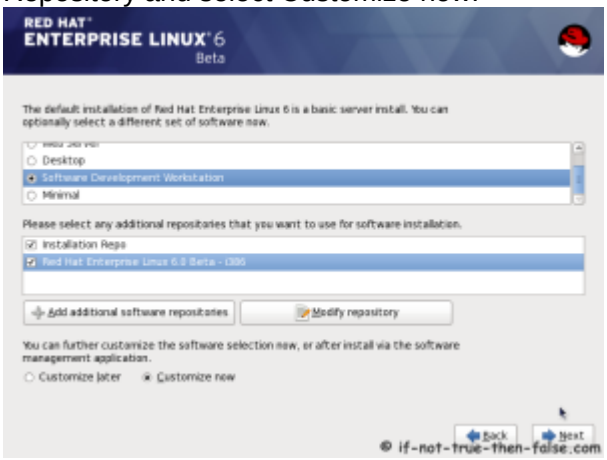
16. Configure boot loader options

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



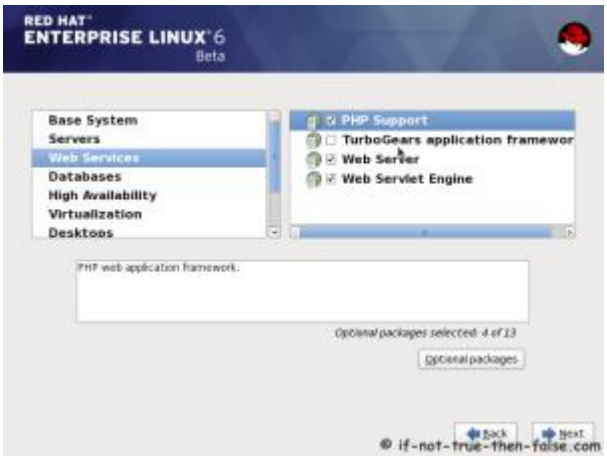
17. Select softwares to install and enable repositories

This case we select *Software Development Workstation* and enable Red Hat Enterprise Linux 6.0 Beta Repository and select Customize now.

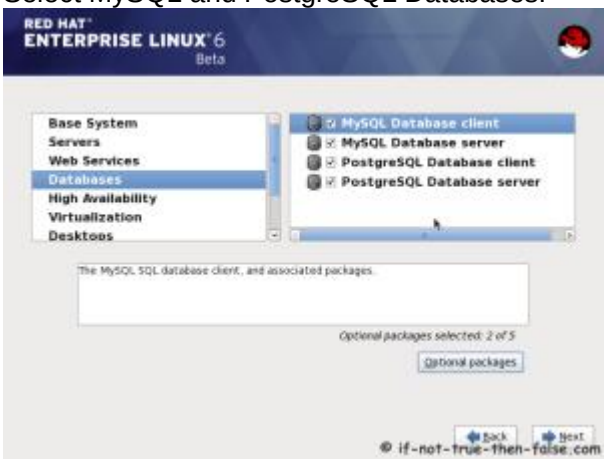


18. Customize package selection

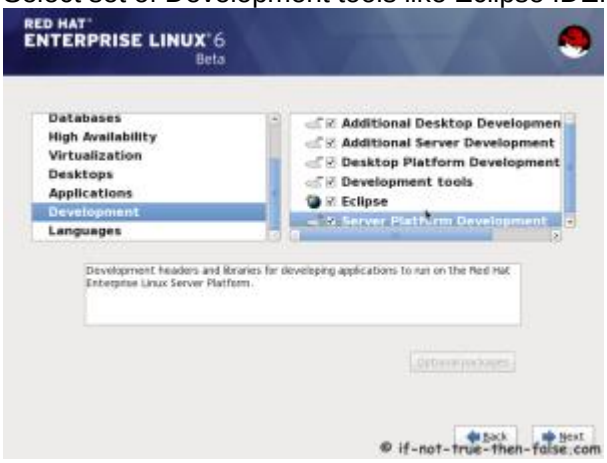
Select PHP and Web Server to installation.



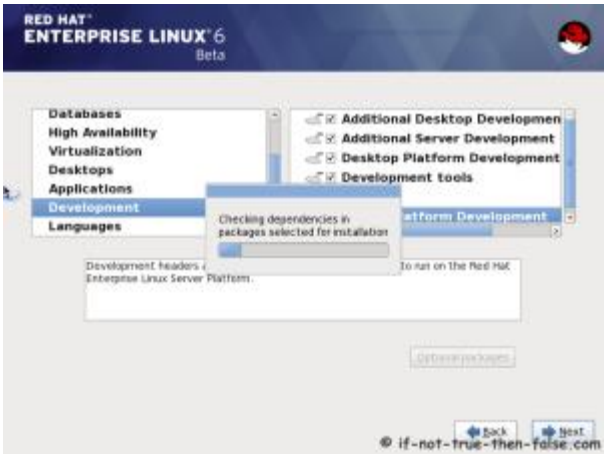
Select MySQL and PostgreSQL Databases.



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.

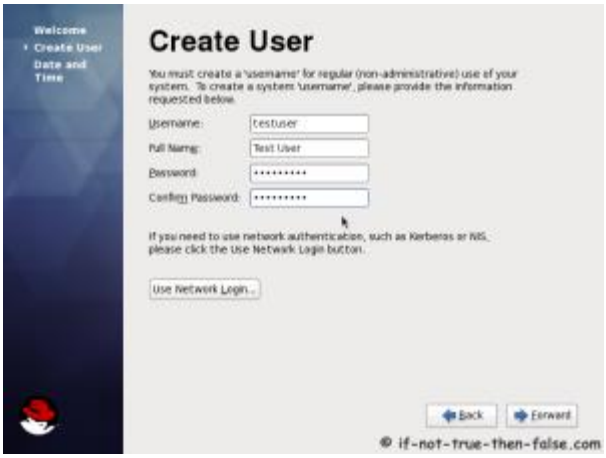
Red Hat 6 RHEL Finishing Installation

23. Selecting RHEL 6 from grub

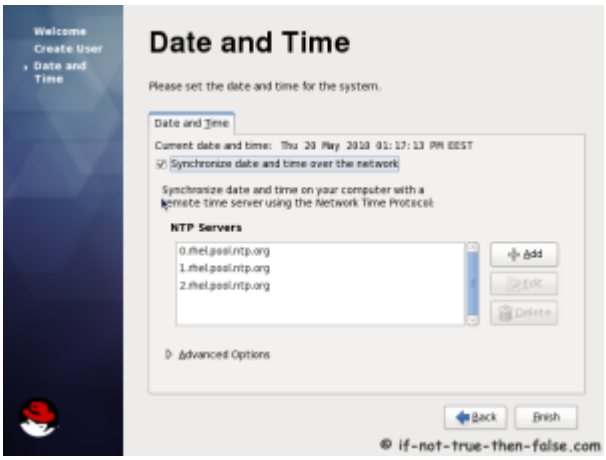
24. Booting Red Hat 6

25. Red Hat 6 Welcome screen

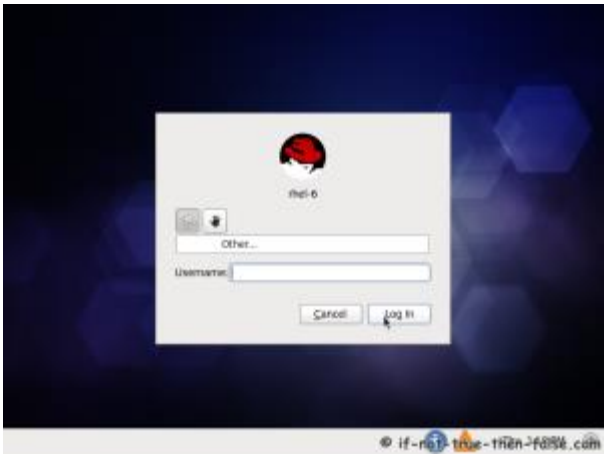
26. Create normal user



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



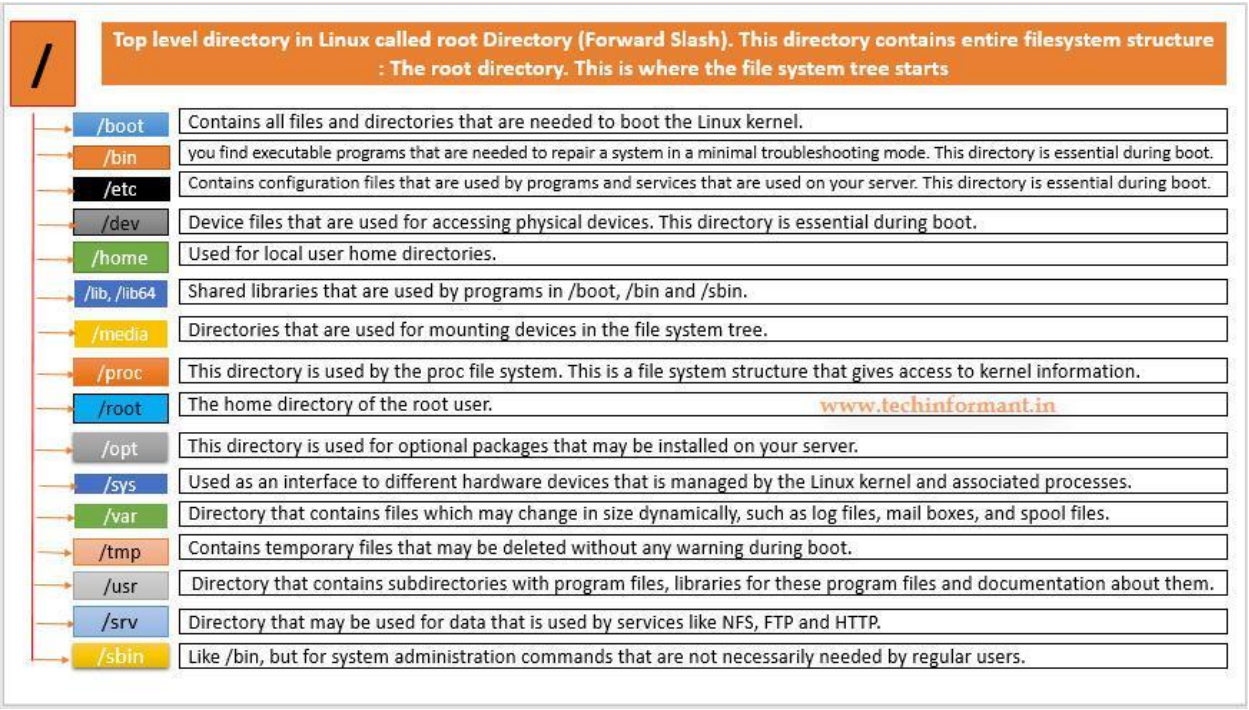
28. Login Red Hat 6 Gnome Desktop



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

File System (Inverted root tree structure)

RHEL 7.x




```
File  Edit  View  Search  Terminal  Help
[root@localhost ~]# cd /
[root@localhost /]# ll
total 32
lrwxrwxrwx.   1 root root    7 Jul 21  2016 bin -> usr/bin
dr-xr-xr-x.   4 root root 4096 Dec 22 09:45 boot
drwxrwxrwx.   2 root root   25 Aug  1  2016 datashare
drwxr-xr-x.  21 root root 3360 Dec 22 10:03 dev
drwxr-xr-x. 132 root root 8192 Dec 22 10:03 etc
drwxr-xr-x.   4 root root   30 Aug  1  2016 home
lrwxrwxrwx.   1 root root    7 Jul 21  2016 lib -> usr/lib
lrwxrwxrwx.   1 root root    9 Jul 21  2016 lib64 -> usr/lib64
drwxr-xr-x.   2 root root    6 Jun  9  2014 media
drwxr-xr-x.   3 root root   15 Jul 21  2016 mnt
drwxr-xr-x.   3 root root   15 Jul 21  2016 opt
dr-xr-xr-x. 364 root root    0 Dec 22 10:03 proc
dr-xr-x---.   6 root root 4096 Dec 22 12:09 root
drwxr-xr-x.  34 root root 1020 Dec 22 13:19 run
lrwxrwxrwx.   1 root root    8 Jul 21  2016 sbin -> usr/sbin
drwxr-xr-x.   2 root root    6 Jun  9  2014 srv
dr-xr-xr-x.  13 root root    0 Dec 22 10:03 sys
drwxrwxrwt.  24 root root 4096 Dec 22 12:29 tmp
drwxr-xr-x.  13 root root 4096 Jul 21  2016 usr
drwxr-xr-x.  23 root root 4096 Dec 22 10:03 var
[root@localhost /]#
```

LAB: Explore Linux File System eg. /root, /tmp, /home, /etc, /lib etc

YUM Repository Configuration, & Redhat Package Management (Find, Install, Remove and Update)

YUM (Yellow Dog Updater, Modified) Tool for Package Management

In this topic, we will learn how to **install, update, remove, find packages, manage packages and repositories on Linux systems using YUM** (Yellow Dog Updater Modified) tool developed by RedHat. Versions of Red Hat Enterprise Linux 4 and earlier used up2date.

The example commands shown in this topic are practically tested on our **CentOS & RHEL** server.

YUM

- It is an open source command-line as well as graphical based package management tool for **RPM** (**RedHat Package Manager**) based Linux systems.
- It allows users and system administrator to easily install, update, remove or search software packages on a systems.
- It was developed and released under **GPL** (**General Public License**) as an open source, means anyone can be allowed to download and access the code to fix bugs and develop customized packages.
- Third party repository can be used.

YUM COMMAND CHEAT SHEET

for Red Hat Enterprise Linux

YUM QUERIES

SUBCOMMAND	DESCRIPTIONS AND TASKS
`help`	Display yum commands and options `yum help` Show yum subcommands and options

Individual packages

`list`	List package names from repositories
	`yum list available`
	List all available packages
	`yum list installed`
	List all installed packages
	`yum list all`
	List installed and available packages
	`yum list kernel`
	List installed and available kernel packages

`info`	Display information about a package `yum info vsftpd` List info about vsftpd package
`deplist`	Display dependencies for a package `yum deplist nfs-utils` List dependencies and packages providing them

`provides`	Find packages that provide the queried file
	`yum provides "*bin/top"`
	Show package that contains top command
	`yum provides "*/README.top"`
	Show package containing README.top file

`search`	Search package names and descriptions for a term `yum search samba` Find packages with samba in name or description
`updateinfo`	Get information about available package updates `yum updateinfo security` Get info on available security updates

Groups of packages

`grouplist`	List names of installed and available package groups
`groupinfo`	Display description and contents of a package group `yum groupinfo "Web Server"` See packages in Web Server group
`check-update`	Query repositories for available package updates

MANAGE YUM REPOSITORIES

SUBCOMMAND	DESCRIPTIONS AND TASKS
`repolist`	Display enabled software repositories
`repoinfo`	Display information about enabled yum repositories * `yum repoinfo rhel-7-server-rpms` See info on rhel-7-server-rpms repo

`repo-pkgs`	Work with packages in a particular repository *
	`yum repo-pkgs my-rpms list`
	List packages from my-rpms repo
	`yum repo-pkgs my-rpms install`
	Install all packages from my-rpms repo
	`yum repo-pkgs my-rpms remove`
	Remove all packages from my-rpms repo

`makecache`	Download yum repository data to cache

TROUBLESHOOT AND MAINTAIN YUM

SUBCOMMAND	DESCRIPTIONS AND TASKS
`check`	Check the local RPM database for problems (runs for a long time)

`history`	View and use yum transactions
	`yum history list`
	List all yum install, update and erase actions
	`yum history info 3`
	Show details of yum transaction 3
	`yum history undo 3`
	Undo the yum action from transaction 3
	`yum history redo 3`
	Redo the undone yum action from transaction 3

`clean`	Clear out cached package data
	`yum clean packages`
	Delete packages saved in cache
	`yum clean all`
	Clean out all packages and meta data from cache

`fssnapshot`	List LVM stapshots (helps roll back after package updates)

`fs`	Act on filesystem (prevent doc or language file install on minimal systems)
	`yum fs filters`
	List enabled filesystem filters
	`yum fs documentation`
	Filters all docs from being installed (careful!)

INSTALL, REMOVE AND UPGRADE PACKAGES WITH YUM

SUBCOMMAND	DESCRIPTIONS AND TASKS
`install`	Install a package from a repository to your system `yum install vsftpd` Install the vsftpd package

`update`	Update one or all packages on your system
	`yum update`
	Update all packages with available updates
	`yum update httpd`
	Update the httpd package (if available)
	`yum update --security`
	Apply security-related package updates

`update-to`	Update one or all packages to a particular version
`upgrade`	Update packages taking obsoletes into account

`localinstall`	Install a package from a local file, http, or ftp
	`yum localinstall abc-1-1.i686.rpm`
	Install abc package from local directory
	`yum localinstall http://myrepo/abc-1-1.i686.rpm`
	Install abc from FTP site

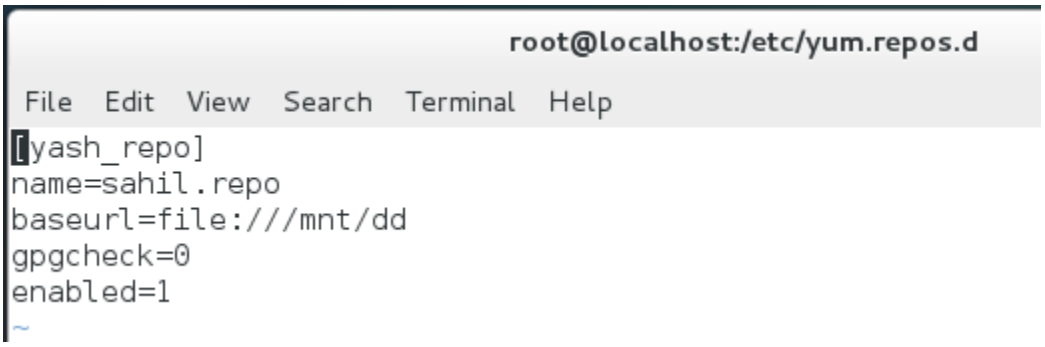
`downgrade`	Downgrade a package to an earlier version `yum downgrade abc` Downgrade the abc package to an earlier version
`reinstall`	Reinstall the current version of a package `yum reinstall util-linux` Reinstall util-linux (to replace any deleted files)
`swap`	Remove one package and install another `yum swap ftp lftp` Remove ftp package and install lftp package
`erase`	Erase a package (and possibly dependencies) from your system `yum remove vsftpd` Remove the vsftpd package and dependencies
`remove`	Same as erase
`autoremove`	Same as erase, plus removes additional unneeded packages * `yum autoremove httpd` Remove httpd and other unneeded packages
`groupinstall`	Install all packages in the selected group `yum groupinstall "Web server"` Install Web Server packages

LAB: Yum configuration

Need to set up yum repository for locally-mounted DVD on Red Hat Enterprise Linux 7

https://access.redhat.com/solutions/1355683

- How to set up yum repository to use locally-mounted DVD with Red Hat Enterprise Linux (RHEL) 7
- Would like to upgrade server from RHEL 7.x to RHEL 7.y
- Have a secure environment that will never be connected to the internet, but still needs to be updated
- Way to update the packages on server, with no satellite server and servers disconnected from internet
- Offline patches for Red Hat systems
- How do I create a local repository in RHEL 7?



LAB: Package Management

Refer above cheat-sheet

How do I use YUM?

