LINUX Administration

Training Material

Rrootshell™

Excellence
Delivered

Rrootshell Technologiiss Pvt Ltd

Corp Head Office (Hyderabad):
202, Trendset Pyla,
Vengal Rao Nagar,
Hyderabad, Andhra Pradesh,
India - 500 038.
Phone : 040- 66464618,
66616704, 66629937
Email : hyd@rrootshell.com

Branch Office (Bangalore):
# 956, NS Plaza,16th Main Road,
B.T.M 2nd Stage (Mico Layout),
Bangalore, Karnataka
India - 560 076
Phone : 080- 32972711,
32965533
Email : blr@rrootshell.com

Branch Office (Gurgaon):
C-201/201, Sushant Arcade,
Sushant Lok 1, Near Huda City
Metro Station
Gurgaon, Haryana
India - 122 002.
Phone : 0124- 4077443,
9650377443
Email : gurgaon@rrootshell.com

INDEX

LINUX

Linux is a free and open-source operating system developed by Linus Torvalds and friends and was first announced by Linus in a post he made August 25, 1991. The Linux kernel runs on numerous different platforms including the Intel and Alpha platform and is available under the GNU General Public License.

The system can be distributed, used and expanded free of charge. In this way, developers have access to all the source codes, thus being able to integrate new functions or to find and eliminate programming bugs quickly. Thereby drivers for new adapters (SCSI controller, graphics cards, etc.) can be integrated very rapidly.
Linux may be obtained in two different ways. All the necessary components can be downloaded free of charge from the Internet. This means that an individual operating system can be assembled for almost nothing. An alternative is to use a so-called Distribution, offered by various companies and including a wide range of applications and installation programs that significantly simplify the installation of Linux.

Presently, Linux is successfully being used by several millions of users worldwide. The composition of user groups varies from private users, training companies, universities, research centers right through to commercial users and companies, who view Linux as a real alternative to other operating systems.

Linux distributions, flavors, and variants

There are hundreds of different distributions of Linux that have been released.

- Chrome OS
- Damn Small Linux (DSL)
- Debian
- Gentoo
- Puppy Linux
- Ubuntu

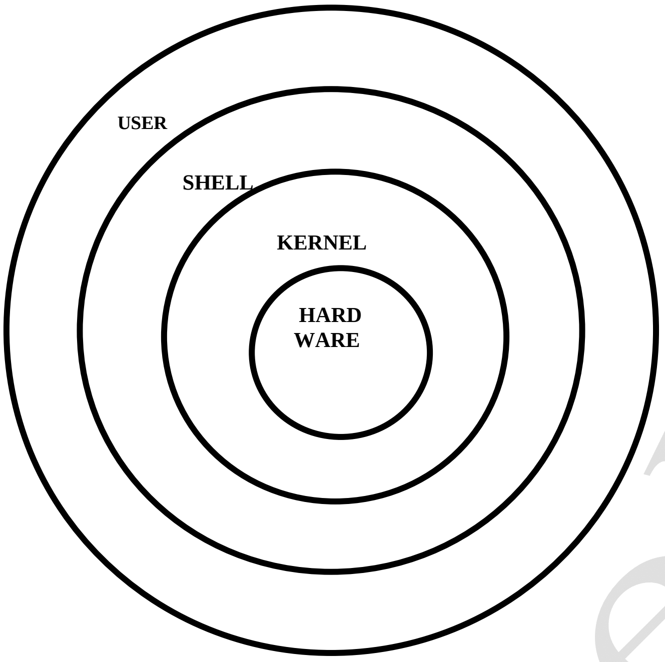
A Linux-based system is a modular Unix-like operating system. It derives much of its basic design from principles established in Unix during the 1970s and 1980s. Such a system uses a monolithic kernel, the Linux kernel, which handles process control, networking, and peripheral and file system access. Device drivers are either integrated directly with the kernel or added as modules loaded while the system is running.

Separate projects that interface with the kernel provide much of the system's higher-level functionality. The GNU userland is an important part of most Linux-based systems, providing the most common implementation of the C library, a popular shell, and many of the common Unix tools which carry out many basic operating system tasks. The graphical user interface (or GUI) used by most Linux systems is built on top of an implementation of the X Window System.

Some components of an installed Linux system are:

- A bootloader, for example GNU GRUB or LILO. This is a program which is executed by the computer when it is first turned on, and loads the Linux kernel into memory.
- An init program. This is the first process launched by the Linux kernel, and is at the root of the process tree: in other terms, all processes are launched through init. It starts processes such as system services and login prompts (whether graphical or in terminal mode).
- Software libraries which contain code which can be used by running processes. On Linux systems using ELF-format executable files, the dynamic linker which manages use of dynamic libraries is "ld-linux.so". The most commonly used software library on Linux systems is the GNU C Library. If the system is set up for the user to compile software themselves, header files will also be included to describe the interface of installed libraries.
- User interface programs such as command shells or windowing environments.

Architecture Of OS:-



User: User is nothing but an individual who uses the available hardware and software resources.

Shell: It is a command line interpreter. Shell access request from user and checks command existence, if command exist then it converts init kernel understandable language and send given request to kernel. Shell acts an interface between user and kernel.

Types Of Shells:

Shell Name	Developed By	Prompt	Interpreter Name
Bourne Shell	Stephen Bourne	$	Sh
Bash Shell	Stephen Bourne	$	Bash
Korn Shell	David Korn	$	Ksh
Z Shell	Paul	$	Zsh
C Shell	Bill Joy	%	Csh

- Advanced Version Of Bourne Shell Is Bash Shell(Bash ➡ Bourne Again Shell)

Default Shell	Flavor Name
Bourne Shell	Linux
Bash Shell	Solaris , Hp-UX
Korn Shell	IBM-AIX
C Shell	IRIX

Kernel: Kernel is the heart of the operating System. Kernel is responsible for interacting with the hardware and producing output to the Screen. It handles memory, file, device, process and network management for the operating system. Linux is truly just the kernel.

Difference between UNIX & WINDOWS:

	UNIX		WINDOWS
1	It is Multiuser and Multitasking O/S.	**1**	Windows also Multiuser and Multitasking O/S.
2	To boot unix o/s , 2 MB Ram is enough.	**2**	12 MB Ram is Required.
3	UNIX is process based concept.	**3**	It is thread based concept.
4	In UNIX any user process is killed It will not	**4**	It effect to all.

	effect to others.		
5	Unlimited users working on the Server.	**5**	Limited users system working on the Server.
6	UNIX is open system.	**6**	It is closed system.
7	It is portable O/S.	**7**	Not portable.
8	No down time.	**8**	Down time is there.

File System: It is method of storing the data in an organized fashion on the disk .Every partition on the disk except MBR and Extended partition should be assigned with some file system in order to make them store the data. File System is applied on the partition by formatting it with a particular type of file system.

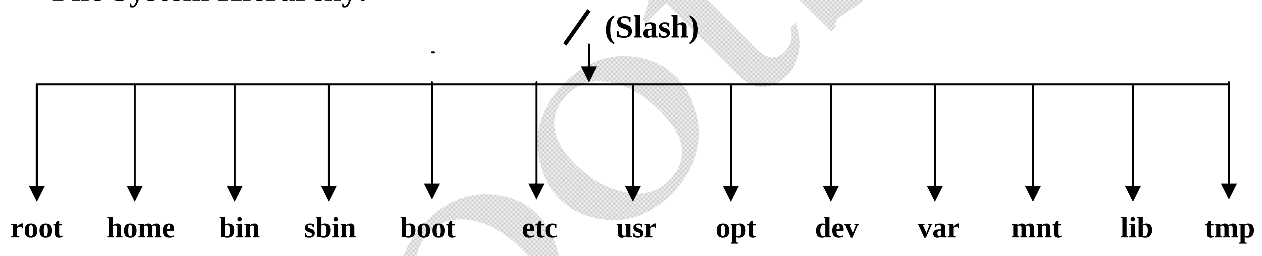
Types Of File Systems:

(a) Disk file system: A disk file system is a file system designed for the storage of files on a data storage device, most commonly a disk drive , which might be directly or indirectly connected to the computer.
Examples: FAT, FAT32, NTFS, CDFS, HFS, EXT2, EXT3, ISO 9660**.**

(b) Network file system: A Network file system is a file system that acts as a client for a remote file access protocol, providing access to files on a server.
Examples: DFS, NFS, SMB, FTP.

(c) Virtual file system: The purpose of VFS is to allow client applications to access different types of concrete file systems in a uniform way it can be used to bridge the differences in windows, macos and unix file system, so that applications can access files on local file systems of those types without knowing what types of file system they are accessing.

File System Hierarchy:



/: This is top level working directory .It is parent directory for all other directories .It is called as "***ROOT***" directory. It is represented by forward slash (**/**).

root: It is home directory for root user (super user). It provides working environment for root user.

home: It is home directory for other users. It provides working environment for other users (except root).

bin (*Binary files*): It contains commands used by all users.

sbin (*Super user binary files*): It contains commands used by only super user (or) root user.

boot: It contains system bootable files, boot loader information, kernel related information for Linux.

etc: It contains all system configuration files.

Example: /etc/hosts , /etc/resolv.conf

usr: By default software's are installed in /usr directory .(UNIX Sharable resources).

opt: It is a optional directory for users .It contains third party soft wares.

dev: It contains all device files information .similar to device manager of windows .In UNIX/LINUX every device treated as a file.

var: It is containing variable files information like mails, log files.

mnt: It is default removable media working directory it is empty by default.

lib: It contains library files which are used by o/s .It is similar to **"*.dll* "** files of windows. Library files in linux are shared object files.

tmp: It contains temporary files information.

media: It contains all of removable media like cd-rom , pendrive.

proc: It contains process files. Its contents are not permanent , they keep changing .Its also called as virtual Directory. Its files contains useful information used by o/s like ***/proc/meminfo*** , ***/proc/cpuinfo,……***

Basic Commands

#	➡	Root user prompt.
$	➡	User working prompt.
logname	➡	Displays current user name.
pwd	➡	Present working directory.
date	➡	Displays current date & time.
cal	➡	Displays current month calendar.
cal 2014	➡	Particular year total months.
cal 10 2014	➡	2014 year 10th month calendar.
who	➡	To displays the information about all the users who have logged into the system.
Whoami	➡	It displays current user name.
Finger	➡	It displays complete information about all the users who are logged in.
Uptime	➡	How long server up & running, how many users connected and load average time.
which (or) **whereis**	➡	Given command location.
tty	➡	Terminal position.
df	➡	Displays disk free size.
du	➡	Disk usage information.
clear	➡	To clear screen.

Creating Files:

cat (concatenate): It is used to create a file and display, appending the contents of a file.

Syntax: cat > filename , cat >> filename , cat filename

Examples:

To create a file:

```
cat > filename ↵
Hello world ↵
Ctrl+d (To save the file)
```

To display the content of the file:

```
cat < filename ↵   (or)
cat  filename ↵
```

To append the data in the existing file:

```
cat >> filename ↵
RrootShell Tech ↵
Ctrl+d(to save)
```

touch: To create multiple files but all are empty

Syntax: touch file1 file2 file3 file4 -----------filen (or) **touch file{1..n}**

Example:

```
touch  file1 file2 file3 file4 ↵    (or)
touch  file{1..10} ↵
```

ls: Display the contents of a directory.

Syntax: ls [options]

Options:

-r → reverse
-a → hidden
-R → recursively
-i → inode
-l → long list

Examples:

```
ls  -l   ↵
ls  -a   ↵
ls  -lr  ↵
ls  -R  /boot  ↵
ls  -i   ↵
```

mkdir: Creates a directory

Syntax: mkdir [options] <Directory name>

Options:

-p → parents

Examples:

To create multiple directories

```
mkdir  abc  xyz  ↵
```

To Create a nested directory

```
mkdir  -p  /x/y/z  ↵
```

Navigation Commands:

cd: Changes the current location.

cd .. → To go one level back.
cd ../.. → To go two levels back.
cd → To change users home directory.

Note: The trailing Slash (/) is optional when you're using the cd command. It indicates that the name being specified is a directory

Example: `cd  xyz  ↵`

cp: Copies files or directories from one location to another.

Syntax: cp [options] source destination

Options:

-R → Copies recursively
-f → Copies recursively
-v → Copies recursively

Examples:

```
cp  file1  file2        ↵   One file to another.
cp  file1 file2   /xyz  ↵   Multiple files into Directory.
cp  –R  /xyz   /abc     ↵   directory to another.
cp  -rvf  /xyz  /abc  /x   ↵   Multiple directories into Directory.
cp   /var/log/messages   .   ↵   (.) represents current location.
```

mv: Moves or renames files and directories

Syntax: mv [options] source destination

Options:

-v → verbose

Examples:

Rename the file by specifying the file name & new name of the file

```
mv  messages  file1   ↵
```

Move it to the *abc* directory for safe keeping

```
mv   file1   /abc   ↵
ls   /abc   ↵
```

rm: Deletes files or directories

Syntax: rm [options] file

Options :

```
-i  → interactive
-r  → recursively
-f  → forcefully
```

*Examples***:**

Delete the *file1* file:

```
cd   /abc   ↵
rm  –i   file1   ↵
```

Delete the *abc* directory:

```
cd  ..   ↵
rm  –rf   /abc   ↵
```

file: Displays the type of a file.

Syntax: file <file name>

Examples:

```
file   file1   ↵
o/p:  file: empty
file  /etc/passwd   ↵
o/p: passwd: ascii test
```

Meta Characters (or) **Wild Card Characters:**

(a) *** :** It matches zero (or) more characters in the given file.

Examples:

```
ls    a*        ↵    Displaying files start with 'a'.
ls    i*g       ↵    Start with 'i' end with 'g' .
ls    *g        ↵    List out end with 'g' only.
rm   i*         ↵    Removes start with 'i'.
cp    a*        ↵    Copies start with 'a'.
cp  –rf    i*g   /abc   ↵
cp  –rvf   *   /backup   ↵    Copies current directory all files.
```

(b) **?:** It matches any single character in the given file.

Examples:

```
ls    ?        ↵    Display single character files.
ls    ??       ↵    Two character files.
ls   a???      ↵    List four character files but first one is 'a'.
rm   ??        ↵    Removes two character files.
cp   ???   /abc /    ↵    Copies three character files.
```

(c) **[ ]**: It matches any single characters in the given list.

Examples:

```
ls   [aeiou]     ↵    Displays given matching character files.
ls   [aeiou]*    ↵    Displays start with a,e,i,o,u files.
rm  [aeiou]*     ↵    Removing start with a,e,i,o,u.
```

cp [aeiou]* /abc ← Copying start with ***a,e,i,o,u***.

(d) **[-] :** It matches any single character in the given range.

Examples:

ls [a–f] ← Displays start with ***a,b,c,d,e,f***.
ls [a–f , o-v] ← Displays start with ***a –e & o-v***.
rm [a–f] ← Removes ***a–f***.
cp [a–f,1-9] ← Copying files **a–f & 1-9**.

Using a Text Editors

Being able to use a text editor is probably one of the most critical skills to have as a system administrator. You constantly need to edit config files write scripts or make changes to system files…… all of which require you to use a text editor.

- The three most popular editors available are
  a) vi (or) vim : Text editor with great flexibility.
  b) emacs : Similar to vi an advanced text editor with many features.
  c)nano : A basic text editor for quick editing.

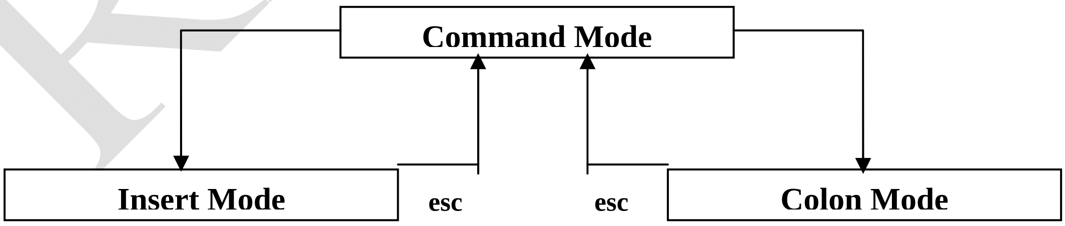
(a) **vi (or) vim :**This editor is used to create new files, open the files and modify contents of files.

- vi editor is most popular .
- It has three modes
  i. Command mode
  ii. Insert mode
  iii. Execution (or) colon mode
- By default mode is command mode.

Syntax: vi [arguments] [file]

Arguments:

-R	Opens a file in read–only mode.
-o	Open two files at a time.
+	Starts at the end of the file.
+**(**num**)**	Starts at line (num).



Insert Mode Options:

I	To begin insert mode at the current cursor position.
I	To insert at the beginning of the current line.
A	To append to the next words letter.
A	To append to the end of the line.

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

Commands For Command Mode:

E	Moves to the end of a word.
B	Moves to the beginning of a word.
$	Moves to the end of a line.
^	Moves to the beginning of a line.
H	Moves to the first line on screen.
M	Moves to the middle line on screen.
L	Moves to the last line on screen.
x(nx)	Deletes current character.
dd(ndd)	Deletes the current line.
dw(ndw)	Deletes current word.
yy(nyy)	Yanks (Copies) the current line.
P	Paste below the cursor line.
P	Past above the cursor line.
U	Undo the last action.
gg(ngg)	Go to beginning of the file.
G	End of the file.
w(n)	To move the cursor forward word by word.
b(n)	To move the cursor backward word by word.
ctrl+f	To forward one page.
ctrl+b	To backward one page.
/	To search a word in the file.
N	Find next occurrence of search word.
N	Find previous occurrence of search word.
.	Repeat last command action.

Commands For Last Line Mode:

:q	To quit without saving.
:w	To save the changes.
:wq	To save & quit.
:wq! (or) **x!**	To save & quit with force fully
:set nu (or) **:se nu**	To setting line numbers.
:set nonu (or) **:se nonu**	To remove line numbers.
:n	Jumps to line n.
:$d	To delete last line.
:!<unix cmd>	To execute unix cmds.
:X	To give password to the file and remove password.
:/string/	To search a word in the file.

To Find & Replace:

:%s/sachin/dravid/	↵	To replace string “dravid” for the first on a line instance.
:%s/sachin/dravid/g	↵	For each instance of a line.
:%s/sachin/dravid/gi	↵	To ignore case sensitive.
:%s/sachin/dravid/gc	↵	Ask for confirmation.

Executing unix Commands in vi: Any unix command can be executed from the vi command line by typing an **“/”** before the unix command.

Example:

:! Pwd ↵
:r! date ↵ Reads the results from the date command into a new line following the Cursor

- I want to copy 1 , 4 lines to paste after 10th line
  :1,4 co 10 ↵
- I want to move 3 , 7 lines after 8th line
  : 3,7 mo 8 ↵
- I want to copy 1, 30 lines create a new file
  :1,30w test1 ↵
- I want to append the data into a existing file
  :8,20w >> test1 ↵
- I want to insert end of the line (or) we required line
  :r /etc/passwd ↵

Managing Two Files At Time:

vim –o file1 file2 ↵
(or)
vim file1 file2 ↵

Options:

: n edit next file (file2)
: rew rewind to the file(file1)
(or)

- To move one file to another file (ctrl+w)
  └► Press two times

I/O Redirection:

- Some times you need to use the output from a command more than once .To accomplish this you can redirect the output of commands using some neat command line tricks.
- There are also a few characters you can use to direct or redirect output of commands these characters are

> Directs output to a file or device (override if the file exists).
< Directs input from the file of device.
>> Appends output or text to a file (creates if the file doesn't exist).
| Pipes the output of one command to another.
&& Combines commands.
(or)

STDIN - File Description (FD) – 0

STDOUT - File Description (FD) – 1

STDERR - File Description (FD) – 2

Examples:

1) cat file1 > backup ↵
   cat backup ↵ To verify
2) cat >test2<test1 ↵ Input from the test1
   cat test2 ↵ To verify
3) cat test1 test2 test3 2>error ↵
   cat error ↵
4) cat sample test test3 > out-file 2>>error ↵
   cat out-file ↵
   cat error ↵
   echo: Outputs or displays a string.

Example:

```
echo   " Hello World"   ↵
```

- **To output some text to a file:**

```
echo   "Hello World " > file2   ↵
cat     file2     ↵
```

cut: Divides a string or output.

Syntax: cut [options] file

Options:

-d   Specifies a delimiter.
-f   Displays a particular field.
-c   Displays a character.

- **Displays the third field of the text using space as a delimiter:**

```
cut  –d  " "  - f3  file2   ↵
```

- **Displays third & fourth fields:**

```
cut   f3,4  file2   ↵
```

- **Displays 1st to 5th character:**

```
cut  –c  1–5  file2   ↵
```

paste: To join two or more files horizontally by using delimiters.

- **To join two files horizontally:**

```
paste  states  capitals   ↵
                AP : Hyderabad
                MP : Bhopal
                KN : Banglore
paste  –d " :"  states  capitals > example   ↵
```

wc: Provides a word or line count.

Syntax: wc [options] file

Options:

-l   lines
-w   words
-c   characters

Examples:

```
wc  example    ↵        Displays lines words and characters
wc   –l   example   ↵          only lines
wc  –w   example     ↵         only words
```

diff: Displays different lines between two files.

Example:

```
diff  file1  file2   ↵
```

cmp: It compress two files character by character.

Example:

```
cmp  file1  file2   ↵
```

NOTE: If files are same it doesn't return any output otherwise it displays line numbers and character position.

tr: It translate character by character.

Examples:

```
tr  " aeiou"  "AEIOU"  < example   ↵
```

- **Translate lower to upper**

```
tr  “a –z “ “A –Z” < sample     ↵
```

- **Translate upper to lower**

```
tr  “A –Z”  “a –z” < sample       ↵
```

- **Squeeze**

```
tr  –s  “  ”< sample    ↵
```

- **To delete *aeiou***

```
tr  –d  “ aeiou”  <  sample     ↵
```

- **Replace with tab space**

```
tr   “ ,”   “\t”  < sample        ↵
```

aspellcheck: To check the spelling mistakes but not grammatical mistakes.

Examples:

```
aspellcheck  sample    ↵
aspellcheck   test     ↵
```

head: Displays top 10 lines of the file.

Examples:

```
head  sample    ↵
head   -5  sample    ↵
```

tail: Displays last 10 lines of file.

Examples:

```
tail  sample    ↵
tail  -5  sample    ↵                last 5 lines
tail  –f  sample        ↵            file is open continuously
```

piping(|): Combine the two or more commands into a single line.

Examples:

- Here the first command output is taken as the next command input

```
ls  –l | wc  –l    ↵
cat example | cut  –d  “  ”  -f3 file _example    ↵
cat example | head -20    ↵
```

&&: Combines commands.

