**RedHat Enterprise Linux Basics Commands**

1. 1. 2.6x kernel (2.6.18)

a. 'uname -a' returns OS/Kernel information

**Note:** 'uname -a' returns the following useful info:

1. OS – Linux

2. Fully Qualified Domain Name (FQDN)

3. Kernel version - 2.6.18...

a. 2.6 = major version

b. .18 = minor version

c. anything else after the minor version indicates that the kernel was patched by the distributor

4. Date and time that the kernel was compiled

2. Supports multiple versions:

a. Basic - Red Hat Enterprise Linux Server

a1. supports 2 physical (Socket) CPUs

a2. Up to 4 virtual guests

b. Advanced Platform

b1. supports unlimited physical CPUs

b2. supports unlimited virtual guests

**Note:** Virtualization limits pertain to the virtualization technology included with Red Hat Enterprise Linux. NOT third-party software (VMWare)

3. Supports the following platforms:

a. Intel 32/64-bits

b. AMD 32/64-bits

c. IBM - POWER and z-series, S/390

**Note:** Memory limitation is based on hardware

**Common uses of the various versions of RHEL**

1. RHEL Basic Version

a. File & Print

b. Web server

c. Infrastructure server (DHCP, DNS, Proxy, etc.)

2. RHEL Advanced Version

a. Application server (Apache Tomcat, JBOSS, Weblogic, WebSphere, etc.)

b. Database server (MySQL, PostgreSQL, Oracle, Ingres, etc.)

c. Clustering

**Basic LINUX Commands**

1. tty - reveals the current terminal

2. whoami - reveals the currently logged-in user

3. which - reveals where in the search path a program is located

4. echo - prints to the screen

a. echo $PATH - dumps the current path to STDOUT

b. echo $PWD - dumps the contents of the $PWD variable

c. echo $OLDPWD - dumps the most recently visited directory

5. set - prints and optionally sets shell variables

6. clear - clears the screen or terminal

7. reset - resets the screen buffer

8. history - reveals your command history

a. !690 - executes the 690th command in our history

b. command history is maintained on a per-user basis via:

~/.bash\_history

~ = user’s $HOME directory in the BASH shell

9. pwd - prints the working directory

10. cd - changes directory to desired directory

a. 'cd ' with no options changes to the $HOME directory

b. 'cd ~' changes to the $HOME directory

c. 'cd /' changes to the root of the file system

d. 'cd Desktop/' changes us to the relative directory 'Desktop'

e. 'cd ..' changes us one-level up in the directory tree

f. 'cd ../..' changes us two-levels up in the directory tree

11. Arrow keys (up and down) navigates through your command history

12. BASH supports tab completion:

a. type unique characters in the command and press 'Tab' key

13. You can copy and paste in GNOME terminal windows using:

a. left button to block

b. right button to paste OR Ctrl-Shift-v to paste

14. ls - lists files and directories

a. ls / - lists the contents of the '/' mount point

b. ls -l - lists the contents of a directory in long format:

Includes: permissions, links, ownership, size, date, name

c. ls -ld /etc - lists properties of the directory '/etc', NOT the contents of '/etc'

d. ls -ltr - sorts chronologically from older to newer (bottom)

e. ls --help - returns possible usage information

f. ls -a - reveals hidden files. e.g. '.bash\_history'

**Note:** files/directories prefixed with '.' are hidden. e.g. '.bash\_history'

15. cat - catenates files

a. cat 123.txt - dumps the contents of '123.txt' to STDOUT

b. cat 123.txt 456.txt dumps both files to STDOUT

c. cat 123.txt 456.txt > 123456.txt - creates new catenated file

16. mkdir - creates a new directory

a. mkdir testRH5 - creates a 'testRH5' directory

17. cp - copies files

a. cp 123.txt testRH5/

By default, 'cp' does NOT preserve the original modification time

b. cp -v 456.txt testRH5/

18. mv - moves files

a. mv 123456.txt testRH5/ - moves the file, preserving timestamp

19. rm - removes files/directories

a. rm 123.txt

b. rm -rf 456.txt - removes recursively and enforces

20. touch - creates blank file/updates timestamp

a. touch test.txt - will create a zero-byte file, if it doesn't exist

b. touch 123456.txt - will update the timestamp

c. touch -t 200801091530 123456.txt - changes timestamp

21. stat - reveals statistics of files

a. stat 123456.txt - reveals full attributes of the file

22. find - finds files using search patterns

a. find / -name 'fstab'

Note: 'find' can search for fields returned by the 'stat' command

23. alias - returns/sets aliases for commands

a. alias - dumps current aliases

b. alias copy='cp -v'