Yum must be running as root. Here are some useful commands:

1) Install a package:

```
yum install package
```

Example:

```
yum install httpd
```

2) Remove a package:

```
yum remove package
```

Example:

```
yum remove httpd
```

3) Update a package:

```
yum update package
```

Example:

```
yum update httpd
```

4) Search for a package:

```
yum search package
```

Example:

```
yum search httpd
```

5) Find information about a package:

```
yum info package
```

Example:

```
yum info httpd
```

6) List packages containing a certain term:

```
yum list term
```

Example:

```
yum list httpd
```

7) Find what package provides a particular file:

```
yum whatprovides 'path/filename'
```

Example:

```
yum whatprovides 'etc/httpd.conf'
yum whatprovides '*/libXp.so.6'
```

8) Update all installed packages with kernel package :

```
yum update
```

Example:

```
yum update
```

9) To update a specific package:

```
yum update <package-name>
```

Example:

```
yum update openssh-server
```

Basic Linux Commands:

Getting Help

Note: No need to memorize all commands
Commands to get help
#whatis <command>

- Displays short description of commands
- Uses a database that is updated nightly
- If whatis does not work, run ***makewhatis*** as root

- #makewhatis

```
[root@localhost ~]# whatis chkconfig
chkconfig            (8)  - updates and queries runlevel information for system services
chkconfig           (rpm) - A system tool for maintaining the /etc/rc*.d hierarchy.
[root@localhost ~]#
```

#<command> --help

- Displays usage summary and argument list
- Used by most, but not all, commands

```
[root@localhost ~]# chkconfig --help
chkconfig version 1.3.30.2 - Copyright (C) 1997-2000 Red Hat, Inc.
This may be freely redistributed under the terms of the GNU Public License.

usage:   chkconfig --list [name]
         chkconfig --add <name>
         chkconfig --del <name>
         chkconfig [--level <levels>] <name> <on|off|reset|resetpriorities>
[root@localhost ~]#
```

#man [<chapter>] <command>

- Provides documentation for commands
- Every command has a man page
- Linux manual is divided into section
    - 1 → User commands
    - 2 → System calls
    - 3 → Library calls
    - 4 → Special files
    - 5 → File formats
    - 6 → Games
    - 7 → Misc
    - 8 → Administrative commands

Example:

man 7 man
#man 5 vsftpd.conf

man page navigation

- Use arrow keys, PgUP, and PgDn for navigation
- For searching text use **/text** with n/N to next and previous match
- Use q to quit

Info command

#info <command>

- Similar to man command, but it provides more details
- Info pages are structured like a web site
- Each page is divided into nodes
- Links to nodes are preceded by *

Organizing files and directory (command line)

- Listing (ls)
    - ls -a (include hidden files)
    - ls -l (display extra information)
    - ls -R (Recurse through directories)
    - ls -ld (directory and symbolic information)
- Copying (cp)
- Deleting (rm, rmdir)
- Moving (mv)
- Renaming or files and directories (mv)
- Creating empty file (touch)
- Creating file and writing data into it by using cat command
- Seeing present working directory (pwd)

Instructions to practice above said command to organize files and directories

1. Create directory under /home with name mydir
2. Change directory to /home/mydir
3. Create a few empty files in /home/mydir
4. Copy /etc/passwd file into /home/mydir
5. Create a file using cat command in /home/mydir directory
6. Move some files from mydir to /tmp
7. Copy some files from mydir to /tmp
8. Change directory to /tmp
9. Rename file name
10. Delete file
11. Delete myfile directory under /home

Organizing files and directory (nautilus)

Nautilus is a GNOME graphical filesystem browser.
#nautilus

Discussing absolute and Relative pathname

Absolute pathname: Complete to file location starts with forward slash
Relative pathname: Specify location relative to your current working directory and does not starts with "/"

Editing file with vi editor

- Opening file

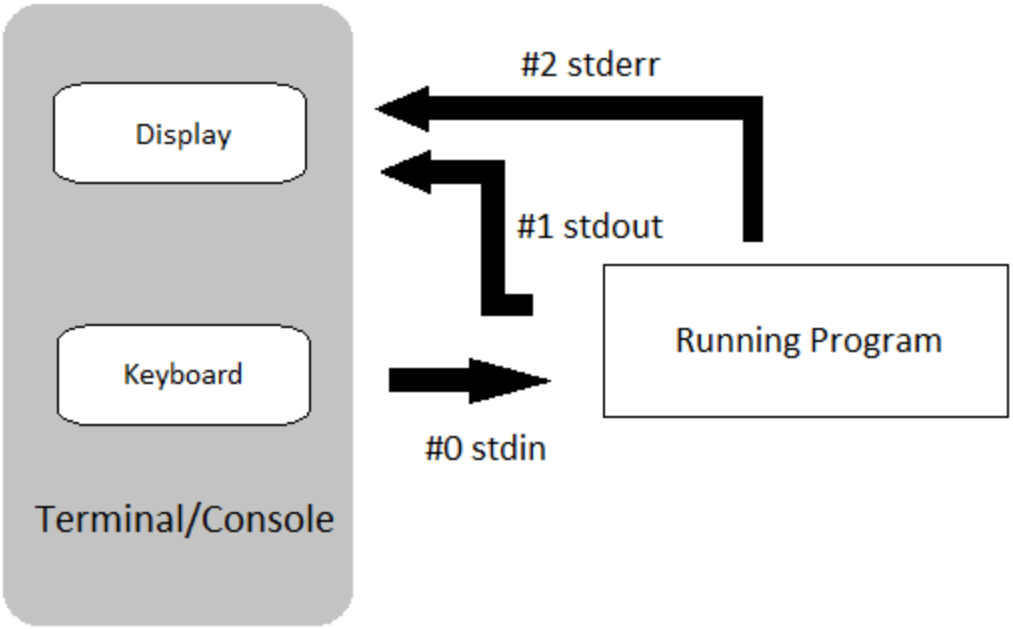
- Modifying file
- Saving file
- Exiting without saving
- Searching text
- Deleting lines
- Copying lines
- Making line numbers visible

I/O Redirection – STDIN, STDOUT, STDERR streams

In Linux everything is file. Devices like keyboard, cpu, hard disk are considered as
By default there are always three files open,
stdin (keyboard),
stdout (display) and
stderr (error messages outputted on the screen).

As with every open file, it is possible to redirect the three files.
For instance you can redirect (capture) the output of a command, file, script and send it to another command, file or script as input.

In UNIX and Linux every open file is assigned a file descriptor. The operating system (UNIX and Linux) needs to keep track of which files are open and it does this by assigning a number to an open file. (A file descriptor can be considered as a simplified type of **file pointer** or **file handle in C programming language** The file descriptors for *stdin*, *stdout*, and *stderr* are 0, 1, and 2. There are also file descriptors 3 to 9, that can be used to open additional files. These remaining file descriptors can be useful to assign to *stdin*, *stdout*, or *stderr* as a temporary duplicate link.



STDOUT and STDERR can be redirected to files

1. Command > file (direct std output of command to a file)

```
[root@localhost ~]# ifconfig eth0 > myfile
[root@localhost ~]# cat myfile
eth0      Link encap:Ethernet  HWaddr 00:0C:29:07:E3:27
          inet addr:192.168.192.100  Bcast:192.168.192.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe07:e327/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:166 errors:0 dropped:0 overruns:0 frame:0
          TX packets:44 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:16935 (16.5 KiB)  TX bytes:6470 (6.3 KiB)
          Interrupt:67 Base address:0x2000

[root@localhost ~]#
```

2. Command >> file (append std output)

```
[root@localhost ~]# ifconfig lo >> myfile
[root@localhost ~]# cat myfile
eth0      Link encap:Ethernet  HWaddr 00:0C:29:07:E3:27
          inet addr:192.168.192.100  Bcast:192.168.192.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe07:e327/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:166 errors:0 dropped:0 overruns:0 frame:0
          TX packets:44 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:16935 (16.5 KiB)  TX bytes:6470 (6.3 KiB)
          Interrupt:67 Base address:0x2000

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:16436  Metric:1
          RX packets:1373 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1373 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2323818 (2.2 MiB)  TX bytes:2323818 (2.2 MiB)
```

3. Command < file (sends file as input to command)

4. Command 2> file (redirect error message)

   To redirect stderr to a file named /tmp/result:

```
[root@localhost ~]# ls /usr/user 2> /tmp/result
[root@localhost ~]# cat /tmp/result
ls: /usr/user: No such file or directory
[root@localhost ~]#
```

5. Command 2>> file (append error message)

6. Command &> file (redirect all output to the file)

   #ls /usr/user /usr &> /tmp/allresult

```
[root@localhost ~]# ls /usr /urs &> /tmp/allresult
[root@localhost ~]# cat /tmp/allresult
ls: /urs: No such file or directory
/usr:
bin
etc
games
include
kerberos
lib
libexec
local
sbin
share
src
tmp
X11R6
[root@localhost ~]#
```

Pipe [“|”]

Pipes are used to connect the commands

A *pipe* is a form of *redirection* that is used in Linux and other Unix-like operating systems to **send the output of one program/command to another program/command** for further processing.

Command1 | Command2

Above command sends STDOUT of command1 to SDTIN of command2

Examples of pipe “|” commands:

Example 1:

```
[root@localhost man]# ls -C
bg  de  es  hr  it  man0p  man1x  man3   man4   man5x  man7   man8x  mann  pt     ru  sv     zh_TW
cs  el  fi  hu  ja  man1   man2   man3p  man4x  man6   man7x  man9   nl    pt_BR  sk  tr
da  en  fr  id  ko  man1p  man2x  man3x  man5   man6x  man8   man9x  pl    ro     sl  zh_CN
[root@localhost man]# ls -C |tr 'a-z' 'A-Z'
BG  EL  FR  IT     MAN1   MAN2X  MAN4   MAN6   MAN8   MANN  PT_BR  SL     ZH_TW
CS  EN  HR  JA     MAN1P  MAN3   MAN4X  MAN6X  MAN8X  NL    RO     SV
DA  ES  HU  KO     MAN1X  MAN3P  MAN5   MAN7   MAN9   PL    RU     TR
DE  FI  ID  MAN0P  MAN2   MAN3X  MAN5X  MAN7X  MAN9X  PT    SK     ZH_CN
[root@localhost man]#
```

dmesg

(Display message or driver message) is a command which will show Kernel ring buffers.

These messages contain valuable information about device drivers loaded into the kernel at the time of booting as well as when we connect a hardware to the system on the fly.

In other words dmesg will give us details about hardware drivers connected to, disconnected from a machine and any errors when hardware driver is loaded into the kernel.

Display all the devices drivers loaded in to kernel.

#dmesg

Display hardware information related to Ethernet port eth0

```
dmesg | grep -i eth0
```

To display USB stuff

dmesg | grep -i usb

To display total memory available and shared memory details

dmesg | grep -i Memory

cut

To extract the desired column of file or from command output. For more information use man pages

To extract the desired line which matches the given word or phrase

#chkconfig –list|grep vsftpd
#grep root /etc/passed |cut –d: -f7

wc (Word count)

The wc (word count) command in Unix/Linux operating systems is used to find out number of newline count, word count, byte and characters count in a files specified by the file arguments.

#wc <filename>

Grep

The `grep command` is used to search text or searches the given file for lines containing a match to the given strings or words.

Example:

A command to search root user from file /etc/passwd

#grep root /etc/passwd

Recursive search using grep command

#grep –r “192.168.1.1” /etc/

grep command often used with shell pipes. In this example, show the name of the hard disk devices:

```
# dmesg | egrep '(s|h)d[a-z]'
```

Display cpu model name:

```
# cat /proc/cpuinfo | grep -i 'Model'
```

Display cpu model name:

```
# cat /proc/cpuinfo | grep -i 'Model'

The same command be rewrite

#grep model < /proc/cpuinfo
```

#grep bash /etc/passwd|sort

Example

```
[root@localhost man]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:07:E3:27
          inet addr:192.168.192.100  Bcast:192.168.192.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe07:e327/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:897 errors:0 dropped:0 overruns:0 frame:0
          TX packets:44 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:96288 (94.0 KiB)  TX bytes:6470 (6.3 KiB)
          Interrupt:67 Base address:0x2000

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:16436  Metric:1
          RX packets:1373 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1373 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2323818 (2.2 MiB)  TX bytes:2323818 (2.2 MiB)

[root@localhost man]# ifconfig|grep Bcast
          inet addr:192.168.192.100  Bcast:192.168.192.255  Mask:255.255.255.0
[root@localhost man]# ifconfig|grep Bcast|cut -d":" -f2
192.168.192.100  Bcast
[root@localhost man]# ifconfig|grep Bcast|cut -d":" -f2|cut -d" " -f1
192.168.192.100
[root@localhost man]#
```

Head

It displays first 10 lines of file by default

#head /etc/passwd
#head –n 4 /etc/passwd

Tail

It displays last 10 lines of file by default
#tail /etc/passwd
#tail –n 5 /etc/passwd
#tail –f /var/log/messages
Above command will continue to show updates to the file until Ctrl-C is pressed.

Searching file(examples of '*find*' command)

Find command searches files from Linux machine in real time environment under the criteria or command line arguments.
#find /tmp -perm -002 -exec chmod o-w {} \ ;
#find -name network
#find / -name '*.conf'
#find /home -user sandi -group sandi
#find -user sandi -not -group sandi
#find -user joe -o -user sandi
#find -not \(-user joe -o -user sandi\)
#find / -user sandi -o -uid 500
#find -perm -002
#find -size +10M
#find -size -10M

Practice: Retrieves files via HTTP or FTP

#wget http://www.myfavouritewebsite.com/index.html
#wget --tries=10 --wait=20 ftp://ftp.myftplink.com/files

Practice: Copying file/s over network

#scp <source file> <destination file>
#scp /etc/passwd root@192.168.X.Y:/remote

Practice: rsync (It allows to transfer only the difference between two sets of files.

#rsync <source file> <destination file>

Filesystem

In computing, a **file system** (or **filesystem**) is used to control how information is stored and retrieved. Without a file system, information placed in a storage area would be one large body of information with no way to tell where one piece of information stops and the next begins.

There are many different kinds of file systems. Each one has different structure and logic. Each one has different properties of speed, flexibility, security, size and more. Some file systems have been designed to be used for specific applications. For example the ISO 9660 file system is designed specifically for optical disks.

File systems can be used on many different kinds of storage devices. Each storage device uses a different kind of media. The most common storage device in use today is a hard drive whose media is a disc that has been coated with a magnetic film. The film has ones and zeros 'written' on it sending electrical pulses to a magnetic "read-write" head. Other media that are used are magnetic tape, optical disc, and flash memory. In some cases, the computer's main memory (RAM) is used to create a temporary file system for short term use.

Some file systems are used on local data storage devices; others provide file access via a network protocol (e.g. NFS, SMB, or 9P clients).

File System in Linux

Linux supports many different file systems, but common choices for the system disk on a block device include the ext* family (such as ext2, ext3 and ext4), XFS, JFS,ReiserFS and btrfs.

Adding New File System

Identifying device

The first step in adding a new filesystem to the filesystem tree is to identify the device to be used.

For Example: /dev/hda and /dev/sda

/proc/partitions will list all the block devices and partitions that the system recognizes.
You can then try using "**file -s <device>**" to determine what kind of filesystem is present on the partition, if any.

Another very useful command to list the device
#fdisk –l

Important:

Backup the partition table before any change

#sfdisk -d /dev/sda > /tmp/mypartition.sda

Restore partition table after major mistake

#sfdisk /dev/sda < /tmp/mypartition.sda

Practice: Creating partition, making filesystem, and mounting

fdisk

Making filesystem

Making file system with a label

#mkfs –t ext4 –L my_localdata /dev/sdax

Another command to format partition with ext3 fs

mkfs.ext3

Mounting filesystem with mount command

Make mount point using mkdir

Next mount the filesystem
#mount -o -rw LABEL=mylocaldata /<mountpoint directory>

Make entry in /etc/fstab for permanent mount
LABEL=/mnt/data /mnt/data ext3 defaults 1 2

Using mount -a
Using umount

#blkid ← to see UUID of the device (e.g./dev/sda1)

Use UUID to mount partition in /etc/fstab file

```
UUID=41c22818-fbad-4da6-8196-c816df0b7aa8  /disk2p2      ext3
defaults,errors=remount-ro 0       1
```

After adding hard disk in VMWare Vitual Machine in Linux, to detect the hard disk without getting restart the machine, run the below command:

echo "- - -" > /sys/class/scsi_host/host0/scan

Inodes

The “inode” is sometimes referred to as an index node. It is a file structure on a file system.
It contain metadata of file.
More easily, it is a “database” of all file information **except the file contents and the file name**.

In a file system, inodes consist roughly negligible total disk space, whether it is a whole storage unit (hard disk,thumb drive, etc.) or a partition on a storage unit. **The inode space is used to “track” the files stored on the hard disk**. The inode entries store metadata about each file, directory or object, but only points to these structures rather than storing the data. **Each entry is 128 bytes in size**. The ***metadata contained*** about each structure can include the following:

- Inode number
- Access Control List (ACL)
- Extended attribute
- Direct/indirect disk blocks
- Number of blocks
- File access, change and modification time
- File deletion time
- File generation number

- File size
- File type
- Group
- Number of links
- Owner
- Permissions
- Status flags

NOTE: the metadata does not include the file's name.
to get a listing of an inode number, use the following command:

Code:
```
$ ls –i filename
```
You can use the "stat" command to get more information than the inode number:

Code:
```
$ stat filename
```

Implementing Disk Quotas

Disk space can be restricted by implementing disk quotas which alert a system administrator before a user consumes too much disk space or a partition becomes full.

Disk quotas can be configured for individual users as well as user groups. This makes it possible to manage the space allocated for user-specific files (such as email) separately from the space allocated to the projects a user works on (assuming the projects are given their own groups).

In addition, quotas can be set not just to control the number of disk blocks consumed but to control the number of inodes (data structures that contain information about files in UNIX file systems). Because inodes are used to contain file-related information, this allows control over the number of files that can be created.

The **quota** RPM must be installed to implement disk quotas.

To implement disk quotas, use the following steps:

1. Enable quotas per file system by modifying the **/etc/fstab** file.
2. Remount the file system(s).
3. Create the quota database files and generate the disk usage table.
4. Assign quota policies.

As root, using a text editor, edit the **/etc/fstab** file. Add the **usrquota** and/or **grpquota** options to the file systems that require quotas:

```
/dev/VolGroup00/LogVol00 /          ext3    defaults          1 1
LABEL=/boot              /boot      ext3    defaults          1 2
none                     /dev/pts   devpts  gid=5,mode=620    0 0
none                     /dev/shm   tmpfs   defaults          0 0
none                     /proc      proc    defaults          0 0
none                     /sys       sysfs   defaults          0 0
/dev/VolGroup00/LogVol02 /home      ext3    defaults,usrquota,grpquota  1 2
/dev/VolGroup00/LogVol01 swap       swap    defaults          0 0 . . .
```

In this example, the **/home** file system has both user and group quotas enabled.

Note

The following examples assume that a separate **/home** partition was created during the installation of Red Hat Enterprise Linux. The root (/) partition can be used for setting quota policies in the **/etc/fstab** file.

Remounting the File Systems

After adding the **usrquota** and/or **grpquota** options, remount each file system whose **fstab** entry has been modified. If the file system is not in use by any process, use one of the following methods: Issue the **umount** command followed by the **mount** command to remount the file system.(See the **man** page for both **umount** and **mount** for the specific syntax for mounting and unmounting various filesystem types.)

Issue the **mount -o remount *<file-system>*** command (where ***<file-system>*** is the name of the file system) to remount the file system. For example, to remount the **/home** file system, the command to issue is **mount -o remount /home**.
If the file system is currently in use, the easiest method for remounting the file system is to reboot the system.

Creating the Quota Database Files

After each quota-enabled file system is remounted, the system is capable of working with disk quotas. However, the file system itself is not yet ready to support quotas. The next step is to run the **quotacheck** command.

The **quotacheck** command examines quota-enabled file systems and builds a table of the current disk usage per file system. The table is then used to update the operating system's copy of disk usage. In addition, the file system's disk quota files are updated.

To create the quota files (**aquota.user** and **aquota.group**) on the file system, use the **-c** option of the **quotacheck** command. For example, if user and group quotas are enabled for the **/hom e** file system, create the files in the **/hom e** directory:
quotacheck -cug /home

The **-c** option specifies that the quota files should be created for each file system with quotas enabled,

the **-u** option specifies to check for user quotas, and the **-g** option specifies to check for group quotas. If neither the **-u** or **-g** options are specified, only the user quota file is created. If only **-g** is specified, only the group quota file is created.

After the files are created, run the following command to generate the table of current disk usage per file
system with quotas enabled:

quotacheck -avug

T he options used are as follows:
a — Check all quota-enabled, locally-mounted file systems
v — Display verbose status information as the quota check proceeds
u — Check user disk quota information
g — Check group disk quota information

After **quotacheck** has finished running, the quota files corresponding to the enabled quotas (user and/or group) are populated with data for each quota-enabled locally-mounted file system such as **/hom e**.

Assigning Quotas per User

The last step is assigning the disk quotas with the **edquota** command.
T o configure the quota for a user, as root in a shell prompt, execute the command:
edquo ta *username*

Perform this step for each user who needs a quota. For example, if a quota is enabled in **/etc/fstab** for the **/hom e** partition (**/dev/VolGroup00/LogVol02** in the example below) and the command **edquota testuser** is executed, the following is shown in the editor configured as the default for the system:

```
Disk quotas for user testuser (uid 501):
Filesystem                   blocks     soft     hard     inodes   soft   hard
/dev/VolGroup00/LogVol02  440436        0        0      37418      0      0
```

Note

The text editor defined by the **EDITOR** environment variable is used by **edquota**. To change the editor, set the **EDITOR** environment variable in your **~/.bash_profile** file to the full path of the editor of your choice.

The first column is the name of the file system that has a quota enabled for it. T he second column shows how many blocks the user is currently using. T he next two columns are used to set soft and hard
block limits for the user on the file system. T he **inodes** column shows how many inodes the user is currently using. T he last two columns are used to set the soft and hard inode limits for the user on the file system.

The hard block limit is the absolute maximum amount of disk space that a user or group can use. Once this limit is reached, no further disk space can be used.

The soft block limit defines the maximum amount of disk space that can be used. However, unlike the hard limit, the soft limit can be exceeded for a certain amount of time. T hat time is known as the *grace period*. T he grace period can be expressed in seconds, minutes, hours, days, weeks, or months.

If any of the values are set to 0, that limit is not set. In the text editor, change the desired limits. For example:

```
Disk quotas for user testuser (uid 501):
Filesystem                  blocks     soft     hard   inodes   soft   hard
/dev/VolGroup00/LogVol02  440436   500000   550000    37418      0      0
```

T o verify that the quota for the user has been set, use the command:
quota testuser

Practice to implement quota:

```
#fdisk –l
#mount
#vi /etc/fstab
LABEL=/home          /home          ext4            defaults,usrquota     1  2
#mount
#mount  -o  remount   /home
#mount
#quotacheck   -cuf   /home
#quotaon   /home
#repquota   /home
#edquota  <username>
Or
#setquota  -u  username     400     800     0       0       /home
#su  -  username
$quota
$dd   if=/dev/zero   of=testfile      bs=1k  count=400
$ll
$quota
$dd   if=/dev/zero   of=testfile      bs=1k  count=800
$quota
$exit
#repquota   /home
```

User and Group Administration

The control of users and groups is a core element of RHEL system administration. The user of the system is either a human being or an account used by specific applications.