Examples:

```
echo “Hello World” > file2  && cut  –f3 example    ↵
cat    file2      ↵                To verify
echo “My  text” >> file3  &&  cat  file3    ↵
```

more: To see the contents of a file in the form of page wise.

Example:

```
more  example    ↵
```

less: To display file contents in page wise .But we can go to all directions.

Syntax: less   [options] <filename>

Options:

f	forward direction
b	Backward direction
v	vi editor mode
q	To quit

Example:

```
less   example    ↵
```

tee: It is used to write the data into the files as well as on the screen.

Examples:

```
cat  sample   | tee  file1 file2 file3   ↵
cat  file1    ↵            To verify
cat   file2   ↵
```

sort: Sorts the output of a command or file.

Syntax: Sort [options]   file

Options:

```
-r     sorts in reverse order
-b    Ignores leading blanks
-n    compares according to numerical string value
```

Examples:

```
sort   example   ↵
sort  –r   example    ↵          Reverse order
sort  –n   example    ↵          Display numeric
sort   -u   example    ↵         Unique lines
sort   -f    example    ↵        Ignores case
```

uniq: Lists all the unique lines in a file as.

Examples:

```
uniq  example    ↵
uniq  –u  example   ↵      Displays  non duplicated lines.
uniq  –d  example   ↵    Displays only duplicated lines.
uniq  file4 > uniq_file  && cat  uniq _file   ↵
```

In above command to view uniq lines in the sample file create a new file based on the output and view the contents of this new file.

Sed: (Stream Editor): To search and replace strings or patterns in the given file.

- Sed is a multipurpose filter command.

Syntax: sed   “s/old string name/ new string name/g” <file name>

```
s           substitution
g          global occurrence in every line
```

Examples:

```
sed  “s/unix/linux/g”  sample    ↵
sed  “unix/linux/gi”  sample   ↵       Ignore case
sed  “s/unix/linux/”   sample   ↵
sed  “s/^unix/linux/gi”  sample   ↵
sed  “s/unix/gi”  sample        ↵    Delete  a word from a file
sed  -e  “s/unix/sas/gi”  -e  “s/linux/dba/gi”  sample   ↵
sed  –n  “2p” sample   ↵    To print 2nd row
sed  –n  “3,5p”  sample   ↵    To print 3rd ,4th,5th rows
sed  –n  “1p”   sample   ↵
sed  ‘-3d’   sample   ↵     Delete 3rd row
sed  ‘2,5d’  sample   ↵     Delete 2 to 5 lines
sed  ‘2,5 w file’ sample   ↵    It copies 2nd to 5th rows from sample
                                 file to file.
sed  ‘=’  sample   ↵     To get line numbers
```

Regular Expressions (or) Regex (Grep):

- Globally research a regular expression & print.
- To search a string or regular expression in a file(s).

Syntax: grep [options] PATTERN FIEL(S)

Examples:

```
grep  root  sample   ↵
grep  root  sample  example  backup   ↵
grep  root  *   ↵        search all files in a current directory
grep  –i   root  sample   ↵    Ignore case
grep  -c  root  sample   ↵    counts no of lines
grep  -n  root  sample   ↵    print the lines along with line no’s
grep  –l  root  sample   ↵    List file names only the given pattern
grep  –r  root   *    ↵    search  the pattern  Recursively
grep  –v  root  sample   ↵    prints non matching lines
grep  –o  root  sample   ↵    prints only the given pattern
grep   root  sample  --color   ↵    Displays output in color
grep  “it technology”  sample   ↵
grep  “exam*”   sample   ↵    prints start with exam pattern
grep   “b[aeiou]ll”  sample   ↵
         o/p: ball
              bell
grep   “b..d”  sample   ↵
         o/p: band
              book
```

Note: “ .” & “ * ” are wild characters , it matches any single character.

```
grep  c[on]  example   ↵
grep  [0–9]  example   ↵
```

word pattern :

\< \> → word boundary

\< → starting of the word

\> → ending as the word

Examples:

```
grep  “\< root \>”  sample   ↵
grep   “\<root\>”  sample   ↵
grep  “root\>”   sample    ↵
grep  “\<[0 – 9][0 -9][0 -9]\>”   sample   ↵
```

Line pattern:

Anchors:

^ → Start of the line.

$ → End of the line.

Examples:

```
grep  “^d”    sample   ↵    Line start with d
grep “^me”  sample   ↵    Line start with me
grep  “me$”  sample   ↵    Line end with me
grep  “^[^aei]  sample   ↵    Not start with a,e,i.
grep  “^unix $”  sample   ↵    Line should contain only unix
grep   “^$”  sample   ↵    Displays empty lines
grep  “^…$”   sample   ↵    Line should contain three characters
```

fgrep: To search the string more faster than the grep commands .

Examples:

```
fgrep   “unix   ↵
         >sas   ↵
         >dba”   sample   ↵
```

egrep (Extended grep): It is a combination of grep & fgrep plus some additional regular Expressions.

Examples:

```
egrep  "(unix/oracle/sas)"  sample   ↵
egrep  ab{3}c  sample  ↵    Extract occurrence of preceding character
egrep  "/<[0 -9 ]{4,7}\>"  sample   ↵
egrep   ab{3}c   ↵
```

o/p:abbbc

find: This filter is used to search the results by depending on requirements may be on name, inode, permission, user……etc.

Syntax: find <search path> <criteria> <action>

(a) Based On Name:

Examples:

```
find  /  -name  passwd   ↵
find  /home  -name  passwd   ↵
find  /etc  -name  'pass*'  ↵
find  .  –name  linux  ↵
```

(b) Based On Size:

+n → for greater than
-n → for less than n
n → for exactly n

Examples:

```
find  /  -size  4c   ↵    4 character files
find  /  -size  +4c  ↵     more  than 4 character files
find  /  -size  -4c  ↵    less than  4 characters
find  .  –size  +50m  ↵    more than 50m
find  /etc/Backup  -size  -50m  ↵    less than  50M
find  /  -size  +30M  -size  -50M  ↵    between  30 to 50M
```

(c) Based On Permission:

Examples:

```
find  /  -perm  644  ↵
find  /  -perm  665  ↵
find  /  -perm  777  ↵
```

(d)Based On Type:

Examples:

```
find  /  -type  f  ↵    To find files
find  /  -type  d  ↵    To find directories
```

(e) Based On Inode:

Examples:

```
find  /   -inum  14553   ↵
find  /root  -inum  23412   ↵
find  /home  -inum  76425   ↵
```

(f) Based On Time:

mtime → modification time
ctime → change time
atime → Access time

Examples:

```
find   /  -mtime  +10   ↵
find   /  -mtime   -10  ↵
find   /   -mtime    10  ↵
```

- **To find the file with access time**

```
find   /  root  -atime  +5  ↵
find  /root   -atime  -5  ↵
find  /root   -atime   5  ↵
```

- **To find the file with change time**

```
find   /    -ctime  +5  ↵
find   /    -ctime   -5  ↵
find   /    -ctime    5  ↵
find   /    -amin  +5  ↵   file was last accessed  5 minutes ago.
find  /root  -cmin  +5  ↵   files status was last changed 5 minutes ago.
find  /home  -mmin  +5  ↵
find  /root  -2min  -5  ↵
find  .       –amin  5  ↵
```

(g) Based On User:

Examples:

```
find  /  -user  <user name>  ↵
find  /  -user  user1  ↵   particular user  file
```

(h) Based on group:

Examples:

```
find  /   group  <groupname>  ↵
find  /    group   rrootshell  ↵   particular group files
```

Path: It is the way of representing files & directories in the system.

There are two types of paths

a) Absolute path
b) Relative path

a) Absolute path: It is the way of representing files and directories from the top of hierarchy.

Examples:

```
ls   /var/ftp/pub   ↵
cp  /home/user1/*   /home/user2/  ↵
```

b) Relative path: It is the way of representing files and directories which are related to current directory.

Examples:

```
cd  /home/user1  ↵
cp  file1  unix/sas  ↵
cd  unix/sas  ↵
ls  ↵
```

Monitoring System Performance (or) **Process Management:**

- A process is a program under execution.

  (or)

  A process is a instance of a running program.

- Process have their own address space in memory, thread of execution, and characters such as security context environment and current priority.
- The linux kernel tracks every aspect of a process by its process Id number information about each process is advertised by the kernel to user program through the /process/pid directories.
- When a process starts another program, the new process is called child process. This original process is the parent process of its child process. Child process inherit characteristics from its parent, such as its environment and the user and groups its as which its run.

There are two types of process:

(a) Foreground Process
(b) Background Process

Foreground Process: In foreground user can execute only one process (or) job.

Example:

```
Firefox ↵
```

- ▪ **To kill foreground process (or) job.**

```
Ctrl+c ↵
```

Background process: In Background user can execute many jobs at a time.

Example:

```
Firefox & ↵
cp file1 file2 & ↵
```

- ▪ **To check the jobs list**

```
jobs ↵
```

ps: Displays information about running process.

```
ps ↵
```

- ▪ **To view process with more detailed information**

```
ps u ↵
```

- ▪ **I can detailed about a particular process**

```
ps aux | grep ssh ↵
      (or)
ps aux ↵
```

kill: Terminates a process.

Syntax: kill pid ↵

Example:

```
kill 352 ↵
```

- Sometimes if the kill command doesn't work the way you intended it to you can also call it with the -9 option to give it priority on the system.

```
kill -9 352 ↵
```

signal:

- The operating system communicates to process through signals. These signals reports events or error situations to process. In many cases signals will result in the process existing.
- One typical signal is SIGTERM, which terminals the process, it asks it to exit cleanly.
- Another is SIGKILL ,which kills the process, the process is required to exit immediately.

- ▪ **To Find The PID(s) Belonging To The SSH Service**

```
pidof sshd ↵
```

(or)

```
pgrep  sshd  ↵
```

Top: Monitors system resources (similar to task manager in windows).

```
top  ↵
```

Options:

S → To change the time internal for updating toop results(sec 's)
R → To sort by pid number
U → username to get only that user process details
p → To sort by cpu utilization.
M → To sort by ram utilization.
c → To display or hide command full path.
r → To renice a process
k → To kill a process
w → To save the modified configuration
q → To quit

- When you are comfortable working with process, you can then make some more advanced adjustments, such as changing the priority of a particular process.

renice: Adjusts the priority of a particular process

Syntax: renice <priority> [options]

Options:

-p → changes process priority for a particular PID.
-u → changes process priority for a particular user(s).

- The priority value range from -20 (first priority) to 20 (dead last priority) only the root user may set process to use a priority under 0.

Example:

```
renice  -1  4547  ↵
```

Note: If all ready process have the same priority ,they will share the process equally. priority only has an effect when two process at different priority levels are computing for cpu time in which case the lower priority process will get less time & appear to run more slowly

nohup: The nohup jobs will create in server in server account so nohup jobs will execute even the user disconnects from his account .

Examples:

```
nohup cp file1 file2  ↵
nohup Firefox  &  ↵
```

Communication Commands:

- The main concept of communication facility exchanging of information or files from one user to another user.

write: It is used for to write message to another user account but he should be logged into the user.

Examples:

```
write  username    terminalname  ↵
-----------------
------------------  ↵
Ctrl+d( save & quit )
```

- To deny messages

```
mesg   n  ↵
```

- To allow messages

```
mesg   y  ↵
```

Wall: It is used for to send broadcast message to all users who are connected to server.

Example:

```
wall ↵
Welcome to linux….. ↵
 Ctrl+d( save & quit)
```

mail: Using mail command you can quickly and efficiently circulate memos and other written information to your co-workers you can even send and receive mails from people outside your organization.

Examples:

```
mail  user1  redhat4.rootshell.com  ↵    Single user
mail  user1  user2  user3  ↵   Multiple user at a time
mail user1 < stud  ↵      }  It translates files to user
mail user2 < file2  ↵     }
```

- **Mails are stored in mailbox**

  /var/spool/mail/username

- **To open the mail box**

  mail ↵

Note: By default all mails will store in primary mail box ***(/var/spool/mail).*** It will open mails in primary mail box transferred to secondary mail box (i.e., mbox).

- **To open secondary mail box**

  mail –f ↵

- The primary mailbox only maintains unread mails.

Mail box options:

q	→	Quit
r	→	Reply
p	→	Print
d	→	Delete
d 2	→	delete 2nd mail
w filename	→	It writes to new file

One Bit Equals To How Many Bytes: It is the smallest component of data and byte is larger than bit size. The size 1000 can be replaced with 1024 and still be correct using the other acceptable standards. Both of these standards are correct depending on what type of storage you are referring.

Processor (or) Virtual storage		**Disk storage**	
1bit	Binary digit	1bit	Binary digit
8bit	1Byte	8bit	1Byte
1024bytes	1Kilobyte	1000bytes	1Kilobyte
1024kilo	1Megabyte	1000kilo	1Megabyte
1024mega	1Gigabyte	1000meg	1Gigabyte
1024giga	1Terabyte	1000giga	1Terabyte
1024tera	1Petabyte	1000tera	1Petabyte
1024peta	1Exabyte	1000peta	1Exabyte
1024exa	1Zettabyte	1000exa	1Zettabyte
1024zetta	1Yotta	1000zetta	1Yotta
1024yotta	1Bronto	1000yotta	1Bronto
1024bronto	1Geobyte	1024bronto	1Geobyte

Links: To give a pointer to the source file called as a link.

- In unix/linux two types of files

  (A) soft links **(B)** hard links

(a) Soft links:

- The inode number of the source file, link file are different.
- It can be a created across the file system.
- Editing of original file will be replicated in the link files.
- Size of soft link file equals to number of characters in original file path.
- If source file is deleted the file will not be accessible
- It is also be called as shortcut link.

Syntax: ln –s <source file > < link file > ↵

Example: ln –s /home/user1/abc.txt /softlink ↵

(b) Hard links:

- Source file, link file has same inode numbers.
- It can't be created across file system.
- Editing of original file will replicate in the link files.
- Size of hard link file is same as original file.
- The source of hard link file is same as original file.
- It is a backup link.

Syntax: ln <source file> <link file> ↵

Example:

ln /root/abc.txt /hardlink ↵

Shell concept: Shell is a command line interpreter, The shell access request from user and the checks command Existence, if the command exist then it converts into kernel understandable language (Machine language) and it send the given request to kernel.

- The shell access interface b/w user and kernel

Types Of Shells:

Shell Name	Developed By	Prompt	Interpreter Name
Bourne Shell	Stephen Bourne	$	Sh
Bash Shell	Stephen Bourne	$	Bash
Korn Shell	David Korn	$	Ksh
Z Shell	Paul	$	Zsh
C Shell	Bill Joy	%	Csh

- Advanced Version Of Bourne Shell Is Bash Shell(Bash ➡ Bourne Again Shell)
- Bash shell is default shell in linux.

Default Shell	Flavor Name
Bourne Shell	Linux
Bash Shell	Solaris , Hp-UX
Korn Shell	IBM-AIX
C Shell	IRIX(Silicon Graphics)

Features Of Shells:

- The following are features of shells:
  - **i.** Word Completion
  - **ii.** Command History
  - **iii.** Command Alias

- **To check the shell**

```
cat    /etc/shells   ↵
```

- **To check parent shell of current user**

```
echo   $SHELL   ↵
```

- **To view the available shells**

```
ls    /bin/*sh   ↵
```

- **To shift from bash shell to sh shell**

```
sh   ↵
```

- **To shift from sh shell to k shell**

```
ksh   ↵
```

- **To check current working shell**

```
echo   $0   ↵
```

- **To Exist current working shell**

```
exit   ↵
```

Command Completion: Linux automatic command completion is a tool or program that can identify what you are typing in the linux command line terminal and can complete that command, words or sentence for you. This is really cool feature in linux.

- When <TAB> key is pressed, any command starting with the given string will be completed by the system automatically.
- For multiple commands that starting with the given string, pressing <TAB> key twice will list down all these matched files or commands.
- If there are no matched commands, files or folders then the automatic word completion will not shown , a 'ting' sound will buzzed.

Example:

```
cd   /etc/pas<tab>   ↵
```

Command History: The history command performs one of several operations related to recently-executed commands recorded in a history list. Each of these recorded commands are referred to as an 'event', when specifying an event to the history commands.

Examples:

```
history         ↵
history 10      ↵
history  –c     ↵    Lock the history
history   –r    ↵     unlock the history
rm    .bash_history   ↵     Removes the history
```

Command Alias:

- Alias is a built in shell command in Linux/Unix operating systems.
- It can save you a lot of typing by assigning a name to long commands.
- The alias command can be useful if want to create a 'shortcut' to a command.

Syntax: alias aliasname ='command'

Examples:

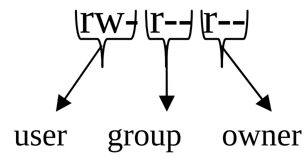
```
alias  u=useradd    ↵
alias   c= clear    ↵
alias               ↵    Display alias list
unalias   u         ↵    Disable  alias
vi   .bashrc        ↵    Permanent  alias names
```

FILE PERMISSIONS

- Just like every operating system Linux comes with a set of permissions that it uses to protect files, directories and devices on the system.
- These permissions can be a manipulated to allow (or) disallow access to files and directories on different parts of the system.

Basic File Permission:

- Let's look at how permissions work first. Linux permissions are implemented through the properties of files defined by three separate categories.



User: Person who owns the file.

Group: Group that owns the file.

Other: All other users on the system.

- Permissions in linux can be assigned one of two ways. you can use the mnemonic or a single digit to represent the permission level.

Operation	Digit	Mnemonic	Description
Read	R	4	View file contents
Write	W	2	Write or change
Execute	X	1	Run the file

Default File Permissions:

umask: Universal mask is a default value that always gets dedicated from maximum file permission allocated for every file & Directory.

- For super user umask value is ***#022***
- For Normal user umask value is ***$002***

For Super User:

```
     Maximum permission of a file   666
                            Umask   022
                                  -----------
       Default file permission     644
                                  -----------
  Maximum permission of a directory    777
                                 Umask   022
                                       -----------
     Default directory permission    755
                                       -----------
```

For Normal user:

```
     Maximum permission of a file    666
                          Umask     002
                                 ------------
    Default directory permission     664
                                 ------------

  Maximum Permission of a directory   777
```

```
                Umask   002
                   ------------
Default directory permission    755
```

- **To see the umask**

```
umask  ↵
```

- **To change the umask**

```
umask  ↵
```

- **To view umask value from the file**

```
vi   /etc/bashrc  ↵
```

Operators:

+ → To add a permission.

- → To remove a permission.

= → To override the permission.

- Here are some of the commands you can use to work with permission.

  (a) chmod
  (b) chgrp
  (c) chown

(a) chmod: It is used to change the permission of a file and directory. It can be used by the owner of a file (or) by root.

Syntax: chmod [options] [permission] [File]

Options:

-R → Acts recursively.

-v → Provides verbose output.

Example:

```
chmod  u+rw,g+r,o+x  Linux  ↵
          (or)
chmod  641  Linux  ↵

chmod ugo=rw  backup  ↵
          (or)
chmod  666   backup  ↵

chmod  u–w,g-r,o-x  Linux  ↵
          (or)
chmod   400 Linux  ↵

chmod –R u+w,g+r,o+x Linux  ↵

chmod  -R 777 unix  ↵

chmod 755 unix  ↵
```

(b) chgrp: By using this command we can change group of the file.

Syntax: chgrp [options] [groupname] [file]

Options:

-R → Recursively

-v → Verbose

Examples:

```
ls  –l   Linux  ↵
chgrp  sales  Linux  ↵
```

(c) chown: This command is used to we can change the owner of the file, as well as owner & group at a time.

Syntax: chown [options] [user :group] [file]

Options:

-R → Recursively   -v → Verbose

Examples:

chown user1 Linux ↵ To change only owner.
chown user1:sales Linux ↵ To change owner & group.

- **To view the symbolic as well as numeric mode of permission.**

  stat Linux ↵

- **To change the permission in GUI mode**

  nautilus & ↵

- **Assign the permission in GUI mode**

  Right click on file → properties →permissions.

JOB AUTOMATION

- In any operating system, it is possible to create jobs that you want to reoccur. This process known as job scheduling is usually done based on user-defined jobs.
- As an administrator however you can define your own jobs and allow your users to create them as well.
- The importance of the job scheduling is that the critical tasks like taking backups which the clients usually wants to be taken in nights can easily be performed without the intervention of the administrator by scheduling a ***cronjob***. If cron job is scheduled carefully then backup will be taken at any given time of client and there will be no need for administrator to remain back at nights to take the backup.
- For Red Hat or any other Linux, this process is handled by the ***cron service*** or ***crond daemon***, which can be used to schedule Jobs.
- By default, Red Hat comes with a set of predefined jobs that occur on the system(hourly, daily, weekly, monthly, and with arbitrary periodicity).
- There are two tools to scheduling jobs.
  (a) At
  (b) Crontab

AT Jobs:

"AT" is used to schedule the job for a particular time or interval, in other words it is used only for one time or only for one interval.

Syntax: at [option]

Option:

-l	Lists all jobs in the queue
-d	Removes a job from the queue.
-f	Reads input from the file.
-m	Sends mail to the user when the job is complete.

Examples:

```
1) at 9am   ↵
   at >date
   ctrl+d (save&quit)
2) at  now+3days   ↵
   at>/bin/echo  "Hello World"
   ctrl+d(save & quit)
3) at  03222013   ↵
   at>ls
   ctrl+d(save & quit)
4) at 1:30  3/22/2013   ↵
  at>cp  file1  file2   ↵
  ctrl+d(save & quit)
5) at –f  filename  11pm   ↵
  at>/bin/echo  "Hello World"
  ctrl+d(save & quit)
```

View the currently queued jobs:

```
at -l  ↵        (or)        atq  ↵
```

Delete the job from the queue:

```
at –d  1   ↵       (or)              atrm  1   ↵
```

Verify that the job is truly gone:

```
atq   ↵
```

To View the job details:

```
at –c 2   ↵
```

Restricting a user from using at jobs:

The atd service uses two files to control access to the services.

i. /etc/at.allow
ii. /etc/at.deny

/etc/at.allow file:

- If it exists only these users are allowed(at.deny is ignored).
- If it doesn’t exist all users except at.deny are permitted.

/etc/at.deny file:

- If it exists and is empty all users are allowed (Red Hat default)

For both files:

- If neither file exists, allows root user only.

Cron Jobs:

- The default setting for Red Hat allows any user to create a cron job. As the root user, you also have the ability to edit and remove any cron job you want.
- Lets jump into creating a cron job for the system. You can use the crontab command to create, edit, and delete jobs.