**Linux Redirection & Pipes**

**Features:**

1. Ability to control input and output

Input redirection '<':

1. cat < 123.txt

**Note:** Use input redirection when program does NOT default to file as input

Output redirection '>':

1. cat 123.txt > onetwothree.txt

**Note:** Default nature is to:

1. Clobber the target file

2. Populate with information from input stream

Append redirection '>>':

1. cat 123.txt >> numbers.txt - creates 'numbers.txt' if it doesn't exist, or appends if it does

2. cat 456.txt >> numbers.txt

Pipes '|':

Features: Connects the output stream of one command to the input stream of a subsequent command

1. cat 123.txt | sort

2. cat 456.txt 123.txt | sort

3. cat 456.txt 123.txt | sort | grep 3

**Command Chaining**

**Features:**

1. Permits the execution of multiple commands in sequence

2. Also permits execution based on the success or failure of a previous command

1. cat 123.txt ; ls -l - this runs first command, then second command without regards for exit status of the first command

2. cat 123.txt && ls -l - this runs second command, if first command is successful

3. cat 1234.txt && ls -l

4. cat 123.txt || ls -l - this runs second command, if first command fails

24. more|less - paginators, which display text one-page @ a time

1. more /etc/fstab

2. less 1thousand.txt

25. seq - echoes a sequence of numbers

a. seq 1000 > 1thousand.txt - creates a file with numbers 1-1000

26. su - switches users

a. su - with no options attempts to log in as 'root'

27. head - displays opening lines of text files

a. head /var/log/messages

28. tail - displays the closing lines of text files

a. tail /var/log/messages

29. wc - counts words and optionally lines of text files

a. wc -l /var/log/messages

b. wc -l 123.txt

30. file - determines file type

a. file /var/log/messages

**Tar, Gzip, Bzip2, Zip**

**Features:**

1. Compression utilities (gzip, bzip2, zip)

2. File rollers (the ability to represent many files as one)

**Gzip:**

Includes:

1. gzip - compresses/decompresses files

2. gunzip - decompresses gzip files

**Example:**

1. compress '1million.txt' file using gzip

a. gzip -c 1million.txt > 1million.txt.gz

**Note:** gzip auto-dumps to STDOUT, by default

b. gzip -l 1million.txt.gz - returns status information

c. gunzip 1million.txt.gz - dumps to file, and removes compressed version

d. gzip -d 1million.txt.gz

e. zcat 1million.txt.gz - dumps the contents to STDOUT

f. less 1million.txt.gzip - dumps the contents of gzip files to STDOUT

**Bzip2:**

1. bzip2 -c 1million.txt > 1million.txt.bz2

**Note:** Bzip2 tends to outperform gzip on larger files

2. bunzip2 1million.txt.bz2

3. bzip2 -d 1million.txt.bz2

4. bzcat 1million.txt.bz2 - dumps contents to STDOUT

5. less 1million.txt.bz2 - also dumps the contents to STDOUT

**Zip & unzip:**

1. zip filename.zip path/ - general usage

2. zip 1million.txt.zip 1million.txt

**Note:** zip differs slight from gzip and bzip2 in that the destination file (resultant zip file) is specified before the source

3. unzip 1million.txt.zip

**Tar & Gzip/Bzip2:**

1. tar -cvf filename.tar path/ - creates a non-compressed archive

2. tar -cvf 1million.txt.tar 1million.txt

**Note:** tar, requires a small overhead for itself in each file

3. tar -czvf 1million.txt.tar.gz 1million.txt - creates, tar/gzip document

4. tar -cjvf 1million.txt.tar.bz2 1million.txt - creates, tar/bzip2 document

5. tar -tzvf

6. tar -cjvf 1million.txt.tar.bz2 1million.txt testRH5/- creates, tar/bzip2 document for the text file and 'testRH5' directory tree

**GREP**

**Features:**

1. The ability to parse lines based on text and/or RegExes

2. Post-processor

3. Searches case-sensitively, by default

4. Searches for the text anywhere on the line

1. grep 'linux' grep1.txt

2. grep -i 'linux' grep1.txt - case-insensitive search

3. grep '^linux' grep1.txt - uses '^' anchor to anchor searches at the beginning of lines

4. grep -i '^linux' grep1.txt

5. grep -i 'linux$' grep1.txt - uses '$' anchor to anchor searches at the end of lines

**Note:** Anchors are RegEx characters (meta-characters). They're used to match at the beginning and end of lines

6. grep '[0-9]' grep1.txt - returns lines containing at least 1 number

7. grep '[a-z]' grep1.txt

8. rpm -qa | grep grep - searches the package database for programs named 'grep'