Every process on the system run as a particular user.
Every file is owned by a particular user. Access of files and directories are restricted by user.

Every user has a numerical identification called UID and similarly each group is linked with a group ID.

Id command is to use see the information about current logged in user.

#id

To view user associated with directory or file, use ls –l <dir> command

#ls –l /tmp

To see process user or process information

#ps au

Option a to view all process with a terminal
Option u to view process associated with the user.

When we create a user, and give password, user's data mainly stored in two files

/etc/passwd

/etc/shadow

Similarly, when we create a group its information stored in file /etc/group and /etc/gshadow.
Explaining user database files:

The /etc/passwd file contains seven fields:

User name, password placeholder, uid number, gid number, of the user's primary group, Comment, home directory, and shell to be run when a user logs in.

```
❶username:❷password:❸UID:❹GID:❺GECOS:❻/home/dir:❼shell
```

The /etc/group file contains:

Group name, group password placeholder, gid number, and a comma separated list of group members.

```
groupname:password:GID:list,of,users,in,this,group
```

The /etc/shadow files referenced when someone logs in and it contains:

A mapping of a user name to a password

Important:

1. System users and groups all have uid and gid number between 1 and 499.

2. To avoid ID conflict, local users have UID's in the range of 500-999, and UIDs for central directory service such as LDAP in the range of 10001 to 30000.

Running commands as root with sudo

While su (switch user) is available to switch the used

The sudo command allows us run a command as root, or as an another user, which depends on setting configured in the /etc/sudoers file.

$ sudo usermod –L username
[sudo] passowd for student: *password*

An additional advantage of using sudo, all commands run using sudo will be logged by default to /var/log/secure

#tail -5 /var/log/massages

Creating new user account:

#useradd <username>

#passwd <username>

Let us analyse /etc/passwd and /etc/shadow files for the new created user.

Modifying User

usermod [option] username

Options

-c	comment
-d	change home directory
-g	change the primary group of user
-G	Give group for the user
-L	Lock the password
-U	Unlock the password

Deleting/Removing user
Manually can be removed user from /etc/passwd, /etc/shadow, /etc/group, /etc/gshadow, /var/spool/mail, etc.
Or
userdel –r username

Shadow password and password policy

In the distant past, encrypted passwords were stored in the world-readable /etc/passwd file. This was thought to be reasonably secure until dictionary attacks on encrypted passwords became common. At that point, the encrypted passwords, or "password hashes," were moved to the more secure /etc/shadow file. This new file also allowed password aging and expiration features to be implemented.

There are three pieces of information stored in a modern password hash:

1gCjLa2/Z$6Pu0EK0AzfCjxjv2hoLOB/

1. **1**: The hashing algorithm. The number 1 indicates an MD5 hash. The number 6 appears when a SHA-512 hash is used.
2. **gCjLa2/Z**: The *salt* used to encrypt the hash. This is originally chosen at random. The salt and the unencrypted password are combined and encrypted to create the encrypted password hash. The use of a salt prevents two users with the same password from having identical entries in the **/etc/shadow** file.
3. **6Pu0EK0AzfCjxjv2hoLOB/**: The encrypted hash.

When a user tries to log in, the system looks up the entry for the user in **/etc/shadow**, combines the salt for the user with the unencrypted password that was typed in, and encrypts them using the hashing algorithm specified. If the result matches the encrypted hash, the user typed in the right password. If the result doesn't match the encrypted hash, the user typed in the wrong password and the login attempt fails. This method allows the system to determine if the user typed in the correct password without storing that password in a form usable for logging in.

Note

Red Hat Enterprise Linux 6 and 7 support two new strong password hashing algorithms, SHA-256 (algorithm **5**) and SHA-512 (algorithm **6**). Both the salt string and the encrypted hash are longer for these algorithms. The default algorithm used for password hashes can be changed by the **root** user by running the command **authconfig --passalgo** with one of the arguments **md5**, **sha256**, or **sha512**, as appropriate.

Red Hat Enterprise Linux 7 defaults to using SHA-512 encryption.

/etc/shadow format

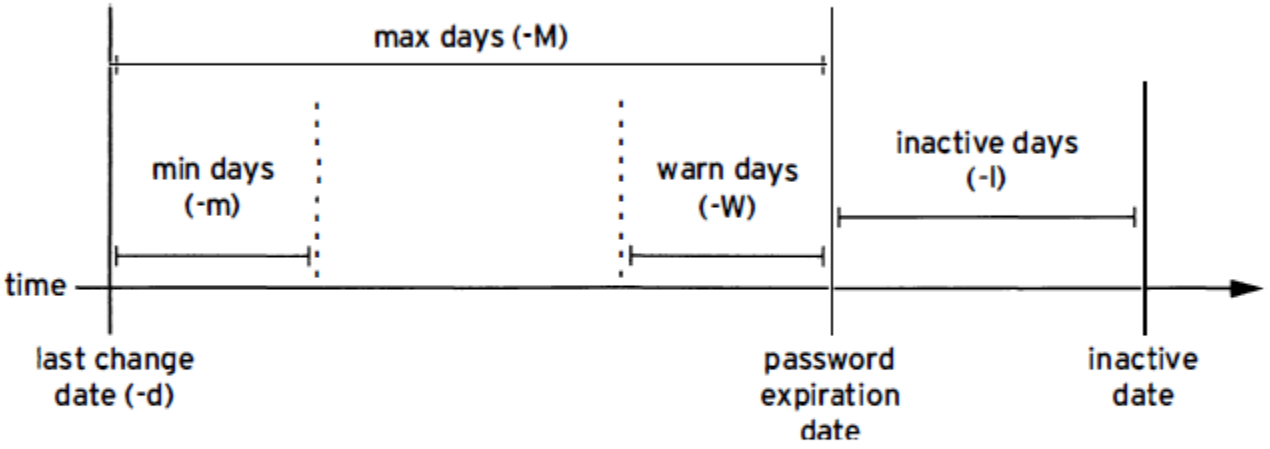
The format of **/etc/shadow** follows (nine colon-separated fields):

❶name:❷password:❸lastchange:❹minage:❺maxage:❻warning:❼inactive:❽expire:❾blank

1. The login *name*. This must be a valid account name on the system.
2. The encrypted *password*. A password field which starts with a exclamation mark means that the password is locked.
3. The date of the last *password change*, represented as the number of days since 1970.01.01.
4. The *minimum* number of days before a password may be changed, where 0 means "no minimum age requirement."
5. The *maximum* number of days before a password must be changed.
6. The *warning* period that a password is about to expire. Represented in days, where 0 means "no warning given."
7. The number of days an account remains active after a password has expired. A user may still log into the system and change the password during this period. After the specified number of days, the account is locked, becoming *inactive*.
8. The account *expiration* date, represented as the number of days since 1970.01.01.
9. This *blank* field is reserved for future use.

Password aging

The following diagram relates the relevant password-aging parameters, which can be adjusted using **chage** to implement a password-aging policy.




```
# chage -m 0 -M 90 -W 7 -I 14 username
```

chage -d 0 *username* will force a password update on next login.

chage -l *username* will list a username's current settings.

chage -E *YYYY-MM-DD* will expire an account on a specific day.

For practice: page No. 154 for book RHEL 1

For Practice Lab: Managing Local Linux User and Group : Performance Checklist, Book 1, page 155

Access Control List (ACL)

Red Hat Enterprise Linux supports *access control lists* (ACLs) for files and directories which allow permissions for specific users outside of the owner to be set.
Grant rwx access to files to multiple users or groups.

```
#vi /etc/fstab
LABEL=/home /home          ext4          default,acl      1        2
:wq
#mount  -o  remount  /home
#mount
#cd  /home
#touch  filefoo
#setfacl        -m      u:<username>:rw      filefoo
#getfacl filefoo
#setfacl        -x      u:<username>  filefoo
```

On directories, default access control lists can be used, if we wanted all contents of a directory to be writable by the user student.

```
#setfacl  -m  d:u:student:rw  /home/share
```

Managing Users and Groups

Controlling Access to Files and Linux File System Permissions

Linux is Unix kind of operating system, it's multi user and can run on various types of hardware. This feature is excellent but vulnerable and open door for malign user to change or remove crucial data.

Security of Linux is segregated into 2 levels.

1. Ownership
2. Permission

Ownership of Linux files

Every file and directory on your Unix/Linux system is assigned 3 types of owner, given below.

User

A user is the owner of the file. By default, the person who created a file becomes its owner. Hence, a user is also sometimes called an owner.

Group

A user- group can contain multiple users. All users belonging to a group will have the same access permissions to the file. Suppose you have a project where a number of people require access to a file. Instead of manually assigning permissions to each user, you could add all users to a group, and assign group permission to file such that only this group members and no one else can read or modify the files.

Other

Any other user who has access to a file. This person has neither created the file, nor he belongs to a usergroup who could own the file. Practically, it means everybody else. Hence, when you set the permission for others, it is also referred as set permissions for the world.

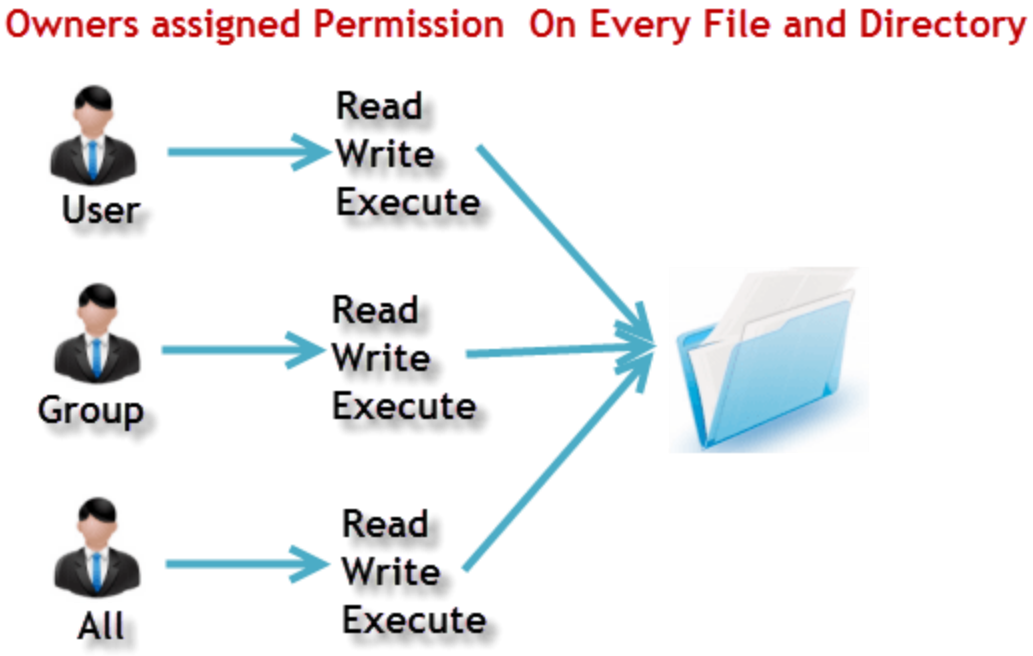
Now, the big question arises how does **Linux distinguish** between these three user types so that a user 'A' cannot affect a file which contains some other user 'B's' vital information/data. It is like you do not want your colleague, who works on your Linux computer, to view your images. This is where **Permissions** set in, and they define **user behavior**.

Let us understand the **Permission system** on Linux.

Permissions

Every file and directory in your UNIX/Linux system has following 3 permissions defined for all the 3 owners discussed above.

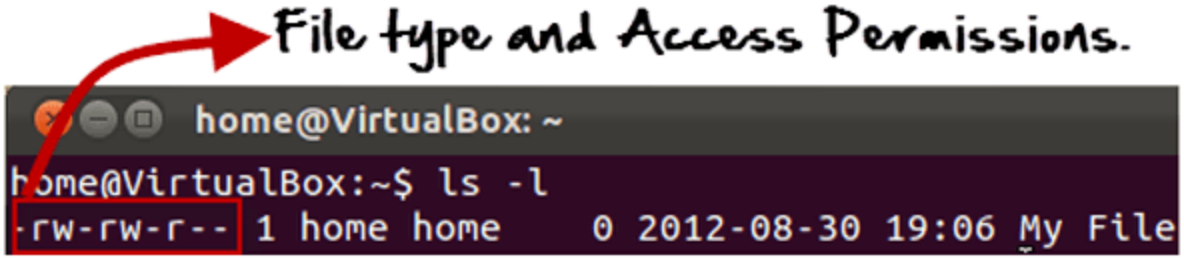
- **Read:** This permission give you the authority to open and read a file. Read permission on a directory gives you the ability to lists its content.
- **Write:** The right permission gives you the authority to modify the contents of a file. The write permission on a directory gives you the authority to add, remove and rename files stored in the directory. Consider a scenario where you have to write permission on file but do not have write permission on the directory where the file is stored. You will be able to modify the file contents. But you will not be able to rename, move or remove the file from the directory.
- **Execute:** In Windows, an executable program usually has an extension ".exe" and which you can easily run. In Unix/Linux, you cannot run a program unless the execute permission is set. If the execute permission is not set, you might still be able to see/modify the program code(provided read & write permissions are set), but not run it.



Let's see this in action

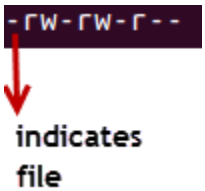
ls - l on terminal gives

```
ls - l
```



Here, we have highlighted **'-rw-rw-r--'**and this weird looking code is the one that tells us about the permissions given to the owner, user group and the world.

Here, the first '**-**' implies that we have selected a file.p>



Else, if it were a directory, **d** would have been shown.

The characters are pretty easy to remember.

r = read permission
w = write permission
x = execute permission
- = no permission

Let us look at it this way.

The first part of the code is **'rw-'**. This suggests that the owner 'Home' can:

- Read the file
- Write or edit the file
- He cannot execute the file since the execute bit is set to '-'.

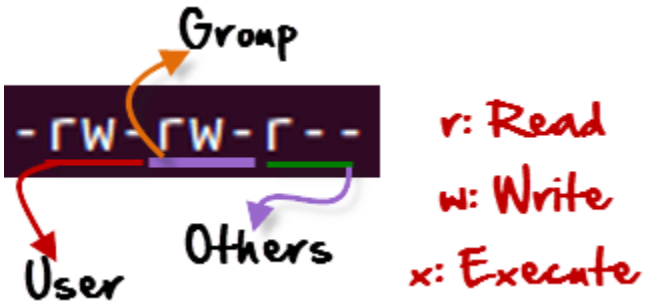
By design, many Linux distributions like Fedora, CentOS, Ubuntu, etc. will add users to a group of the same group name as the user name. Thus, a user 'tom' is added to a group named 'tom'.

The second part is **'rw-'.** It for the user group 'Home' and group-members can:

- Read the file
- Write or edit the file

The third part is for the world which means any user. It says **'r--'.** This means the user can only:

- Read the file



Changing file/directory permissions with 'chmod' command

Say you do not want your colleague to see your personal images. This can be achieved by changing file permissions.

We can use the '**chmod'** command which stands for 'change mode'. Using the command, we can set permissions (read, write, execute) on a file/directory for the owner, group and the world. **Syntax:**

```
chmod permissions filename
```

There are 2 ways to use the command -

1. **Absolute mode**
2. **Symbolic mode**

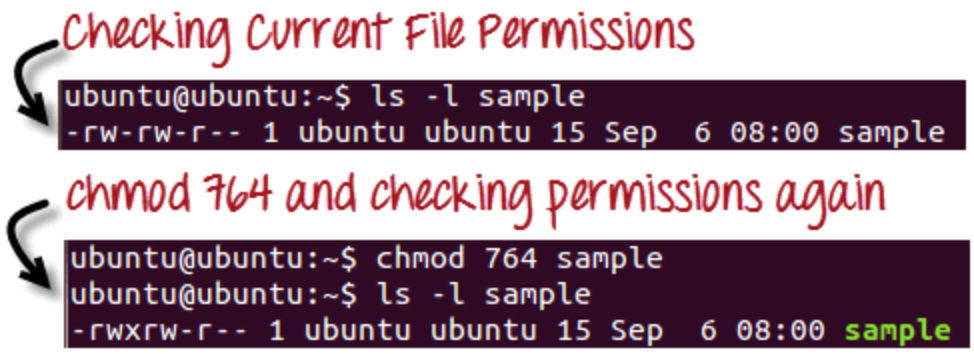
Absolute(Numeric) Mode

In this mode, file **permissions are not represented as characters but a three-digit octal number**.

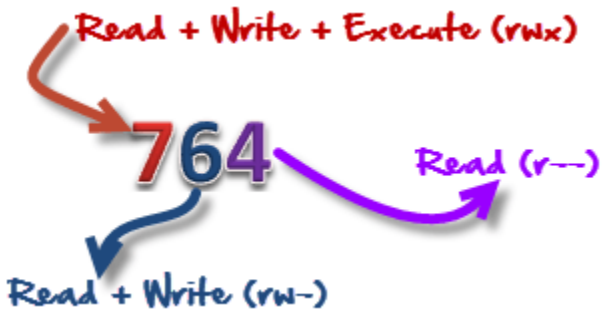
The table below gives numbers for all for permissions types.

Number	Permission Type	Symbol
0	No Permission	---
1	Execute	--x
2	Write	-w-
3	Execute + Write	-wx
4	Read	r--
5	Read + Execute	r-x
6	Read +Write	rw-
7	Read + Write +Execute	rwx

Let's see the chmod command in action.



In the above-given terminal window, we have changed the permissions of the file 'sample to '764'.



'764' absolute code says the following:

- Owner can read, write and execute
- Usergroup can read and write
- World can only read

This is shown as '-rwxrw-r-

This is how you can change the permissions on file by assigning an absolute number.

Symbolic Mode

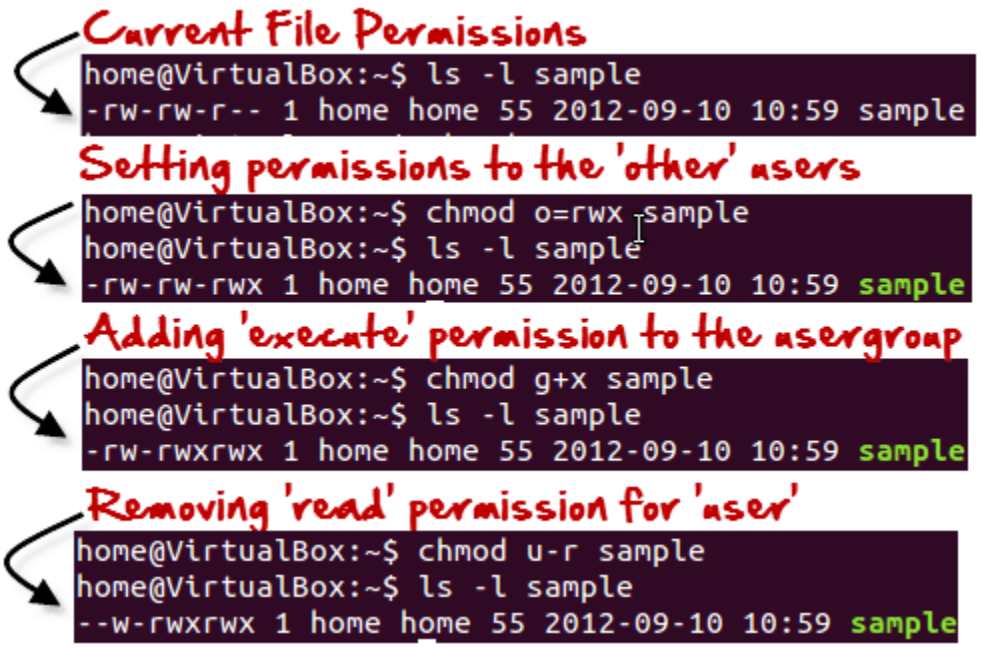
In the Absolute mode, you change permissions for all 3 owners. In the symbolic mode, you can modify permissions of a specific owner. It makes use of mathematical symbols to modify the file permissions.

Operator	Description
+	Adds a permission to a file or directory
-	Removes the permission
=	Sets the permission and overrides the permissions set earlier.