Syntax: crontab [-u user] [option]

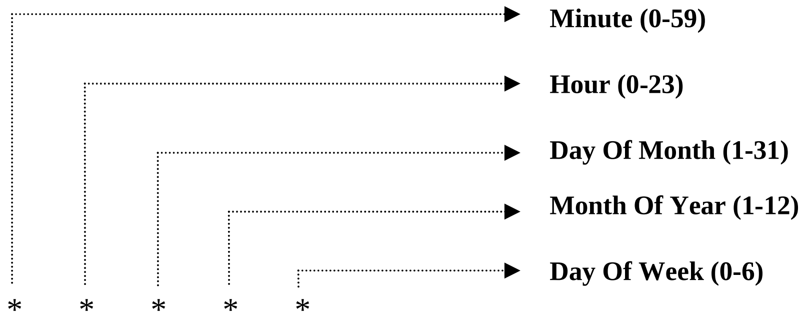
Option:

-e	Edits the users crontab
-l	Lists the users crontab
-d	Deletes the users crontab
-i	Prompts before deleting the users crontab

- Before you start using crontab command, however you should look over the format it uses so you understand how to create and edit cron jobs. Each user has his own crontab file in ***/var/spool/cron,*** based on the username of each user. Any ***“allow”*** actions taken by the cron service are logged to ***/var/log/cron***.

To view the /etc/crontab file to understand its syntax:

```
cat   /etc/crontab   ↵
```



Examples:

```
crontab   -e   ↵
                    */01      *        *        *        *                    date
                    */30      11       *        *        *               cp   file1   file2
                     45     10,22      03       *        *          /bin/echo  “Hello RrootShell”
                     50       22      01,11     10       *                    logname
                     59       23       31       12       *          bin/echo  “Happy New Year”
                    */01      *        *        *       0,6        /bin/echo/  “Today is weekend”
:wq!   ↵
```

To check assigned cronjob:

```
cronjob  -l   ↵
```

To check the cronjob service:

```
service crond status   ↵
```

Restart the cron service:

```
service crond restart   ↵
chkconfig crond on   ↵
```

To setup user1’s crontab:

```
crontab  -u  user1  -e   ↵
 *      *       *       *       *        /tmp/sample_script
 :wq!
```

To remove a user1’s crontab jobs:

```
crontab   -u  user1  -r   ↵
```

You can verify the log file:

```
tail   /var/log/cron   ↵
```

Note:

What do you think happens if you set up cron jobs to run during the night (say to run some reports) and you shut down the system right before you go home? Well, it runs out that there is another great feature of cron. The ***/etc/anacrontab*** file defines jobs that should be run every time the system is started. If your system is turned off during the time that a cron job should have run, when the system boots again, the cron service will call ***/etc/anacrontab*** to make sure that all missed cron jobs are run.

Let’s look at the /etc/anacrontab file:

```
cat   /etc/anacrontab   ↵
```

Control access to the cron service:

- To start working with cron, first need to know at the two configuration files that control access to the cron service.
- These two files are
  1. /etc/cron.allow
  2. /etc/cron.deny

/etc/cron.allow file:

- If it exists, only these users are allowed (cron.deny is ignored)
- If it doesn’t exist, all users except cron. Deny are permitted.

/etc/cron.deny file:

- If it exists and is empty all users are allowed(Red Hat default)

For Both files:

- If neither file exists root only.

USER ADIMINISTRATION

- User is nothing but an individual who uses the available hardware and software resources.
- In Red Hat there are three different types of users.

  a) Super User [or] root user
  b) System Users
  c) Normal User

(a)Super User [or] root user:

- The root user account is the equivalent of the administrator (or) enterprise admin account in the windows world.
- It is the most powerful account on the system and has access to everything.
- As an root user you can assign access rights to different files & directories allowing your users to gain access to different areas of the system(outside their home directory).

(b) System Users:

- A system account is similar to a natural user account. The main difference is that system users normally don't have a home directory and can't login the way normal users do.
- Many system users are created or associated with applications or service to help run them more securely.
- For example if we install Apache it will create a user apache. These kinds of users are known as system users.

(c) Normal Users:

- Normal users are the users created by the root user. They are normal users like Rama , Ravi , Siva , …… etc.
- Normal user accounts have no write access to anything on the system except their home directory (they can read and explore much of the system, however) which is created when the user account is added.

Some important points related to users:

- Users & Groups are used to control access to files and resources.
- Users login to the system by supplying their username & password.
- Every file on the system is owned by a user and associated with a group.
- Every process has an owner and group affiliation and can only access the resources owner or group can access.
- Every user of the system is assigned a unique user ID .
- ***Users name*** and ***user ID*** are stored in " ***/etc/passwd"*** file.
- ***Users password*** is stored in ***"/etc/shadow"*** in encrypted form.
- Users are assigned a home directory and a program that is run when they login (usually a shell).
- Users can't read, write or execute each other's files without permissions.

Types of Users in Linux and their Attributes:

Type	Example	UID	GID	Home Directory
Super User	Root	0	0	/root
System User	ftp , nobody ,…	1-499	1-499	/var/ftp
Normal User	Ravi , Siva	500-60,000	500-60,000	/home/user

Whenever a user is created in Linux thing created by default:

- A home directory is created (/home/username)
- A mail box is created (/var/spool/mail)
- Unique UID & GID are given to user.

User Private Groups(UPG):

- Red Hat Linux is having User Private Group(UPG) schema.
- According to UPG , if you add the any user without any group , the users primary group is created with username and group ID same as user ID .
- For Example if a user is created with the name Ravi , then a primary group of that user will be Ravi only.

User information maintained by the two database files:

a) /etc/passwd
b) /etc/shadow

(a) /etc/passwd:

- This file maintains user related information.

Fields Of Passwd file:

- ***/etc/passwd*** file has ***seven*** fields

<username>:*<password>*:*<UID>*:*<GID>*:*<comments>*:*<homedirectory>*:*<shell>*

(b) /etc/shadow:

- This file maintains user related password information.

Fields Of Shadow file:

<username>:*<encrypted password >*:*<last password change>*:<min>:
<max>:<warn>:*<inactive>*:<expires>:*<not used>*

Complexity Requirements of password:

- A root user can change password of self and any user in the system, there are no rules for root to assign a password. Root can assign any length of password either long or short it can be alphabet or numeric or both on the whole there is no limitation for root for assigning a password.
- A normal user can change only his/her password. Valid password for a normal user should add here to the following rules.
  - It should be at least 7 characters but not more than 255 characters
  - At least one character should be upper case.
  - At least one character should be lower case.
  - At least one character should be a symbol and number.
  - It should not match the previous password.
  - The login name and the password can’t be same.

To manage user accounts you can use the following commands:

useradd:- create user (or) system accounts

Syntax: useradd [options] LOGIN

Options:

-u	User ID
-c	Comment

-e	Expiration Date
-s	Shell
-r	Creates a System Account
-d	Home Directory
-g	Primary Group ID
-G	Secondary Group ID

Examples:

- *create a user*

```
useradd ravi ↵
```

- *To check the user you just created*

```
cat /etc/password | grep ravi ↵
```

- *Creating user with our own attributes*

```
useradd -u 501 –g 500 -c “System Admin” -d /home/siva
  -s /bin/bash siva ↵
```

- *To check the user:*

```
cat /etc/password | grep siva ↵
```

***usermod*:-** Modifies user accounts

Syntax: usermod [options] LOGIN

Options:

-l	To change login name
-L	To lock account
-U	To unlock account

Example:

- Changing the name of the user

```
usermod –l newname oldname ↵
```

- To lock the user account

```
usermod –L username ↵
```

- To unlock the user account

```
usermod –U username ↵
```

Note: when an account is locked it will show ***!(exclamation mark)*** in ***/etc/shadow*** file.

***userdel*:-** Removes a user or system account.

Syntax: userdel [options] LOGIN

Options:

-f	Forces deletion of the user even if he’s still logged in
-r	Removes the user’s home directory and mail spool

Example:

```
userdel username ↵
```

***password*:-** Sets a password or resets a password for a user account.

Syntax: passwd [options] LOGIN

Options:

-l	locks a users account

-u unlocks a user’s account

Example:

```
passwd  siva   ↵
cat /etc/shadow | grep siva   ↵
```

***chage*:-**Enables you to modify the parameters surrounding passwords (complexity , age , expiration)

Syntax: chage [options] user

Options:

-d	Indicates the day the password was last changed
-E	Sets the account expiration date
-I	Change the password in an inactive state after the account expires
-l	shows account aging information
-m	sets minimum number of days between password changes
-**M**	sets maximum number of days a password is valid
-**W**	sets the number of days to warn before the password expires

Example:

- Finds the users password information.

```
chage  -l  user   ↵
```

- Sets user account to expire in one week

```
chage  -E  2013-03-28 siva   ↵
```

***pwck*:-** Verifies the consistency of passwords across database files.

- When you create or delete users, sometimes things don’t always work out properly. This can cause the password file to become inconsistent
- Use the ***pwck*** command to verify the consistency between the /etc/passwd & /etc/shadow file.

```
pwck   ↵
```

GROUP ADMINISTRATION

The control of users and groups is a core element of Red Hat Enterprise Linux system administration

Group Membership

In any UNIX environment, a user can be a member of two different kinds of groups: the primary group and all other groups. Every user must be a member of a primary group. If one user on your system does not have a primary group setting, no one will be able to login, so membership in a primary group is vital. On a Red Hat server, all users are by default a member of a group that has the same name as the user. This is done for security reasons to make sure that no files are shared with other users by accident.

Users can be members of more than just the primary group, and they will automatically have access to the rights granted to these other groups. The most important difference between a primary group and other groups is that the primary group will automatically become group owner of a new file that a user creates.

UID

Another major type of information used when creating a user is the user ID (UID). For your server, this is the only way to identify a user. (Usernames are just a convenience because we, as humans, tend not to handle being identified by numbers well.) In general, all users need a unique UID. Red Hat Enterprise Linux starts generating local UIDs at 500, as explained in a moment. The highest UID available by default is 60000. This is because of a restriction that is defined in /etc/login.defs, which can be changed if needed. Typically, UIDs below 500 are reserved for system accounts that are needed to start services.
The UID 0 is also special—the user who has it has complete administrative permissions to the server. UID 0 is typically reserved for the user root.

Shell

To log in to a server, every user needs a shell. The shell will enable interpretation of commands the user enters from their console. The default shell in Linux is /bin/bash, but several other shells are available. One of the more common alternative shells is /bin/tcsh, which has a scripting language similar to the C programming language, which makes tcsh the perfect shell for C programmers.

- Group is nothing but collection of users using which one can reduce the administration task in the operating system environment.
- Group are divided into two types
  a) Primary Group
  b) Secondary Group

(a) Primary Group: It is a group in which a user initially belongs in this group the user can access the resource with default permissions.

(b) Secondary Group: Apart from primary if a user have an account in the other group i.e., then it is called as secondary group to the user.

- Group information maintained by the two database files

  i. /etc/group
  ii. /etc/gshadow

(i) /etc/group:

- This file maintains group related information.

Syntax: *<group name>***:***<password placeholder>***:***<GID>***:***<members>*

(ii) /etc/gshadow:

- This file maintains group password related information.

Syntax: *<group name>***:***<password placeholder>***:***<group admin>***:***<members>*

groupadd: Creates a group.

Syntax: groupadd [options] groupname

Options:

-r	creates a system group
-g	groupid

Example:

- Let's create a group called sales.

```
groupadd sales ↵
```

- To verify the group

```
cat /etc/group | grep sales ↵
```

- To add the user:

```
usermod –G sales user1 ↵
cat /etc/group | grep sales ↵
```

- To add another user to the sales group

```
usermod -G sales user2 ↵
```

- Now if you verify you should see two user accounts in the last field

```
cat /etc/group | grep sales ↵
```

Another way to verify, what groups a user belongs to

Syntax: id [options] username

Options:

-G	Shows the GID
-n	Shows the name instead of the ID
-U	shows the UID

Examples:

```
id -Gn user1 ↵
```

- If ***id*** Command is called without any options you can also see what UID & GID the user has

```
id user1 ↵
```

groupmod**:**- Modifies the properties of a group

Syntax: groupmod [options] groupname

Options:

-g	groupid
-n	new group name

Examples:

- To change the groupid:

```
groupmod -g 888 sales ↵
```

- To change groupname:

```
groupmod  -n  <newname>   <Existing name>
```

***gpasswd*:**-Assigns password to the group

Syntax: gpasswd   **[**groupname]

Example:

- assigns password to the group

```
gpasswd    <groupname>
```

- Adding and removing members to a group:

Syntax: gpasswd   [options] <arguments>   <Groupname>

Options:

-a	Add single User
-M	Add Multiple Users
-A	Group Administrator
-d	Removing a user from group

Examples:

```
gpasswd  -a  user1  sales   ↵
gpasswd   -M  user2,user3,user4  color   ↵
gpasswd   -d  user2 color   ↵
gpasswd    -A user3 color   ↵
```

***groupdel*:** Deletes a group

Syntax: groupdel   [groupname]

Example:

- To remove a group

```
groupdel   <groupname>
```

Note: If the group has empty (or) secondary users you can delete the group. In case the group maintains single primary user then you can't delete the group account.

Switching Accounts:

***su*:**- Enables you to run a command as another user or switch user accounts.

Syntax: su   [username]

Examples:

- To switch accounts use this command:

```
su   user1   ↵
```

- You can also login as the root user using the SU command:

```
su   – root   ↵
```

TIP:- You can login to the root user account using the 80 command with no parameters. So what is the difference between using ***su*** & ***su* -** . The ***su*** command moves you into the root user's account without initializing any of root's path or shell variables. When you use ***su* -**, everything is initialized as if you were logging in form the console.

User Account Initialization:

- When a user is created, everything from the "***/etc/skel"*** directory _is copied to the user's newly created home directory(usually "***/home/<username>"***).
- You can modify these "skeleton" files or can add your own Custom files. The benefit here is that user creation becomes standardized, ensuring that policies are adhered to.
- Customizable files are broken down in to two different sections.

a) User specific files
b) Global user Configuration

(a) User specific files: After a user is created and his home directory is populated that user can now customize those file to fit his own personal needs.

.bashrc	Defines functions and aliases.
.bash_profile	Sets environment variables.
.bash_logout	Defines any commands that should be executed before the users log out.

(b) Global user Configuration:

/etc/bashrc	Defines functions and aliases.
/etc/profile	Sets environment variables
/etc/profile.d	Specifies a directory that contains scripts that are called by the "***/etc/profile"*** file.
/etc/login.defs	This file controls specifies relating to system wide user logins and Passwords .

```
grep  -v  ^# /etc/login.defs   ↵
        (or)
cat  /etc/login.defs   ↵
```

Group Collaboration:

Group collaboration is an essential part of any business and for any system administrator who deals with users. Here we look at three key features about file and directory permissions.

a)Setuid
b) Setgid
c)Sticktybit

(a) Setuid: This flag is used to allow multiuser access.

- For example, if you have a script that generates reports for your company, but the script must be run as user1 to succeed you can set the ***setuid*** bit to enable other users to run this command as through they were user1.

Example:

- Create a file to hold the report script:

```
touch  reporting_script
```

Set the setuid bit:

```
chmod  4755  reporting _script
                (or)
chmod  u+s   reporting _script
```

Now view the permissions of the file:

```
ls   –l    reporting_script
```

o/p:-

```
-rwsr-xr-x. 1  root  root  0 Jan  8 11:59 reporting_script
```

- In the file's owner permissions, notice that there is an ***'S'*** in place of ***'x'*** .This shows that this file has the setuid flag set.
- To find all setuid files:

```
find  /  -perm  4000   ↵
```

(b) Setgid: This flag is used to allow multigroup access.

- Which is similar to setuid but set as the group level instead with this bit set, all users of the group are able to execute the file instead of just the user who owns it. The setgid bit allows users to collaborate on files.

Example:

- As root, create the directory:

```
mkdir   /tmp/oracle   ↵
```

- Create the group and add users to it

```
groupadd   sales   ↵
usermod  -G sales  user1   ↵
usermod  -G sales  user2   ↵
```

- Assign the permissions for collaboration:

```
chown    root:sales   /tmp/oracle
```

Set the setgid bit:

```
chown   2770  /tmp/oracle
```

- Verify

```
ls  -ls  /tmp/oracle   ↵
```

- Now all members of the sales group are able to read/write to files within this folder. Also, notice that access to this folder is denied for anyone who is not a members of the sales group.

(c) Sticky bit: This flag prevents accidental delete by users & groups.

Example:

- Set the sticky bit on the /tmp directory:

```
chmod  1777   /tmp   ↵
```

- Verify

```
ls  –ld    /tmp   ↵
```

- For the sticky bit there is ***'t'*** on the end of the permissions listed. Now other users are not able to delete your files only you can delete your files .
- This feature might be helpful when you are sharing files and there are particular files you don't want other users to delete.

PACKAGE MANAGEMENT

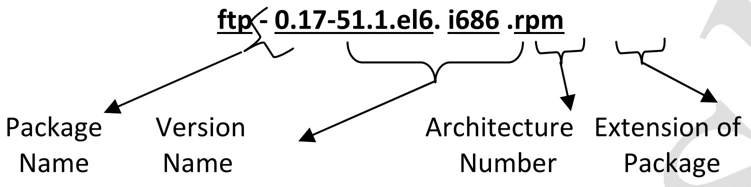
All software on a Red Hat Enterprise Linux system is divided into RPM packages which can be installed, upgraded, or removed. This part describes how to manage the RPM packages on a Red Hat Enterprise Linux system using graphical and command line tools .

- Software is the basic of any operating system, allowing you to install and use different utilities.
- In Linux, Software is distributed through the use of packages which contain the actual software files.
- Each distribution of Linux has its own package management system.
- For Red Hat, there are two package management tools
  a) RPM
  b) YUM

(a) RPM→(Redhat Package Manager)

- RPM is default package installation tool in Linux Operating System.
- By using RPM we can install, upgrade, query, verify and Remove the packages.
- Before diving into package management, let's look at the naming convention used by the system to describe packages.

Package Parameters:



Methods of Installation:

- Two ways of Installations
  a) Standalone
  b) Network

(a) Standalone: In this type, we can install the packages through any removable media (or) through dump.

Installing and Removing Packages:

Syntax: ***rpm <options> <package name> --force --aid***

Options:

i	Installs a given package
v	Verbose output
h	Shows hash progress when installing
U	Upgrades a given package
e	Removes a given package
--force	Installs packages forcefully
--aid	Installs packages along with Dependencies
--nodeps	To erase an package without dependencies

Examples:

- For installing a package

```
rpm  -ivh  tftp-server-0.49-5.1.el6.i686.rpm  ↵
```

- For installing a package forcefully

```
rpm -ivh tftp-server-0.49-5.1.el6.i686.rpm  --force  ↵
```

- To upgrade a package

```
rpm  -Uvh  tftp-server-0.49-5.1.el6.i686.rpm  ↵
```

- To uninstall a package

```
rpm  -e  tftp-server   ↵
```

Query Options: Querying for packages using *“-q”*

Syntax: ***rpm   -q<options> <package name>***

Options:

c Lists all config files
d Lists all documentation files.
i Displays information about the package
l Lists the files in a package
s Status of the package

Verify options: Verifying for packages using ***“-V”***

Syntax: ***rpm   -V<options> <package name>***

Options:

a Queries all packages

f Displays information about the specified file.

Examples:

- For installing an application

```
rpm   -ivh   tftp-server-0.49-5.1.el6.i686.rpm    ↵
```

- For installing an application with forcefully

```
rpm   -ivh   tftp-server-0.49-5.1.el6.i686.rpm  --force   ↵
```

- If you want to upgrade this package (because you know that it is already installed), you can substitute the ***-i*** option for ***-u***.

```
rpm  -Uvh  tftp-server-0.49-5.1.el6.i686.rpm    ↵
```

- For uninstalling a package

```
rpm   -e  tftp-server    ↵
```

- You can always reinstall the package at a later date if you keep the ***.rpm*** file on your system.

```
rpm  -ivh  -replacepkgs  tftp-server-0.49-5.1.el6.i686.rpm    ↵
```

Tip: A common situation to run into on the job has to install a package that is already installed. You don’t have to go through the trouble of uninstalling the package first only to reinstall it. You can use the “***-replacepkgs”*** option alongside the regular install options to override an existing installation.

- Query the installed system package for ***tftp-server***

```
rpm  -qa | grep  tftp-server   ↵
```

- Query the information from the ***tftp-server*** package

```
rpm  -qi  tftp-server   ↵
```

- Suppose you are looking around on your new Red Hat installation and a file but can’t sure what it does. You can use the -f option to query the package that the file belongs to, possibly giving you a better idea of what that file might be used for

- Find out where the   “***/etc/syslog.conf*** ” file came form

```
rpm   -qf   /etc/syslog.conf   ↵
```

- Use the ***-c*** option to find all config files.

```
rpm  -qc  rsyslog   ↵
```

- To the find the documentation files for a given package.

```
rpm  -qd  rsyslog   ↵
```

- To listing of all files that came with the package.

```
rpm   -ql  rsyslog   ↵
```

- To find out whether a package has any dependencies.

```
rpm   -qR  rsyslog   ↵
```

Network Installation: In this level we can install the package from server through NFS (or) FTP services.

NFS Service:

- First check the communication.

```
ping  192.168.30.66   ↵
```

- For accessing the data shared form server

```
mount 192.168.30.66:/var/ftp/pub/Server  /mnt   ↵
cd   /mnt   ↵
rpm   -ivh  vsftpd*   --force   --aid   ↵
```

FTP Service: In this method to install the package the ftp server should have the dump of ***operating system*** under the ftp default sharable location.

- First check the communication.

```
ping   192.168.30.66   ↵
```

- For accessing the data shared from the server.

```
rpm  -ivh  ftp://192.168.30.66/pub/Server/<pkg-name>   --force
--aid   ↵
```

WORKING WITH YUM:

- YUM Stands for “***Yellowed Update Modifier*** ”.
- In this section, we look at the exact same tasks, except this time We use the more flexible YUM utility.
- The YUM command has access to repositories where tons of packages are kept and can install, upgrade, or remove them for you automatically.
- YUM also takes care of resolving and installing any dependencies for you, which the rpm command can’t do.
- YUM is an interactive tool which waits for the confirmation of a user.
- YUM is a default package management tool in Red Hat o/s
- YUM this tool we can install the required packages with dependencies.

Syntax: yum   <Options> <Command> <Package Name>

Options:

C	Specifies the location of the config file.
Q	Specifies quit, no output.
y	To always answer yes to prompts.
v	Provides verbose output.

Commands:

Clean	Removes cached data.
Erase	Removes a package from the systems.
List	Display available packages.

Install	Install a package on the systems.
Search	Enable you to search for a package.
Update	Updates a package.
grouplist	Displays available package groups.
groupinstall	Install a package with in a group.
groupremove	Remove a package with in a group.