9. rpm -qa | grep -i xorg | wc -l - returns the number of packages with 'xorg' in their names

10. grep sshd messages

11. grep -v sshd messages - performs and inverted search (all but 'sshd' entries will be returned)

12. grep -v sshd messages | grep -v gconfd

13. grep -C 2 sshd messages - returns 2 lines, above and below matching line

**Note:** Most, if not all, Linux programs log linearly, which means one line after another, from the earliest to the current

**Note:** Use single or double quotes to specify RegExes

Also, execute 'grep' using 'egrep' when RegExes are being used

**Awk**

**Features:**

1. Field/Column processor

2. Supports egrep-compatible (POSIX) RegExes

3. Can return full lines like grep

4. Awk runs 3 steps:

a. BEGIN - optional

b. Body, where the main action(s) take place

c. END - optional

5. Multiple body actions can be executed by separating them using semicolons. e.g. '{ print $1; print $2 }'

6. Awk, auto-loops through input stream, regardless of the source of the stream. e.g. STDIN, Pipe, File

**Usage:**

1. awk '/optional\_match/ { action }' file\_name | Pipe

2. awk '{ print $1 }' grep1.txt

**Note:** Use single quotes with awk, to avoid shell interpolation of awk's variables

3. awk '{ print $1,$2 }' grep1.txt

**Note:** Default input and output field separators is whitespace

4. awk '/linux/ { print } ' grep1.txt - this will print ALL lines containing 'linux'

5. awk '{ if ($2 ~ /Linux/) print}' grep1.txt

6. awk '{ if ($2 ~ /8/) print }' /var/log/messages - this will print the entire line for log items for the 8th

7. awk '{ print $3 }' /var/log/messages | awk -F: '{ print $1}'

**Sed - Stream Editor**

**Features:**

1. Facilitates automated text editing

2. Supports RegExes (POSIX)

3. Like Awk, supports scripting using '-F' option

4. Supports input via: STDIN, pipe, file

**Usage:**

1. sed [options] 'instruction[s]' file[s]

2. sed -n '1p' grep1.txt - prints the first line of the file

3. sed -n '1,5p' grep1.txt - prints the first 5 lines of the file

4. sed -n '$p' grep1.txt - prints the last line of the file

5. sed -n '1,3!p' grep1.txt - prints ALL but lines 1-3

6. sed -n '/linux/p' grep1.txt - prints lines with 'linux'

7. sed -e '/^$/d' grep1.txt - deletes blank lines from the document

8. sed -e '/^$/d' grep1.txt > sed1.txt - deletes blank lines from the document 'grep1.txt' and creates 'sed1.txt'

9. sed -ne 's/search/replace/p' sed1.txt

10. sed -ne 's/linux/unix/p' sed1.txt

11. sed -i.bak -e 's/3/4' sed1.txt - this backs up the original file and creates a new 'sed1.txt' with the modifications indicated in the command

**Note:** Generally, to create new files, use output redirection, instead of allowing sed to write to STDOUT

**Note:** Sed applies each instruction to each line

**Perl**

**Features:**

1. Parses text

2. Executes programs

3. CGI - Web forms, etc.

4. Supports RegExes (Perl and POSIX)

5. etc.

**Example:**

1. Print 'Hello World' to STDOUT

a. perl -c helloworld.pl - checks the syntax of the script

b. perl helloworld.pl - executes the script

c. chmod +x helloworld.pl && ./helloworld.pl

2. Parse RegExes from the command line

**System Utilities**

**Features:**

1. Process listing

2. Free/available memory

3. Disk utilization

1. ps - process status/listing

a. ps -ef or ps -aux

2. top - combines, ps, uptime, free and updates regularly

3. uptime - returns useful system utilization information:

a. current time

b. uptime - days, hours and minutes

c. connected users

d. load averaged - 1,5,15 minute values

4. free - returns memory utilization

a. RAM

b. SWAP

free -m - for human readable format

5. df - returns disk partition/mount point information

a. df - returns info. using kilobytes

b. df -h - returns info. using megabytes/human readable (gigs/teray/etc.)