The various owners are represented as -

User Denotations	
U	user/owner
G	group
O	other
A	all

We will not be using permissions in numbers like 755 but characters like rwx. Let's look into an example



Changing Ownership and Group

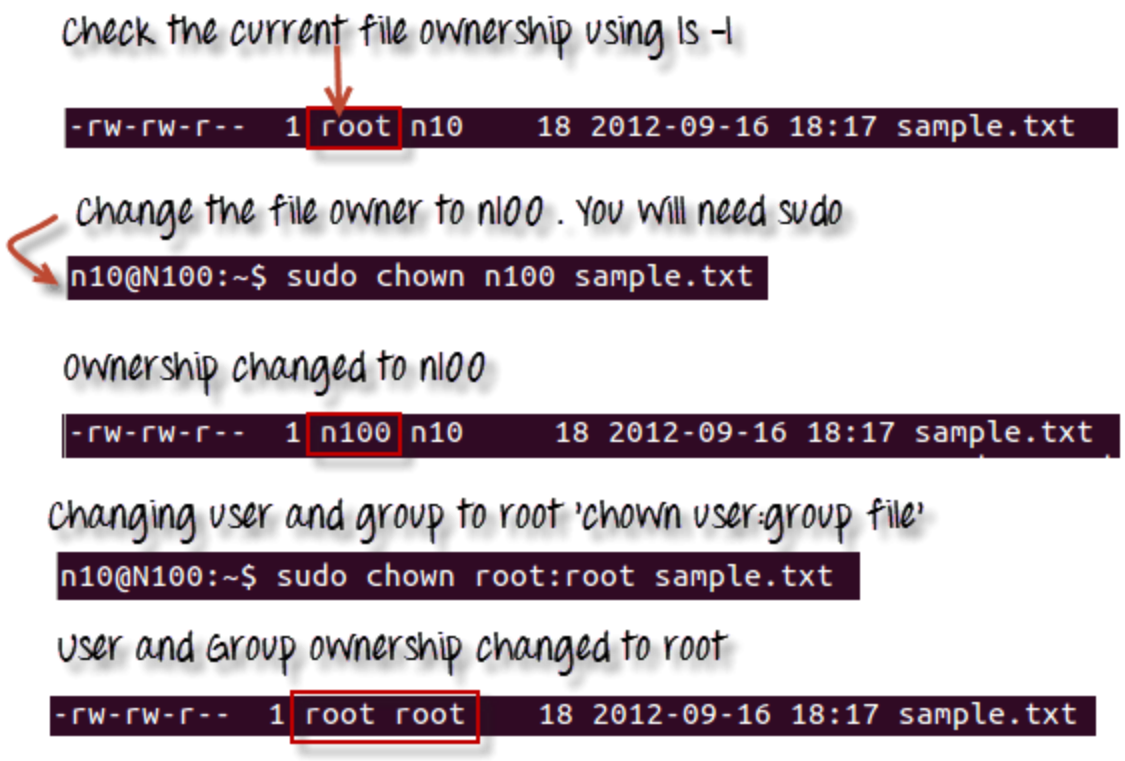
For changing the ownership of a file/directory, you can use the following command:

```
chown user
```

In case you want to change the user as well as group for a file or directory use the command

```
chown user:group filename
```

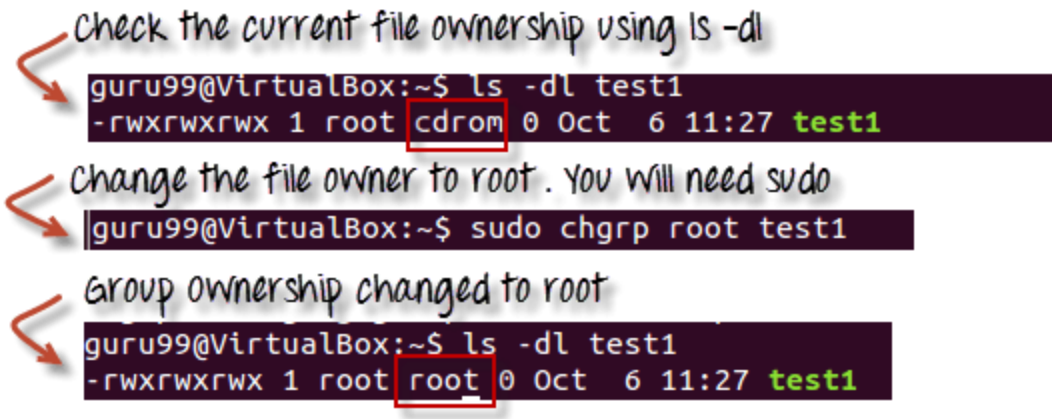
Let's see this in action



In case you want to change group-owner only, use the command

```
chgrp group_name filename
```

'**chgrp'** stands for change group.



Tip

- The file /etc/group contains all the groups defined in the system
- You can use the command "groups" to find all the groups you are a member of

```
guru99@VirtualBox:~$ groups
cdrom guru99 adm sudo dip plugdev lpadmin sambashare
guru99@VirtualBox:~$
```

- You can use the command newgrp to work as a member a group other than your default group

```
guru99@VirtualBox:~$ newgrp cdrom
guru99@VirtualBox:~$ cat > test
this is a test to change group
^C
guru99@VirtualBox:~$ ls -dl test
-rw-rw-r-- 1 guru99 cdrom 31 Oct 11 16:39 test
guru99@VirtualBox:~$
```

- You cannot have 2 groups owning the same file.
- You do not have nested groups in Linux. One group cannot be sub-group of other
- x- eXecuting a directory means Being allowed to "enter" a dir and gain possible access to sub-dirs
- There are other permissions that you can set on Files and Directories which will be covered in a later advanced tutorial

Summary:

- Linux being a multi-user system uses permissions and ownership for security.
- There are three user types on a Linux system viz. User, Group and Other
- Linux divides the file permissions into read, write and execute denoted by r,w, and x
- The permissions on a file can be changed by 'chmod' command which can be further divided into Absolute and Symbolic mode
- The 'chown' command can change the ownership of a file/directory. Use the following commands: chown user file or chown user:group file
- The 'chgrp' command can change the group ownership **chrgrp group filename**
- What does x - eXecuting a directory mean? A: Being allowed to "enter" a dir and gain possible access to sub-dirs.

LAB: Design a LAB to manage file permission and ownership.

Special Permission:

Special permissions on files and directories in linux are : **SetUID**, **SetGID** and **Sticky bit.**With the help of **"chmod"**command we can implement the special permissions on file and directories.

SUID / Set User ID :

A program is executed with the file owner's permissions (rather than with the permissions of the user who executes it).

SGID / Set Group ID :

Files created in the directory inherit its GID, i.e When a directory is shared between the users , and sgid is implemented on that shared directory , when these users creates directory, then the created directory has the same gid or group owner of its parent directory.

Sticky Bit :

It is used mainly used on folders in order to avoid deletion of a folder and its content by other user though he/she is having write permissions. If Sticky bit is enabled on a folder, the folder is deleted by only owner of the folder and super user(root). This is a security measure to suppress deletion of critical folders where it is having full permissions by others.
When we implement these permissions, we get the below symbols in permissions field :

Permissions	Meaning
–S——	SUID is set, but user (owner) execute is not set.
–s——	SUID and user execute are both set.
——S—	SGID is set, but group execute is not set.
——s—	SGID and group execute are both set.
———T	Sticky bit is set, bot other execute is not set.
———t	Sticky bit and other execute are both set.

SUID Example : passwd command
When normal user try to change his/her password , **passwd command is used ,** which is owned by root. This **passwd command** file will try to edit some system config files such as /etc/passwd, /etc/shadow etc. So passwd command is set with SUID to give root user permissions to normal user so that it can update /etc/shadow and other files.
Assign suid to a File :

```
# chmod  u+s testfile.txt
```

OR

```
#  chmod 4750  testfile.txt
```

In this example , 4 indicates SUID bitset, 7 for full permissions for owner, 5 for write and execute permissions for group, and no permissions for others.

SGID Example :

```
# chmod g+s <file/Directory> OR # chmod 2750 <file/Directory>
```

Here in 2750, 2 indicates SGID bitset, 7 for full permissions for owner, 5 for write and execute permissions for group, and no permissions for others.

StickyBit Example :

```
# chmod o+t /opt/ftp-data
```

or

```
# chmod +t /opt/ftp-data
```

or

```
# chmod 1757 /opt/ftp-dta
```

In this example , 1 indicates Sticky Bit set, 7 for full permissions for owner, 5 for read and execute permissions for group, and full permissions for others.

Note : To check the special permissions , use these commands :

```
# ls -l  <file-name>

# ls -ld   <directory/folder-name>
```

What is Sticky Bit?

The sticky bit is used to indicate special permissions for files and directories. If a directory with sticky bit enabled will restrict deletion of the file inside it. It can be removed by root, owner of the file or who have to write permission on it. This is useful for publically accessible directories like /tmp.

Here is the implementation of Sticky bit on file on Linux system.
Method 1:

```
$ chmod +t tecadmin.txt

$ ls -l tecadmin.txt

-rw-r--r-T 1 root root 0 Mar  8 02:06 tecadmin.txt
```

Mothod 2:

```
# chmod 1777 tecadmin.txt

# ls -l tecadmin.txt

-rwxrwxrwt 1 root root 0 Mar  8 02:06 tecadmin.txt
```

In above output it showing sticky bit is set with character t or T in permissions filed. Small t represent that execute permission also enable and capital T represent that execute permission are not enabled.

What is SUID (setuid)?

If SUID bit is set on a file and a user executed it. The process will have the same rights as the owner of the file being executed.

For example: **passwd** command have SUID bit enabled. When a normal user changes his password this script update few system files like /etc/passwd and /etc/shadow which can't be updated by non-root account. So that **passwd** command process always run with root user rights.
Here is the implementation of SUID on file under Linux system.

Mehtod 1:

```
# chmod u+s tecadmin.txt

# ls -l tecadmin.txt

-rwsr-xr-x 1 root root 0 Mar  8 02:06 tecadmin.txt
```

Method 2:

```
# chmod 4655 tecadmin.txt

# ls -l tecadmin.txt

-rwSr-xr-x 1 root root 0 Mar  8 02:06 tecadmin.txt
```

What is SGID (setgid)?

Same as SUID, The process will have the same group rights of the file being executed. If SGID bit is set on any directory, all subdirectories and files created inside will get same group ownership as the main directory, it doesn't matter who is creating.

Here is the implementation of SGID on directory on Linux system.

```
# chmod g+s /test/

# ls -ld /test

drwxrwsrwx 2 root root 4096 Mar  8 03:12 /test
```

Now swich to other user and create a file in /test directory.

```
# su - tecadmin
```

```
$ cd /test/

$ touch tecadmin.net.txt

$ ls -l tecadmin.net.txt

-rw-rw-r-- 1 tecadmin root 0 Mar  8 03:13 tecadmin.net.txt
```

UMASK

When user create a file or directory under Linux or UNIX, she create it with a default set of permissions. In most case the system defaults may be open or relaxed for file sharing purpose. For example, if a text file has 666 permissions, it grants read and write permission to everyone. Similarly a directory with 777 permissions, grants read, write, and execute permission to everyone.

Default umask Value

The user file-creation mode mask (umask) is use to determine the file permission for newly created files. It can be used to control the **default file permission for new files**. It is a four-digit octal number. A umask can be set or expressed using:

- Symbolic values
- Octal values

Procedure To Setup Default umask

You can setup umask in **/etc/bashrc** or **/etc/profile** file for all users. By default most Linux distro set it to 0022 (022) or 0002 (002). Open /etc/profile or ~/.bashrc file, enter:

```
# vi /etc/profile
```

OR

```
$ vi ~/.bashrc
```

Append/modify following line to setup a new umask:

```
umask 022
```

Save and close the file. Changes will take effect after next login. All UNIX users can override the system umask defaults in their /etc/profile file, ~/.profile (Korn / Bourne shell) ~/.cshrc file (C shells), ~/.bash_profile (Bash shell) or ~/.login file (defines the user's environment at login).

Explain Octal umask Mode 022 And 002

As I said earlier, if the default settings are not changed, files are created with the access mode 666 and directories with 777. In this example:

1. The default **umask 002** used for normal user. With this mask default directory permissions are 775 and default file permissions are 664.
2. The default **umask for the root user is 022** result into default directory permissions are 755 and default file permissions are 644.
3. For directories, the **base permissions** are (rwxrwxrwx) 0777 and for files they are 0666 (rw-rw-rw).

In short,

1. A umask of **022** allows only you to write data, but anyone can read data.
2. A umask of **077** is good for a completely private system. No other user can read or write your data if umask is set to 077.
3. A umask of **002** is good when you share data with other users in the same group. Members of your group can create and modify data files; those outside your group can read data file, but cannot modify it. Set your umask to **007** to completely exclude users who are not group members.

But, How Do I Calculate umasks?

The octal umasks are calculated via the bitwise AND of the unary complement of the argument using bitwise NOT. The octal notations are as follows:

- **Octal value** : Permission
- **0** : read, write and execute
- **1** : read and write
- **2** : read and execute
- **3** : read only
- **4** : write and execute
- **5** : write only
- **6** : execute only
- **7** : no permissions

Now, you can use above table to calculate file permission. For example, if umask is set to 077, the permission can be calculated as follows:

Bit	Targeted at	File permission
0	Owner	read, write and execute
7	Group	No permissions
7	Others	No permissions

To set the umask 077 type the following command at shell prompt:

```
$ umask 077
$ mkdir dir1
```

```
$ touch file
$ ls -ld dir1 file
```
Sample outputs:

```
drwx------ 2 vivek vivek 4096 2011-03-04 02:05 dir1

-rw------- 1 vivek vivek    0 2011-03-04 02:05 file
```

TASK: CALCULATING THE FINAL PERMISSION FOR FILES

You can simply subtract the umask from the base permissions to determine the final permission for file as follows:
666 – 022 = 644

- File base permissions : 666
- umask value : 022
- subtract to get permissions of new file (666-022) : 644 (rw-r–r–)

TASK: CALCULATING THE FINAL PERMISSION FOR DIRECTORIES

You can simply subtract the umask from the base permissions to determine the final permission for directory as follows:
777 – 022 = 755

- Directory base permissions : 777
- umask value : 022
- Subtract to get permissions of new directory (777-022) : 755 (rwxr-xr-x)

How Do I Set umask Using Symbolic Values?

The following symbolic values are used:

1. **r** : read
2. **w** : write
3. **x** : execute
4. **u** : User ownership (user who owns the file)
5. **g** : group ownership (the permissions granted to other users who are members of the file's group)
6. **o** : other ownership (the permissions granted to users that are in neither of the two preceding categories)

The following command will set umask to 077 i.e. a umask set to u=rwx,g=,o= will result in new files having the modes -rw——-, and new directories having the modes drwx——:

```
$ umask u=rwx,g=,o=
$ mkdir dir2
$ touch file2
$ ls -ld dir2 file2
```

Sample umask Values and File Creation Permissions

If umask value set to	User permission	Group permission	Others permission
000	all	all	all
007	all	all	none
027	all	read / execute	none

all = read, write and executable file permission

Limitations of the umask

1. The umask command can restricts permissions.
2. The umask command cannot grant extra permissions beyond what is specified by the program that creates the file or directory. If you need to make permission changes to existing file use the chmod command.

umask and level of security

The umask command be used for setting different security levels as follows:

umask value	Security level	Effective permission (directory)
022	Permissive	755
026	Moderate	751
027	Moderate	750

077	Severe	700

For more information about the umask read the man page of bash or ksh or tcsh shell:

```
man bash
```

```
help umask
man chmod
```

Monitoring & Managing Linux Processes

Process: A process is a running instance of an executable program. Linux is a multitasking OS, therefore many processes are running at the same time. And other users on the system are also running programs or using Linux. To observer running processes, 'top' command is popular.

What's running in your system?

Terminal

```
user@bash: top
Tasks: 174 total, 3 running, 171 sleeping, 0 stopped
KiB Mem: 4050604 total, 3114428 used, 936176 free
Kib Swap: 2104476 total, 18132 used, 2086344 free

 PID USER %CPU %MEM COMMAND
6978 ryan 3.0  21.2 firefox
  11 root 0.3   0.0 rcu_preempt
6601 ryan 2.0   2.4 kwin
...
```

Explanation:

- **Line 2** Tasks is just another name for processes. It's typical to have quite a few processes running on your system at any given time. Most of them will be system processes. Many of them will typically be sleeping. This is ok. It just means they are waiting until a particular event occurs, which they will then act upon.
- **Line 3** This is a breakdown of working memory (RAM). Don't worry if a large amount of your memory is used. Linux keeps recently used programs in memory to speed up performance if they are run again. If another process needs that memory, they can easily be cleared to accommodate this.
- **Line 4** This is a breakdown of Virtual memory on your system. If a large amount of this is in use, you may want to consider increasing it's size. For most people with most modern systems having gigabytes of RAM you shouldn't experience any issues here.
- **Lines 6 - 10** Finally is a listing of the most resource intensive processes on the system (in order of resource usage). This list will update in real time and so is interesting to watch to get an idea of what

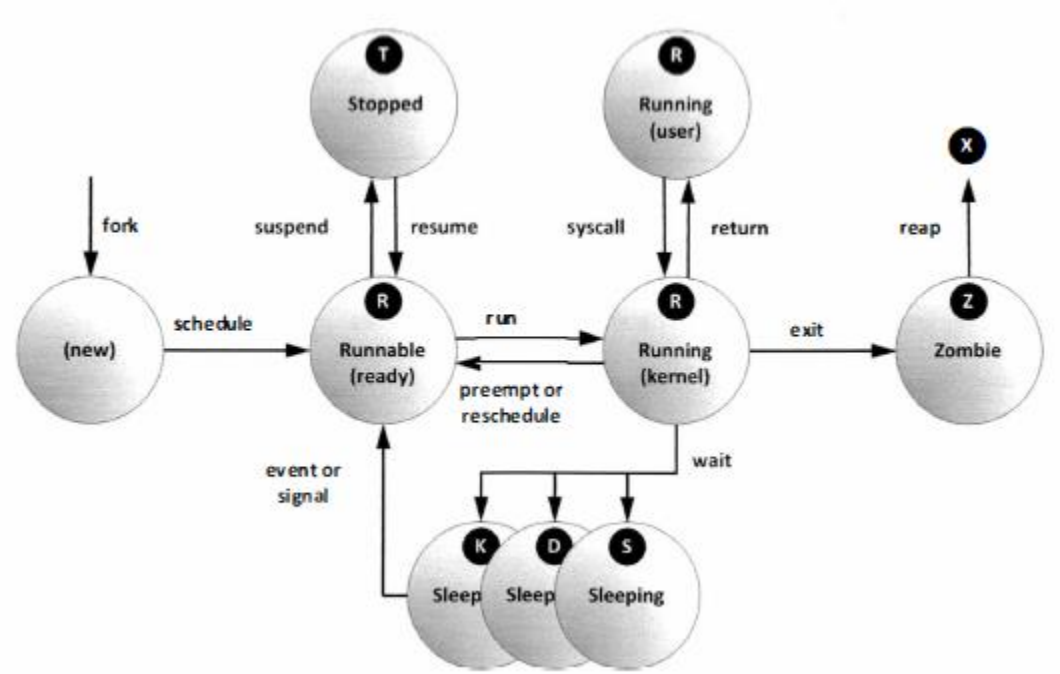
is happening on your system. The two important columns to consider are memory and CPU usage. If either of these is high for a particular process over a period of time, it may be worth looking into why this is so.The USER column shows who owns the process and the PID column identifies a process's Process ID which is a unique identifier for that process.

Process Types:

Foreground Processes:
Background Processes:

As a Linux system administrator you may sometimes want to run process in background to continue working on your command while the background process finishes its work. Linux system allows for a simultaneous process execution and ability to run programs in foreground, background. This tutorial will teach you some basic ins and outs of the foreground and background bash shell feature.

Process State:



Running State:

A **process** moves into the running **state** when it is chosen for execution.

The **process's** instructions are executed by one of the CPUs (or cores) of the system. There is at most one running **process** per CPU or core. A **process** can run in either of the two modes, namely kernel mode or user mode.

Zombie State:

On **Unix** and **Unix**-like computer operating systems, a **zombie process** or defunct **process** is a **process** that has completed execution (via the exit system call) but still has an entry in the **process** table: it is a **process** in the "Terminated state".

ps: command is used for listing current processes.

Details provided by ps:

PID of process

CPU and real time already expanded

Memory allocated

Current process state etc.

#ps –aux or aux

A common display listing displays all process

#ps lax ← long listing

Similar command

#ps –ef ← to display all processes

How to kill the process? Page 197

1. Display all processes

The following command will give a full list of processes

```
$ ps ax

$ ps -ef
```

Pipe the output to "less" to make it scrollable.

Use the "u" option or "-f" option to display detailed information about the processes

```
$ ps aux

$ ps -ef -f
```

2. Display process by user

To filter the processes by the owning user use the "-u" option followed by the username. Multiple usernames can be provided separated by a comma.

```
$ ps -f -u www-data
UID        PID  PPID  C STIME TTY          TIME CMD
www-data  1329  1328  0 09:32 ?        00:00:00 nginx: worker process
www-data  1330  1328  0 09:32 ?        00:00:00 nginx: worker process
www-data  1332  1328  0 09:32 ?        00:00:00 nginx: worker process
www-data  1377  1372  0 09:32 ?        00:00:00 php-fpm: pool a.localhost
www-data  1378  1372  0 09:32 ?        00:00:00 php-fpm: pool a.localhost
www-data  4524  2359  0 10:03 ?        00:00:00 /usr/sbin/apache2 -k start
www-data  4527  2359  0 10:03 ?        00:00:00 /usr/sbin/apache2 -k start
www-data  4528  2359  0 10:03 ?        00:00:00 /usr/sbin/apache2 -k start
```

3. Show process by name or process id

To search the processes by their name or command use the "-C" option followed by the search term.

```
$ ps -C apache2
  PID TTY          TIME CMD
 2359 ?        00:00:00 apache2
 4524 ?        00:00:00 apache2
 4525 ?        00:00:00 apache2
...
```

To display processes by process id, use the "-p" option and provides the process ids separated by comma.

```
$ ps -f  -p 3150,7298,6544
```

The "-C" must be provided with the exact process name and it cannot actually search with a partial name or wildcard. To search the process list more flexibly, the usual grep command has to be used

```
$ ps -ef | grep apache
```

4. Sort process by cpu or memory usage

System administrators often want to find out processes that are consuming lots of memory or CPU. The sort option will sort the process list based on a particular field or parameter.