YUM Server Configuration:

- Mount dvd

```
mount   /dev/cdrom   /mnt   ↵
```

- Install RPM package (vsftpd)

```
rpm  -ivh   vsftpd*   --force  --aid   ↵
```

- Copy the dump of o/s into the default sharable location ftp.

```
cp  -rvf   /mnt/*     /var/ftp/pub   ↵
```

- Install the createrepo package

```
rpm  -ivh createrepo*   --force  --aid   ↵
```

- Create the new Repository

```
createrepo  -g   /var/ftp/pub/server/repodata/comps-rhel5-
server-core.xml  /var/ftp/pub   ↵
```

Note: If any errors are showing then remove ***repodata***.

```
rm   -rf   /var/ftp/pub/repodata    ↵
```

- Open the YUM configuration file

```
 vi   /etc/yum.repos.d/linux.repo   ↵
[LINUX]
  name =Repository Server
  baseurl=ftp://192.168.30.66/pub
  enabled =1
   gpgcheck =0
 :wq!    ↵
```

- Restart the ftp service

```
service  vsftpd  restart   ↵

yum cleanall    ↵
```

- To list out package

```
yum  list   ↵
```

- To install the package

```
yum  install  postfix  -y   ↵
```

- You could also update the postfix package

```
yum  update  postfix  -y   ↵
```

- To remove the package

```
yum   remove  postfix    ↵
```

Note: One great feature about yum is that instead of updating a single package, you can list all updates that need to be installed for the systems.

```
yum  list  updates   ↵
```

- From this list, you can choose to update packages individually or as a whole. If you want to install all the updates.

```
yum  update  ↵
```

- You can install that "groups"

```
yum  groupinstall  "Development Tools"  ↵
```

Note: Don't forget that linux doesn't handle whitespace the way that Windows does. If you specify a group name, you need to enclose it in quotation marks (" ").

Searching For Packages:

- Find the postfix package to install.

```
yum  search  postfix  ↵
```

- To find out more information about the postfix package:

```
yum  info  postfix  ↵
```

- To flush the cache, do the following.

```
yum  cleanall  ↵
```

Configuring Additional Repositories:

- Sometimes you might want to install a package that isn't in the repositories that come preconfigured with Red Hat. If the package is available in someone else's repository, you can add that person's repository to your yum config File.
- You can either add your own custom repositories to that main config file "***/etc/yum.conf*"** **(or)** create a "***.repo*"** file in the "***/etc/yum.repos.d*"** directory which will be added automatically. Here is what a sample entry for a custom repository looks like

```
 vi   /etc/yum.repos.d/<filename>.repo   ↵
[unique title]
  name=my custom yum repository.
  baseurl=ftp://192.168.30.68/opt/yum/myrepos
  enabled=1
  gpgcheck=0
:wq!   ↵
```

- One neat trick that you can do is to create a repository based on an ISO.
- This trick can be useful in an environment whether there is no internet access and you'd like to install packages from the Red Hat DVD.

- create a folder for the temporary mount

```
mkdir   /mnt/cd  ↵
```

- Mount the ISO image (or) cd

```
mount  -o  loop   /dev/cdrom   /mnt/cd  ↵
```

Note: To create a ISO image

```
dd   if =/dev/scd0  of =/os/Rhel5.iso  ↵
```

├→Check it which device is mount.

- Read the image file.

```
mount  -o loop  /os/Rhel5.iso  /dvd  ↵
```

- create a repository in the temporary directory

```
cd   /mnt  ↵

createrepo  .  ↵
```

This creates an XML file of all the package from the mounted CD.

- you also need to clean the current repository cache:

```
yum   cleanall   ↵
```

- Create a ***".repo"*** file for your temporary repository

```
vi   /etc/yum.repos.d/iso.repo   ↵

 [ISO Repo]
name= My Repository
baseurl = file:///mnt/cd
enabled=1
gpgcheck=0
:wq!   ↵
```

Adding Your Custom Packages:

- Just as you have already created two different types of custom repositories, you can add packages you have created to them as well.
  - copy the file over to your private directory.

```
cp   /root/mysample-1.0-0.x86_64.rpm   /opt/yum/myrepos   ↵
```

  - Update the repository to recognize your new package

```
createrepo   -update   ↵
```

- Now you should be able to search for your package in your private repository along with other software.

Registering Your System: To register your system to the Red Hat Network, you must have an active subscription with Red Hat. You can register during the installation of your system or manually after the system has already been installed. Use the ***'rhn_register'*** command to begin the registration process. After you finish, you can visit http://rhn.redhat.com to start managing your system(s) through the web. You need to make sure that the '***rhnsd'*** daemon is running in order for it to be managed.

- Set the daemon to boot on system start

```
chkconfig  rhnsd  on   ↵
```

- You should verify whether the service is currently running

```
service   rhnsd   status   ↵
```

- If it is, you are all set; otherwise, start the service manually

```
service  rhnsd   start   ↵
```

KERNEL

The kernel is the main component, or the heart of an operating system. It controls all of the resources, timings, interrupts, memory allocation, process separation, error handling, and logging in the system. In a typical Linux computer, the kernel is modular, in that it has a core file (or files) and then loads the other device drivers as needed. In some cases, say an embedded device, the kernel may consist of one big image with all of the drivers it needs contained inside a file. This is known as a monolithic kernel.

- Kernel is responsible for interacting with hardware and producing output to the screen. There is also a virtual file system that gets created in the ***/proc*** directory to hold information and parameters for the kernel.
- Linux is truly just kernel. Red Hat and the other distributions in existence today are software and configuration files packaged with Linux kernel to bring you an entire operating system. Because kernel is really what runs everything, understanding how it works is essential.
- Kernel can be used to load new drivers, support new hardware, or even offer a custom kernel for individual needs.
- Linux kernel is modular, and because of this, you can load and unload kernel modules even after system has booted.
- Let's start with '***uname'*** command to find out same info about kernel.

uname: Display information about the kernel.

Syntax: uname [Options] ↵

Options:

-a	Prints all information relating to the kernel.
-s	Show the kernel name.
-r	Request kernel release information.
-v	Request the kernel version.

- Let's check and see which version of the kernel is currently running

```
uname   -a  ↵
```

o/p:-

```
Linux   redhat1  2.6.32-71.el6.x86_64 …………
```

Note: The kernel version numbering is important here.

- The first number **(2)** is the major version of the kernel.
- The second number is the major release of the first number. If the release number is even, which it is **(6),** it means that is a stable release of the kernel. Odd numbers are development kernels and should not be used for production systems.
- The third number is the patch version of the kernel.
- The last number **(71)** is added to Red Hat to represent its release version of the kernel.
- Also note the **el6**, which tells you are running Red Hat enterprise Linux6. If you couldn't tell, this is an **x64 –**bit version of the operating system.
- To get this version of the currently installed kernel

```
rpm  -qa  | grep  kernel  ↵
```

- You could also use the following :

```
rpm  -q     kernel  ↵
```

- When it comes to working with kernels, you should be familiar with different locations, let's look at four of these locations:

/boot	Place where the kernel and boot files are kept.
/proc	Current hardware configurations and status.
/usr/src	Source code of the kernel.
/lib/modules	Kernel modules.

lsmod: Lists currently loaded kernel modules.

Syntax: lsmod ↵

- To look at what is currently loaded by the kernel since you booted the systems, you use the following command.

```
            lsmod ↵
      o/p:-
      Module           Size    Used by
      nls_utf8         1005      1
      autofs4          21604     3
      sunrpc           197617    1
      ext4             322814    2
```

[Output truncated]

modinfo: Displays information about a kernel module

- To see detail information of ext4 kernel module listed previously

```
modinfo   ext4  ↵
modinfo   cdrom  ↵
```

To Find All The Kernel Modules:

- All the kernel modules will be residing in ***'/etc/lib/modules'*** directory.

```
cd /etc /lib/modules  ↵
ls  ↵
```

- To search all the kernel modules in the systems using find cmd.

```
find  /  -name  *.ko  ↵    (Modules in the system will be ending with
                              .ko extension)
```

modprobe: To remove the loaded module

Syntax: modprobe <module name> ↵

Examples:

```
modprobe   -r   vfat  ↵
```

- Now to check

```
lsmod    | grep -i vfat  ↵
```

- To install / re-install a module

```
modprobe   vfat  ↵
lsmod   | grep  -i  vfat  ↵
```

Updating Kernel:

- View the current version of the kernel

```
uname   -r  ↵
```

- To view kernel package information

```
yum  info  kernel  ↵
```

(or)

```
rpm    -qi   /grep kernel   ↵
```

- Using the package manager, you can upgrade the kernel to the latest version:

```
yum update  kernel    -y  ↵
```

(or)

```
rpm   -ivh  kernel -2.6.18-194.3.1el5  ↵
```

Note: When updating a kernel with the rpm command never use the -u option to update. The reason behind this is that the update option erases the prior kernel when updating, whereas the -i option installs the newer kernel alongside the old kernel. If something doesn't work or goes wrong, you have on older kernel to revert to.

Tuning Kernel With '*/proc/sys*':

- Kernel has a virtual file system, ***/proc/sys*** that allows you to tune the kernel while the system is running.
- Kernel creates ***/proc/sys*** virtual file system when the system boots up, which holds all the parameters of the kernel. This virtual file system is then used to manipulate kernel parameters for testing purposes (these changes are valid only until the system reboots).
- When you have kernel tuned the way you would like, you can simply have your settings applied when the system boots (through a special config file), or you can compile your own kernel to have them built in permanently.
- As you are testing kernel changes, make sure you don't rely on any settings made within the ***/proc/sys*** file system because they are erased when the system reboots. During testing you can use the echo cmd to change the values of the kernel while the system is running.
  - View the current value in the kernel

    ```
    cat   /proc/sys/net/ipv4/ip_forward   ↵
    ```

  - Change the kernel option that controls packet forwarding

    ```
    echo 1>/proc/sys/net/ipv4/ip_forward   ↵
    ```

  - Verify the value has changed

    ```
    cat   /proc/sys/net/ipv4/ip_forward   ↵
    ```

- If you want the changes to be persistent across system reboots you can put the parameters you'd like to remain during boot in the /etc/sysctl.conf file.

sysctl: Enable you to tune kernel parameters.

- Before you start tuning things let's look at all the available options

  ```
  sysctl  -a   ↵
  ```

- Now, let's make the same changes to the kernel as before.
  - Query the parameter responsible for forwarding packets within kernel.

    ```
    sysctl   -a   | grep ip_forward   ↵
    ```

  - using ***sysctl*** to change the option.

    ```
    sysctl   -w   net.ipv4.ip_forward=1   ↵
    ```

    (-w : Enables you to change a settings in the sysctl.conf file)

  - Verify the value has been changed:

    ```
    sysctl     -a | grep ip_forward   ↵
    ```

  - Return the parameter to its original value.

    ```
    sysctl   -w  net.ipv4.ip_forward=0   ↵
    ```

DISK PARTITIONING

- Disk partitioning is one of the many steps you must take when preparing a system for use.
- Partitioning means to divide a single hard drive into many logical drives.
- In each system the physical disk drivers are divided up logically into partitions that allow you to store data on them.
- There are also special types of partitions such as RAID that allow for increased performance, redundancy, or the both, LVM is an advanced form of partitioning that eases management of partitions as makes growing them for increased storage capacity simple.
- One of the key points to remember when working with partitions is to always plan ahead.

Disk partitioning criteria:

M B R	P	P	P	EXTENDED				
				L	L	L	……………	L

MRB	: Master Boot Record.
P	: Primary Partition.
EXTENDED	: Extended Partition.
L	: Logical Partitions.

- Every disk can have only 3 primary partitions.
- A primary partition is a partition which usually holds the operating system.
- Extended partition is a special type of primary partition which can be subdivided into multiple logical partitions.
- Logical partitions are the partitions which are created under extended partitions.
- As in the real world it is the results that matter. It doesn’t matter whether you use Disk Druid, fdisk, or parted to create partitions. You can create new partitions at the command line or use GUI front ends to these tools such as the disk utility.
- Remember disk druid is available only during the installation process.
- You can use two different utilities when partitions disks:

  fdisk ⟶ Disk partitioning utility
  parted ⟶ Another disk partitioning utility.
- While fdisk is more common, it is slowly being replaced by parted, which is more flexible.

Disk Identification:

- Different types of disks will be having different initials in Linux

  IDE drive will be shown as /***dev/had.***
  SCSI/SATA drive will be shown as ***/dev/sda.***
  Virtual drive will be shown as ***/dev/vda***

Note: The first two letters represent whether the disk is a ***SCSI (sd)*** **(or)** ***IDE (hd)*** disk. The third letter represents which disk it actually is if there is a number often the three letters, it is the number of the partitions.

- To view information about current partition layout,

  #cat /proc/partitions | grep hd ↵ for IDE.
  #cat /proc/partitions | grep sd ↵ for SCSI/SATA .
- You need to view their current partitions to see if any Exist.

fdisk:

Syntax: fdisk [options] [Device] ↵

Options:

-b	Specifies the sector of the disk.
-h	Number of heads on the disk.
-l	Lists current partitions table.

Note: There are some limitations when it comes to working with partitions you can have only four primary partitions to a physical disk with one exception. If you want to make more than the four, you need to create their primary partitions and one extended partition, although the primary partitions aren't required for extended partition creation. The extended partition can then hold || logical partitions (5-16) on it.

Creating a Partition:

```
fdisk   /dev/sda   ↵
```

- View all the options available to you

```
Command (m for help )  :m
     p          print the partition table.
     n          add a new  partition.
     d          delete a partition.
     m          print this menu.
     q          quit without saving changes.
     t          change  a partitions systems id.
     w          write table to disk and exit.
```

- Creating a new partitions

```
fdisk   /dev/sda   ↵

                 :n   ↵
First cylinder (1-1044, default 1):
Using default value 1
Last cylinder or +size ------ (1-1044, ----) +500M   ↵
```

- Create a second partition

```
                 :n   ↵
 ------
 ------
 ------
 ------
```

- Verify newly created partitions\

```
:p   ↵
```

- Write the changes to disk

```
:w   ↵
```

- Now that two new partitions have been created and written to disk, you should verify their existence. Before doing that however, you want the kernel to reread the partition table to make sure that it recognizes all disks and partitions correctly. To do this, you use the '***partprobe***' command.

partprobe:

Syntax: partprobe [options] [device] ↵

Options:

-d	Do not update the kernel.
-s	Prints a summary of contents.

- Now call the partprobe command.

```
#partprobe  /dev/sda   ↵
```

- Now that you have created partitions with the fdisk utility, do it again using the parted command.
  - Create a partition:

```
parted /dev/sda   ↵
Menu
(parted)help   ↵
```

  - Create your first partition in a similar manner to fdisk

```
(parted)mkpart   ↵
Partition type ? Primary /extended?
File system type?[ext2]?
Start?
End?
```

  - Make your second partition again.

```
(parted)mkpart   ↵
```

  - Before writing changes to disk, you should verify that they have been created the way you want them

```
(Parted) print   ↵
```

  - Exit the program to save your changes

```
(Parted) quit   ↵
```

Note: There are a few things you should notice here. First, you need to specify exactly where you want the start and end of the partition to be. If you don't plan this out ahead of time, you will end up with incorrect partition sizes. You should also take note of the fact that you don't have to write the changes to disk manually; this is done or you when you quit the pated program.

  - Again, you need to force the kernel to reread the partition table

```
partprobe   ↵
```

  - Once again, verify that your partitions have been created successfully:

```
parted   -l   ↵
```

Deleting a partition:

- Deleting a partition is much easier than creating one because you need to specify only the partition number that you want to delete.
  - Start the fdisk utility

```
fdisk   /dev/sda   ↵

   :p   ↵        printout the current partition.
   :d   ↵        delete a partition.
   :6   ↵        want to delete 6th partitions.
   :w   ↵        write changes to disk.
```

  - Don't forget to reread the partition table

```
partprobe   /dev/sda   ↵
```

  - Stand the parted utility

```
parted   /dev/sda   ↵
(parted) print     ↵
(parted) rm  5     ↵
(parted) quit      ↵
```

FILE SYSTEM

File system refers to the files and directories stored on a computer. A file s ys tem can have different formats called file system types. These formats determine how the information is stored as files and directories . Some file system types s tore redundant copies of the data, while some file system types make hard drive access faster. T his part discusses the ext3, swap, RAID, and LVM file system types . It also discusses the parted utility to manage partitions and access control lists (ACLs) to customize file permissions .

- File System is a data structure which organizes data in a structured format. Every partition on the disk except MBR and extended partition should be assigned with same file system in order to make them store the data.
- File system is applied on the partition by formatting it with a particular type of a file system.
- The number of file system types many exceed the number of operating systems. While RHEL can work many of these formats the default is ext4. While many users enable other file systems such as ReiserFs, RedHat may not support them.
- Before the partitions can be used, however you need to create a file system for each one.
- The default file system for **RHEL5** is ‘***ext3***’ and has been changed to ‘***ext4’*** for **RHEL6**. Both of these file systems offers a journaling option, which has two main advantages.
    i. It can help speed up recovery if there is a disk failure, because journaling File system keeps a “journal” of the file systems metadata.
    ii. It can check drivers faster during the system boot process.
- The ***journaling*** feature ***isn’t available*** on older file systems such as ***ext2***.
- The first Linux operating system used the Extended file system (ext) until the last few years, RedHat Linux operating systems formatted their partitions by default to the Seconded Extended file system (ext2). For RHEL5, the default was the third extended file system (ext3). The new default for RHEL6 is the fourth extended file system (ext4).
- Ext file System is the widely used file System in Linux, where as ***vfat*** is the file system to maintain storage between Linux and Windows (in case of multiple o/s).

ext2	ext3	ext4
1) Stands for Second Extend file System.	1) Stands for Third Extend file System.	1) Stands for Fourth Extend file System.
2) Introduced in 1993.	2) Introduced in 2001.	2) Introduced in 2008.
3) Doesn’t have journaling.	3) Support Journaling.	3) Support Journaling.
4) Maximum file size can be from **16GB** to **2TB.**	4) Maximum file size can be from **16GB** to **2TB.**	4) Maximum file size can be from **16GB** to **16TB.**
5) Maximum File System Size can be From **2TB** to **32TB.**	5) Maximum File System Size can be From **2TB** to **32TB.**	5) Maximum File System Size can be **1EB.**

- There are many Types of File systems.

Swap: The Linux swap file system is associated with dedicated swap partitions. You’ve probably created at least one swap partition when you installed RHEL.

Ms-Dos & Vfat: These file systems allows you to read MS-DOS formatted file systems. MS-DOS lets you read pre-windows 95 partitions, or regular windows partitions within the limits of short file names. VFAT lets you read windows 9x/NT/2000/vista/7 partitions formatted to the FAT16 or FAT32 File systems.

ISO 9660: The standard file system for CD-ROMs. It is also known is the High Sierra File System or HSFS, on the UNIX SYSTEMS.

/Proc: A Linux virtual File System. Virtual means that it doesn't occupy real disk space, Instead files are created as needed. Used to provide information on kernel configuration and device status.

/dev/pts: A Linux implementation of the Open Group's Unix98 PTY supports.

FS: IBM's journaled File System, commonly used on IBM Enterprise servers.

ReiserFS: The ReiserFS file system is resizable and supports fast journaling it's more efficient when most of the file are very small and very large .it's based on the concepts of "balanced trees" it is no longer supported by RHEL, or even by its formet main proponent ,SUSE.

Xfs: Developed by silicon Graphics as a journaling file system, it supports vary large file as of this writing, xfs file are limited to 9*10power18 bytes Don't confuse this file system with the x font server, both use the same Acronym.

NTFS: The current Microsoft Windows file System.

Creating a file system:

- When you've creating a file system, there are many different ways to complete the same task.
- They're all based on the mkfs command, which work as a front end to file system-specific commands such as mkfs.ext2, mkfs.ext3 and mkfs.ext4.

Syntax: mkfs [options] [device] ↵

Options:

-j	Creates a journal option.
-m	Specific a reserved percentage of blocks of blocks on a file system.

- There are two ways to apply formatting on a volume. For example, if you've just created a partition on ***sdb*** disk i.e., ***'/dev/sdb5'***

```
mkfs  -t  ext4  /dev/sdb5   ↵
mke2fs -t  ext4  /dev/sdb5   ↵
mkfs.ext4  /dev/sdb5   ↵
```

- If you want to reformat an existing partition, Logical Volume, or RAID array, take the following precautions.
  - Backup any existing data on the partition data on the partition.
  - Unmount the partition.
- You can format partitions, Logical Volume, and RAID arrays to other file systems. The options available in RHEL 6 include:

mkfs.cramfs	Creates a compressed ROM File System.
Mkfs.ext2	Formats a volume to the ext2 file system.
Mkfs.ext3	Formats a volume to the ext3 file system.
Mkfs.ext4	Formats a volume to the ext4 file system.

mkfs.msdos (or) mkfs.vfat (or) Mkdosfs Formats a partition to the Microsoft compatible VFAT file system, It does not create bootable File System.

mkfs.xfs Formats a volume to the xfs file system developed by the former silicon graphics.

Mkswap Formats a volume to the linux swap file system.

- One advantage of same rebuild distribution is the availability of useful package not supported by or available from RedHat. For example, cento 6 includes the ntfs progs package, which supports the mounting of NTFS Partitions.

Creating a swap: In Linux, a swap space is used as a ***"scratch space"*** for the system. When the system runs low on memory, it uses this swap as a virtual memory area to swap items in and out of physical memory. Although it should not be used in place of physical memory because it is much slower, it's critical piece of any system.

- There are two different types of swap that you can have:

(a) partition swap
(b) file swap

(a) **partition swap:**

- Create a partition

```
fdisk  /dev/sdb  ↵
```

- Update to kernel

```
partprobe  /dev/sdb  ↵
```

- Use the '***mkswap***' command to create a swap space

Syntax: mkswap [option] [device] ↵

Option:

-c Checks the device for bad blocks before creating the swap area.

Examples:

```
mkswap   /dev/sdc5  ↵
```

- Use the '***swapon***' command to enable a swap partition.

Syntax: swapon [options] [device] ↵

Options:

-a Enables all swap devices.
-e Silently skips devices that don't exit.
-s Verifies that swap is running.

Examples:

- Enable the swap partition

```
swapon   /dev/sdc5  ↵
```

- Verify the swap is running correctly

```
swapon   -s  ↵
```

- Use the '***swapoff***' command to disable a swap partition.

Syntax: swapoff [options] [device] ↵

Options:

-a	Enables all swap devices.
-e	Silently skips devices that don't exit.
-s	Verifies that swap is running.