6. vmstat - reports on: processes, memory, paging, block I/O, traps, CPU activity

a. vmstat

b. vmstat -p /dev/hda1 - returns partitions stats for /dev/hda1 (/boot)

7. gnome-system-monitor - GUI, combining most system utilities

8. ls -ltr /proc

a. cat /proc/cpuinfo

9. kill PID - kills the process with a given PID

10. runlevel - returns runlevel information using 2 fields:

a. represents previous runlevel

b. represents current runlevel

**User/Group Management**

**Features:**

1. The ability to control users and groups

**Primary tools:**

1. useradd - used to add users and modify group membership

2. system-config-users

**Example:**

1. Create a user named 'student1' using 'useradd'

**Note:** Default user settings derive from: /etc/login.defs

a. useradd student1

b. set password for user 'student1': passwd student1

Default User Accounts DB: /etc/passwd

student1:x:501:501::/home/student1:/bin/bash

username:shadow\_reference:uid:gid:Description(GECOS):$HOME:$SHELL

**Note:** /etc/passwd is a world-readable file

**Note:** /etc/shadow now stores passwords in encrypted form

**Note:** /etc/shadow is NOT world-readable

Fields in /etc/shadow:

student1:$1$XSFMv2ru$lfTACjN.XxaxbHA0EkB4U0:13891:0:99999:7:::

1. username:

2. encrypted\_password:

3. Days\_since\_Unix\_epoch\_password\_was\_changed (01/01/1970)

4. Days before password may be changed

5. Days after which the password MUST be changed

6. Days before password is to expire that user is warned

7. Days after password expires, that account is disabled

8. Days since Unix epoch, that account is disabled

9. Reserved field (currently unused)

2. Modify user 'student1' to have password expire after 45 days

a. usermod

**Groups:**

1. groupadd - adds new group

2. groups - lists groups on the system: /etc/group

/etc/group - maintains group membership information

**Example:** Create a 'sales' group and add 'linuxusr' and 'student1' as members

1. groupadd sales

2. usermod -G sales linuxusr

3. usermod -G sales student1

**Note:** 2 types of groups exist:

1. Primary - used by default for a user's permissions

2. Supplemental - used to determine effective permissions

**Note:** use 'id' to determine the group information of user

Note: Create a new shell session to realize new group membership information

userdel/groupdel are used to delete users and groups, respectively

**File Types - Permissions – Symlinks**

**Features:**

1. The ability to restrict/control access to files

**Note:** 10 bits represent permissions for files (including directories)

**Note:** use 'ls -l' to examine permissions or GUI application like 'Nautilus'

-rwxrwxr-x 1 linuxusr linuxusr 681 Jan 13 11:31 regextest.pl

1st bit = file type. '-' = file, 'd' = directory

2nd - 4th bits = owner's permissions

r = read = 4

w = write = 2

x = execute = 1

- = none = 0

5th - 7th bits = group owner's permissions

r = read = 4

w = write = 2

x = execute = 1

- = none = 0

8th - 10th bits = everyone (world)

r = read = 4

w = write = 2

x = execute = 1

- = none = 0

**Example:**

1. Manipulate file permissions using 'chmod'

a. chmod -x regextest.pl

-rw-rw-r-- 1 linuxusr linuxusr 681 Jan 13 11:31 regextest.pl

rw = 6 or 4+2 for owner

rw = 6 or 4+2 for group owner

r = 4 for everyone else (world)

Octal notation: 664 for file 'regexetest.pl'

chmod 664 regextest.pl - removes execution for ALL users

chmod 775 regextest.pl - enables execution for ALL users

2. Ensure that 'regextest.pl' is rw by owner and no one else

a. chmod 600 regextest.pl

**Note:** File will now be rw by owner (linuxusr) and 'root'

3. Ensure that 'regextest.pl' is r by owner and no one else

a. chmod 400 regextest.pl && ls -l regextest.pl

**Note:** chmod supports string values, which represent octal values

chmod +/- x file

chmod +/- w file

chmod +/- r file

chmod +/- u+x file - updates owner's execute permissions on the file

chmod +/- o+x file - updates other's execute permissions on the file

chmod +/- g+x file - updates group's execute permissions on the file

chmod a+rwx = chmod 777

chown - permits changing of ownership of files

a. chown root regextest.pl - changes ownership to 'root'

b. chown linuxusr:sales regextest.pl - changes owner and group to 'linuxusr:sales'

**Example:**

Update 'regextest.pl' so that owner and group owner may modify the file

a. chmod 660 regextest.pl

**SETUID:**

**Features**:

1. ability to execute file as owner

chmod 4760 regextest.pl - this will ensure that the perl script always executes as the user 'linuxusr'