Multiple fields can be specified with the "--sort" option separated by a comma. Additionally the fields can be prefixed with a "-" or "+" symbol indicating descending or ascending sort respectively. There are lots of parameters on which the process list can be sorted. Check the man page for the complete list.

```
$ ps aux --sort=-pcpu,+pmem
```

Display the top 5 processes consuming most of the cpu.

```
$ ps aux --sort=-pcpu | head -5

USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND

root          1  2.6  0.7  51396  7644 ?        Ss   02:02   0:03
/usr/lib/systemd/systemd --switched-root --system --deserialize 23

root       1249  2.6  3.0 355800 30896 tty1     Rsl+ 02:02   0:02 /usr/bin/X -
background none :0 vt01 -nolisten tcp

root        508  2.4  1.6 248488 16776 ?        Ss   02:02   0:03 /usr/bin/python
/usr/sbin/firewalld --nofork

silver     1525  2.1  2.3 448568 24392 ?        S    02:03   0:01 /usr/bin/python
/usr/share/system-config-printer/applet.py
```

5. Display process hierarchy in a tree style

Many processes are actually forked out of some parent process, and knowing this parent child relationship is often helpful. The '--forest' option will construct an ascii art style tree view of the process hierarchy.

The following command will search for processes by the name apache2 and construct a tree and display detailed information.

```
$ ps -f --forest -C apache2

UID         PID  PPID  C STIME TTY          TIME CMD

root       2359     1  0 09:32 ?        00:00:00 /usr/sbin/apache2 -k start

www-data   4524  2359  0 10:03 ?        00:00:00  \_ /usr/sbin/apache2 -k start

www-data   4525  2359  0 10:03 ?        00:00:00  \_ /usr/sbin/apache2 -k start

www-data   4526  2359  0 10:03 ?        00:00:00  \_ /usr/sbin/apache2 -k start

www-data   4527  2359  0 10:03 ?        00:00:00  \_ /usr/sbin/apache2 -k start

www-data   4528  2359  0 10:03 ?        00:00:00  \_ /usr/sbin/apache2 -k start
```

Try not to use any sorting with the tree style display, as they both effect the order of display in different ways.

6. Display child processes of a parent process

Here is an example of finding all forked apache processes.

```
$ ps -o pid,uname,comm -C apache2

  PID USER     COMMAND

 2359 root     apache2

 4524 www-data apache2

 4525 www-data apache2

 4526 www-data apache2

 4527 www-data apache2

 4528 www-data apache2
```

The first process that is owned by root is the main apache2 process and all other apache2 processes have been forked out of this main process. The next command lists all child apache2 processes using the pid of the main apache2 process

```
$ ps --ppid 2359

  PID TTY          TIME CMD

 4524 ?        00:00:00 apache2

 4525 ?        00:00:00 apache2

 4526 ?        00:00:00 apache2

 4527 ?        00:00:00 apache2

 4528 ?        00:00:00 apache2
```

7. Display threads of a process

The "-L" option will display the threads along with the processes. It can be used to display all threads of a particular process or all processes.

The following command shall display all the threads owned by the process with id 3150.

```
$ ps -p 3150 -L
```

8. Change the columns to display

The ps command can be configured to show a selected list of columns only. There are a large number of columns to to show and the full list is available in the man pages.

The following command shows only the pid, username, cpu, memory and command columns.

```
$ ps -e -o pid,uname,pcpu,pmem,comm
```

It is possible to rename the column labels

```
$ ps -e -o pid,uname=USERNAME,pcpu=CPU_USAGE,pmem,comm
  PID USERNAME CPU_USAGE %MEM COMMAND
    1 root           0.0  0.0 init
    2 root           0.0  0.0 kthreadd
    3 root           0.0  0.0 ksoftirqd/0
    4 root           0.0  0.0 kworker/0:0
    5 root           0.0  0.0 kworker/0:0H
    7 root           0.0  0.0 migration/0
    8 root           0.0  0.0 rcu_bh
    9 root           0.0  0.0 rcuob/0
   10 root           0.0  0.0 rcuob/1
```

Quite flexible.

9. Display elapsed time of processes

The elapsed time indicates, how long the process has been running for. The column for elapsed time is not shown by default, and has to be brought in using the "-o" option

```
$ ps -e -o pid,comm,etime
```

10. Turn ps into an realtime process viewer

As usual, the watch command can be used to turn ps into a realtime process reporter. Simple example is like this

```
$ watch -n 1 'ps -e -o pid,uname,cmd,pmem,pcpu --sort=-pmem,-pcpu | head -15'
```

The output on my desktop is something like this.

```
Every 1.0s: ps -e -o pid,uname,cmd,pmem,pcpu --...  Sun Dec  1 18:16:08 2013
```

```
 PID USER     CMD                         %MEM %CPU
3800 1000     /opt/google/chrome/chrome -  4.6  1.4
7492 1000     /opt/google/chrome/chrome -  2.7  1.4
3150 1000     /opt/google/chrome/chrome    2.7  2.5
3824 1000     /opt/google/chrome/chrome -  2.6  0.6
3936 1000     /opt/google/chrome/chrome -  2.4  1.6
2936 1000     /usr/bin/plasma-desktop      2.3  0.2
9666 1000     /opt/google/chrome/chrome -  2.1  0.8
3842 1000     /opt/google/chrome/chrome -  2.1  0.8
4739 1000     /opt/google/chrome/chrome -  1.8  1.0
3930 1000     /opt/google/chrome/chrome -  1.7  1.0
3911 1000     /opt/google/chrome/chrome -  1.6  0.6
3645 1000     /opt/google/chrome/chrome -  1.5  0.4
3677 1000     /opt/google/chrome/chrome -  1.5  0.4
3639 1000     /opt/google/chrome/chrome -  1.4  0.4
```

The output would be updated every 1 second to refresh the stats. However do not think that this is similar to top.

You would notice that the output of top/htop command changes much more frequently compared to the above ps command.

This is because the top output sorts on a value that is a mix of cpu usage and memory usage. But the above ps command sorts in a more simpler manner, taking 1 column at a time (like school maths). So it would not update rapidly like top.

Monitor Process Activity

Interpreting displayed load average values

The three values represent the weighted values over the last 1, 5, and 15 minutes. A quick glance can indicate whether system load appears to be increasing or decreasing. Calculate the approximate *per-CPU* load value to determine whether the system is experiencing significant waiting.

- **top**, **uptime**, **w**, and **gnome-system-monitor** display load average values.

```
[student@serverX ~]$ uptime
 15:29:03 up 14 min,  2 users,  load average: 2.92, 4.48, 5.20
```

- Divide the displayed load average values by the number of logical CPUs in the system. A value below 1 indicates satisfactory resource utilization and minimal wait times. A value above 1 indicates resource saturation and some amount of service waiting times.

```
# From /proc/cpuinfo, system has four logical CPUs, so divide by 4:
#                              load average: 2.92, 4.48, 5.20
#          divide by number of logical CPUs:    4     4     4
#                                             ----  ----  ----
#                       per-CPU load average: 0.73  1.12  1.30
#
# This system's load average appears to be decreasing.
# With a load average of 2.92 on four CPUs, all CPUs were in use ~73% of the time.
# During the last 5 minutes, the system was overloaded by ~12%.
# During the last 15 minutes, the system was overloaded by ~30%.
```

SSH key Authentication

SSH, which is an acronym for **Secure Shell**, was designed and created to provide the best security when accessing another computer remotely. Not only does it encrypt the session, it also provides better authentication facilities, as well as features like secure file transfer, X session forwarding, port forwarding and more. SSH works on port number 22.

Method:

User$ ssh-keygen –t rsa ← it will create both public and private key

RSA – is asymmetric cryptography algorithm
Public key to enrypt
Private key to decrypt

$ ssh-copy root@<IP address of Remote Server>

$ssh root@<Remote Server IP Address>

JOB Scheduling

at and crond

Setting Up Cron Job Using *crontab*:

Step 1: Open a Terminal Window (Command Line) in Linux.

Step 2: The following is a list of cron directories:

- /etc/cron.hourly
- /etc/cron.daily
- /etc/cron.weekly
- /etc/cron.monthly

Copy your shell script 'script.sh` or 'script' into one of the directories above. If you need to run the script hourly, place your script file in the "cron.hourly" folder. For daily, place it inside the "cron.daily" and so forth.

Step 3: Give the shell script the correct permission. For example, if script is called "script.sh", set permission as follows:

cd /etc/cron.daily/
chmod 755 script.sh

Step 4: Add new cron job to crontab:

crontab –e

This opens vi editor for you. Create the cron command using the following syntax:

1. The number of minutes after the hour (0 to 59)
2. The hour in military time (24 hour) format (0 to 23)
3. The day of the month (1 to 31)
4. The month (1 to 12)
5. The day of the week(0 or 7 is Sun, or use name)
6. The command to run

More graphically they would look like this:

```
*     *     *     *     *  Command to be executed
-     -     -     -     -
|     |     |     |     |
|     |     |     |     +----- Day of week (0-7)
|     |     |     +------- Month (1 - 12)
|     |     +--------- Day of month (1 - 31)
|     +----------- Hour (0 - 23)
+------------- Min (0 - 59)
```

An example command would be **"0 0 * * * /etc/cron.daily/script.sh"**. This would mean that the shell script will exactly execute at midnight every night.

```
To save the changes to the crontab that you just made, hit ESC key, and
then type :w followed by :q to exit.
```

To list existing cron jobs:

crontab –l

To remove an existing cron job:

- **Enter:** crontab –e
- Delete the line that contains your cron job
- Hit ESC > :w > :q

Analyzing and Storing Log

Log directory → /var/log

The **/var/log** directory holds various system- and service-specific log files maintained by **rsyslog**:

Overview of system log files

Log file	Purpose
/var/log/messages	Most syslog messages are logged here. The exceptions are messages related to authentication and email processing, that periodically run jobs, and those which are purely debugging-related.
/var/log/secure	The log file for security and authentication-related messages and errors.
/var/log/maillog	The log file with mail server-related messages.
/var/log/cron	The log file related to periodically executed tasks.
/var/log/boot.log	Messages related to system startup are logged here.

Sample rules section of `rsyslog.conf`

```
#### RULES ####

# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.*                                                 /dev/console

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none;cron.none                /var/log/messages

# The authpriv file has restricted access.
authpriv.*                                              /var/log/secure

# Log all the mail messages in one place.
mail.*                                                  -/var/log/maillog


# Log cron stuff
cron.*                                                  /var/log/cron

# Everybody gets emergency messages
*.emerg                                                 :omusrmsg:*

# Save news errors of level crit and higher in a special file.
uucp,news.crit                                          /var/log/spooler

# Save boot messages also to boot.log
local7.*                                                /var/log/boot.log
```

Log file rotation

Logs are "rotated" by the **`logrotate`** utility to keep them from filling up the file system containing **`/var/log/`**. When a log file is rotated, it is renamed with an extension indicating the date on which it was rotated: the old **`/var/log/messages`** file may become **`/var/log/messages-20141030`** if it is rotated on October 30, 2014. Once the old log file is rotated, a new log file is created and the service that writes to it is notified.

Analyze a syslog entry

The system logs written by **rsyslog** start with the oldest message on top and the newest message at the end of the log file. All log entries in log files managed by **rsyslog** are recorded in a standard format. The following example will explain the anatomy of a log file message in the **/var/log/secure** log file:

```
❶Feb 11 20:11:48 ❷localhost ❸sshd[1433]: ❹Failed password for student from
 172.25.0.10 port 59344 ssh2
```

❶ The time stamp when the log entry was recorded.
❷ The host from which the log message was sent.
❸ The program or process that sent the log message.
❹ The actual message sent.

Monitor a log file with `tail`

It is especially helpful for reproducing problems and issues to monitor one or more log files for events. The **tail -f /path/to/file** command outputs the last 10 lines of the file specified and continues to output new lines as they get written to the monitored file.

To monitor for failed login attempts on one terminal, run **ssh** as user root while a user tries to log in to the serverX machine:

```
[root@serverX ~]$ tail -f /var/log/secure
...
Feb 10 09:01:13 localhost sshd[2712]: Accepted password for root from 172.25.254.254
 port 56801 ssh2
Feb 10 09:01:13 localhost sshd[2712]: pam_unix(sshd:session): session opened for user
 root by (uid=0)
```

Output only **systemd** journal messages that originate from the **systemd** process that always runs with process id 1 on serverX.

```
[root@serverX ~]# journalctl _PID=1
```

Display all **systemd** journal messages that originate from a system service started with user id 81 on serverX.

```
[root@serverX ~]# journalctl _UID=81
```

Output the journal messages with priority **warning** and above on serverX.

```
[root@serverX ~]# journalctl -p warning
```

Create a **journalctl** query to show all log events recorded in the previous 10 minutes on serverX. The command assumes a current time of 9:15:00.

```
[root@serverX ~]# journalctl --since 9:05:00 --until 9:15:00
```

Display only the events originating from the **sshd** service with the system unit file **sshd.service** recorded since 9:00:00 this morning on serverX.

```
[root@serverX ~]# journalctl --since 9:00:00 _SYSTEMD_UNIT="sshd.service"
```

The systemd journal can be made persistent by creating the directory **/var/log/journal** as user root:

```
[root@serverX ~]# mkdir /var/log/journal
```

Ensure that the **/var/log/journal** directory is owned by the root user and group systemd-journal, and has the permissions 2755.

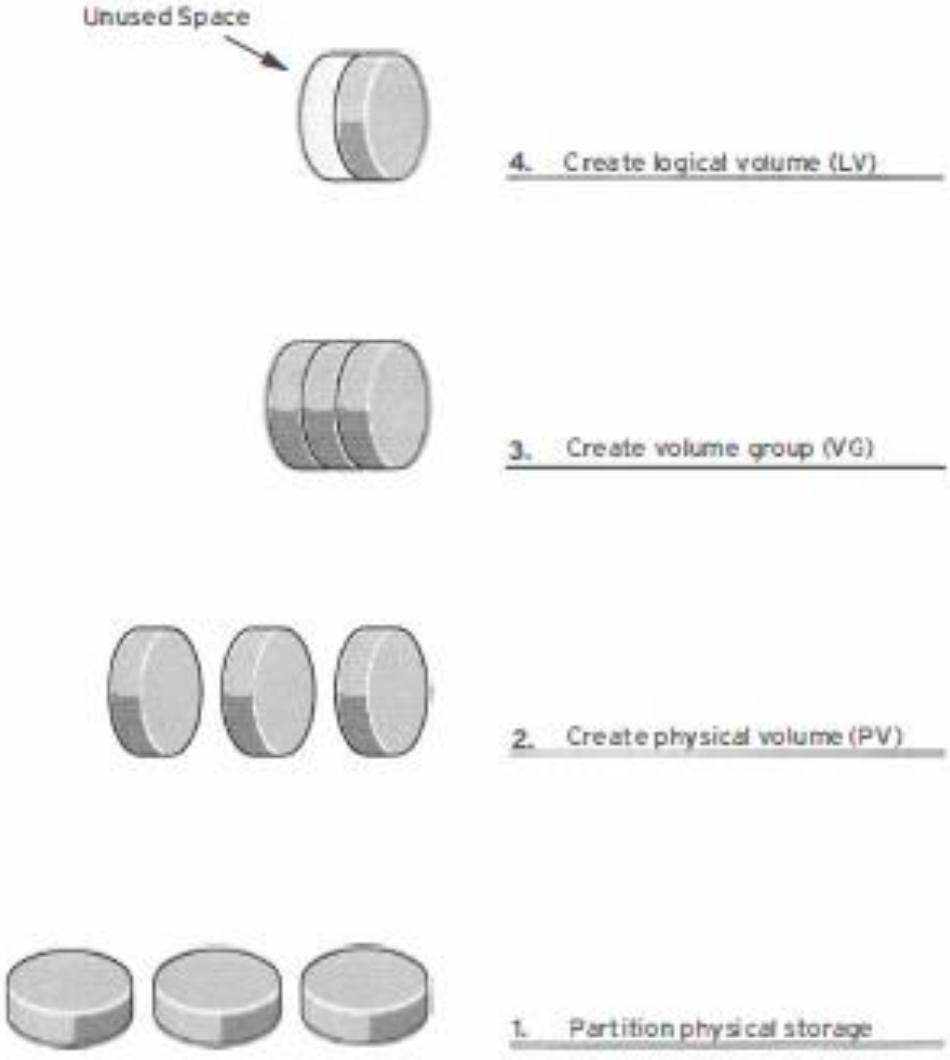
```
[root@serverX ~]# chown root:systemd-journal /var/log/journal
[root@serverX ~]# chmod 2755 /var/log/journal
```

Either a reboot of the system or sending the special signal **USR1** as user root to the **systemd-journald** process is required.

Logical Volume Management (LVM)

Logical volume management (LVM) concepts

Logical volumes and logical volume management make it easier to manage disk space. If a LVM-hosted file system needs more space, it can be allocated to its logical volume from the free space in its volume group and the file system can be resized. If a disk starts to fail, a replacement disk can be registered as a physical volume with the volume group and the logical volume's extents can be migrated to the new disk.



LVM Definitions

- *Physical devices* are the storage devices used to persist data stored in a logical volume. These are block devices and could be disk partitions, whole disks, RAID arrays, or SAN disks. A device must be initialized as an LVM physical volume in order to be used with LVM. The entire "device" will be used as a physical volume.
- *Physical volumes (PV)* are used to register underlying physical devices for use in volume groups. LVM automatically segments PVs into *physical extents (PE)*; these are small chunks of data that act as the smallest storage block on a PV.
- *Volume groups (VG)* are storage pools made up of one or more physical volumes. A PV can only be allocated to a single VG. A VG can consist of unused space and any number of logical volumes.
- *Logical volumes (LV)* are created from free physical extents in a volume group and provide the "storage" device used by applications, users, and the operating system. LVs are a collection of *logical extents (LE)*, which map to physical extents, the smallest storage chunk of a PV. By default, each LE will map to one PE. Setting specific LV options will change this mapping; for example, *mirroring* causes each LE to map to two PEs.

Creating logical volume:

1. **Prepare physical device**
   Add one or more disk with the host machine and identify physical device
   #fdisk –l
   #fdisk /dev/xdva
   Create partition using fdisk

2. **Create a physical volume**
   #pvcreate /dev/sdb1 /dev/sdc1
   #pvdisplay

3. **Create a volume Group**
   #vgcreate <vgname eg vg001> /dev/sdb1 /dev/sdc1
   #vgdisplay

4. **Create a logical volume**
   #lvcreate –n <lvname> -L <Size eg 2G> <vgname vg001>
   #lvdisplay

5. **Add the file system**
   #mkfs –t xfs /dev/vg001/lv001
   #mkdir /mnt/lvvol01

   Make entry in /etc/fstab
   /dev/vg001/lv001 /mnt/lvvol01 xfs defaults 1 2

#mount -a

Prepare the File System

Unmount file system

Remove the logical Volume

#lvremove /dev/vg001/lvvol01

Remove the volume group

#vgremore vg001

Remove the physical volumes

#pvremove /dev/sdb1 /dev/sdc1

Extending and reducing a volume group

1. Check the space of current volume
2. Check the free space with df –h command
3. With fdisk, create an additional partition or add block
4. Set partition type 8e
5. Pvcreate new partition
6. Extend the volume group
   #vgextend <volume name> /dev/sdd1 ← partition name
   #vgdisplay <volume name>
7. Now extend the logical volume
   #lvextend –L <70M or any available space> /dev/vol01/lvvol01 ← LV Name
8. Resize the File System
   #xfs_growfs /storage ← LV name
   #df –h /storage ← LV name to verify the space has been added

NIC Channel Bonding or Link Aggregation (Load Balancing/Fault Tolerance) RHEL 6.X

In order to provide Redundancy/Fault Tolerance Bonding of NICs are used in Linux

#cat /etc/redhat-release

Check directory and ensure we have two NICs available to make bond

eth1 and eth2 must be available

Start bond configuration in ~network-scripts directory

#vim ifcfg-bond0

DEVICE=bond0
IPADDR=192.168.0.30
NETMASK=255.255.255.0
GATEWAY=192.168.0.1
BONDING_OPTS="mode=1 miimom=100"
BOOTPROTO=none
!wq

Important: Bond0 is considered or configured as master and eth1 and eth2 would be configured as slave interfaces, master will send packets to these slaves in every 100 msec to check connectivity or eth1 and eth2

Now configure slave interfaces eth0 and eth1

#vim ifcfg-eth1

DEVICE=eth1
TYPE=Ethernet
NM_CONTROLLED=no
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
!wq

#vim ifcfg-eth2

DEVICE=eth2
TYPE=Ethernet
NM_CONTROLLED=no
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
!wq

#service network restart
#ifconfig

To see the current status of the bond0 and Ethernet just created.