Examples:

- Enable the swap partition

```
swapoff   /dev/sdc5   ↵
```

(b) file swap:

- You can use the dd command to reserved space for another swap on the '***/dev/sdc6'*** partition.
- The dd command can be used for many different purposes and has a huge syntax
  - Reserve 1GB of space for the swap

```
dd  if =/dev/zero  of=/mnt/file_swap  bs=1024  count=1000000   ↵
```

  - Just as with partition swaps, you can create a swap space specifying the device file just created

```
mkswap     /mnt/file_swap   ↵
```

  - Enable the swap

```
swapon   /mnt/file_swap   ↵
```

  - Again you can verify that the swap is enabled

```
swapon  -s   ↵
```

Note: The big difference between the two swap type is that file swap is easier to manage because you can just move the swap file to another disk if you want. The swap partition would need to be removed, re-created, and so on. Although Redhat recommends using a partition swap, file swaps are fast enough these days with less administrative overhead to not use them instead. One word of caution, tough, is that you can use only one swap (of either type) per physical disk.

Mounting a file System:

- After formatting a partition we cannot add the data into the partition. In order to add the data in the partition is it required to be mounted.
- Mounting is a procedure where we attach a directory to the file system.
- They can be mounted to any directory, which is referred to as a mount point .Every mount point before is a directory.
- If you mount a file system on a directory that is not empty everything within that directory becomes inaccessible. Therefore, you should create a new directory as a mount point for each of your file systems.
- There are only two commands for mounting a file system:

mount ⟶ Mounts a file system.
umount ⟶ Unmounts a file system.

  - Start by going to the /opt directory where you can make some directories to serve as a mount points.

```
cd /opt   ↵

mkdir   abc   ↵
mkdir  backup   ↵
```

Syntax: mount [options] [device] [mountpoint] ↵

Options:

-r	Mount as read-only.
-w	Mounts as read/write (default).
-l LABEL	Mounts the file system with the name LABEL.
-v	Provides verbose output.

- Mount the two file Systems

```
mount  /dev/sdc5  /opt/abc   ↵
mount  /dev/sdb6  /opt/backup   ↵
```

Note: Don’t specify a file system type or any mount option .the reason is that the mount command automatically detects the file system type and mounted with the defaults option **(rw)**.

- To unmount a file systems.

Syntax: umount [options] [mountpoint] ↵
Options:

-f	Force unmounts.
-v	Provides verbose output.

- You can use the “fuser and lsof” commands to check for open files and users that are currently using files on a file system

Syntax: fuser [option] [mountpoint/file system] ↵
Options:

-c	Checks the mounted file system.
-k	Kills processes using the file system.
-m	Shows all processes using the file system.
-u	Displays user id’s.
-v	Verbose output.

- Checks to see what user are currently using the file system.

```
fuser  -cu  /dev/sdc6   ↵
```

(or)

```
lsof   /dev/sdc6   ↵
```

- To kill the open Connections , you can use the fuser cmd again

```
fuser  -ck  /opt/backup   ↵
```

- Now you should be able to unmount the file system

```
umount   /opt/backup   ↵
```

- Now you know how to mount and unmount file system, but there is something else you need to look at. If you reboot your system right now, all the file systems that you just mounted are not persistent so anything that is mounted with mount command will no longer be available across system reboots. If suppose you want to know how to fix that?
- The system looks at two config files

/etc/mtab	Contains a list of all currently mounted file system.
/etc/fstab	Mounts all listed file system with give option at boot time.

- View the ***/etc/mtab*** file

```
cat  /etc/mtab   ↵
```

- Every time you mount or unmount a file system this file is updated to always reflect what is currently mounted on the system.

- You can also query to check whether a partition file system is mounted.

```
cat  /etc/mtab | grep backup   ↵
```

- You can use the mount command with no options to also view the currently mounted file systems:

```
mount   ↵
```

- Go through the /etc/fstab file.

Syntax:*<device> <Mountpoint> <file system type> <Mount options> <Write data during shutdown> <check sequence>*

- View the /etc/fstab file

```
cat   /etc/fstab   ↵
```

- The first three fields should be fairly obvious because you have been working with them throughout the chapter.
- The fourth field defines the option that you can use to mount the file system.
- The fifth field defines whether data should be backup (also called dumping) before a system shutdown or reboot occurs. This field commonly use a value of 1.A value of 0 might be used if the file system is temporary storage space for files such as /tem.
- The last field defines the order in which file system checking should take place. For root file system the value should be 1 everything else should be 2 if you have a removable file system (CD-ROM or External)
- you can define a value of 0 and skip the checking altogether.
- If you want file system created earlier to be mounted when the system boots, you can add two definitions for them here.

- Open the /etc/fstab file for editing

```
vi    /etc/fstab   ↵
/dev/sdc6     /opt/backup     ext3     defaults   0  0
:wq!    ↵
```

- You can use the mount command with -a option to remount all file systems defines in the /etc/fstab file.

```
mount -a    ↵
```

Extra File System Commands:

Label: Labels you to determine a specific file system more easily with a common name, instead of /dev/sdc6.An added benefit is the system's being able to keep its label even if the underlying disk is switched with a new one.

- Take you file system offline

```
umount  /dev/sda6   ↵
```

- Let's Label the file system data to denote that it's the Company _data file System.

```
e2label    /dev/sdc6   abc_data   ↵
```

- You can use the same command to verify

```
e2label    /dev/sdc6   ↵
```

- Find the file system you just labeled

Syntax: findfs LABEL=<label> | UUID=<uuid> ↵

```
findfs  LABEL=abc_data   ↵
```

- You can also query more information about the device using the ***blkid*** command.

Syntax: **blkid [options]** ↵

Options:

-s	Shows specification tag(s).
dev	Specifies the device to probe.

- Combine the blkid cmd with grep for specific results.

```
blkid | grep  abc_data   ↵
```

- When you finish your maintenance you can remount the file system with the new Label instead of the device path

```
mount   LABEL=abc_data   /opt/abc   ↵
```

- You could even update the /etc/fstab file to use the label information instead of the device path.

```
vi   /etc/fstab   ↵
LABEL=abc_data    /opt/abc    ext3    defaults   0  0
:wq!    ↵
```

- You can use the ***mount*** command to verify the label names

```
mount    -l   ↵
```

- You also can use the ***df*** command to view the usage information for your file system.

Syntax: df [options] ↵

Options:

-h	Specifies human-readable format
-l	Local file system only
-T	Point the File system type

Examples:

```
df    -h   ↵
df    -l   ↵
df    -Th   ↵
```

Managing File System Quotas:

- Quotas are used to restrict the amount of disk space occupied by user or groups.
- Quotas regulates disk consumption of users. It improves system performance.
- Quotas are two types
  - **i.** User level
  - **ii.** Group level
- If we apply quotas on a group level it will effected to only the primary users of that group.
- Quotas can be applied only quotas enabled partitions.
- You need to install the required packages before you can use quotas on your system.
  - To install the Quota package

```
yum    install   –y    quota   ↵
```

  - Verify that the package was installed successfully

```
rpm   -qa | grep  quota   ↵
```

  - You can query quota support from the kernel with the following command.

```
grep  -i  config_quota   /boot/config- 2.6.32-71.el6.i686   ↵
```

- Now that you have a listing of the commands you can use, you first need to edit the /etc/fstab file to specify which file systems you want to utilize quotas.
  - Open the /etc/fstab file edit the following line

```
vi   /etc/fstab
         /dev/sdc5    /opt/abc_data  ext3   defaults, usrquota, grpquota
:wq!    ↵
```

- Now you need to remount the /opt/abc_data file system before the changes take effect.

- You can accomplish this by using the mount command:

```
mount  -o  remount    /opt/abc_data  ↵
```

- You can verify that the mount and quota options took correctly.

```
mount | grep abc_data  ↵
```

- There are two files that maintain quotas for users and groups.

aquota.users	User’s quota file
aquota.group	Group quota file

- These two files are automatically created in the top level directory of the file system where you are turning on quotas-in this case, the ***/opt/abc_data*** file system.
  - To start the quota system, you use the quotacheck cmd

Syntax: quotacheck [options]   [partition] ↵

Options:

-c	Don’t read existing quota files.
-u	Checks only user’s quotas.
-g	Checks only group quotas.
-m	Doesn’t remount the file system as read-only.
-v	Provides verbose output.

```
quotacheck   -ugm   /opt/abc_data  ↵
```

- To verify that the quota files were created successfully

```
ls    /opt/abc_data  ↵
```

Enabling Quotas: Normally, you would have to call the quota on and quota off cmds to have the quota system enforced, but they are automatically called when the system boots up and shuts down.

- Run the cmd manually the first time just to make sure that quota, turned on :

```
quotaon  -v  /opt/abc_data  ↵
```

- Lets briefly discuss the two different limits you can have when dealing with quotas

Soft limit: Has a grace period that acts as an alarm, signaling when you are reaching your limit. If your grace period expires, you are required to delete files until you are once again under your limit. If you don’t specify a grace period, the soft limit is the maximum number of files you can have.

Hard limit: Required only when a grace period exists for soft limit if the hard limit dose exist, it is the maximum limit that you can hit before your grace period expires on the soft limit.

- To work with quotas for users and groups, you need to do some conversions in your head here. Each block is equal to 1kb. (1,000 kb =1 mb)
  - Set the limits for user1 by using the edquota cmd.

Syntax: edquota   [-u/-g]   [username/groupname] ↵

```
edquota    -u    user1  ↵
Filesystem         blocks     soft        hard     inodes    soft      hard
 /dev/sdc6            0         2000       2500        0         0          0
```

- Again, you use the edquota cmd, but with a different option

```
edquota    -t   ↵
```

- Here, the current value is seven days for the block grace period. You should not give your users that much time to get their act together, so drop that limit to two days.

TIP: The Edquota and offers a pretty cool feature. After you configure a Quota and your limits for a single user you can actually copy this over to other users as if it were a template. To do this specify the user you want to use as a template first call the ***edquota*** cmd with the ***-p*** option.

```
edquota  -up  users1 user2   ↵
```

Quota Usage Reports:

Syntax: repquota [options] [partitions] ↵

Options:

-a	Reports on all non-NFS file Systems with Quotas turned on.
-u	Reports on user quotas.
-g	Reports on group quotas.
-v	Verbose output.

Example:

```
repquota   -uv   /opt/abc_data   ↵
```

File System Security: Linux, like most operating systems has a standard set of file permissions. A side from these, it also has a more refined set of permissions implemented through access control Lists.

- This section covers both of these topics how they are used to implement file system security for files directories and more.
  - Installing the required package

    ```
    yum   install   -y   acl   ↵
    ```

  - Verify the package installation

    ```
    rpm   -qa | grep  acl   ↵
    ```

  - Before you can use ACL's however, you need to make sure that the file system has been mounted with ACL parameter

    ```
    mount  | grep acl   ↵
    ```

  - You can accomplish this using the following

    ```
    mount  -t ext3 -o  acl  /dev/sdc6   /opt/backup   ↵
    ```

  - If your file system isn't already mounted, you could also use the following.

    ```
    mount   -t  ext3  -0 acl   /dev/sdc6  /opt/backup   ↵
    ```

  - To verify you can use the previous cmd.

    ```
    mount  |  grep  acl   ↵
    ```

  - Adjust the following line in your ***/etc/fstab*** file

    ```
    vi    /etc/fstab   ↵
    /dev/sdc6   /opt/backup   ext3   defaults, acl   1   2
    :wq!    ↵
    ```

  - To make the changes take effect, you need to remount the file system.

    ```
    mount   -o  remount   /opt/backup   ↵
    ```

  - Now verify that your file system has the ACL Option

    ```
    mount  | grep  -i  acl   ↵
    ```

- The file system is now mounted properly with the ACL option, so can start to look at the management cmds that pertain to ACL's.

getfacl	Obtains the ACL from a file or directory
setfacl	Sets or modifies an ACL.

  - Create sample file on which you can test an ACL in the /opt/backup.

    ```
    cd   /opt/backup   ↵
    touch   file1   ↵
    ```

- Now you can use the getfacl cmd to view the ACL currently associated with file

Syntax: getfacl [options] file ↵

Options:

-d Displays the defaults ACL.

-R Recourses into subdirectories.

Example:

```
getfacl file1 ↵
```

Syntax: setfacl [options] file ↵

Options:

-m Modifies an ACL.

-x Removes an ACL.

-r Doesn’t recalculate the mask.

-R Recourses into subdirectories.

Examples:

- Set the test file so that user1 also has access to this file

  ```
  setfacl   -m   u:user1:rwx   /opt/backup/file1 ↵
  ```

- To check the ACL permission again:

  ```
  getfacl file1 ↵
  ```

- To remove the ACL for user1:

  ```
  setfacl   -x    u:user1    /opt/backup/file1 ↵
  ```

- Verify the ACL has been removed:

  ```
  getfacl   file1 ↵
  ```

- If you have multiple ACL set up on a single file, you can remove then all with the ‘***-b***’ option instead of removing them one by one

  ```
  setfacl   -b   file ↵
  ```

- File permissions and ACL’S can get really complex if they aren’t throughout ahead of time.

RAID

- RAID means Redundant Array of independent Disk.
- RAID partitions allows for more advanced features such as redundancy and better performance.
- There are two types of RAID'S
  - **i.** Hardware raid
  - **ii.** Software raid
- While RAID can be implemented at the hardware level, the Red Hat exams are not hardware based and therefore focus on the software implementation of RAID through the MD driver.
- Before we describe how to implement RAID, let's look at the different Types of Raids

RAID0: (Striping)

- Disks are grouped together to from one large drive. This offers better performance at the cost of availability. If any single disk in the RAID fail; the entire set of disks becomes unusable.
  - Minimum 2, max 32 hard disks
  - Data is written alternatively
  - No fault tolerance.
  - Read & write speed is fast.

RAID1: (Mirroring)

- Disks are copied from one to another, allowing for redundancy. If one disk fail, the other disk takes over, having an exact copy of data from the original disks.
  - Min 2,max 32 hard disks
  - Data is written simultaneously
  - Fault tolerance available
  - Read fast, write slow

RAID5: (Striping with Parity)

- Disks are similar to RAID0 and are joining together to from one large drive. The difference here is that 25% of the disk is used for a parity bit, which allows the disks to be recovered should a single disk fail.
  - Min 3, max 32 hard disks
  - Data is written alternatively
  - Parity is written on all disks
  - Read & Write speed is fast.
  - Fault tolerance is available.

Implementation of RAID 5:

- Install the following package

```
yum   install   -y   mdadm   ↵
```

- Verify the install

```
rpm   -qa  | grep   mdadm   ↵
```

- To start you first need to create partitions on the disk you want to use. You start with a RAID5 setup, so you need to make partitions on at least three different disks.

Creating a Raid Array:

- Create three partitions

```
fdisk    /dev/sda   ↵
```

- To verify when you're done

```
fdisk   -l   ↵
```

- Now you can begin to set up the RAID 5 array with the three partitions.

Syntax: mdadm   [options]   ↵

Options:

-a	Add a disk into a current array.
-C	Creates a new RAID array.
-D	Prints the details of array.
-f	Fails a disk in the array.
-l	Specifies level of RAID array to create.
-n	Specifies the devices in the RAID array.
-S	Stops an array.
-A	Active an array.
-v	Provides verbose output.

- Create Raid5 using below command

```
mdadm  --create  /dev/md0 --level=5  --raid-devices=3  /dev/sdb1
 /dev/sdb2   /dev/sdb3     ↵
```

- Again to verify that RAID array has been created successfully.

```
mdadm   -D   /dev/md0   ↵
```

- View the status of the newly created RAID array:

```
cat     /proc/mdstat   ↵
```

- This output shows that have an active RAID5 array with three disks in it .the last few lines here show the state of each disk and partition in the RAID array. You can also see that the RAID is in “Recovery” mode or creating itself.
  - If you wait the estimated 2.9 minutes and then query again you see the following

```
cat    /proc/mdstat   ↵
```

- You now see that the RAID is good to go as it has finished building itself

What to Do When a Disk Fails:

- Suppose that a disk in the array failed. In that case, you need to remove that disk from the array and replace it with a working one.
  - Manually fail a disk in the array.

```
mdadm  /dev/md0 -f   /dev/sdb3   ↵
```

- Verify that the disk in the array has failed

```
mdadm   -D    /dev/md0   ↵
```

- To remove a disk from the array

```
mdadm    /dev/md0   -r    /dev/sdb3   ↵
```

- Look at the last few lines of the RAID details again

```
mdadm   -D   /dev/md0   ↵
```

- If you want, you could combine the previous two commands.

```
mdadm  -v  /dev/md0   -f    /dev/sdb3   -r    /dev/sdb3   ↵
```

- You can add new disk back to the array

```
mdadm   /dev/md0    -a   /dev/sdb4   ↵
```

- Verify that it has been added properly

```
mdadm    -D   /dev/md0   ↵
```

- Query the kernel

```
cat     /proc/mdstat   ↵
```

- Should something go seriously wrong and you need to take RAID array offline completely.

```
mdadm   -vs   /dev/md0   ↵
```

Deleting a RAID Array:

- TO delete an array, first stop it

```
mdadm    -vs    /dev/md0   ↵
```

- Then remove the RAID array device

```
mdadm    -r     /dev/md0
```

LOGICAL VOLUME MANAGER (LVM)

LVM is a tool for logical volume management which is used to allocating disks, striping, mirroring and resizing logical volumes. With LVM, a hard drive or set of hard drives is allocated to one or more physical volumes. LVM physical volumes can be placed on other block devices which might span two or more disks. Since a physical volume cannot span over multiple drives, to span over more than one drive, create one or more physical volumes per drive. The volume groups can be divided into logical volumes, which are assigned mount points, such as /home and / and file system types, such as ext2 or ext3 or ext4. When "partitions" reach their full capacity, free space from the volume group can be added to the logical volume to increase the size of the partition. When a new hard drive is added to the system, it can be added to the volume group, and partitions that are logical volumes can be increased in size.

On the other hand, if a system is partitioned with the ext4 file system, the hard drive is divided into partitions of defined sizes. If a partition becomes full, it is not easy to expand the size of the partition. Even if the partition is moved to another hard drive, the original hard drive space has to be reallocated as a different partition or not used.

- LVM is a form of advanced partition management. The benefit to using LVM is ease of management due to the way disks are setup.
- LVM is the method of allocating hard drive space in to logical volumes that can be easily resize of partition with LVM, the hard drive **(or)** set of hard drive are allocated to one **(or)** more physical volumes.
- The physical volumes are combined into volume groups such volume group is divided into logical volumes which are assigned mount points as "/home" ,'/' etc. These logical volumes are formatted to exits file systems.
- The LVM must follow the bellow sequence

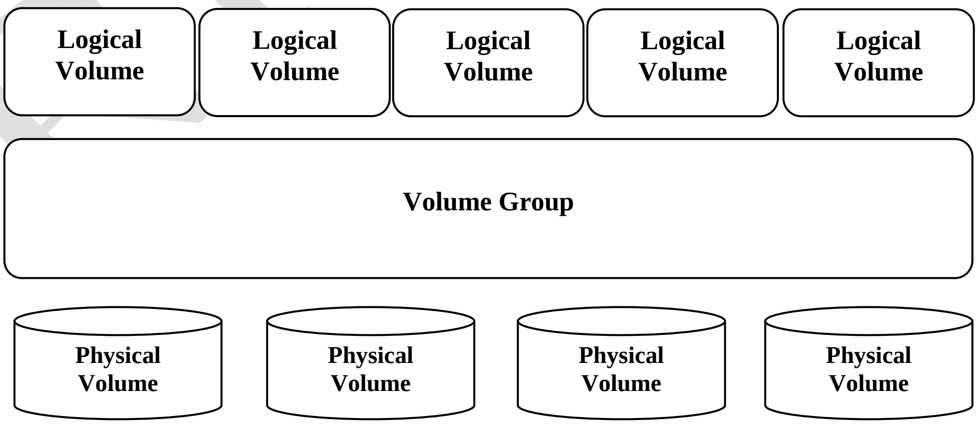
1. Physical volume.
2. Volume group.
3. Logical volume.

1. Physical volume: The collections of individual physical drives are called as physical volumes.

2. Volume group: It is a collection of physical volumes and assigns a name through which we can create logical volumes.

3. Logical volume: The logical volumes are specified from the volume group. These are logical partitions which can resize, format, mount etc.

LVM STRUCTURE



Implementation of LVM:

- Install the required packages

```
yum   install   -y   lvm*   ↵
```

- Verify that it is installed

```
rpm  -qa   | grep  -i  lvm   ↵
```

- Creating an LVM partitions (four partitions)

```
fdisk      /dev/sdb   ↵
```

- To update to kernel for rereading

```
partprobe     /dev/sdb   ↵
```

Creating an LVM Partition:

- To create physical volumes:

```
pvcreate    /dev/sdb{6,7,8,9}   ↵
```

- Verify that the physical volume was created successfully:

```
pvdisplay       ↵
     (or)
     pvs   ↵
```

Creating Volume Group:

- To create Volume Group

```
vgcreate   vg1   /dev/sdb{6,7,8,9}   ↵
```

- Verify that the volume group was created successfully

```
vgdisplay    -v   vg1    ↵
```

- When volume groups are created and initialized. The physical volumes are broken down into physical extends (The unit of measurement for LVM). This is significant because you can adjust the size of how data is stored based on the size of each physical extend defined when the volume group is created (Default value of PE is **4 MB).**

Creating Logical Volumes:

- To create logical volumes, use the lvcreate command and specify the size of the partition that you'd like to create. The size can be specified in kilobytes, megabytes, gigabytes, or logical extents (LE).
- Like physical extents, logical extents are a unit of measure when dealing with logical volumes.

- Create a partition of 2GB size.

```
lvcreate   -L  2000   vg1   -n   lv1   ↵
```

- To verify logical volume info

```
lvdisplay       ↵
   (or)
   lvs   ↵
```

- To create one more logical volume

```
lvcreate   -L  3000   vg1  -n   lv2   ↵
```

- Using the lvrename command you can change the name of a logical partition.

```
lvrename    /dev/vg1/lv2    /dev/vg1/lvol2   ↵
```

- Verify with the following cmd

```
lvdisplay   ↵
```

Adjusting the Size Of LVM Partitions:

- The single best feature of LVM is that you can reduce or expand your logical volumes and volume groups. If you are running out of room on a particular logical volume (or) volume group, you can add another physical volume to the volume group and then expand and logical volume to give you more room.

