***RedHat Enterprise linux 7.3 – RHCE preparation kit***

***Linux Principles***

***1.* Everything is a file :**

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

***2.* Configuration Data stored in Text:**

Text is a universal interface, and many Linux utilities exist to manipulate text. Storing configuration in text allows an administrator to move a configuration from one machine to another easily. There are several revision control applications that enable an administrator to track which change was made on a particular day, and provide the ability to roll back the system configuration to a particular date and time.

***3.* Small, single purpose programs*:***

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

***4*. Ability to chain programs together to perform complex tasks:**

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

***5.* Avoid Captive user interface.**

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

***Features of Linux***

**Multi User and Multi-tasking OS**

Linux is a multi-user and multi-tasking OS. That means that more than one person can be logged on to the same Linux computer at the same time from different terminals.

Linux is also multi – tasking: a user can have more than one process executing at the same time.

***Components of Linux User Environment***

**1. Kernel**: Kernel is the core of Linux. It loads into RAM when a machine boots, where it runs until the computer is shutdown. Its job is to perform many low-level and system-level functions, to interpret and send instructions to the system’s hardware, to schedule and run processes, and to manage input and output.

**2. Shell:**

The Shell provides an interface by which the user can communicate with the kernel. It examines and evaluates commands typed at its prompt, then passes them to the kernel. It then receives the results back from the kernel and displays the results. The shell also provides a mechanism by which these results can be captured in a file, and by which the input to a command can be directed from a file. It enables the output of one command to be supplied as the input to another command. The shell most often used in Linux is bash

**Files**

Linux supports several different types of files: 1.Regular files, 2.Directories 3. Symbolic links, 4.Sockets, 5.Pipes and 6 .Character and Block devices.

**Device files**

Linux accesses hardware devices through files.

The two file types that Linux recognizes are character devices and block devices. By character special files and block special files.

**Character devices:**

Character special files interact with their respective I/O devices on a character by character (or byte by byte) basis. Character devices must be read and written to in order. (Serial)

**Block devices:**

Block devices such as hard disk, transfer data in blocks. Blocks typically range in size from 512 bytes to 32 KB. Block devices are also capable of being randomly accessed.

**File System Hierarchy Concepts**

* Files and directories are organized into a single-rooted inverted tree-structure, including distinct physical volumes such as floppy disk and CD-ROMs and multiple hard disks.
* The base of the inverted tree-hierarchy is known as root or / - the top of the file system.
* A forward slash separates elements of a pathname, for example /usr/bin/X11/X.
* Files and directory names in Linux file hierarchy are case sensitive.
* Each shell and process has a current working directory.
* .. refers to the parent directory of any particular directory.
* . refers to the current directory.
* Files and directories whose names which begin with . are hidden, that is they are not displayed by default.
* A user’s path is a list of directories that are searched for commands typed at the command line.

**Linux Commands v 1.0 (FC3)**

$man <command> : To get help for commands

$info <command> : To get help for commands

~ : A user’s home directory

- : The previous working Directory

$date : To get date

$cal : To get the current calendar

$cal –y 2003 : To get the calendar for year 2003

$cal 2 2003 : To get the calendar for month 2 year 2003

$id : Gives the user id

$groups : To get the groups to which user belongs

$whoami : To give the username

$logname : To give the logname of the user

$id username : Gives the username’s id

$groups user : Gives the users group

$who : Lists all users who have logged in

$w : Long listing of command who

$last : last login with times

$last user : last login time of user

$finger username : user’s details

$passwd : To change the password

$id : To get the user id

$id username : To get the username’s id

/usr/share/doc : Help file documents

$pwd : To get to the present working directory

$mkdir : To create to directory

$mkdir dirname ; cd dirname : two commands can be used simultaneously.

Absolute Path name begins with the /

Relative Path name begins from the current directory.

**CD - Change Directory**

$cd : Change to home directory

$cd dirname : Change to dirname

$cd .. : Change to parent directory

$cd - : Change to previous working directory

$cd ~ : Change to home directory

**Ls - List Contents**

$ls : List the directory contents

$ls dirname : List the contents of dirname directory

$ls –a : List the hidden files too

$ls –l : List the files and folders in long listing format

$ls –d : List the directory names

$ls –R : Recursive listing.list contents of subdirectories

$ls –i : List the contents with inode value

$ls –h : List the contents in human readable format

**Copy**

$cp source destination : copy source to destination

$cp –r : Recursive copying

$cp –p : To preserve time stamps… while copying.