#cat /proc/net/bonding/bond0
#mii-tool

#dmesg |grep eth
#ip link show

Bond Option Modes
balance-rr or 0 : Sets a round-robin policy for fault tolerance and load balancing. Transmissions are received and sent out sequentially on each bonded slave interface beginning with the first one available.

active-backup or 1: Sets an active-backup policy for fault tolerance. Transmissions are received and sent out via the first available bonded slave interface. Another bonded slave interface is only used if the active bonded slave interface fails.

balance-xor or 2: Sets an XOR(exclusive-or) policy for fault tolerance and load balancing. Using this method the interface matches up the incoming request's MAC Address with the MAC Address for one of the slave NICs. Once the link is established, transmissions are sent out sequentially beginning with the first available interface.

What is Swap Space?

Swap space in Linux is used when the amount of physical memory (RAM) is full. If the system needs more memory resources and the RAM is full, inactive pages in memory are moved to the swap space. While swap space can help machines with a small amount of RAM, it should not be considered a replacement for more RAM. Swap space is located on hard drives, which have a slower access time than physical memory.

Swap space can be a dedicated swap partition (recommended), a swap file, or a combination of swap partitions and swap files.

Swap should equal 2x physical RAM for up to 2 GB of physical RAM, and then an additional 1x physical RAM for any amount above 2 GB, but never less than 32 MB.

So, if:

M = Amount of RAM in GB, and S = Amount of swap in GB, then

```
If M <= 2
        S = M *2
Else
```

```
S = M + 2
```

Using this formula, a system with 2 GB of physical RAM would have 4 GB of swap, while one with 3 GB of physical RAM would have 5 GB of swap. Creating a large swap space partition can be especially helpful if you plan to upgrade your RAM at a later time.

For systems with really large amounts of RAM (more than 32 GB) you can likely get away with a smaller swap partition (around 1x, or less, of physical RAM).

5.2. Adding Swap Space

Sometimes it is necessary to add more swap space after installation. For example, you may upgrade the amount of RAM in your system from 128 MB to 256 MB, but there is only 256 MB of swap space. It might be advantageous to increase the amount of swap space to 512 MB if you perform memory-intense operations or run applications that require a large amount of memory.

You have three options: create a new swap partition, create a new swap file, or extend swap on an existing LVM2 logical volume. It is recommended that you extend an existing logical volume.

Adding New Swap Space

You can either use a dedicated hard drive partition to add new swap space, or create a swap file on an existing filesystem and use it as swap space.

How much swap space is currently used by the system?

Free command displays the swap space. free -k shows the output in KB.

```
# free -k
             total       used       free     shared    buffers     cached
Mem:       3082356    2043700    1038656          0      50976    1646268
-/+ buffers/cache:     346456    2735900
Swap:      4192956          0    4192956
```

Swapon command with option -s, displays the current swap space in KB.

```
# swapon -s
Filename                        Type            Size    Used    Priority
/dev/sda2                       partition       4192956 0       -1
```

Swapon -s, is same as the following.

```
# cat /proc/swaps
Filename                        Type            Size    Used    Priority
/dev/sda2                       partition       4192956 0       -1
```

Method 1: Use a Hard Drive Partition for Additional Swap Space

If you have an additional hard disk, (or space available in an existing disk), create a partition using fdisk command. Let us assume that this partition is called /dev/sdc1

Now setup this newly created partition as swap area using the mkswap command as shown below.

```
# mkswap /dev/sdc1
```

Enable the swap partition for usage using swapon command as shown below.

```
# swapon /dev/sdc1
```

To make this swap space partition available even after the reboot, add the following line to the /etc/fstab file.

```
# cat /etc/fstab
/dev/sdc1               swap                    swap    defaults        0 0
```

Verify whether the newly created swap area is available for your use.

```
# swapon -s
Filename                        Type            Size    Used    Priority
/dev/sda2                       partition       4192956 0       -1
/dev/sdc1                       partition       1048568 0       -2

# free -k
```

```
             total       used       free     shared    buffers     cached
Mem:       3082356    3022364      59992          0      52056    2646472
-/+ buffers/cache:     323836    2758520
Swap:      5241524          0    5241524
```

Note: In the output of swapon -s command, the Type column will say “partition” if the swap space is created from a disk partition.

Method 2: Use a File for Additional Swap Space

If you don’t have any additional disks, you can create a file somewhere on your filesystem, and use that file for swap space.

The following dd command example creates a swap file with the name “myswapfile” under /root directory with a size of 1024MB (1GB).

```
# dd if=/dev/zero of=/root/myswapfile bs=1M count=1024
1024+0 records in
1024+0 records out

# ls -l /root/myswapfile
-rw-r--r--    1 root     root     1073741824 Aug 14 23:47 /root/myswapfile
```

Change the permission of the swap file so that only root can access it.

```
# chmod 600 /root/myswapfile
```

Make this file as a swap file using mkswap command.

```
# mkswap /root/myswapfile

Setting up swapspace version 1, size = 1073737 kB
```

Enable the newly created swapfile.

```
# swapon /root/myswapfile
```

To make this swap file available as a swap area even after the reboot, add the following line to the /etc/fstab file.

```
# cat /etc/fstab


/root/myswapfile               swap                    swap    defaults        0 0
```

Verify whether the newly created swap area is available for your use.

```
# swapon -s
Filename                        Type            Size    Used    Priority
/dev/sda2                       partition       4192956 0       -1
/root/myswapfile                file            1048568 0       -2

# free -k
             total       used       free     shared    buffers     cached
Mem:       3082356    3022364      59992          0      52056    2646472
-/+ buffers/cache:     323836    2758520
Swap:      5241524          0    5241524
```

Note: In the output of swapon -s command, the Type column will say “file” if the swap space is created from a swap file.

If you don’t want to reboot to verify whether the system takes all the swap space mentioned in the /etc/fstab, you can do the following, which will disable and enable all the swap partition mentioned in the /etc/fstab

```
# swapoff -a

# swapon -a
```

Configuring Network

Basic Network Configuration
Completed

Network Configuration Utilities

Completed

Network Configuration Files
Completed

HostName
Completed

DNS Configuration (Client Side)
Local name resolution is done via **/etc/hosts** file. If you have small network, use /etc/hosts file. DNS (domain name service is accountable for associating domain names with ip address, for example domain yahoo.com is easy to remember than IP address 202.66.66.12) provides better name resolution.

To configure Linux as DNS client you need to edit or modify **/etc/resolv.conf** file. This file defines which name servers to use. You want to setup Linux to browse net or run network services like www or smtp; then you need to point out to correct ISP DNS servers:

/etc/resolv.conf file

In Linux and Unix like computer operating systems, the /etc/resolv.conf configuration file contains information that allows a computer connected to the Internet to convert alpha-numeric names into the numeric IP addresses that are required for access to external network resources on the Internet. The process of converting domain names to IP addresses is called "resolving."
The resolv.conf file typically contains the IP addresses of nameservers (DNS name resolvers) that attempt to translate names into addresses for any node available on the network.

Setup DNS Name resolution

Steps to configure Linux as DNS client, first login as a root user (use su command):

Step # 1: Open /etc/resolv.conf file:

```
# vi /etc/resolv.conf
```

Step #2: Add your ISP nameserver as follows:

```
search isp.com
nameserver 202.54.1.110
nameserver 202.54.1.112
nameserver 202.54.1.115
```

Note Max. three nameserver can be used/defined at a time.
Step # 3: Test setup nslookup or dig command:

```
$ dig www.example.com
$ nslookup www.example.com
```

Gateway Configuration

The Default Gateway

The default gateway is specified by means of the GATEWAY directive and can be specified either globally or in interface-specific configuration files. Specifying the default gateway globally has certain advantages especially if more than one network interface is present and it can make fault finding simpler if applied consistently. There is also the GATEWAYDEV directive, which is a global option. If multiple devices specify GATEWAY, and one interface uses the GATEWAYDEV directive, that directive will take precedence. This option is not recommend as it can have unexpected consequences if an interface goes down and it can complicate fault finding.
Global default gateway configuration is stored in the **/etc/sysconfig/network** file. This file specifies gateway and host information for all network interfaces.

/etc/sysconfig/network

The **/etc/sysconfig/network** file is used to specify information about the desired network configuration. By default, it contains the following options:

NETWORKING=*boolean*

A Boolean to enable (**yes**) or disable (**no**) the networking. For example:

```
NETWORKING=yes
```

HOSTNAME=*value*

The hostname of the machine. For example:

```
HOSTNAME=server1.example.com
```

GATEWAY=*value*

The IP address of the network's gateway. For example:

```
GATEWAY=192.168.1.1
```

Configuring Routing Table

Red Hat Linux Static Routing Configuration

Q. I've two network interface connected to two different routers as follows:

[a] eth0 LAN network 10.0.0.0/8 - gateway IP - 10.8.2.65

[b] eth1 ISP assigned network 202.54.22.128/28 - gateway IP - 202.54.22.129

I can only ping to public server but not to another servers inside LAN? I'm not able to route traffic via 10.8.2.65. How do I configure static routing under Red Hat Enterprise Linux or CentOS.

Ans:Under Red Hat you need to define static routing using route command. The configuration is stored under /etc/sysconfig/network-scripts/route-eth0 for eth0 interface.

Update route using route command

Type the following command:

```
# route add -net 10.0.0.0 netmask 255.0.0.0 gw 10.8.2.65 eth0
```

```
# route -n
```

Create static routing file

The drawback of abive 'route' command is that, when RHEL reboots it will forget static routes. So store them in configuration file:

```
echo '10.0.0.0/8 via 10.8.2.65' >> /etc/sysconfig/network-scripts/route-eth0
```

Restart networking:

```
# service network restart
```

Network Monitoring Utilities (ip, netstat, nmap, tcpdump)
Nmap: See Presentation (***ppt: nmap-Port Scanner***)

Short for network mapper, nmap is a network exploration tool and security port scanner.

Here are some Nmap usage examples, from the simple and routine to a little more complex and esoteric. Some actual IP addresses and domain names are used to make things more concrete. In their place you should substitute addresses/names from *your own network*. While I don't think port scanning other networks is or should be illegal, some network administrators don't appreciate unsolicited scanning of their networks and may complain. Getting permission first is the best approach.

For testing purposes, you have permission to scan the host scanme.nmap.org. This permission only includes scanning via Nmap and not testing exploits or denial of service attacks. To conserve bandwidth, please do not initiate more than a dozen scans against that host per day. If this free scanning target service is abused, it will be taken down and Nmap will report Failed to resolve given hostname/IP: scanme.nmap.org. These permissions also apply to the hosts scanme2.nmap.org, scanme3.nmap.org, and so on, though those hosts do not currently exist.

nmap -v scanme.nmap.org

This option scans all reserved TCP ports on the machine scanme.nmap.org . The -v option enables verbose mode.

nmap -sS -O scanme.nmap.org/24

Launches a stealth SYN scan against each machine that is up out of the 256 IPs on the class C sized network where Scanme resides. It also tries to determine what operating system is running on each host that is up and running. This requires root privileges because of the SYN scan and OS detection.

nmap -sV -p 22,53,110,143,4564 198.116.0-255.1-127

Launches host enumeration and a TCP scan at the first half of each of the 255 possible eight-bit subnets in the 198.116 class B address space. This tests whether the systems run SSH, DNS, POP3, or IMAP on their standard ports, or anything on port 4564. For any of these ports found open, version detection is used to determine what application is running.

nmap -v -iR 100000 -Pn -p 80

Asks Nmap to choose 100,000 hosts at random and scan them for web servers (port 80). Host enumeration is disabled with -Pn since first sending a couple probes to determine whether a host is up is wasteful when you are only probing one port on each target host anyway.

nmap -Pn -p80 -oX logs/pb-port80scan.xml -oG logs/pb-port80scan.gnmap 216.163.128.20/20

This scans 4096 IPs for any web servers (without pinging them) and saves the output in grepable and XML formats.

Netstat:

Verifying Which Ports Are Listening

After configuring network services, it is important to pay attention to which ports are actually listening on the system's network interfaces. Any open ports can be evidence of an intrusion.

There are two basic approaches for listing the ports that are listening on the network. The less reliable approach is to query the network stack by typing commands such as netstat -an or lsof -i. This method is less reliable since these programs do not connect to the machine from the network, but rather check to see what is running on the system. For this reason, these applications are frequent targets for replacement by attackers. In this way, crackers attempt to cover their tracks if they open unauthorized network ports.

A more reliable way to check which ports are listening on the network is to use a port scanner such as nmap.

The following command issued from the console determines which ports are listening for TCP connections from the network:

nmap -sT -O localhost

The output of this command looks like the following:

Starting nmap V. 3.00 (www.insecure.org/nmap/) Interesting ports on localhost.localdomain (127.0.0.1): (The 1596 ports scanned but not shown below are in state: closed) Port State Service 22/tcp open ssh 111/tcp open sunrpc 515/tcp open printer 834/tcp open unknown 6000/tcp open X11 Remote OS guesses: Linux Kernel 2.4.0 or Gentoo 1.2 Linux 2.4.19 rc1-rc7) Nmap run completed -- 1 IP address (1 host up) scanned in 5 seconds

This output shows the system is running portmap due to the presence of the sunrpc service. However, there is also a mystery service on port 834. To check if the port is associated with the official list of known services, type:

cat /etc/services | grep 834

This command returns no output. This indicates that while the port is in the reserved range (meaning 0 through 1023) and requires root access to open, it is not associated with a known service.

Next, check for information about the port using netstat or lsof. To check for port 834 using netstat, use the following command:

netstat -anp | grep 834

The command returns the following output:

tcp 0 0 0.0.0.0:834 0.0.0.0:* LISTEN 653/ypbind

The presence of the open port in netstat is reassuring because a cracker opening a port surreptitiously on a hacked system would likely not allow it to be revealed through this command. Also, the [p] option reveals the process id (PID) of the service which opened the port. In this case the open port belongs to ypbind (NIS), which is an RPC service handled in conjunction with the portmap service.
The lsof command reveals similar information since it is also capable of linking open ports to services:

lsof -i | grep 834

Below is the relevant portion of the output for this command:

ypbind 653 0 7u IPv4 1319 TCP *:834 (LISTEN) ypbind 655 0 7u IPv4 1319 TCP *:834 (LISTEN) ypbind 656 0 7u IPv4 1319 TCP *:834 (LISTEN) ypbind 657 0 7u IPv4 1319 TCP *:834 (LISTEN)

These tools reveal a great deal about the status of the services running on a machine. These tools are flexible and can provide a wealth of information about network services and configuration. Consulting the man pages for lsof, netstat, nmap, and services is therefore highly recommended.

Tcpdump:

tcpdump command is also called as packet analyzer.

tcpdump command will work on most flavors of unix operating system. tcpdump allows us to save the packets that are captured, so that we can use it for future analysis. The saved file can be viewed by the

same tcpdump command. We can also use open source software like wireshark to read the tcpdump pcap files.

In this tcpdump tutorial, let us discuss some practical examples on how to use the tcpdump command.

1. Capture packets from a particular ethernet interface using tcpdump -i

When you execute tcpdump command without any option, it will capture all the packets flowing through all the interfaces. -i option with tcpdump command, allows you to filter on a particular ethernet interface.

```
$ tcpdump -i eth1

14:59:26.608728 IP xx.domain.netbcp.net.52497 > valh4.lell.net.ssh: . ack 540 win 16554

14:59:26.610602 IP resolver.lell.net.domain > valh4.lell.net.24151:  4278 1/0/0 (73)

14:59:26.611262 IP valh4.lell.net.38527 > resolver.lell.net.domain:  26364+ PTR? 244.207.104.10.in-addr.arpa. (45)
```

In this example, tcpdump captured all the packets flows in the interface eth1 and displays in the standard output.

Note: Editcap utility is used to select or remove specific packets from dump file and translate them into a given format.

2. Capture only N number of packets using tcpdump -c

When you execute tcpdump command it gives packets until you cancel the tcpdump command. Using -c option you can specify the number of packets to capture.

```
$ tcpdump -c 2 -i eth0

listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
```

```
14:38:38.184913 IP valh4.lell.net.ssh > yy.domain.innetbcp.net.11006: P 1457255642:1457255758(116) ack 1561463966 win 63652

14:38:38.690919 IP valh4.lell.net.ssh > yy.domain.innetbcp.net.11006: P 116:232(116) ack 1 win 63652

2 packets captured

13 packets received by filter

0 packets dropped by kernel
```

The above tcpdump command captured only 2 packets from interface eth0.

Note: Mergecap and TShark: Mergecap is a packet dump combining tool, which will combine multiple dumps into a single dump file. Tshark is a powerful tool to capture network packets, which can be used to analyze the network traffic. It comes with wireshark network analyzer distribution.

3. Display Captured Packets in ASCII using tcpdump -A

The following tcpdump syntax prints the packet in ASCII.

```
$ tcpdump -A -i eth0

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes

14:34:50.913995 IP valh4.lell.net.ssh > yy.domain.innetbcp.net.11006: P 1457239478:1457239594(116) ack 1561461262 win 63652

E.....@.@..]..i...9...*.V...]...P....h....E...>{..U=...g.

......G..7\+KA....A...L.
```

```
14:34:51.423640 IP valh4.lell.net.ssh > yy.domain.innetbcp.net.11006: P 116:232(116) ack 1 win 63652

E.....@.@..\..i...9...*.V..*]...P....h....7......X..!....Im.S.g.u:*..O&....^#Ba...

E..(R.@.|.....9...i.*...]...V..*P..OWp.......
```

Note: Ifconfig command is used to configure network interfaces

4. Display Captured Packets in HEX and ASCII using tcpdump -XX

Some users might want to analyse the packets in hex values. tcpdump provides a way to print packets in both ASCII and HEX format.

```
$tcpdump -XX -i eth0

18:52:54.859697 IP zz.domain.innetbcp.net.63897 > valh4.lell.net.ssh: . ack 232 win 16511

        0x0000:  0050 569c 35a3 0019 bb1c 0c00 0800 4500  .PV.5.........E.

        0x0010:  0028 042a 4000 7906 c89c 10b5 aaf6 0f9a  .(.*@.y.........

        0x0020:  69c4 f999 0016 57db 6e08 c712 ea2e 5010  i.....W.n.....P.

        0x0030:  407f c976 0000 0000 0000 0000            @..v........

18:52:54.877713 IP 10.0.0.0 > all-systems.mcast.net: igmp query v3 [max resp time 1s]

        0x0000:  0050 569c 35a3 0000 0000 0000 0800 4600  .PV.5.........F.

        0x0010:  0024 0000 0000 0102 3ad3 0a00 0000 e000  .$......:.......

        0x0020:  0001 9404 0000 1101 ebfe 0000 0000 0300  ................
```

```
0x0030:  0000 0000 0000 0000 0000 0000          ............
```

5. Capture the packets and write into a file using tcpdump -w

tcpdump allows you to save the packets to a file, and later you can use the packet file for further analysis.

```
$ tcpdump -w 08232010.pcap -i eth0

tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes

32 packets captured

32 packets received by filter

0 packets dropped by kernel
```

-w option writes the packets into a given file. The file extension should be **.pcap**, which can be read by any network protocol analyzer.

6. Reading the packets from a saved file using tcpdump -r

You can read the captured pcap file and view the packets for analysis, as shown below.

```
$tcpdump -tttt -r data.pcap

2010-08-22 21:35:26.571793 00:50:56:9c:69:38 (oui Unknown) > Broadcast, ethertype Unknown (0xcafe), length 74:

        0x0000:  0200 000a ffff 0000 ffff 0c00 3c00 0000  ............<...

        0x0010:  0000 0000 0100 0080 3e9e 2900 0000 0000  ........>.).....

        0x0020:  0000 0000 ffff ffff ad00 996b 0600 0050  ...........k...P
```

```
        0x0030:  569c 6938 0000 0000 8e07 0000           V.i8.....

2010-08-22 21:35:26.571797 IP valh4.lell.net.ssh > zz.domain.innetbcp.net.50570: P
800464396:800464448(52) ack 203316566 win 71

2010-08-22 21:35:26.571800 IP valh4.lell.net.ssh > zz.domain.innetbcp.net.50570: P 52:168(116) ack 1
win 71

2010-08-22 21:35:26.584865 IP valh5.lell.net.ssh > 11.154.12.255.netbios-ns: NBT UDP
PACKET(137): QUERY; REQUEST; BROADC
```

7. Capture packets with IP address using tcpdump -n

In all the above examples, it prints packets with the DNS address, but not the ip address. The following example captures the packets and it will display the IP address of the machines involved.

```
$ tcpdump -n -i eth0

15:01:35.170763 IP 10.0.19.121.52497 > 11.154.12.121.ssh: P 105:157(52) ack 18060 win 16549

15:01:35.170776 IP 11.154.12.121.ssh > 10.0.19.121.52497: P 23988:24136(148) ack 157 win 113

15:01:35.170894 IP 11.154.12.121.ssh > 10.0.19.121.52497: P 24136:24380(244) ack 157 win 113
```

8. Capture packets with proper readable timestamp using tcpdump -tttt

```
$ tcpdump -n -tttt -i eth0



2010-08-22 15:10:39.162830 IP 10.0.19.121.52497 > 11.154.12.121.ssh: . ack 49800 win 16390

2010-08-22 15:10:39.162833 IP 10.0.19.121.52497 > 11.154.12.121.ssh: . ack 50288 win 16660
```

```
2010-08-22 15:10:39.162867 IP 10.0.19.121.52497 > 11.154.12.121.ssh: . ack 50584 win 16586
```

9. Read packets longer than N bytes

You can receive only the packets greater than n number of bytes using a filter ‘greater’ through tcpdump command

```
$ tcpdump -w g_1024.pcap greater 1024
```

10. Receive only the packets of a specific protocol type

You can receive the packets based on the protocol type. You can specify one of these protocols — fddi, tr, wlan, ip, ip6, arp, rarp, decnet, tcp and udp. The following example captures only arp packets flowing through the eth0 interface.

```
$ tcpdump -i eth0 arp

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes

19:41:52.809642 arp who-has valh5.lell.net tell valh9.lell.net

19:41:52.863689 arp who-has 11.154.12.1 tell valh6.lell.net

19:41:53.024769 arp who-has 11.154.12.1 tell valh7.lell.net
```

11. Read packets lesser than N bytes

You can receive only the packets lesser than n number of bytes using a filter ‘less’ through tcpdump command

```
$ tcpdump -w l_1024.pcap  less 1024
```

12. Receive packets flows on a particular port using tcpdump port

If you want to know all the packets received by a particular port on a machine, you can use tcpdump command as shown below.

```
$ tcpdump -i eth0 port 22

19:44:44.934459 IP valh4.lell.net.ssh > zz.domain.innetbcp.net.63897: P 18932:19096(164) ack 105 win 71

19:44:44.934533 IP valh4.lell.net.ssh > zz.domain.innetbcp.net.63897: P 19096:19260(164) ack 105 win 71

19:44:44.934612 IP valh4.lell.net.ssh > zz.domain.innetbcp.net.63897: P 19260:19424(164) ack 105 win 71
```

13. Capture packets for particular destination IP and Port

The packets will have source and destination IP and port numbers. Using tcpdump we can apply filters on source or destination IP and port number. The following command captures packets flows in eth0, with a particular destination ip and port number 22.

```
$ tcpdump -w xpackets.pcap -i eth0 dst 10.181.140.216 and port 22
```

14. Capture TCP communication packets between two hosts

If two different process from two different machines are communicating through tcp protocol, we can capture those packets using tcpdump as shown below.

```
$tcpdump -w comm.pcap -i eth0 dst 16.181.170.246 and port 22
```

You can open the file comm.pcap using any network protocol analyzer tool to debug any potential issues.