- Add **2GB** more to the **lv1** logical volume.

```
lvextend    -L   +2000     /dev/vg1/lv1
```

- Verify the change with the following cmd

```
lvdisplay   lv1     ↵
```

- To decrease a logical volume

```
lvresize      -L   2000     /dev/vg1/lv1   ↵
                   (or)
lvresize      -L   -2000    /dev/vg1/lv1   ↵
```

- Suppose you want to add a new physical volume so that you can extend your volume group.
  - Create a new physical volume some where

```
pvcreate    /dev/sdb10   ↵
```

  - Now extended your volume group to incorporate the new physical volume.

```
vgextend   vg1   /dev/sdb10   ↵
```

  - Now verify the detail of the newly increased vg

```
vgdisplay  -v   vg1    ↵
```

- To reduce the volume group to no longer include the physical volume /dev/sda15, you can use the vgreduce command

```
vgreduce   vg1   /dev/sdb10   ↵
```

  - Now verify reduction of volume group

```
vgdisplay   vg1   ↵
```

  - Now create File system on lv1

```
mkfs.ext4    /dev/vg1/lv1   ↵
```

  - Mount it on /mnt directory.

```
mount    /dev/vg1/lv1   /mnt   ↵
```

Note: Use the ***resize2fs*** cmd to extend the file system. Before extending the file system, however you should always ensure the integrity of the file system first with e2fsck cmd.

Syntax: e2fsck [options] [Device] ↵

Options:

-p	Automatically repairs.
-n	Makes no changes to the file system
-y	Assumes "yes" to all questions
-f	Force checking of the file system
-v	Provides verbose output

Example:

- Check the file system

```
e2fsck  -f   /dev/vg1/lv1   ↵
```

Syntax: resize2fs [options] [device] ↵

Options:

-p	Points percentage as task completes

-f Force the cmd to proceed.

Example:

- Extend the underlying logical volume

```
lvextend  -L  3000  /dev/vg1/lv1   ↵
```

- Now you can extend the file system

```
resize2fs   -p   /dev/vg1/lv1   ↵
```

- Now that your maintenance is complete, remount the file system:

```
mount   /dev/vg1/lv1   /mnt   ↵
```

- You can use the mount cmd to verify it mounted it mounted successfully.

```
mount   ↵
```

- Now you can use the df cmd to view the usage information for your file system. This should also reflect the additional space that you just added to the /mnt file system.

Syntax: df [options] ↵

Options:

```
-h        specifies human-readable format
-T        prints the file system type
```

Example:

```
df   -h   ↵
```

Deleting an Lvm Partition:

- If just as important to understand how to delete LVM partitions is it to create them. This is a common task when you are upgrading or redesigning a file system layout.
- **Note:** Make sure you back up any data before deleting anything within the LVM structure.

- To remove a logical volume

```
lvremove   /dev/vg1/lv1   ↵
lvremove   /dev/vg1/lvol2   ↵
```

- To remove the volume group

```
vgremove    vg1   ↵
```

- You can also do both steps in one cmd by using the **-f** option.

```
vgremove   -f   vg1   ↵
```

- Wipe all current Physical Volumes

```
pvremove   /dev/sdb6   ↵
pvremove   /dev/sdb7   ↵
pvremove   /dev/sdb8   ↵
pvremove   /dev/sdb9   ↵
pvremove   /dev/sdb10   ↵
```

SYSTEM INITIALIZATION

Boot Process:

Boot process consists the set of processes from powering on the pc to login prompt comes.

- When a computer boots up, the BIOS is the first program that is run. After it is loaded, the BIOS begins it test the system through the power on self-test(POST) and then starts loading peripheral devices. The BIOS then looks for the boot device and passes control it. The boot device contains the master boot record (MBR), which starts to boot the system via the boot loader. From here, the grand unified boot loader (GRUB) looks to boot into the kernel that is labeled as the default. Finally, the kernel calls the init process, which boots up the rest of the system.
- The GRUB has become the default boot loader for Red Hat, Ubuntu, and many other versions of Linux as well.
- When GRUB loads, you are given a list of kernels and additional operating systems from which you can choose to boot.
- By default, there is configurable 5-second time cut value that chooses the default kernel if you don't make a selection and the timeout threshold is reached. After GRUB loads the kernel, it passes control over to the kernel, which it run begins to initialize and configure the computer's hardware.
- During the boot process, everything is logged to the /var/log/dmesg file. You can also use the dmesg and to query information about the boot process. After the system has booted.
- When the system's drives are in place, the kernel executes the /sbin/init program.
- In RHEL6, the boot process has been replaced by anew utility called uistart instead of the traditional sysVInit style scripts. This utility decreases the time that it takes the system to boot and is already currently being used on other versions of Linux such as Ubuntu.
- The init program is the first process created by the kernel. It is responsible for the rest of the boot process and setting up the environment for the user.

- First , it consults the /etc/inittab file, which define how the rest of the boot process will go. The /etc/inittab file lists the default run level to boot into and the initialization script (/etc/re.d/rc.sysinit).

- Let's look at the etc/inittab file to see what init process goes through.

```
#cat   etc /init tab
```

- You can see that the default run level is set to 5, although six different run levels corn. /etc/inittab file also defines how to handle power failures and virtual terminals. After in the process is done consulting the /etc/rc.d/rc.sysinit script is run, which handles setting the system clock net eorking setting up the user environment & more.
- In redhat linux, the default run level is 5. This default run level is passed to the / etc rc.d/rcscript, which calls all the programs in the /etc/rc.d/rc#.d directory.
- The last thing that you see is the long prompt . if you have a screen where you to the system; other wise, you see a text mode iogin.

- **Working with grub:** the grub boot loader is broken into different stges. The code contained on the MBR is considered GRUB stage 1.it loads GRUB stage 1.5, which tries to identify the file system type (optional), or it can call Grub stage 2 directly. Stage 2 is what call the kernel and loads it in to memory. In stage GRUB need to search the MBR looking for an active partitions form which to boot the kernel. GRUB has it own format for looking through hand disks.

- The syntax of this format is

  (xdn[,m]) where Xd is the drive
  N is the number of the disk
  M denotes the partition number.

- The syntax is very useful when trouble shooting issues with GRUB because you need to know GRUB searches for disk drives when trying to find the primary partition. When the primary partition found GRUB loads the kernel, which is where you more on to stage 2. Stage 2.is the place where you will tend to speed the most time troubleshooting boot issues with the system it present you e hat a list of kernel's that you can boot from, along with a listing of options that you can use to modify the parameters passed to the kernel during boot up

- **GRUB BOOT OPTIONS:**

e	Edit the ands before booting
a	Modify (or) append the kernel argument before booting
c	open the GRUB and line.

- You can use 'a' option to modify any parameter you want to pass to the kernel. This includes changing the run level that the system will boot into
- After choosing the 'a' option you can pass mode as a parameter to enter into the mode. Here are the different modes that you can boot into.

Single user mode	perform maintenance tasks (or) forget the root pass wore
Run level 2 (or) 3	load only partial services during the boot process
Emergency mode	used to perform tasks on an unbootable system
Rescue mode	used to fix issues (or) reinstall GRUB…

The config file:-

GRUB has only a single config file /boot/grub/grub.confg.Two other files actually have soft links to this main config file as well:
/boot/grub/menu.lst and /etc/grub. Config when GRUB starts ,it reads its configuration from the main config file

```
#cat /boot/grub/grub. config
```

using the c option to enter the and line,you can make changes to the configfile,reinstall GRUB or repair a broken config file

The GRUB command line:

How to repair a broken MBR.

you need to make use of the rescue environment, which can be found by booting from the RHEL installation DVD. When you are in the rescue environment,you,can repair your broken MBR.
step 1: Load up the GRUB and line to find the disk and partition that contain the grub.config file using the find cmd :

```
grub>find /grub/grub.cong
        (hdo,0)
```

you could also run:

```
grub>root
        (hdo,0)
```

step 2: Install GRUB on the drive is rerurned:

```
grub>step (hdo)
```

Now that your MBR is fired you should be able to boot the system once again

Runlevel l:

1).when the system boots up, it queries for the default runlevel, which is defined in the /etc/init tab file. when the default runlevel is located,the system boobs into the paricular run level

2.There are sex run levels in total, which are shown in the /etc/inittab file. Each runlevel also has as directory called etc/rc.d/rc.d#. where # is the runlevel (from 1 to 6)
3 let's look at the different run levels

0. Halt
1. Single user mode
2. Multiuser with partial services
3. Full multiuser with networking (text mode)
4. Not used
5. Pull multiuser graphical mode(gui desktop login)
6. reboot

4.The easiest run levels to understand are 0 and 6 these two run levels are called by the same and with different input in runlevel 0,essentially the system is off in runlevel 6 the system is restarting runlevel 1 is used to enter single user mode which you would enter if there are issues with the system and you'd like to perform maintenances you can also reset the root user's password in this runlevel. The remaining run levels provide various states in different services to run it
5.Don't forget the upstart is the program that actually starts & stops services at eactrunlevelnow.The /etc/init/rc.config file shows how each set of scripts is called

```
#cat /etc/init/rc.conf
```

Run level Utilities:
Let’s now look at the many system utilities that help you manage the system in different runlevel these management cmds are

Syn: shutdown [options] time

```
-k    doesn't shutdown; rust warns
-h    halts the system after shutdown
r     reboots instead of turning off the system
-f    forces a file system check on reboot
-n    kills all process quickly (not recommended)
-t    SECS stands a shutdown message but delays shutdown by x sec
```

ex:

```
#shutdown  -h now
#shutdown  -r now
# reboot
#shutdown  -in 120   delay the shutdown by 2 minutes
```

Halt :powers down the system.
to turn of the system #shutdown now (or) #halt
Power off: works the same as the halt cmd.
to check the current runlevel #runlevel
(or)
#who -r
to change runlevel3 #init 3

BACK UP & RESTORE

- In information technology, a backup or the process of backing up is making copies of data which may be used to restore the original after a data loss event.
- Back up have two distinct purposes.
- The primary purpose is to recover data after its loss, be it by data deletion or corruption. Data loss is a very common experience of computer users. 67% of internet users have suffered serious data loss.
- The secondary purpose of backup's is to recover data from an earlier time, according to a user defined data retention policy, typically configured with in a backup application for how long copies of data are required.
- Backup is the most important job of a system administrator, as a system admin it is your duty to take backup of the data every day.
- Many companies have gone out of the market because of poor backup planning.
- The easiest way to backup your first is just copying. But if your have too many files to backup, copying and restoring may take too long time and it is not convenient. If there is a tool that can put many files into one file, the world will be better fortunately, 'tar' is used to create archive files.

Compression and Archiving:

- As you will learn when you become a system Administrator, backups are the number one priority.
- If something should crash or become corrupt and you can't restore it because you aren't keeping up with your backups or you just don't keep any, you may be looking for a new job. Although we don't address backup programs here, this is good lead into archiving and compression.

tar: Tar means tape archiving.

- It is used for compressing and archiving files and directories.
- A more common use for tar is to simply combine a few files into a single file, for any storage distribution.

Syntax: tar [options] [file]

Options:

-c → Creators a new archive.
-v → Provides verbose output.
-f → Specifies the archive file to use.
-t → Lists the files in an archive.
-x → Extract the backup.
-z → Zipping.

Example:

- **Create some random blank files**

```
touch  file1  file2  file3  abc1   ↵
```

- **Create a simple archive containing these files**

```
tar  –cvf  sample.tar  file1  file2  file3  abc1   ↵
```

- When an archive is created you can also apply compression to reduce the amount of space the archive files takes up. Although multiple types of compression are supported with the use of tar, we look only at gunzip (.gz) and bzip2 (bz2) here.

- **Let's re-create the archive using the gunzip compression**

```
tar  –cvzf  sample.tar.gz  file1  file2  file3  abc1   ↵
```

- **View the current directory to see the two current archive files**

```
ls   ↵
```

- **To see all the contents within the build file**

```
tar  - tvf   sample.tar   ↵
```

- **Now extract this build file verbosely**

```
tar  - xvf   sample. tar   ↵
```

- **To extract files on different location**

```
tar  –xvf  sample.tar   -c  /root/   ↵
```

cpio:cpio is a tool for creating and extracting archives or copying files from one place to another.

- It handles a number of cpio formats as well as reading and writing tar files.
- cpio like tar but can read input from the ***"find"*** command.

Syntax: find   –name   file   **|** cpio [options] [controller] <Destination>

Options:

-o   (out)
-i   (in)

Controllers:

O (or) **>** -out
I (or) < -in

- **To take the backup files**

```
ls  file * | cpio –acvf  >  /root/bkp.cpio   ↵
```

- **To see the backup content**

```
cpio  –it < /root/bkp.cpio   ↵
cpio  –it  –I  /root/bkp.cpio   ↵
```

- **To restore the backup file**

```
cpio  –icuvd < /root/bkp.cpio   ↵
```

o → Reads the standard input
i → Extract files from the standard input
c → Read or write header information in ASCII character
u → Copy unconditionally (older file will not replace a new file)

dd: dd means Disk to Disk

- Used to take the backup of one partition to another, here source partition should be given to "if", destination partition should be passed to "of".

- **To take the backup**

```
dd   if=/dev/sdb6   of=/dev/sdb7   ↵
```

- **To recovery**

```
dd  if=/dev/sdb7   of=/dev/sdb6   ↵
```

scp: scp means Secure Copy

- SCP is used to copy data from one Unix or Linux System to another Unix or Linux Server.
- SCP uses secured shell(ssh) to transfer the data between the remote hosts.
- The features of SCP are:
  - Copies files with in the same machine.
  - Copies files from local machine to remote machine.
  - Copies files from remote machine to local machine.

- Copies files between two different remote servers.

Syntax: scp [options] [user from_Host: source_file] [user to_Host : Destination_file]

Options:

-r ⟶ Recursively
-q ⟶ Progress bar not displayed
-v ⟶ Verbose mode
-p ⟶ Copy files using the specified port number.

- **Copy file from local host to remote server:**

  scp filename root@redhat5.rootshell.com:/root ↵

- **Copy files from remote host to local server:**

  scp root@redhat5.rootshell.com:/root/backup* **.** ↵

- **Current working directory**

- **Copying a directory:**

  scp -r directory root@redhat5.rootshell.com:/root ↵

Improving performance of scp command:

- Using blowfish or arcfour encryption will improve the performance of the scp command

  scp -c blowfish filename root@redhat5.rootshell.com: **.** ↵

- Specifying the port number

  scp -p 6001 bkp_file root@redhat5.rootshell.com:/tmp ↵

NETWORKING

- One of the key elements of connecting to different systems in the network configuration involved.
- A network is a set of hardware devices connected together either physically or logically to allow them to exchange information.
- Network management is fairly easy when it comes to Red Hat.
- Most of the network configuration is kept in files; therefore, adjusting these settings is simple.
- In the network system maintains the fully qualified domain name.

```
FQDN=Host name + Domain name
```

Let’s start by looking at the information about host name & Networking:

- To Check the hostname

```
hostname ↵
```

- To managing hostname temporarily

```
hostname   redhat1 ↵
```

- To managing permanently

```
vi   /etc/sysconfig/network ↵
        Networking = yes
        Hostname    = redhat1.rootshell.com
        Networking_IPV6=no
```

- For avoiding graphical problems

```
vi  /etc/hosts ↵
        192.168.30.61    redhat1    redhat1.rootshell.com
        :wq!
```

To manage IP Address:

- To Check IP Address:

Syntax: ifconfig   [options]   [interface]

Options: netmask
up
down

Example:

```
ifconfig eth0 ↵
```

- To display all interface on the system

```
ifconfig ↵
```

- To change ip address temporarily:

Syntax: ***ifconfig*** ***eth0*** <ipaddress> ***netmask*** <subnetmask> <up/down>

```
ifconfig eth0 192.168.30.61 netmask 255.255.255.0 up ↵
```

- To verify

```
ifconfig  eth0 ↵
```

- To change permanently:

  i) setup ↵
  ii) system-config-network-tui ↵

- You could also check the output of the interface config file

```
vi   /etc/sysconfig/network-scripts/ifcfg-eth0   ↵
     DEVICE=eth0
     HWADDR=00:0c:29:5b:14:4d
     ONBOOT=yes
     IPADDR=192.168.30.61
     BOOTPROTO=none
     NETMASK=255.255.255.0
     TYPE=Ethernet
     GATEWAY=192.168.30.254
     :wq!   ↵
```

- To check the interface detected (or) not:

Syntax: ethtool <interface>

```
ethtool   eth0   ↵
```

- To bring the single interface down:

```
ifdown   eth0   ↵
```

- To restore the interface that you just brought down:

```
ifup  eth0   ↵
```

Note: Any time you make a change to an interface's settings you need to bring down that interface and then bring it up again.

Warning: Restarting the network service interrupts all network connections and any client that is currently connected .

- Restart the network service as follows

Syntax: service <***service name***> <***stop | start | restart | status***>

```
service  network  restart   ↵
```

- To check the all service status

```
service  --status–all   ↵
```

- To manage permanently:

Syntax: chkconfig --level <***run levels***> <***service name***> <***on*** | ***off***>

```
chkconfig  network  on   ↵
```

- To check the status of the service:

```
chkconfig   --list   network   ↵
```

- To disable the service at boot time:

```
chkconfig   network   off   ↵
```

Routing: When you have a system that has two or more network interface, they are called dual bomed (or) multibomed systems you need to make sure that each interface has a gateway that it can route through.

Syntax: route [options]

Options:

add	Add a net route
del	Deleted an existing route
flush	Flushes any temporary routes

- Let's look at the current routes on the system

```
route   ↵
```

- To set default gateway

```
route   add   default  gw  192.168.30.254 eth0   ↵
```

- To Verify

```
route   ↵
```

Networking Utilities:

ping: Tests the connectivity between two hosts.

```
ping   192.168.30.62   ↵
```

- When you ping something on a Linux host, unlike in windows, the ping continues until you cancel it.
- You can limit the number of ping requests sent by prefixing "***–c" number_count*** in front of the destination host.

```
ping  -c  3  192.168.30.62   ↵
```

netstat: Shows information about connections (open, closed and listening)

- netstat command used to obtain information or routing tables, listening sockets and established connections.

Syntax: netstat [options]

Options:

-r	Displays the routing table
-I	Displays interface statistics
-t	Shows top connections
-u	shows udp connections
-a	Displays all sockets(tcp ,udp or local)
-p	Displays process ID's
-e	Displays Extended information.

- To check that the connection is available for your clients:

```
netstat  - tuape  | grep  ssh   ↵
```

Client DNS Troubleshooting:

/etc/sysconfig/network	:	Contains the hostname of the system.
/etc/hosts	:	Contains the local IP to hostname mapping
/etc/resolv.conf	:	Contains the IP Address of the DNS servers
nslookup	:	queries (or) looks up a domain name or system.
Ping	:	Test connectivity between two hosts.

Ethernet Bonding:

Ethernet bonding is used to combine multiple interfaces into one creating an increase in available bandwidth and redundancy. This is done by creating a special network interface file called as a channel bonding interface . (i.e.; ***bond0***)

Lab:

Change Directory and List:

```
[root@redhat2 /]# cd   /etc/sysconfig/network-scripts/
```

```
[root@redhat2 network-scripts]# ls
                ifcfg-eth0
                ifcfg-eth1
                ifcfg-lo
```

Create "Bond0" File and give entries as below

```
[root@redhat2 network-scripts]# cat > ifcfg-bond0
              DEVICE=bond0
              IPADDR=192.168.30.62
              NETMASK=255.255.255.0
              GATEWAY=192.168.30.254
              ONBOOT=yes
```

Now Edit "ifcfg-eth0" and "ifcfg-eth1" files

```
[root@redhat2 network-scripts]# vi   ifcfg-eth0
```

```
                    # Intel Corporation 82545EM Gigabit Ethernet Controller (Copper)
                      DEVICE=eth0
                      HWADDR=00:0C:29:0E:1D:A7
                      MASTER=bond0
                      SLAVE=yes
                      ONBOOT=yes
                      TYPE=Ethernet

[root@redhat2 network-scripts]# vi   ifcfg-eth1
                   # Intel Corporation 82545EM Gigabit Ethernet Controller (Copper)
                      DEVICE=eth1
                      HWADDR=00:0C:29:0E:1D:B1
                      ONBOOT=yes
                      MASTER=bond0
                      SLAVE=yes
                      TYPE=Ethernet
```

List

```
[root@redhat2 network-scripts]# ls
                     ifcfg-bond0
                     ifcfg-eth0
                     ifcfg-eth1
                     ifcfg-lo
```

Load Bond Driver

Change Directory and List

```
[root@redhat2 /]# cd   /etc/modprobe.d/
[root@redhat2 modprobe.d]# ls
                    blacklist    blacklist-compat    blacklist-firewire    modprobe.conf.dist
```

Create "bonding.conf" File

```
[root@redhat2 modprobe.d]# cat>bonding.conf
                    alias  bond0  bonding
                     options  bond0  mode=0  milmon=100

                 here mode=0---->Balance Round Robin
                      mode=1---->Active Backup
                      mode=2---->Balance XOR
                      mode=3---->BroadCast
                      mode=4---->803.3ad
                      mode=5---->balance-tlb
                      mode=6---->balance-alb
[root@redhat2 modprobe.d]# modprobe  bonding
```

Restart Network

```
[root@redhat2 ~]# service   network   restart

                    Shutting down interface bond0:                      [  OK  ]
                     Shutting down loopback interface:                  [  OK  ]
                     Bringing up loopback interface:                   [  OK  ]
                     Bringing up interface bond0:                      [  OK  ]
```

Check Status Of bond0,etho,eth1

```
[root@redhat2 /]# ifconfig   -a

                 bond0     Link encap:Ethernet  HWaddr 00:0C:29:0E:1D:A7
                       inet addr:192.168.30.62  Bcast:192.168.30.255  Mask:255.255.255.0
                       inet6 addr: fe80::20c:29ff:fe0e:1da7/64 Scope:Link
                       UP BROADCAST RUNNING MASTER MULTICAST  MTU:1500  Metric:1
                       RX packets:24963 errors:0 dropped:0 overruns:0 frame:0
```

```
          TX packets:35253 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1924578 (1.8 MiB)  TX bytes:25062291 (23.9 MiB)
-------------------------------------------------------
eth0      Link encap:Ethernet  HWaddr 00:0C:29:0E:1D:A7
          UP BROADCAST RUNNING SLAVE MULTICAST  MTU:1500  Metric:1
          RX packets:24472 errors:0 dropped:0 overruns:0 frame:0
          TX packets:35139 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1876882 (1.7 MiB)  TX bytes:25047945 (23.8 MiB)
-------------------------------------------------------
eth1      Link encap:Ethernet  HWaddr 00:0C:29:0E:1D:A7
          UP BROADCAST RUNNING SLAVE MULTICAST  MTU:1500  Metric:1
          RX packets:491 errors:0 dropped:0 overruns:0 frame:0
          TX packets:116 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:47696 (46.5 KiB)  TX bytes:14670 (14.3 KiB)
-------------------------------------------------------
```