$cp –i : To copy interactively.

**Move**

$mv source destination : Moving source to destination

$mv source source : Renaming source to source2

**Remove**

$rm : Removes a file

$rm –i : Interactive Removing files

$rm –r : Recursive removal (includes subdirectories)

$rm –f : Forcible removable

$mkdir dirname : creating new directory

$rmdir : remove a empty directory

$rm –r : remove a non-empty directory with files

$touch file1 file2 : To set the time stamp to current time and if no file exists with that name, it will create a new empty file.

$file filename : To determine file contents/type/properties

$cat filename : To view the contents of a file

$cat –A : Show all characters

$cat –b : number each line of output

$cat –s : squeeze multiple blank lines into single line.

$cat filename.txt | more : show multiple lines with more piping.

$cat filename.txt | less : show multiple lines with less piping.

<Spacebar> : Move ahead one page

<return> : Move ahead one line

<Ctrl+d> : Move ahead half a screen

<Ctrl+C> : To save and quit the file

$cat > file.txt : To create a new file with cat command

$cat >> file.txt : To append the file with new contents

/search : To search text in files

n : To repeat the search

q : Quit

$head filename : To view first 10 lines of file

$head –n 3 filename : To view first 3 lines of a file

$tail filename : To view the last 10 lines of a file

$tail –n 4 : To view the last 4 lines of a file

$tail –f filename : Will continue to show update of the file until Ctrl+C

$pico : Text Editor

$pico –w : To turn off wrapping of text

**Slocate**

$slocate /path filename : To locate the filename(uses a database )

$locate /path filename : To softlink to slocate

#updatedb : To update the slocate database

/var/lib/slocate/slocate.db : Slocate database file

**Bash Shell**

\* : For multiple characters of text  
? : For single character of text

$history

$history | less

$history | more

~/.bash\_history == history file

$HISTSIZE

echo $HISTSIZE : to get the current history size

Change HISTSIZE in /etc/profile for increasing the history size.

!! To repeat the last command

!c to repeat the last command that started with c

!n to repeat a command by its number in history output

Use CTRL + R for search for a command in command history

$last 1 reboot : Last system reboot

$su : Substitute user to root without root profile.

$su root : Substitute user to root without root profile.

$su - : Substitute user to root with root profile.

$su - username : Substitute for username with profile.

$file filename : To view the type of file.

$stat filename : To view the data stored in the inode for a file

**Mounting Removable Storage media**

**CDROM:**

$mount –t iso9660 /dev/cdrom /mnt/cdrom

$umount /mnt/cdrom

$eject /mnt/cdrom

**Mounting Floppy**

Mounting floppy

$mount /dev/fd0 /mnt/floppy

$umount /mnt/floppy

**Mounting FAT filesystem**

$mount –t vfat /dev/hda1 /mnt/winc

$umount /mnt/winc

**Mounting \*.iso file**

Mounting a \*.iso file

#mount –t iso9660 –ro loop /tmp/firstiso.iso /mnt/cdrom

**FILE COMPRESSION TOOLS**

**GZIP**

$gzip filename : to get filename.gz

$gunzip filename.gz : to get back filename

$gzip –v filename : To get verbose output

**BZIP**

$bzip2 filename : to get filename.bz2

$bunzip2 filename.bz2 : to get filename

**ZIP**

$zip files+dirs : to zip files and directories \*.zip

$unzip \*.zip : to get back filename

$compress : ??

compress – 50 %

gzip - 1/3 rd of original

bzip2 - 1/4th of original

bzip2 better than gzip better than compress

**TAR**

$tar cvf file1,file2,file3 : To archive files

$tar dirname : To archive the directory and its contents

$tar tf archive.tar : To display the contents of files inside \*.tar

$tar tvf archive.tar : causes long listing of each files inside \*.tar

$tar xvf archive.tar : To extract files inside \*.tar

$tar –xvzf ??

**Disk Usage**

$df : disk Usage

e.g. #df /home

$df –h : Disk usage in human readable format

$du filename : Space occupied by file filename

$du –sh

**Formatting a floppy**

fdformat /dev/fd0 to make file system.

mkfs -t ext2 /dev/fd0 (or)

mkfs2fs /dev/fd0 (or)

mkfs -t vfat /dev/fd0 (or)

mformat a:

**File systems:**

Ext2 : The Extended Linux file system. Widely used in Linux systems.