15. tcpdump Filter Packets – Capture all the packets other than arp and rarp

In tcpdump command, you can give “and”, “or” and “not” condition to filter the packets accordingly.

```
$ tcpdump -i eth0 not arp and not rarp

20:33:15.479278 IP resolver.lell.net.domain > valh4.lell.net.64639:  26929 1/0/0 (73)

20:33:15.479890 IP valh4.lell.net.16053 > resolver.lell.net.domain:  56556+ PTR? 255.107.154.15.in-addr.arpa. (45)

20:33:15.480197 IP valh4.lell.net.ssh > zz.domain.innetbcp.net.63897: P 540:1504(964) ack 1 win 96

20:33:15.487118 IP zz.domain.innetbcp.net.63897 > valh4.lell.net.ssh: . ack 540 win 16486

20:33:15.668599 IP 10.0.0.0 > all-systems.mcast.net: igmp query v3 [max resp time 1s]
```

Creating and Configuring Sub-interfaces

Create sub interfaces on CentOS and Redhat
Submitted by Linux 101 on Sun, 12/12/2010 - 22:33
Sub interfaces or virtual interfaces are used for a number of reasons. Normally for VLANs, but also if you want your machine to have multiple IP addresses.

This is relatively straight forward to do.

It can be done from the command line like this:

```
# ifconfig eth0:1 192.168.111.1
```

The above command has just created a virtual / sub interface on eth0 called eth0:1 and assigned it the IP 192.168.111.1

This however is not a permanent solution because when you reboot, this interface will be lost. To make it permanent we need to create a file in*/etc/sysconfig/network-scripts/* called ifcfg-eth0:1

```
DEVICE=eth0:1
BOOTPROTO=none
HWADDR=00:16:17:90:a5:15
ONBOOT=yes
IPADDR=192.168.111.1
NETMASK=255.255.255.0
```

```
TYPE=Ethernet
```

Very similar to ifcfg-eth0 but note there is no default gateway set. Always remove the gateway line from the cfg file you will inevitably copy to create this.

The MAC or Hardware address must also match the parent interface.

If you need more than one virtual / sub interface, simply create more config files.

To bring an interface up after creating the config file use:

```
# ifup eth0:1
```

Monitoring System Logs

Log files are files that contain messages about the system, including the kernel, services, and applications running on it.

There are different log files for different information. For example, there is a default system log file, a log file just for security messages, and a log file for cron tasks.

Log files can be very useful when trying to troubleshoot a problem with the system such as trying to load a kernel driver or when looking for unauthorized log in attempts to the system. This chapter discusses where to find log files, how to view log files, and what to look for in log files.
Some log files are controlled by a daemon called **syslogd**. A list of log messages maintained by **syslogd**can be found in the **/etc/syslog.conf** configuration file.

Locating Log Files
Most log files are located in the **/var/log/** directory. Some applications such as **httpd** and **samba** have a directory within **/var/log/** for their log files.

You may notice multiple files in the log file directory with numbers after them. These are created when the log files are rotated. Log files are rotated so their file sizes do not become too large. The **logrotate**package contains a cron task that automatically rotates log files according to the **/etc/logrotate.conf** configuration file and the configuration files in the **/etc/logrotate.d/** directory. By default, it is configured to rotate every week and keep four weeks worth of previous log files.

- Kernel boot messages
- Mail system messages
- Kernel auditing messages
- Standard system error messages

Viewing Log Files

Most log files are in plain text format. You can view them with any text editor such as **Vi** or **Emacs**. Some log files are readable by all users on the system; however, root privileges are required to read most log files.

To view system log files in an interactive, real-time application, use the **Log File Viewer**.

Installing the gnome-system-log package

In order to use the **Log File Viewer**, first ensure the *gnome-system-log* package is installed on your system by running, as root:

```
~]# yum install gnome-system-log
```

For more information on installing packages with Yum, refer to Section 6.2.4, "Installing Packages". After you have installed the *gnome-system-log* package, you can open the **Log File Viewer** by clicking on **Applications** → **System Tools** → **Log File Viewer**, or type the following command at a shell prompt:

```
~]$ gnome-system-log
```

The application only displays log files that exist; thus, the list might differ from the one shown in Figure 20.1, "Log File Viewer".

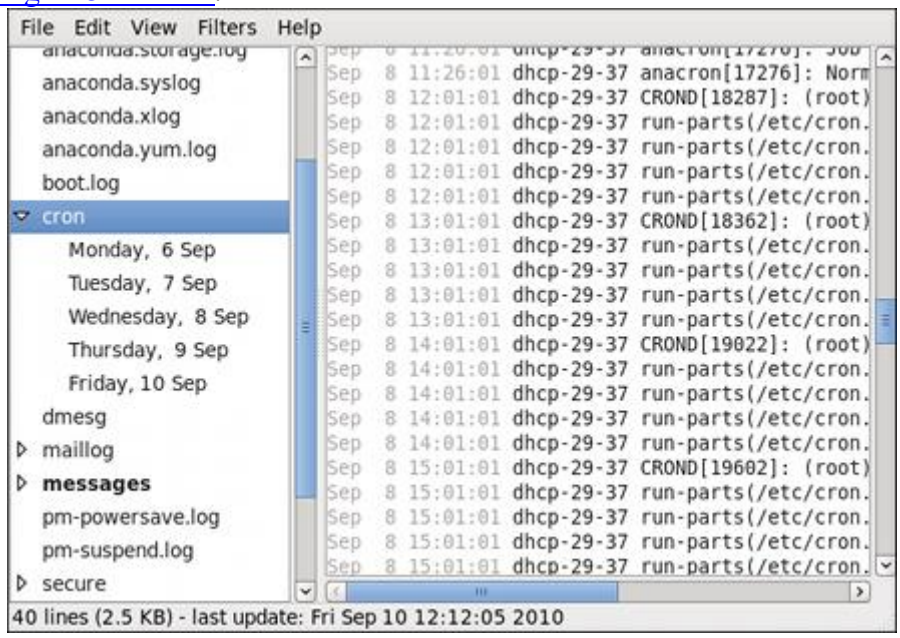


Figure 20.1. Log File Viewer

The **Log File Viewer** application lets you filter any existing log file. Click on **Filters** from the menu and select **Manage Filters** to define or edit your desired filter.

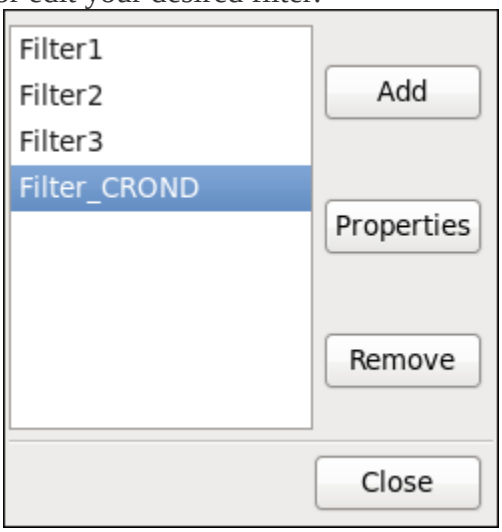


Figure 20.2. Log File Viewer - Filters

Adding or editing a filter lets you define its parameters as is shown in Figure 20.3, "Log File Viewer - defining a filter".

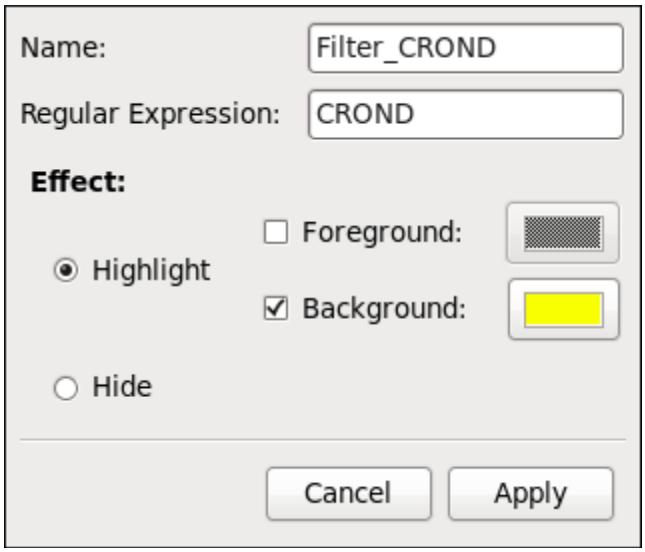


Figure 20.3. Log File Viewer - defining a filter

When defining a filter, you can edit the following parameters:

- **Name** — Specifies the name of the filter.
- **Regular Expression** — Specifies the regular expression that will be applied to the log file and will attempt to match any possible strings of text in it.
- **Effect**
  - **Highlight** — If checked, the found results will be highlighted with the selected color. You may select whether to highlight the background or the foreground of the text.
  - **Hide** — If checked, the found results will be hidden from the log file you are viewing.

When you have at least one filter defined, you may select it from the **Filters** menu and it will automatically search for the strings you have defined in the filter and highlight/hide every successful match in the log file you are currently viewing.

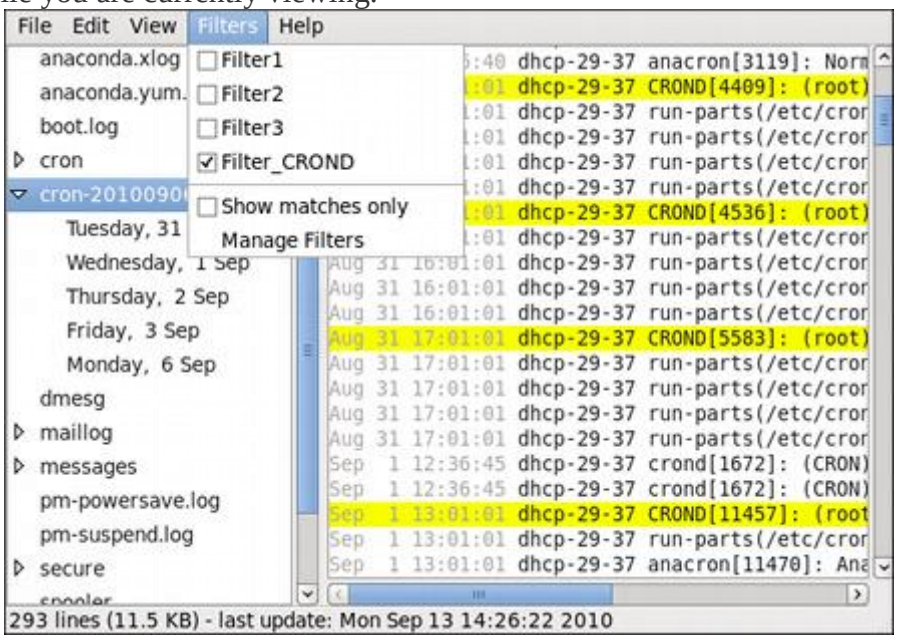


Figure 20.4. Log File Viewer - enabling a filter

When you check the **Show matches only** option, only the matched strings will be shown in the log file you are currently viewing.

5. **Automating Tasks with cron**
   Cron tables
   Cron access control

6. **Security**
   - SELINUX (introduction, modes, queries)
     Notes
   - TCP Wrapper
     Notes
   - iptables (introduction)
     Notes
   - Host Security
     Using scp, rsync and secure ftp for file transfer
     GPG (An introduction)
     GNU Privacy Guard (GPG, also GnuPG), the GNU project's free alternative to PGP, is encryption software that's compliant with the OpenPGP (RFC4880) standard. Using GPG you can encrypt (and decrypt) files that contain sensitive data, such as electronic protected health information (ePHI) regulated by the Health Insurance Portability and Accountability Act (HIPAA) privacy and security rules. For more on GPG, see the GNU Privacy Guard web site.

o encrypt and decrypt files with a password, use gpg command. It is an encryption and signing tool for Linux/UNIX like operating system such as FreeBSD/Solaris and others.

gnupg

GnuPG stands for GNU Privacy Guard and is GNU's tool for secure communication and data storage. It can be used to encrypt data and to create digital signatures. It includes an advanced key management facility.

Encrypting a file in linux

To encrypt a single file, use command gpg as follows:

```
$ gpg -c filename
```

To encrypt myfinancial.info.txt file, type the command:

```
$ gpg -c myfinancial.info.txt
```

Sample output:

```
Enter passphrase:<YOUR-PASSWORD>
Repeat passphrase:<YOUR-PASSWORD>
```

This will create a *myfinancial.info.txt.gpg* file. Where,

- **-c :** Encrypt with symmetric cipher using a passphrase. The default symmetric cipher used is CAST5, but may be chosen with the --cipher-algo option. This option may be combined with --sign (for a signed and symmetrically encrypted message), --encrypt (for a message that may be decrypted via a secret key or a passphrase), or --sign and --encrypt together (for a signed message that may be decrypted via a secret key or a passphrase).

Please note that if you ever forgot your password (passphrase), you cannot recover the data as it use very strong encryption.

Decrypt a file

To decrypt file use the gpg command as follow:

```
$ gpg myfinancial.info.txt.gpg
```

Sample outputs:

```
gpg myfinancial.info.txt.gpg


gpg: CAST5 encrypted data


Enter passphrase:<YOUR-PASSWORD>
```

Decrypt file and write output to file vivek.info.txt you can run command:

```
$ gpg myfinancial.info.gpg –o vivek.info.txt
```

○

Listing Linux host hardening security

Securing your Linux server is important to protect your data, intellectual property, and time, from the hands of crackers (hackers). The system administrator is responsible for security Linux box. In this first part of a Linux server security series, I will provide some hardening tips for default installation of Linux system.

#1: Encrypt Data Communication

All data transmitted over a network is open to monitoring. Encrypt transmitted data whenever possible with password or using keys / certificates.

1. Use scp, ssh, rsync, or sftp for file transfer. You can also mount remote server file system or your own home directory using special sshfs and fuse tools.
2. GnuPG allows to encrypt and sign your data and communication, features a versatile key managment system as well as access modules for all kind of public key directories.
3. Fugu is a graphical frontend to the commandline Secure File Transfer application (SFTP). SFTP is similar to FTP, but unlike FTP, the entire session is encrypted, meaning no passwords are sent in

cleartext form, and is thus much less vulnerable to third-party interception. Another option is FileZilla - a cross-platform client that supports FTP, FTP over SSL/TLS (FTPS), and SSH File Transfer Protocol (SFTP).

4. OpenVPN is a cost-effective, lightweight SSL VPN.
5. Lighttpd SSL (Secure Server Layer) Https Configuration And Installation
6. Apache SSL (Secure Server Layer) Https (mod_ssl) Configuration And Installation

#1.1: Avoid Using FTP, Telnet, And Rlogin / Rsh

Under most network configurations, user names, passwords, FTP / telnet / rsh commands and transferred files can be captured by anyone on the same network using a packet sniffer. The common solution to this problem is to use either OpenSSH , SFTP, or FTPS (FTP over SSL), which adds SSL or TLS encryption to FTP. Type the following command to delete NIS, rsh and other outdated service:

```
# yum erase inetd xinetd ypserv tftp-server telnet-server rsh-serve
```

#2: Minimize Software to Minimize Vulnerability

Do you really need all sort of web services installed? Avoid installing unnecessary software to avoid vulnerabilities in software. Use the RPM package manager such as yum or apt-get and/or dpkg to review all installed set of software packages on a system. Delete all unwanted packages.

```
# yum list installed
```

```
# yum list packageName
```

```
# yum remove packageName
```

#3: One Network Service Per System or VM Instance

Run different network services on separate servers or VM instance. This limits the number of other services that can be compromised. For example, if an attacker able to successfully exploit a software such as Apache flow, he / she will get an access to entire server including other services such as MySQL, e-mail server and so on.

#4: Keep Linux Kernel and Software Up to Date

Applying security patches is an important part of maintaining Linux server. Linux provides all necessary tools to keep your system updated, and also allows for easy upgrades between versions. All security update should be reviewed and applied as soon as possible. Again, use the RPM package manager such as yum.

```
# yum update
```

You can configure Red hat / CentOS / Fedora Linux to send yum package update notification via email.

Another option is to apply all security updates via a cron job. Under Debian / Ubuntu Linux you can use apticron to send security notifications.

#5: Use Linux Security Extensions

Linux comes with various security patches which can be used to guard against misconfigured or compromised programs. If possible use SELinux and other Linux security extensions to enforce limitations on network and other programs. For example, SELinux provides a variety of security policies for Linux kernel.

#5.1: SELinux

I strongly recommend using SELinux which provides a flexible Mandatory Access Control (MAC). Under standard Linux Discretionary Access Control (DAC), an application or process running as a user (UID or SUID) has the user's permissions to objects such as files, sockets, and other processes. Running a MAC kernel protects the system from malicious or flawed applications that can damage or destroy the system. See the official Redhat documentation which explains SELinux configuration.

#6: User Accounts and Strong Password Policy

Use the useradd / usermod commands to create and maintain user accounts. Make sure you have a good and strong password policy. For example, a good password includes at least 8 characters long and mixture of alphabets, number, special character, upper & lower alphabets etc. Most important pick a password you can remember. Use tools such as "John the ripper" to find out weak users passwords on your server. Configure pam_cracklib.so to enforce the password policy.

#6.1: Password Aging

The chage command changes the number of days between password changes and the date of the last password change. This information is used by the system to determine when a user must change his/her password. The /etc/login.defs file defines the site-specific configuration for the shadow password suite including password aging configuration. To disable password aging, enter:

```
chage -M 99999 userName
```

To get password expiration information, enter:

```
chage -l userName
```

Finally, you can also edit the /etc/shadow file in the following fields:

```
{userName}:{password}:{lastpasswdchanged}:{Minimum_days}:{Maximum_days}:{Warn}:{Inactive}:{Expire}:
```

Where,

1. **Minimum_days**: The minimum number of days required between password changes i.e. the number of days left before the user is allowed to change his/her password.
2. **Maximum_days**: The maximum number of days the password is valid (after that user is forced to change his/her password).
3. **Warn** : The number of days before password is to expire that user is warned that his/her password must be changed.
4. **Expire** : Days since Jan 1, 1970 that account is disabled i.e. an absolute date specifying when the login may no longer be used.

I recommend chage command instead of editing the /etc/shadow by hand:

```
# chage -M 60 -m 7 -W 7 userName
```

7. Working with SSH (Secure Shell)

How do I install and configure ssh server and client under CentOS Linux operating systems?

You need to install the following packages (which are installed by default until and unless you removed it or skipped it while installing CentOS)

- openssh-clients : The OpenSSH client applications
- openssh-server : The OpenSSH server daemon

OpenSSH Installations under CentOS Linux

To install the server and client type:

```
# yum -y install openssh-server openssh-clients
```

Start the service:

```
# chkconfig sshd on
```

```
# service sshd start
```

Make sure port 22 is opened:

```
# netstat -tulpn | grep :22
```

Firewall Settings

Edit /etc/sysconfig/iptables (IPv4 firewall),

```
# vi /etc/sysconfig/iptables
```

Add the line

```
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
```

If you want to restict access to 192.168.1.0/24, edit it as follows:

```
-A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p tcp --dport 22 -j ACCEPT
```

If your site uses IPv6, and you are editing ip6tables, use the line:

```
-A RH-Firewall-1-INPUT -m tcp -p tcp --dport 22 -j ACCEPT
```

Save and close the file. Restart iptables:

```
# service iptables restart
```

OpenSSH Server Configuration

Edit /etc/ssh/sshd_config, enter:

```
# vi /etc/ssh/sshd_config
```

To disable root logins, edit or add as follows:

```
PermitRootLogin no
```

Restrict login to user tom and jerry only over ssh:

```
AllowUsers tom jerry
```

Change ssh port i.e. run it on a non-standard port like 1235

```
Port 1235
```

Save and close the file. Restart sshd:

```
# service sshd restart
```

What is Automounting?