Now Manually bring down eth0 interface and check status of all inerfaces

```
[root@redhat2 /]# ifdown  eth0
[root@redhat2 /]# ifconfig  -a

bond0     Link encap:Ethernet  HWaddr 00:0C:29:0E:1D:A7
          inet addr:192.168.30.62  Bcast:192.168.30.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe0e:1da7/64 Scope:Link
          UP BROADCAST RUNNING MASTER MULTICAST  MTU:1500  Metric:1
          RX packets:639 errors:0 dropped:0 overruns:0 frame:0
          TX packets:150 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:57646 (56.2 KiB)  TX bytes:18626 (18.1 KiB)
--------------------------------------------------------
eth0      Link encap:Ethernet  HWaddr 00:0C:29:0E:1D:A7
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:24513 errors:0 dropped:0 overruns:0 frame:0
          TX packets:35163 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1880228 (1.7 MiB)  TX bytes:25051613 (23.8 MiB)
--------------------------------------------------------
eth1      Link encap:Ethernet  HWaddr 00:0C:29:0E:1D:A7

UP BROADCAST RUNNING SLAVE MULTICAST  MTU:1500  Metric:1
          RX packets:639 errors:0 dropped:0 overruns:0 frame:0
          TX packets:152 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:57646 (56.2 KiB)  TX bytes:18950 (18.5 KiB)
-------------------------------------------------
```

Now Manually Bring up eth0 and Bring down eth1 interface

```
[root@redhat2 /]# ifup  eth0
[root@redhat2 /]# ifdown  eth1
```

Now Check The Status Of all Interfaces

```
[root@redhat2 /]# ifconfig   -a

bond0     Link encap:Ethernet  HWaddr 00:0C:29:0E:1D:A7
          inet addr:192.168.30.62  Bcast:192.168.30.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe0e:1da7/64 Scope:Link
          UP BROADCAST RUNNING MASTER MULTICAST  MTU:1500  Metric:1
          RX packets:24644 errors:0 dropped:0 overruns:0 frame:0
          TX packets:35198 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1889146 (1.8 MiB)  TX bytes:25055419 (23.8 MiB)
---------------------------------------------------------
eth0      Link encap:Ethernet  HWaddr 00:0C:29:0E:1D:A7
          UP BROADCAST RUNNING SLAVE MULTICAST  MTU:1500  Metric:1
          RX packets:24644 errors:0 dropped:0 overruns:0 frame:0
          TX packets:35199 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1889146 (1.8 MiB)  TX bytes:25055653 (23.8 MiB)
---------------------------------------------------------
eth1      Link encap:Ethernet  HWaddr 00:0C:29:0E:1D:B1
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:758 errors:0 dropped:0 overruns:0 frame:0
          TX packets:180 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:65752 (64.2 KiB)  TX bytes:24178 (23.6 KiB)
---------------------------------------------------------
```

Bring up eth1 also

```
[root@redhat2 /]# ifup  eth1
```

Bonding Kernel Module stored in "/proc/net/bonding/bond0"

NETWORK ADMINISTRATION

- A network is a set of hardware devices connected together, either physically or logically to allow them to exchange information.
- In the network system maintains the FQDN(Fully Qualified Domain Name)

FQDN = Host Name + Domain Name

To change the Host Name:

To check the hostname

```
hostname ↵
```

- **To change the Host Name Temporarily**

```
hostname   redhat4.rootshell.com ↵
```

- **To change the Host Name Permanently**

```
vi   /etc/sysconfig/network ↵
Networking = yes
Host Name = redhat4.rootshell.com
:wq! ↵
```

- **For avoiding graphical problems**

```
vi   /etc/hosts ↵
192.168.30.64    hostname   redhat4
(redhat4.rootshell.com)
```

To change the IP Address:

- **To check the IP Address**

```
ifconfig ↵
```

- **To change the IP Address Temporarily**

  Syntax: ifconfig eth0 <ipaddress> netmask <default submask> up ↵

  Example:

```
ifconfig eth0 192.168.30.70 netmask 255.255.255.0 up ↵
```

- **To change the IP Address Permanently**

  For GUI: (i) neat ↵

  (iii) system-config-network ↵

  (iv) **For CLI:** (i) setup ↵

  (ii) neat-tui ↵

  (iii) system-config-network-tui ↵

- **To check the NIC Card detected (or) not:**

  Syntax: ethtool <ethernet card> ↵

  Example:

```
ethtool  eth0 ↵
```

To Manage The Services:

- **To Manage The Services Temporarily:**

  Syntax: service <service name > <stop/start/restart> ↵

  Example:

```
service vsftpd   restart ↵
```

- **To check the particular service status:**

```
service vsftpd  status ↵
```

- **To Check the all services status:**

```
service   --status-all ↵
```

- **To Manage The Services Permanently:**

Syntax: chkconfig --level <run levels> <service name> <on/off> ↵

Example:

```
chkconfig  --level 235  vsftpd   on   ↵
```

- **To check the particular service status:**

  Syntax: chkconfig --list <service name>

  Example:

  ```
  chkconfig  --list  vsftpd   ↵
  ```

- **To Check the all services status:**

  ```
  chkconfig  --list   ↵
  ```

- **To see the port numbers of all services:**

  ```
  vi   /etc/services   ↵
  ```

File Transfer Protocol –FTP

- FTP Protocol is used to download and upload the files over the internet (or)
- It is standard method for sharing the files over the internet for many years
- FTP servers are the most common way to make directories of documents and software available to the public over the internet

Types of FTP Servers:

1. FTP : By default it is available in Solaris
2. Pro-FTP : For anonymous logins it is a third party tool
3. SFTP : Secure FTP
4. VSFTP : Very secure FTP
5. WU-FTP : Washington FTP
6. Cute FTP

- In Linux by default SFTP,VSFTP are available

Requirements:

Packages	:	vsftp……rpm
Port No	:	20 – FTP Data Transfer , 21 – FTP Control Connection
Config File	:	/etc/vsftpd/vsftpd.config
Sharing loc	:	/var/ftp
Service	:	vsftpd
Daemon	:	vsftpd

Configuration:

a) Install the package : # rpm –ivh vsftpd * --force --aid **(or)**

: # yum install vsftpd * -y

b) Testing to see if vsftpd is running: # netstat -a |grep ftp

c) Open configuration file : # vi /etc/vsftpd/vsftpd.conf

- By default anonymous users are enable. If you want to disable login permission to anonymous users then **anonymous_enable = no [# in line no : 12]**
- Once logged in to VSFTP server ,you will automatically have access to only the default anonymous ftp directory **/var/ftp** and all its sub directories .

➢ Go to sharing location :- # cd /var/ftp

#mkdir upload [for uploading files]

➢ Create some files in pub directory to download :- # cd pub (enter)

# touch f1 f2 f3 (enter)

- Restart the service:- # service vsftpd restart (enter)

  #chkconfig vsftpd on (enter)
- Now go to client side we can check it:

Accessing the private users: open the config file

- To disable the anonymous users : # vi/etc/vsftpd/vsftpd.conf
  - local_enable = yes (line no:15)
  - anon_upload.enable = yes (line no:27)
  - ftpd_banner = welcome to ftp (line no:85)

- Create the Local Users:
  - #useradd ftp1 #passwd ftp1
  - #useradd ftp2 #passwd ftp2

- Create a user group and shared directory. In this case we will use " **/home/ftp-users**" and a user group name of "**ftp-users**" for the remote users
  - #groupadd ftp-users
  - #mkdir /home/ftp-docs

- Make the Directory accessible to the **ftp-users** group
  - #chmod 750 /home/ftp-docs
  - #chown root: ftp-users /home/ftp-docs

- Add users and make their default directory as /home/ftp-docs
  - #useradd –g ftp-users –d /home/ftp-docs user1
  - #useradd –g ftp-users –d /home/ftp-docs user2
  - #passwd user1
  - #passwd user2
- Change the permissions of the files in the /home/ftp-docs directory for read only access by the group
  - #chown root:ftp-users /home/ftp-docs/*
  - #chmod 740 /home/ftp-docs/*

- Now users should be able to login via ftp to the server using their new user name and passwords
- If you don't want any FTP users to be able to write to any directory then you should comment out the **write_enable** line in you config file
  - #write_enable = yes line no:18
  - Now restart the service : #service vsftpd restart

Restricting local user:

- If you want to restrict any normal users to login into the FTP server them mention their name in the file.
  - #vi /etc/vsftd/ftpusers
    - ftp1 {restrict ftp1}
    - user1 {restrict user1}
  - Accessing ftp service by specified users
    - #vi /etc/vsftpd/user-list

        - ftp2
        - user2
    - open the config file
        - #vi /etc/vsftpd/vsftpd.conf
            - Go to the last line – userlist_deny=no
- FTP Users can't login telnet
    - #usermod –s /sbin/nologin ftp1
    - #usermod –s /sbin/nologin user1

Now telnet users should ………

Client side:

- To connect FTP server : #ftp <ftp server ip>
    - Example : #ftp 192.168.1.254
    - Type user name and password

FTP Prompt Commands:

- ftp>pwd -> display server side directory (present working directory)
- ftp>!pwd -> display client side working directory
- ftp>ls -> list the server side info
- ftp>!ls -> list the client side info
- ftp>cd -> change directory at server side
- ftp>!cd ->change directory at client side
- ftp>get <filename> -> to download single file
- ftp>mgt -> to download multiple files
- ftp>put -> to upload a single file
- ftp>mput ->to upload multiple files
- ftp>help ->display the prompt commands
- ftp>bye -> to quit ftp prompt

-----------------x---------------

SCP –Secure copying

- SCP is a command which copies/pushes the files into remote locations
- SCP is available as a part of secure shell package that is normally installed by default on Redhat
- There is a Windows SCP client called WinSCP
- Secure copy is installed in parallel with SSH and that will always run simultaneously on the same tcp port
- SCP does not support anonymous downloads like FTP

- To pull the file:
    - #scp 192.168.1.254:/root/file1 .
- To Push the file:
    - #scp file1 192.168.1.254:/root/desktop

Create ISO Image of your OS

- Insert the DVD:
- Mount the DVD: #mount /dev/dvd/mnt
- To create image: #dd if-/dev/scd0 of=/os/Rhe/5.iso
- Read the image file: #mount –o loop /os/Rhel5.iso /mnt

#cd /mnt #ls

NETWORK FILE SYSTEM (NFS)

- The network file system is certainly one of the most widely used network services.
- NFS is based on the remote procedure call.
- It allows the client to automount and transparently access the remote file systems on the network.
- Using NFS we can share the files among homogenous environment. Hence we can't share the files between Unix to Windows environment.
- NFS was developed by Sun Micro Systems.
- NFS was developed to allow the user to access remote directory as a mapped directory.
- NFS is not a single program, It is a suite of related programs.

Features:

- Everyone can access same data this eliminates the need to have a redundant data on server systems.
- Reduces storage cost.
- Provides data consistency.
- Reduces the system administrator overhead.
- Data Transfer rate in 3.0 is 32kb bit but in 2.0 it is only 8kb.

Prerequisites:

(i) Verify NFS daemon Service: Here we assume that the NFS Service daemon is already on our system, including portmap daemon on which NFS setup depends. One more thing, our system needs to support the NFS file system. For this we needs to issue the following command:

```
cat   /proc/filesystems   ↵
```

- **Another way to check NFS Functioning**

```
rpcinfo   -p   ↵
```

(ii) Server export file:

- **All NFS Server Exports need to be defined in "*/etc/exports*" file.**

Most Common Exports Options:

i. **/home/nfs/   192.168.30.65(rw,sync):** Export /home/nfs directory for host with IP 192.168.30.65 with read, write permissions and synchronize mode.

ii. **/home/nfs/ 192.168.30.65(ro,sync):** Export /home/nfs directory for network 192.168.30.65 with read only permissions and synchronized mode.

iii. **/home/nfs/   192.168.30.65(rw,sync,no_root_squash):** Export /home/nfs directory for host with IP 192.168.30.65 with read, write permissions synchronized mode and the remote root user will be treated as a root and will be able to change any file and directory.

iv. **/home/nfs   *(ro,sync):** Export /home/nfs directory for any host with a read only   permission and synchronized mode.

v. **/home/nfs   *.rootshell.com(rw,sync):** Export /home/nfs directory for any host with in rrootshell.com domain with read, write permissions and synchronized mode.

Requirements:

Packages:

nfs *…..rpm
portmap*…rpm

Port no:

2049 - NFS
111   - PORT MAP

Config File:

/etc/exports

Service:

portmap and NFS

Daemon:

nfsd, mountd, statd, locked

Server side Configuration:

Install the packages:

```
rpm  –ivh  nfs*  portmap*  --force  -aid  ↵
        (or)
yum  install  nfs*  portmap*  -y  ↵
```

Create the Directory for sharing:

```
mkdir  /home/nfs  ↵
cd   /home/nfs  ↵
touch  abc{1..9}  ↵    creating files
```

Open the Config file:

```
vi  /etc/exports  ↵
/home/nfs   192.168.30.65(rw,sync)
/home/nfs   192.168.30.66 (ro,sync)
/home/nfs   192.168.30.66 (rw,sync,no_root_squash)
/home/nfs    *(ro,sync)
/home/nfs    *.rootshell.com(ro,sync)
:wq!  ↵
```

Restart the service:

```
service  nfs  restart  ↵
service  portmap  restart  ↵
```

- **To make a permanent of this service**

```
chkconfig  nfslock  on  ↵
chkconfig  nfs  on  ↵
```

Client Side Configuration:

To check the NFS server sharing information:

Syntax: showmount -e <nfs server ip address>

Example: showmount -e 192.168.30.64 ↵

Create the local mount point:

```
mkdir  /nfsclient  ↵
```

Now mount to the server sharing information:

```
mount  192.168.30.64:/home/nfs  /nfsclient  ↵

cd   /nfsclient  ↵

ls  ↵
```

Limitations of NFS:

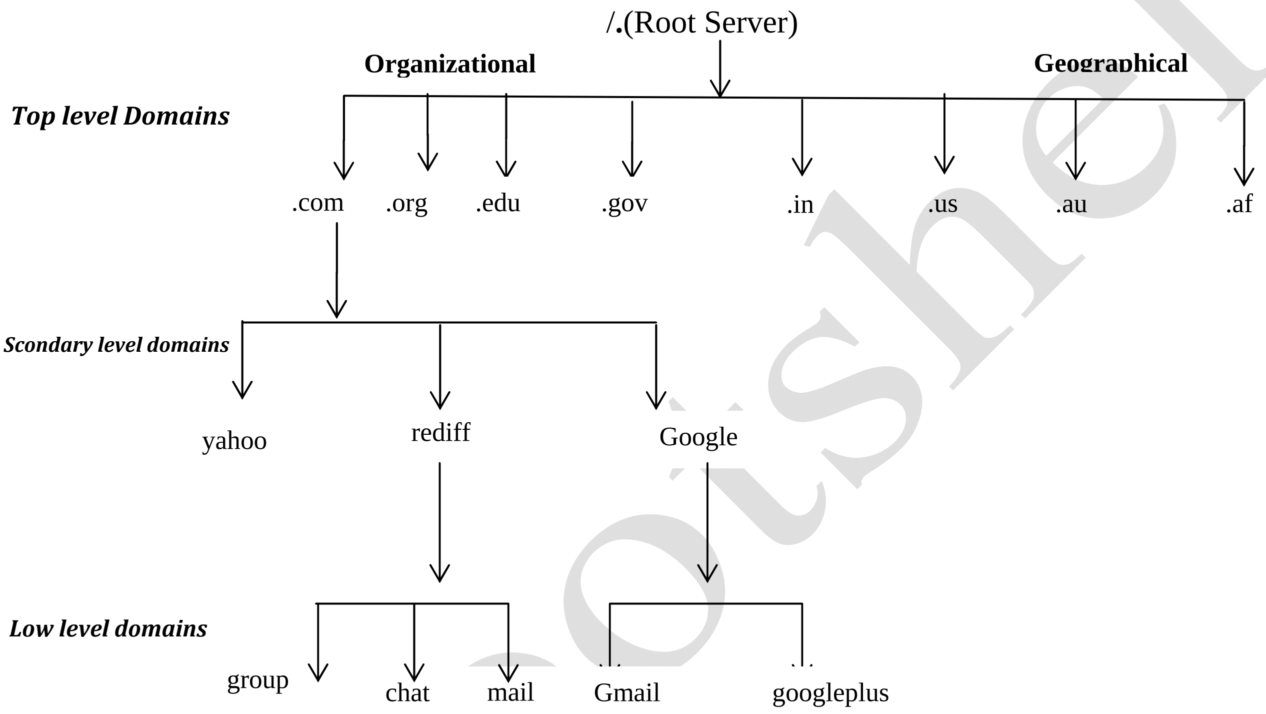
- we can only exports the directories which are started with ***"/"*** .
- We can Exports the files only to the homogeneous environment.
- Only local file systems can be exported.
- It uses only in private network.
- ***/var/lib/nfs/rmtab*** → This file contains exported list which are mounted by list.
- ***/var/lib/nfs/etab*** → This file contains currently exported file system information.

Domain Naming System – DNS

Domain Name System (DNS) is the system that associates hostnames with IP addresses. Thanks to DNS, users and administrators don't have to remember the IP addresses of computers to which they want to connect but can do so just by entering a name, such as www.Rrootshell.com. In this section, you'll learn how DNS is organized.

- Domain naming system is the way in which a URL (or) domain like www.rrootshell.com is converted into an IP Address
- DNS Provides resolution of Names to IP address and IP address to Names

Define a hierarchical name space where each level of the name space is separated by " **.** "



- Top level domains classified into two types
  1. Organizational : Based on purpose (or) function of the domain
  2. Geographical : Based on physical location
- All the root servers are maintained by IANA
- Top level domains are maintained by Inter NIC
- Secondary level domain are maintained by own levels like Yahoo, rediff etc

The DNS Lookup Process

To get information from a DNS server, a client computer is configured with a DNS resolver.This is the configuration that tells the client which DNS server to use. If the client computer is a Linux machine, the DNS resolver is in the configuration file /etc/resolv.conf. When a client needs to get information from DNS, it will always contact the name server that is configured in the DNS resolver to request that information. Because each DNS server is part of the worldwide DNS hierarchy, each DNS server should be able to handle client requests. In the DNS resolver, more than one name server is often configured to handle cases where the first DNS server in the list is not available. Let's assume that a client is in the rrootshell.com domain and wants to get the resource record for www.rrootshell.com. The following will occur:

1. When the request arrives at the name server of example.com, this name server will check its cache. If it has recently found the requested resource record, the name server will issue a recursive answer from cache, and nothing else needs to be done.
2. If the name server cannot answer the request from cache, it will first check whether a forwarder has been configured. A forwarder is a DNS name server to which requests are forwarded that cannot be answered by the local DNS server. For example, this can be the name server of a provider that serves many zones and that has a large DNS cache.
3. If no forwarder has been configured, the DNS server will resolve the name step-bystep. In the first step, it will contact the name servers of the DNS root domain to find out how to reach the name servers of the .fr domain.
4. After finding out which name servers are responsible for the .fr domain, the local DNS server, which still acts on behalf of the client that issued the original request,contacts a name server of the .fr domain to find out which name server to contact toobtain information about the sander domain.
5. After finding the name server that is authoritative for the sander.fr domain, the name server can then request the resource record it needs. It will cache this resource recordand send the answer back to the client.

Host File:

- It provides resolution of hostnames to IP address
- It can only resolve the name provided in the local host file
- You can add the name and IP address in /etc/hosts file but we should have to maintain it all the systems in the network

# /etc/resolve.conf :

This file is used by DNS clients to determine both the location of their DNS server and the domains to which they belong .

(a) **Name server :** Ip address of your DNS name server , there should be only one entry per "name server". If there is more than one name server , you will need to have multiple "name server" lines.

(b) **Domain:** The local domain name to be used by default .if the server is station254.rrootshell.com , then the entry would just be rrootshell.com

Ex: # name server 192.168.1.254

Host File:

- It provides resolution of hostnames to IP address
- It can only resolve the name provided in the local host file.
- You can add the name and IP address in /etc/hosts file but we should have to maintain it all the systems in the network.

/**etc**/**resolv.conf :** This file is used by DNS clients to determine both the location of their DNS server and the domains to which they belong.

(c) **Name Server:** Ipaddress of your DNS name server, there should be only one entry per "name server". If there is more than one name server, you will need to have multiple "name server" lines.

(d) **Domain:** The local domain name to be used by default. If the server is ***"redhat2.rootshell.com"*** , then the entry would just be ***"rootshell.com".***

Ex: nameserver 192.168.30.62

Zone files:

- In all zone files , you can place a comment at the end of any line by inserting a semicolon " ; " character then typing in the text of your comment .
- By default , your zone files are located in the directory ***" /var/named".***

Zone:

- zone is storage database which contains all zone records.

- Two types of zone records are available.

  (a) FLZ (Forward Lookup Zone): Used for resolving hostname to ip address. It maintains host to Ipaddress mapping information.

  (b) RLZ (Reverse Lookup Zone): Used for resolving ip address to hostname.It maintains ip address to host mapping information.

- Each zone file contains a variety of records (**Eg:** SOA, NS , MX , A and CNAME)
  - **SOA (Start Of Authority):** The very first record is the start of authority record of which contains general administrative and control information about the domain.
  - **NS (Name Server):** The NS resource record identifies all name servers that can perform name resolution for the zone. (or) List the name of the name server for the domain.
  - **A (Address):** This record resolves from host name to ip address.
  - **PTR (Pointer):** This record resolves from ip address to host name.
  - **CNAME (Canonical Name):** Provides additional alternate "alias" names for servers listed in the "A" record.
  - **MX (Mail Exchange):** List the mail servers for your domain.