Ext3 : Third extended Linux file system. Enhanced version of ext2 file system, that uses journaling to improve file system data integrity.

**Inode:**

An Inode table contains a list of all files in an ext2 or ext3 filesystem. (It is kind of database, can think it like a excel sheet)

The inode is an entry in the table, containing information about a file including:

file type

permissions

link count

UID, GUID

file size

time stamp

pointers to the file’s data blocks on disk (\*\*\*)

other data

For any partition one inode table will be there.

The computer’s reference for the file is the inode number.

The human way to refer a file is by file name.

Whenever a file name is referenced by a command or application, linux references the inode value and gets the location of the file and pulls the file for viewing or manipulation.

$ls –i

**Copying and inode.**

When you copy file1 to file2 a new inode number is created and a new entry is created in the inode table.

Copies data into the new file(new location in hard disk)

**LAB:** create file1, check the inode value, copy file1 file2, check the inode value for both, two different inode values should be there.

Even if they are in different partitions same inode value creation happens.

**Moving with inode:**

2 types occur: same filesystem, different filesystem.

If both the files are on the same filesystem(partition)

Creates a new filename, deletes the old filename, but keeps the same inode entry, no impact on the inode table.

**LAB**: do it just like the above, but you will see same inode value.

When they are in different file systems(partitions)

Creates a new inode entry in the destination partition.

Data is deleted from the parent partition and moved to the destination partition.

**Remove with inode:**

When using rm file1, deletes the filename from the inode table. Data is not actually removed, but will be overwritten when the data blocks are used by another file.

**Symbolic links:**

A symbolic link points to another file

Syntax: $ln –s filename [linkname]

A new inode entry for the symbolic link is created. Which points to the data of the filename?

The size of the symbolic link is always the number of characters in the path name.

**Hard links:**

One physical file on the filesystem.

Each link references the file’s inode

File is present in the filesystem as long as at least one link remains.

Cannot span drives or partitions.

Syntax: $ln filename [linkname]

All the files will have same inode value.

It is not possible to use the ln command to create additional hard links to directories.

**The Seven Fundamental Filetypes**:

Redhat Linux supports these filetypes

1. Regular file -
2. Directory d
3. Symbolic link l
4. Block special file b
5. Character special file c
6. Named pipe p
7. Socket s

We know about 1. Regular file, 2. Directory, 3. symbolic links

In linux hardware is also considered to be file. We have two types of hardware device files.

C Character special file: & B Block special file

Character special files are used to communicate with hardware one character at a time.

B Block special file:

Block special files are used to communicate with hardware a block of data at a time: 512 bytes, 1024 bytes,.

e.g. run ls –l /dev | less … we can see various block special files and character special files.

p named pipe: A file that process data between processes. It stores no data itself, but passes data between one process writing data into the named pipe and another process reading data from the named pipe. A named pipe can be created using the mknod command

mknod mypipe p

s socket: used in interprocess communication. Very rarely seen.

**Find (dynamic searching, better than slocate)**

$find /etc -name "\*.conf" --- (works)

$find /etc -iname "\*.conf" --- (case insensitive)

$find /home -user arun --- ( it works)

$find /home -group arun

$find /etc -perm 777

$find /etc -type c (character special file) (d), p(pipe) l (symbolic link) s(socket), f(plain file)b(block special file)

$find /etc -size 7 (7 characters long)

$find /home -atime +10 (more than ten days back)

$find /home -atime -10 (from today to ten days in betwen--- last ten days)

$find /home -mtime +10 (more than ten days back)

$find /home -mtime -10 ( last ten days)

**Finding and processing files**

Find with exec

$find /etc -name "\*.conf" -exec ls {} \;

**RUN LEVELS**

init 0 : Shutdown

init 1 : Single User mode without networking

init 2 : Multi User mode without networking

init 3 : Multiuser mode with networking

init 4 : Unused

init 5 : Multiuser mode with GUI

init 6 : Reboot the computer

**How can I shutdown my computer?**

In a text terminal, press <Ctrl><Alt><Del> (the "three-finger salute", you press the three keys simultaneously), wait for the shutdown process to complete, and turn off your machine only after it starts rebooting again. If you are in X-windows, first switch to a text terminal by pressing <Ctr><Alt><F1> (three keys simultaneously).

Linux servers are known to run for more than a year without a reboot.

As root

#init 0

#shutdown -h : shutdown and halt

#shutdown -r : shutdown and reboot

$echo $HOME

$echo $HISTSIZE