An **automounter** is any program or software facility which automatically mounts filesystems in response to access operations by user programs. An automounter system utility (daemon under Unix), when notified of file and directory access attempts under selectively monitored subdirectory trees, dynamically and transparently makes local or remote devices accessible.

The automounter has the purpose of conserving local system resources and of reducing the coupling between systems which share filesystems with a number of servers. For example, a large to mid-sized organization might have hundreds of file servers and thousands of workstations or other nodes accessing files from any number of those servers at any time. Usually, only a relatively small

number of remote filesystems (*exports*) will be active on any given node at any given time. Deferring the mounting of such a filesystem until a process actually needs to access it reduces the need to track such mounts, increasing reliability, flexibility and performance.

Automount Configuration

Installing the RPM packages will get you to this point easily enough, but here's the part you might not be sure about if you haven't done this before.

There are two files in /etc, one called auto.master and one called auto.misc. A sample auto.master looks like this:

```
/auto   /etc/auto.misc  --timeout=60
```

The first entry is not the mount point. It's where the set of mount points (found in the second entry) are going to be. The third option says that the mounted filesystems can try to unmount themselves 60 seconds after use. You will have to stop using the disk before unmounting it.

Auto.misc is a "map file". The map file can have any name; this one is named auto.misc because it originally controlled /misc. Multiple map files can be defined in auto.master. My auto.misc looks like this:

```
Kernel          -ro,soft,intr           ftp.kernel.org:/pub/linux
Cd              -fstype=iso9660,ro              :/dev/cdrom
zip             -fstype=auto                    :/dev/hdd4
floppy          -fstype=vfat                    :/dev/fd0
```

The first column (the "key") is the mount point. In this case it would be /auto/floppy or whatever. The middle set are the options; read the mount manpage for details on this. And the last column specifies where the fs comes from. The "kernel" entry is supposed to be an NFS mount. The : on all the other lines means its a local device.

Hardware and System Clock

CONFIGURING THE DATE AND TIME

Modern operating systems distinguish between the following two types of clocks:

- A *real-time clock* (RTC), commonly referred to as a *hardware clock*, (typically an integrated circuit on the system board) that is completely independent of the current state of the operating system and runs even when the computer is shut down.
- A *system clock*, also known as a *software clock*, that is maintained by the kernel and its initial value is based on the real-time clock. Once the system is booted and the system clock is initialized, the system clock is completely independent of the real-time clock.

The system time is always kept in *Coordinated Universal Time* (UTC) and converted in applications to local time as needed. *Local time* is the actual time in your current time zone, taking into account *daylight saving time* (DST). The real-time clock can use either UTC or local time. UTC is recommended.

Red Hat Enterprise Linux 7 offers three command line tools that can be used to configure and display information about the system date and time: the `timedatectl` utility, which is new in Red Hat Enterprise Linux 7 and is part of `systemd`; the traditional `date` command; and the `hwclock` utility for accessing the hardware clock.

2.1. USING THE `TIMEDATECTL` COMMAND

The **timedatectl** utility is distributed as part of the `systemd` system and service manager and allows you to review and change the configuration of the system clock. You can use this tool to change the current date and time, set the time zone, or enable automatic synchronization of the system clock with a remote server.

For information on how to display the current date and time in a custom format, see also Section 2.2, “Using the date Command”.

2.1.1. Displaying the Current Date and Time

To display the current date and time along with detailed information about the configuration of the system and hardware clock, run the `timedatectl` command with no additional command line options:

```
timedatectl
```

This displays the local and universal time, the currently used time zone, the status of the Network Time Protocol (`NTP`) configuration, and additional information related to DST.

Example 2.1. Displaying the Current Date and Time

The following is an example output of the `timedatectl` command on a system that does not use `NTP`to synchronize the system clock with a remote server:

```
~]$ timedatectl
      Local time: Mon 2013-09-16 19:30:24 CEST
  Universal time: Mon 2013-09-16 17:30:24 UTC
        Timezone: Europe/Prague (CEST, +0200)
     NTP enabled: no
```

```
NTP synchronized: no
 RTC in local TZ: no
      DST active: yes
 Last DST change: DST began at
                  Sun 2013-03-31 01:59:59 CET
                  Sun 2013-03-31 03:00:00 CEST
 Next DST change: DST ends (the clock jumps one hour backwards) at
                  Sun 2013-10-27 02:59:59 CEST
                  Sun 2013-10-27 02:00:00 CET
```

IMPORTANT

Changes to the status of `chrony` or `ntpd` will not be immediately noticed by `timedatectl`. If changes to the configuration or status of these tools is made, enter the following command:

```
~]# systemctl restart systemd-timedated.service
```

2.1.2. Changing the Current Time

To change the current time, type the following at a shell prompt as `root`:

```
timedatectl set-time HH:MM:SS
```

Replace *HH* with an hour, *MM* with a minute, and *SS* with a second, all typed in two-digit form. This command updates both the system time and the hardware clock. The result it is similar to using both the `date --set` and `hwclock --systohc` commands.
The command will fail if an `NTP` service is enabled. See Section 2.1.5, “Synchronizing the System Clock with a Remote Server” to temporally disable the service.

Example 2.2. Changing the Current Time
To change the current time to 11:26 p.m., run the following command as `root`:

```
~]# timedatectl set-time 23:26:00
```

By default, the system is configured to use UTC. To configure your system to maintain the clock in the local time, run the `timedatectl` command with the `set-local-rtc` option as `root`:

```
timedatectl set-local-rtc boolean
```

To configure your system to maintain the clock in the local time, replace *boolean* with `yes` (or, alternatively, `y`, `true`, `t`, or `1`). To configure the system to use UTC, replace *boolean* with `no` (or, alternatively, `n`, `false`, `f`, or `0`). The default option is `no`.

2.1.3. Changing the Current Date

To change the current date, type the following at a shell prompt as `root`:

```
timedatectl set-time YYYY-MM-DD
```

Replace *YYYY* with a four-digit year, *MM* with a two-digit month, and *DD* with a two-digit day of the month.
Note that changing the date without specifying the current time results in setting the time to 00:00:00.

Example 2.3. Changing the Current Date
To change the current date to 2 June 2013 and keep the current time (11:26 p.m.), run the following command as `root`:

```
~]# timedatectl set-time '2013-06-02 23:26:00'
```

2.1.4. Changing the Time Zone

To list all available time zones, type the following at a shell prompt:

```
timedatectl list-timezones
```

To change the currently used time zone, type as `root`:

```
timedatectl set-timezone time_zone
```

Replace *time_zone* with any of the values listed by the `timedatectl list-timezones` command.

Example 2.4. Changing the Time Zone
To identify which time zone is closest to your present location, use the `timedatectl` command with the `list-timezones` command line option. For example, to list all available time zones in Europe, type:

```
~]# timedatectl list-timezones | grep Europe
Europe/Amsterdam
Europe/Andorra
Europe/Athens
Europe/Belgrade
Europe/Berlin
Europe/Bratislava
…
```

To change the time zone to `Europe/Prague`, type as `root`:

```
~]# timedatectl set-timezone Europe/Prague
```

2.1.5. Synchronizing the System Clock with a Remote Server

As opposed to the manual adjustments described in the previous sections, the `timedatectl`command also allows you to enable automatic synchronization of your system clock with a group of remote servers using the `NTP` protocol. Enabling NTP enables the `chronyd` or `ntpd` service, depending on which of them is installed.
The `NTP` service can be enabled and disabled using a command as follows:

```
timedatectl set-ntp boolean
```

To enable your system to synchronize the system clock with a remote `NTP` server, replace *boolean* with `yes` (the default option). To disable this feature, replace *boolean* with `no`.

Example 2.5. Synchronizing the System Clock with a Remote Server
To enable automatic synchronization of the system clock with a remote server, type:

```
~]# timedatectl set-ntp yes
```

Time (Hardware and System)

RHEL 7: https://www.tecmint.com/set-time-timezone-and-synchronize-time-using-timedatectl-command/

RHEL 6:

Fixing Date, Time and Zone on RHEL 6 command line

Had to fix all time related issues on a remote RHEL 6 server which runs without any windowing system. Plain ol' command line. Documenting steps here for future reference:

Check to see if your date and timezone settings are accurate:

```
# date
# cat /etc/sysconfig/clock
```

The server I accessed had wrong settings for both the commands. Here are the steps I used to correct:

Find out your timezone from the folder /usr/share/zoneinfo

```
# ls /usr/share/zoneinfo
```

Mine was pointing to America/EDT instead of ***Asia/Calcutta***

Update and save the /etc/sysconfig/clock file to

```
# sudo vi /etc/sysconfig/clock
ZONE="Asia/Calcutta"
UTC=true
ARC=false
```

Remove the /etc/localtime

```
# sudo rm /etc/localtime
```

Create a new soft link to your time zone

```
# cd /etc
# sudo ln -s /usr/share/zoneinfo/Asia/Calcutta /etc/localtime
# ls -al localtime
```

Now it should show the link to your time zone

Set your hardware clock to UTC

```
# sudo hwclock --systohc --utc
# hwclock --show
```

Update your time from a NTP server (Red Hat NTP server used here)

```
# sudo ntpdate clock.redhat.com
10 Apr 22:08:27 ntpdate[25695]: adjust time server 66.187.233.4 offset 0.004185 sec
```

Finally verify the date now

```
#date --utc
Tue Apr 10 16:39:56 UTC 2012

#date
Tue Apr 10 22:09:58 IST 2012
```

Add the above ntpdate command in your server startup script or in a cron job to automatically set the system time. The list of ntp servers can be found here: https://support.ntp.org/bin/view/Servers/StratumOneTimeServers

Console

The **Linux console** is a system console internal to the Linux kernel (a system console is the device which receives all kernel messages and warnings and which allows logins in single user mode).[1] The Linux console provides a way for the kernel and other processes to send text output to the user, and to receive text input from the user. The user typically enters text with a computer keyboard and reads the output text on a computer monitor. The Linux kernel supports virtual consoles - consoles that are logically separate, but which access the same physical keyboard and display.[2] The Linux console (and Linux virtual consoles) are implemented by the VT subsystem of the Linux kernel, and do not rely on any user spacesoftware.[3] This is in contrast to a terminal emulator, which is a user space process that emulates a terminal, and is typically used in a graphical display environment.

• Virtual Terminals 6 Serial Ports

To access virtual console following key combination is used

Key combination	Console Terminal
Alt +Ctrl+F1	First console (Graphic mode)
Alt +Ctrl+F2	Second console (Text only)
Alt +Ctrl+F3	Third console (Text only)
Alt +Ctrl+F4	Fourth console (Text only)
Alt +Ctrl+F5	Fifth console (Text only)
Alt +Ctrl+F6	Sixth console (Text only)

Key points

- Graphic mode will be available only if we have installed desktop packages (gnome is default).
- Each virtual console represent a separate terminal and provides an independent login session.
- We can switch between virtual consoles. Switching will not terminate the active login se

• SCSI Devices

- Is it possible to add or remove a SCSI device without rebooting a running system?
- Can you scan a SCSI bus for new or missing SCSI devices without rebooting?
- How can I make newly connected SCSI storage devices available without rebooting?
- What is the Linux equivalent to the Solaris command `devfsadm` to add or remove storage devices?
- I am trying to add a LUN to a live system but it is not recognized
- I am trying to add a tape drive to a live system but it is not recognized
- I am trying to add a disk drive to a live system but it is not recognized
- How can I force a rescan of my SAN to find newly associated LUNs?
- What to do if a newly allocated LUN on my SAN is not available?
- Unable to probe for a newly allocated LUN
- Some nodes can't see my new storage device, how can I make it available?
- After SAN maintenance activity, not all devices returned - devices in multipath missing or remain in failed state.
- After SAN failover testing completed, not all devices returned to running state as expected - devices in multipath missing or remain in failed state.
- What is the best way to remove a SCSI disk from the system

How to Rescan SCSI Bus to Add or Remove a SCSI Devices on Linux

1. Identify host bus number :

```
# ls /sys/class/scsi_host/


host0  host1  host2
```

2. Rescan the SCSI Bus to Add a SCSI Devices :

```
# echo "- - -" > /sys/class/scsi_host/host0/scan


# echo "- - -" > /sys/class/scsi_host/host1/scan
```

```
# echo "- - -" > /sys/class/scsi_host/host2/scan
```

• USB Configuration

Detecting USB hard drive

After you plug in your USB device to your USB port, Linux system adds a new block device into `/dev/` directory. At this stage, you are not able to use this device as the USB filesystem needs to be mouted before you can retrieve or store any data. To find out what name your block device file have you can run `fdisk -l` command.

NOTE: `fdisk` command required administrative privileges to access the required information, thus from this reason the commands needs to be executed as a root user or with `sudo` prefix:

```
# fdisk -l

OR

$ sudo fdisk -l
```

Upon executing the above command you will get an output similar to the one below:

```
Disk /dev/sdc: 7.4 GiB, 7948206080 bytes, 15523840 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

Disklabel type: dos

Disk identifier: 0x00000000



Device     Boot Start      End  Sectors  Size Id Type

/dev/sdc1  *     8192 15523839 15515648  7.4G  b W95 FAT32
```

The above output will most likely list multiple disks attached to your system. Look for your USB drive based on its size and filesystem. Once ready, take a note of the block device name of the partition you intent to mount. For example in our case that will be `/dev/sdc1` with FAT32 filesystem.

Create mount point

Before we are able to use `mount` command to mount the USB partition, we need to create a mount point. Mount point can be any new or existing directory within your host filesystem. Use `mkdir` command to create a new mount point directory where you want to mount your USB device:

```
# mkdir /media/usb-drive
```

Mount USB Drive

At this stage we are ready to mount our USB's partition `/dev/sdc1` into `/media/usb-drive` mount point:

```
# mount /dev/sdc1 /media/usb-drive/
```

To check whether your USB drive has been mounted correctly execute `mount` command again without any arguments and use `grep` to search for USB block device name:

```
# mount | grep sdc1


/dev/sdc1 on /media/usb-drive type vfat
(rw,relatime,fmask=0022,dmask=0022,codepage=437,iocharset=utf8,shortname=mixed,erro
rs=remount-ro
```

If no output has been produced by the above `mount` command your USB partition is not mounted. Alternatively, double-check whether you have used a correct block device name in the above command.

Accessing USB Data

If all went well, we can access our USB data simply by navigating to our previously created mount point `/media/usb-drive`:

```
# cd /media/usb-drive
```

USB Unmount

Before we are able to unmount our USB partition we need to make sure that no process is using or accessing our mount point directory, otherwise we will receive an error message similar to the one below:

```
umount: /media/usb-drive: target is busy


(In some cases useful info about processes that


use the device is found by lsof(8) or fuser(1).)
```

Close your shell or navigate away from USB mount point and execute the following command to unmount your USB drive:

```
# umount /media/usb-drive
```

Permanent Mount

In order to mount your USB drive permanently after reboot add the following line into your `/etc/fstab` config file:

```
/dev/sdc1        /media/usb-drive            vfat     defaults         0        0
```

However, the above mount line may fail if you add or remove additional drives from your Linux system. From this reason it is recommend to use partition `UUID` instead of a raw block device name. To do so, first locate a UUID of your USB drive:

```
# ls -l /dev/disk/by-uuid/*


lrwxrwxrwx 1 root root 10 Mar 27 23:38 /dev/disk/by-uuid/2016-08-30-11-31-31-00 ->
../../sdb1


lrwxrwxrwx 1 root root 10 Mar 27 23:38 /dev/disk/by-uuid/3eccfd4e-bd8b-4b5f-9fd8-
4414a32ac289 -> ../../sda1


lrwxrwxrwx 1 root root 10 Mar 27 23:38 /dev/disk/by-uuid/4082248b-809d-4e63-93d2-
56b5f13c875f -> ../../sda5


lrwxrwxrwx 1 root root 10 Mar 28 01:09 /dev/disk/by-uuid/8765-4321 -> ../../sdc1


lrwxrwxrwx 1 root root 10 Mar 27 23:38 /dev/disk/by-uuid/E6E3-F2A2 -> ../../sdb2
```

Based on the above `ls` command output we can see that the UUID belonging to block device `sdc1` is `8765-4321` thus our `/etc/fstab` mount line will be:

```
/dev/disk/by-uuid/8765-4321     /media/usb-drive          vfat    0    0
```

Run `mount -a` command to mount all not yet mounted devices.

```
# mount -a
```

- Defining a Printer

#System-config-printer

Q.How to see installed printer

lpstat -p

• Managing Linux Device Files

What is Udev?

Udev is the device manager for the Linux 2.6 kernel that creates/removes device nodes in the /dev directory dynamically. It is the successor of devfs and hotplug. It runs in userspace and the user can change device names using Udev rules.

Udev depends on the sysfs file system which was introduced in the 2.5 kernel. It is sysfs which makes devices visible in user space. When a device is added or removed, kernel events are produced which will notify Udev in user space.

The external binary /sbin/hotplug was used in earlier releases to inform Udev about device state change. That has been replaced and Udev can now directly listen to those events through Netlink.

• Kernel Hardware Info – /sys/

Uname

Lshw

Lsmod

Lscpu

Lsblk

Lsusb

Fdisk : to know about file system partitions.

Dmidecode : extract hardware information by reading data from the DMI tables.

Lspci: PCI devices may included usb ports, graphics cards, network adapters etc. The lspci tool is used to generate information concerning all PCI controllers on your system plus the devices that are connected to them.

Lsscsi : scsi device information

Lslogins : display information about known users in the system

• /sys/ Structure

• udev

• Kernel Modules

#which <command name> → to find the location of command file in directory

```
[root@localhost ~]# which fdisk
/sbin/fdisk
[root@localhost ~]# rpm -qf /sbin/fdisk
util-linux-2.23.2-21.el7.x86_64
[root@localhost ~]#
```

• System Tools

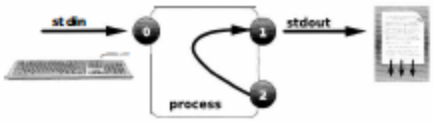
Partition and Filesystem Tools

System Monitoring Tools

Processes

The **sosreport** command is a tool that collects configuration details, system information and diagnostic information from a Red Hat Enterprise **Linux** system. For instance: the running kernel version, loaded modules, and system and service configuration files.

Output Redirection Operators

Usage	Explanation (note)	Visual aid
`>file`	redirect **stdout** to a file [1]	
`>>file`	redirect **stdout** to a file, append to current file content [2]	
`2>file`	redirect **stderr** to a file [1]	
`2>/dev/null`	discard **stderr** error messages by redirecting to **/dev/null**	
`&>file`	combine **stdout** and **stderr** to one file [1]	
`>>file 2>&1`	combine **stdout** and **stderr**, append to current file content [2] [3]	

BOOT PROCESS AND SYSV INIT

- GRUB Configuration 3 Boot Parameters
- /sbin/init
- /etc/inittab
- /etc/rc.d/rc.sysinit
- Runlevel Implementation
- System Configuration Files
- RHEL6 Configuration Utilities
- Typical SysV Init Script@The /etc/rc.local File
- Managing Daemons

- Controlling Service Startup
- Shutdown and Reboot
- Run Level and Kernel Information

FILESYSTEM ADMINISTRATION

- Partitioning Disks with fdisk
- Partitioning Disks with parted
- Filesystem Creation
- Mounting Filesystems
- Filesystem Maintenance
- Resizing Filesystems
- Swap
- Configuring Disk Quotas
- Setting Quotas
- Viewing and Monitoring Quotas
- Filesystem Attributes
- Backup Software
- Backup Examples
- Filesystem Creation and Management

LVM & RAID

- Logical Volume Management
- Implementing LVM
- Creating Logical Volumes
- Manipulating VGs & LVs
- Advanced LVM Concepts
- system-config-lvm
- RAID Concepts

- Array Creation with mdadm
- Software RAID Monitoring
- Software RAID Control and Display
- LVM and RAID: Unix Tool Comparison

USER/GROUP ADMINISTRATION

- User and Group Concepts
- User Administration
- Modifying Accounts
- Group Administration
- Password Aging
- Default User Files
- Controlling Logins

PROCESS ADMINISTRATION

- at & cron Usage
- Anacron
- Viewing Processes
- Managing Processes
- Tuning Process Scheduling
- Process Accounting
- Enabling Process Accounting
- Setting Resource Limits via ulimit

NETWORKING

- Linux Network Interfaces
- Ethernet Hardware Tools
- Network Configuration with ip Command
- Configuring Routing Tables

- IP to MAC Address Mapping with ARP
- Starting and Stopping Interfaces
- NetworkManager
- DNS Clients
- DHCP Clients
- Network Diagnostics
- Information from netstat and ss
- Managing Network-Wide Time
- Continual Time Sync with NTP
- Configuring NTP Clients
- Multiple IP Addresses
- Enabling IPv6
- Interface Bonding
- Interface Bridging
- Network Configuration Tools :