Requirements:

- **Packages:** bind * …………..rpm
  Caching * ………rpm
- **Port no :** 53 --- DNS

- **Configuration files :** /etc/named.caching-nameserver.conf
  /etc/named.rfc1912.zones

Zone file location : /var/named /chroot/var/named
Primary DNS Server Configuration In RHEL-5

Step- 1: Check bind and caching-nameserver rpm package is installed or not by following this command:

```
[root@redhat2 ~]#  rpm –qa bind*   ↵
        (if installed then it will show all the bind  related packages)

bind-chroot-9.3.6-4.P1.el5
bind-libs-9.3.6-4.P1.el5
ypbind-1.19-12.el5
bind-9.3.6-4.P1.el5
bind-utils-9.3.6-4.P1.el5
bind-sdb-9.3.6-4.P1.el5
bind-devel-9.3.6-4.P1.el5
bind-libbind-devel-9.3.6-4.P1.el5
system-config-bind-4.0.3-4.el5

[root@redhat2 ~]# rpm -qa caching-name server*   ↵

Caching-nameserver-9.3.6-4.P1.el5
```

If not installed, then installed the packages using Yum command:

```
[root@redhat2 ~]# Yum Install bind* caching-name server*   ↵
```

Step-2: Check and Configure the Network Card:

```
[root@redhat2 ~]# vi /etc/sysconfig/network-scripts/ifcfg-eth0   ↵
```

(Original File):

```
#Broadcom Corporation Net link BCM57875 Gigabyte Ethernet PCI Express
DEVICE=eth0
BOOTPROTO=static
HWADDR=00:0C:29: EB: B2: CA
ONBOOT=yes
IPV6INIT=no
TYPE=Ethernet
IPADDR=192.168.30.62
NETMASK=255.255.255.0
BROADCAST=192.168.30.254
```

(Modified File):

```
#Broadcom Corporation Net link BCM57875 Gigabyte Ethernet PCI Express
DEVICE=eth0
BOOTPROTO=static
HWADDR=00:0C:29: EB: B2: CA
ONBOOT=yes
TYPE=Ethernet
PEERDNS=no
USERCTL=no
IPV6INIT=no
IPADDR=192.168.30.62
NETMASK=255.255.255.0
BROADCAST=192.168.30.254
```

After changing you have to reload/restart the NIC(eth0) card by following command:

```
     [root@redhat2 ~]# service  network  restart   ↵
```

Step-3: After complete the NIC configure you have to change the host name by following this command:

```
            [root@redhat2 ~]# vi  /etc/sysconfig/network   ↵
```

(Original File):

```
NETWORKING=yes
NETWORKING_IPV6=yes
HOSTNAME=redhat2
```

(Modified File):

```
NETWORKING=yes
NETWORKING_IPV6=no
HOSTNAME=redhat2
```

Step-4: Now Edit hosts file for host name resolution by following this command:

```
    [root@redhat2 ~]# vi /etc/hosts   ↵
```

(Original File):

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1       localhost.localdomain localhost
```

(Modified File):

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
```

```
127.0.0.1       localhost.localdomain localhost
192.168.30.62   redhat2.rootshell.com     redhat2
```

Step-5: Copy & Rename the named.rfc1912.zones file to named.conf file & Change the ownership & permission by following this command:

```
[root@redhat2 ~]# cd /var/named/chroot/etc/   ↵
[root@redhat2 etc]# cp named.rfc1912.zones  named.conf   ↵
[root@redhat2 etc]# chown root:named named.conf   ↵
[root@redhat2 etc]# chmod 777 named.conf   ↵
```

Now Create A Link into /etc directory of named.conf then edit.

```
[root@redhat2 etc]# ln –s /var/named/chroot/etc/named.conf  /etc/named.conf   ↵
[root@redhat2 ~]# vi /etc/named.conf   ↵
```

Original File:

```
// named.rfc1912.zones:
//
// Provided by Red Hat caching-nameserver package
//
// ISC BIND named zone configuration for zones recommended by
// RFC 1912 section 4.1 : localhost TLDs and address zones
//
// See /usr/share/doc/bind*/sample/ for example named configuration files.
//
zone "." IN {
    type hint;
    file "named.ca";
};
zone "localdomain" IN {
    type master;
    file "localdomain.zone";
    allow-update { none; };
};

zone "localhost" IN {
    type master;
    file "localhost.zone";
    allow-update { none; };
};

zone "0.0.127.in-addr.arpa" IN {
    type master;
    file "named.local";
    allow-update { none; };
};

zone "0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6.arpa" IN {
        type master;
    file "named.ip6.local";
    allow-update { none; };
};
```

```
zone "255.in-addr.arpa" IN {
    type master;
    file "named.broadcast";
    allow-update { none; };
};

zone "0.in-addr.arpa" IN {
    type master;
    file "named.zero";
    allow-update { none; };
};
```

Modified File:

```
// named.rfc1912.zones:
//
// Provided by Red Hat caching-nameserver package
//
// ISC BIND named zone configuration for zones recommended by
// RFC 1912 section 4.1 : localhost TLDs and address zones
//
// See /usr/share/doc/bind*/sample/ for example named configuration files.
options {

    directory "/var/named";

};

zone "." IN {
    type hint;
    file "named.ca";
};

zone "localdomain" IN {
    type master;
    file "localdomain.zone";
    allow-update { none; };
};

zone "localhost" IN {
    type master;
    file "localhost.zone";
    allow-update { none; };
};

zone "0.0.127.in-addr.arpa" IN {
    type master;
    file "named.local";
    allow-update { none; };
};

zone "0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6.arpa" IN {
      type master;
    file "named.ip6.local";
    allow-update { none; };
};
```

```
zone "255.in-addr.arpa" IN {
     type master;
     file "named.broadcast";
     allow-update { none; };
};

zone "0.in-addr.arpa" IN {
    type master;
     file "named.zero";
     allow-update { none; };
};

zone "rootshell.com" IN {
     type master;
     file "rootshell.fz";

};

zone "30.168.192.in-addr.arpa" IN {
     type master;
     file "rootshell.rz";
};
```

Step-6: Copy, rename & change the ownership & permission of localhost.zone file for Forward lookup zone.

```
[root@redhat2 ~]# cd /var/named/chroot/var/named/   ↵
[root@redhat2 named]# cp localhost.zone rootshell.fz   ↵
[root@redhat2 named]# chown root:named rootshell.fz   ↵
[root@redhat2 named]# chmod 777 rootshell.fz   ↵
[root@redhat2 named]# vi rootshell.fz   ↵
```

Original File: localhost.zone

```
$TTL 86400
@        IN SOA     @       root (
                        42        ; serial (d. adams)
                        3H        ; refresh
                        15M        ; retry
                        1W        ; expiry
                        1D )      ; minimum

          IN NS @
         IN A      127.0.0.1
          IN AAAA        ::1
```

Modified File: rootshell.fz

```
$TTL 86400
@        IN SOA     redhat2.rootshell.com.       root.rootshell.com. (
                        2011022500 ; serial (d. adams)
                        3H        ; refresh
                        15M        ; retry
                        1W        ; expiry
                        1D )      ; minimum

          IN NS     redhat2.rootshell.com.
redhat2     IN A      192.168.30.62
redhat3     IN A      192.168.30.63
```

Step-7: copy & rename the rootshell.fz file for Reverse lookup zone.

```
[root@redhat2 named]# cp rootshell.fz rootshell.rz ↵
[root@redhat2 named]# chown root:named rootshell.rz ↵
[root@redhat2 named]# chmod 777 rootshell.rz ↵
[root@redhat2 named]# vi rootshell.rz ↵
```

Original File: rootshell.fz

```
$TTL 86400
@         IN SOA     redhat2.rootshell.com.     root.rootshell.com. (
                     2011022500 ; serial (d. adams)
                     3H         ; refresh
                     15M         ; retry
                     1W         ; expiry
                     1D )       ; minimum

         IN NS     redhat2.rootshell.com.
redhat2      IN A       192.168.30.62
redhat3      IN A       192.168.30.63
```

Modified File: rootshell.rz

```
$TTL 86400
@         IN SOA     redhat2.rootshell.com.     root.rootshell.com. (
                     2011022500 ; serial (d. adams)
                     3H         ; refresh
                     15M         ; retry
                     1W         ; expiry
                     1D )       ; minimum

         IN NS     redhat2.rootshell.com.
62        IN PTR     redhat2.rootshell.com.
63        IN PTR     redhat3.rootshell.com.
```

Step-8: Edit the resolv.conf file by following this command

```
[root@redhat2 named]# vi /etc/resolv.conf ↵
search rootshell.com
nameserver 192.168.30.62
```

Step-9: Check the named.conf & zone file by following this commands,

```
[root@redhat2 ~]# named-checkconf /var/named/chroot/etc/named.conf ↵
[root@redhat2 ~]# named-checkzone rootshell.com /var/named/chroot/var/named/
rootshell.fz ↵
[root@redhat2 ~]# named-checkzone rootshell.com  /var/named/chroot/var/named/
rootshell.rz ↵
```

Step-10: DNS check

```
[root@redhat2 ~]# service iptables stop ↵
[root@redhat2 ~]# service network restart ↵
[root@redhat2 ~]# chkconfig named on ↵
```

```
[root@redhat2 ~]# service named status ↵
[root@redhat2 ~]# service named configtest ↵
[root@redhat2 ~]# service named start ↵
[root@redhat2 ~]# nslookup redhat2.rootshell.com ↵
[root@redhat2 ~]# nslookup 192.168.30.61 ↵
```

WEB SERVER

- By using web server we can host the WebPages.
- Number of web servers are available
  - o IIS : Microsoft
  - o Apache : For all platforms
  - o Tomcat, Web logic : Third party tools
- **Apache :**
  - ➢ In the year 1995 "http" daemon was popular to host a web server. This was developed by NCSA (National center for supercomputing applications)
  - ➢ Apache is firmware and is the most popular and widely used web server which consumes 60% of web market that can be configured in both windows and Linux

  Requirements:
  - o Packages : http…rpm
  - o Port no : http – 80
  - o Config file : /etc/httpd/conf/httpd.conf
  - o Service : httpd

- **Configure the DNS:** Remember that you will never receive the correct traffic unless you have configured DNS for your domain to make your new Linux box web server the target of the DNS domains www entry.
  - ➢ See either static DNS or Dynamic DNS pages on how to do this

- **Configure the web server :**
  - ➢ Install the package:
    - ▪ #rpm –ivh http* --force –aid (or)
    - ▪ #yum install http* -y ↵
  - ➢ Open the configuration file:

    #vi /etc/httpd/conf/httpd.conf ↵

  - ➢ Go to Last line:

```
                                    <virtual host*:80>
[Receives mail generated by apache server]  Server Admin root@redhat4.rootshell.com
[Name of the website]                       Server Name redhat4.rootshell.com
[Web page location]                         Document root /var/www/html
[Index (or) home pages]                     Directory Index file.html
                                    <./virtual host>
```

- ➢ Go to website location:

-
    - #cd /var/www/html ↵
    - #vi file.html ↵ here we can write html code
- Restart the service
    - #service httpd restart ↵
- Open the web browser
    - CLI based : #elinks
    - GUI based : #fire fox

- **Authentication of the user:**

    - Open the config file:
        - #vi /etc/httpd/conf/httpd.conf ↵

    - Go to last line:

```
<directory /var/www/html/>
AuthUserFile /etc/httpd/conf/.htpasswd
AuthName “web authentication”
Authe Type Basic
Require valid-user
</directory>
```

    - Now create user and assign the http password:
        - #useradd user ↵
        - #htpasswd –c /etc/httpd/conf/htpasswd user1 ↵
    - Restart the service:
        - #service httpd restart ↵
    - Now open the web browser
        - #elinks (or)
        - #firefox
    - Type URL:
        - http://redhat4.rootshell.com

Now type the authentication user name and password in the dialogue box.

- **IP Based :**
    - The other virtual hosting options is to have one IP address per website which is also known as IP based virtual hosting
    - In this case you will not have a NameVirtualHost directive for the Ip Address and you must only have a single <virtual host> section per IP address.

        - **Create the Virtual IP:**

Go to location

```
#cd /etc/sysconfig/ifcfg-eth0 ↵
#cp ifcfg-eth0        ifcfg-eth0:0
#vi ifcfg-eth0:0 ↵
        Device = eth0:0
        On boot=yes
        IP address = 192.168.30.64
#service network restart
```

- **Open the config file:** #vi /etc/httpd/conf/httpd.conf ↵

Go to last line: <virtual host 192.168.30.64:80>

```
          Server Admin        root@redhat4.rootshell.com
          Server Name         redhat4.rootshell.com
          Document Root       /var/www/html
          Directory index     file.html
</virtual host>
```

- Virtual hosting is used to host multiple websites on a single machine with one or more public IP address
- There are three types of virtual hosting
    - Name Based
    - IP Based
    - Port based

- **Name based :** You can make your web server host more than one site per IP address by using Apache’s “Named Virtual Hosting”

    - Install the package:
        - #yum install httpd* -y ↵
    - Open the config file:
        - vi /etc/httpd/conf/httpd.conf ↵
    - Go to the line 972 remove#:
    - Go to last line:

```
<virtual host*:80>
          ServerAdmin         root@redhat4.rootshell.com
          Server Name         redhat4.rootshell.com
          Document root                 /var/www/html
          Directory Index     file.html
</virtual host>
<virtual host*:80>
          ServerAdmin         root@redhat4.india.com
          Server Name         redhat4.india.com
          Document Root       /var/www/html
          Directory Index     file.html
<virtual host>
```

- **Port Based:** In this we can host more than one website which works on different port numbers.

```
Listen 8000
          <virtual host :192.168.30.64 :8000>
                    ServerAdmin         root@ redhat4.rootshell.com
                    Server Name         redhat4.rootshell.com
                    Document Root       /var/www/html
                    Directory Index     file.html
          <virtual host>
```

- Go to website location:
    - #cd /var/www/html ↵
    - #vi file.html ↵

      Write html code
- Now restart the service:
    - #service httpd restart ↵
    - #chkconfig httpd on ↵
- Open the web browser:

- #elinks (or) #firefox

➢ To test name based: **Error! Hyperlink reference not valid.**

http:// redhat4.india.com

➢ To test IP based: http://192.168.30.64

➢ To test port based: Error! Hyperlink reference not valid.

SAMBA

Samba uses the SMB protocol to share files and printers across a network connection. Operating systems that support this protocol include Microsoft Windows, OS/2, and Linux.

- SAMBA server is used to communicate the shares on windows.
- SAMBA is a free software re-implementation of SMB/CIFS networking protocol, originally developed by Australian Andrew Tridgell.
- As of Version 3, SAMBA provides file and print services for various Microsoft windows clients and can integrate with a windows server domain, either as a Primary Domain Controller (PDC) (or) as a domain member.
- It can also be part of an Active Directory Domain.
- Samba using smb protocol. It also call as “server Message Block” also known as “netbios/tcp-ip”.

Features Of SAMBA:

i. File/Directory Sharing.
ii. Browsing.
iii. Resource Sharing.
iv. User Authentication & Authorization.

Requirements:

Packages:
samba*…..rpm

Port No:
137 – Net BIOS name service.
138 – Net BIOS Datagram service.
139 – Net BIOS Session service.

Config File:
/etc/samba/smb.conf

Service:
smb

Daemons:
smbd (Server Message Block Daemon)
nmbd (Server Message Block Daemon)

Configure SAMBA Server: (192.168.30.64)

- Before configuring the samba server, First we check the samba server is installed or not.

```
rpm –qa  | grep  –i  samba  ↵
```

Note: If it is not installed, then it will not show any output. If the samba server software is installed in the machine, it will show you the output as.

```
rpm  –qa | grep –i samba   ↵
Samba -3.0.33
```

Install the Packages:

```
rpm  -ivh  samba*   --force   --aid   ↵
                  (or)
```

```
yum  install   samba*   -y   ↵
```

Create the source for sharing:

```
mkdir  /sambashare   ↵
cd   / sambashare    ↵
touch 1 2 3   ↵
```

Open the configuration file:

```
vi   /etc/samba/smb.conf   ↵
```

Go to last line:

```
 [common]
 comment  =  This is Samba sharing info
 path         =  / sambashare
valid users  =  smb1 smb2
public        =  no
writable      =  yes
printable     =  no
:wq!    ↵
```

Test the configuration:

```
testparm   ↵
```

- Testparm will parse your configuration file and report any unknown parameters or incorrect syntax. It also performs a check for common misconfigurations and will issue a warning if one is found. So always run testparm again whenever the smb.conf file is changed.
- After checked the syntax don’t forget to restart the service without restarting the service action will not get effect.

Create Samba users and assign samba password:

```
useradd  smb1   ↵
useradd  smb2   ↵
smbpasswd  –a  smb1   ↵
smbpasswd  –a  smb2   ↵
```

Restart the service:

```
service smb restart   ↵
```

Client side (Linux):

To Check the shares are visible or not:

```
smbclient   –L  //192.168.30.64  –N   ↵
                    (or)
smbclient   –L   localhost   ↵
```

We can access the sharing information in two ways:

i. NFS (or) Mount.
ii. FTP.

i. NFS method:

```
mkdir    /smbclient   ↵
mount  //192.168.30.64/sambashare  /smbclient  -0  username=smb1   ↵
     Password:  <smb user password>
     cd  /smbclient   ↵
     ls   ↵
```

ii. FTP method:

```
smbclient  //192.168.30.64/sambashare   –U  smb2   ↵
Password: <smb user password>   ↵
```

```
Smb>quit ↵
```

Client side (Windows):

Open run Prompt:

And type the samba server ip with path \\192.168.30.64/sambashare

(or)

\\192.168.30.64 ↵

Note: This share will not accessible after reboot. For permanent mount just make an entry in /etc/fstab file.

```
vi /etc/fstab ↵
//servername/sharename   mountpoint   smbfs   defaults  0  0
    :wq! ↵
```

KICKSTART INSTALLATION

System administrators would prefer to use an automated installation method to install Red Hat Enterprise Linux on their machines. T o answer this need, Red Hat, Inc created the kickstart installation method. Using kickstart, a system administrator can create a single file containing the answers to all the questions that would normally be asked during a typical installation.

Kickstart files can be kept on a single server system and read by individual computers during the installation. T his installation method can support the use of a single kickstart file to install Red Hat Enterprise Linux on multiple machines, making it ideal for network and system administrators. Kickstart provides a way for users to automate a Red Hat Enterprise Linux installation.

- Kick start allows the installer to read information from a designated file rather than prompting the person doing the installation.
- It especially facilitates the setup of a number of machines that have similar hardware that the installer can auto probe successfully.
- Anaconda automatically generates a kick start file during installation and saves it under /root/anaconda-ks.cfg. This file can be used as a basic for disaster recovery of this same machine (or) as a starting point for your own custom installs.
- ***"system-config-kickstart"*** is a graphical tool for creating and modifying kick start files.
- We can install the O.S in two ways.
  1. Standalone installation
  2. Network installation
- In network installation we can install three ways
  1. NFS
  2. FTP
  3. KICKSTART
- **Requirements:**
  - O.S Dump
  - Pykickstart and system-config-kick start packages
  - NFS (or) FTP service
  - DHCP service
- **Configure the kick start through ftp service:**

  Install the ftp package and copy O.S dump to default sharable location of ftp service (i.e.,/var/ftp/pub)

  - **Mount DVD**

```
 mount /dev/DVD/mnt   ↵
cd /mnt/server   ↵
rpm  -ivh  vsftpd*  --force   --aid   ↵
cp   –rvf   /mnt/*   /var/ftp/pub      ↵
```

- **Install the kick start package :**

```
rpm  -ivh  pykickstart * system-config-kick start *    --force
```

(or)

```
yum install pykickstart * system-config-kick start *    -y
```

- **Now open the kick start dialogue-box :**

```
system-config-kickstart   ↵
```

  - After configure the above dialogue-box save the file with name ks.cfg under the directory /var/ftp/pub.

- **To give full permission to ks.cfg file :**

```
chmod 777 kf.cfg   ↵
```

- **Configure the dhcp service :**
- **Restart the FTP&DHCP services**:
- **Client side configuration:**

Step 1: boot PC with linux bootable DVD

Step 2: The boot prompt type

Use FTP: boot: Linux ks =ftp://192.168.1.254/pub/ks.cfg

Use NFS: boot Linux ks =NFS://192.168.1254:/var/ftp/pub/ks.cfg

```
<html>
<body bgcolor=red>
<marque><h1>………..</h1></marque>
<body>
</html>
```

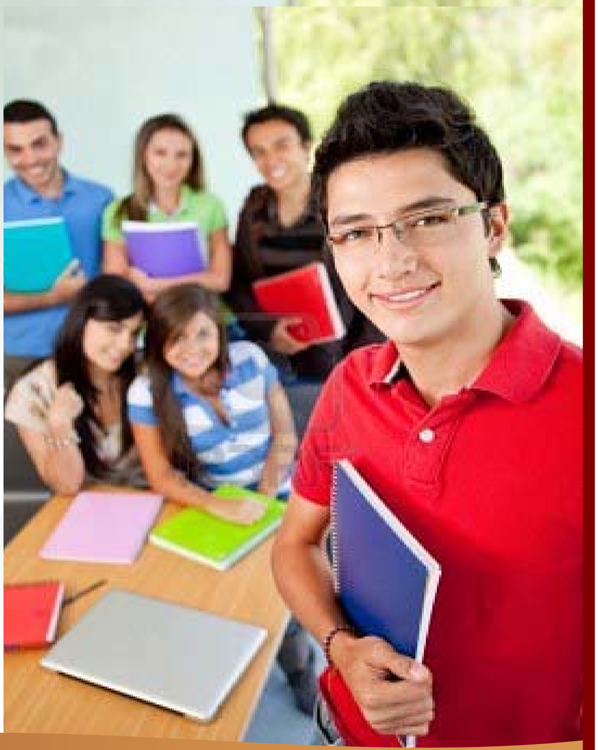
Autofs,
Job scheduling(Cron & At)

An QMS ISO 9001:2008 Certified Company

Now it's the time to INVEST IN YOURSELF...

Train today at Rrootshell for a SECURE tomorrow

Rrootshell™ | Excellence Delivered



Change is in YOUR hands
Shape the FUTURE

Why Rrootshell ...

- *End to End Training and Guidance*
- *Focused on Quality real time trainings*
- *Live Data Centre Practices.*
- *Access to real time servers*
- *No simulations.*
- *Real Time Scenarios Explanation*
- *Placement Assistance.*
- *Experienced Technical Staff Guidance*
- *Abundant Lab Facility*
- *Career Orientation*

Join and be successful

OUR CORE TECHNOLOGY TRAINING OFFERINGS IN

- VCE VBLOCK Systems
- CISCO UCS
- Clariion, Symmetrix, DMX & VMAX.
- VNX & Recover Point
- HITACHI Data Systems Enterprise Storage
- Netapp 7Mode & C-Mode Administration
- Avamar & Bura
- Netbackup, DataDomain & TSM
- Netapp 7Mode & C-Mode Administration
- VMware, LPAR & HACMP
- Hadoop Administration & Development
- Clusters & Volume Manager
- IBM DS Storage & SVC
- UNIX (Solaris, AIX, HP-UX & Linux)

Contact Us
040-66616704
040-66629937
www.rrootshell.com

Rrootshell Technologiiss Pvt Ltd

202, Trendsetpyla, Vengalrao Nagar, Hyderabad, INDIA – 500 0038.:
Ph: Hyderabad : 040- 66616704, 66629937:,
Bangalore :080- 32972711, 32965533,
Emai : hvd@rrootshell.com. info@rrootshell.com. blr@rrootshell.com

Join & Get Transformed as a Trained Software Professional