-rwsrw---- 1 linuxusr sales 787 Jan 13 16:08 regextest.pl

's' in the execute position means that the program will execute as that user

**SETGID:**

**Features**:

1. Ability to enforce permissions to a directory structure

mkdir /sales

chmod 2775 /sales

Create a file in the '/sales' directory as 'linuxusr'

seq 1000000 > linuxusr.1million.txt

**chgrp:**

Permits updating of group permissions

**Sticky Bit:**

**Features:**

1. Ability to ensure that users cannot delete others' files in a directory

drwxrwxrwt 23 root root 4096 Jan 13 15:05 /tmp/

/tmp - users cannot delete other user's files in '/tmp'

chmod 3777 /sales - ensures that /sales will not lose files from incorrect users

**Example:**

1. Set '/sales' using sticky bit and test

a. chmod 3777 /sales && ls -ld /sales OR chmod 777 /sales && chmod +t /sales

**Symlinks**

**Features:**

1. Provides shortcuts to files (including directories)

2. Provides hard links to inode (file system) locations

**Soft Links:**

1. ln -s source\_file target

a. ln -s ./regextest.pl lastscript.pl

**Note:** Soft links may span multiple file systems/hard drives

**Note:** Symlink count is NOT increased when using soft links

2. ln -s /home/linuxusr/testRH5/regextest.pl . - this will symlink (soft) to the /boot file system

**Note:** With soft links, if you change the name or location of the source file, you will break ALL of the symlinks (soft)

**Hard Links:**

**Features:**

1. The ability to reference the same inode/hard drive location from multiple places within the same file system

a. ln source target

ln regextest.pl ./testhardregextest.pl - creates a hard link

**Quotas**

**Features:**

1. Limits disk usage (blocks or inodes)

2. Tied to file systems (set on a per file system basis)

3. Can be configured for users and groups

**Steps to enable quota support:**

1. Enable quota support per file system in: /etc/fstab

a. defaults,usrquota,grpquota

2. Remount the file system(s)

a. mount -o remount /

b. use 'mount' to confirm that 'usrquota,grpquota' support are enabled

3. Create quota database files and generate disk usage table

a. quotacheck -mcug / - this creates /aquota.user & /aquota.group

b. quotacheck –mavug

4. Assign quota policies

a. edquota username - set blocks/inodes soft\_limits hard\_limit

edquota student1 - sets quotas for user 'student1'

export EDITOR=nano - to have edquota default to 'nano' editor

5. Check quotas

a. quota username

quota student1

**Note:** place 'quotacheck -avug' in /etc/cron.\*(hourly,daily)

6. Report on usage

a. repquota -a - this reports on usage

**Note:** The blocks are measured in 1K increments. i.e. 20000 blocks is roughly 20MB

**Basic Provisioning of Partitions and File Systems**

**Features:**

1. Ability to provision extra storage on-the-fly

**Steps:**

1. Identify available storage

a. 'fdisk -l' - returns connected storage

2. Create partitions on desired hard drive:

a. 'fdisk /dev/sdb' - interacts with /dev/sdb drive

b. 'n' - to add a new partition

c. 'p' - primary

d. '1' - start cylinder

e. '+4096M' - to indicate 4 Gigabytes

f. 'w' - to write the changes to the disk

**Note:** use 'partprobe partition (/dev/sdb1)' to force a write to a hard drive's partition table on a running system

**Note:** 'fdisk' creates raw partitions

3. Overlay (format) the raw partition with a file system

a. mke2fs -j /dev/sdb1 - this will write inodes to partition

4. Mount the file system in the Linux file system hierarchy:

a. mkdir /home1 && mount /dev/sdb1 /home1

b. mount OR df -h - either will reveal that /dev/sdb1 is mounted

**Note:** lost+found directory is created for each distinct file system

5. Configure '/home1' to auto-mount when the system boots

a. nano /etc/fstab and copy and modify the '/home' entry

**Swap Partitions & Files**

**Features:**

1. Extra, virtual RAM for the OS

**Steps:**

1. Identify current swap space

a. swapon -s - enumerates partitions and/or files, which constitute swap storage

b. free -m

2. Select target drive and provision swap partition

a. fdisk /dev/sdb

b. n

c. 2

d. 500

e. +512 (cylinder 562) - 63 cylinders are required for 512MB

f. t - change type

g. 82 - Linux Swap/Solaris

h. w - commit changes to disk

3. Create the swap file system on the raw partition: /dev/sdb2

a. mkswap /dev/sdb2

4. Enable swapping - publish the swap space to the kernel

a. swapon /dev/sdb2 - this enables swapping on /dev/sdb2

5. update /etc/fstab

a. /dev/sdb2 swap swap defaults 0 0

swapoff /dev/sdb2 - disables swapping on /dev/sdb2

**Example:**

1. Improve system performance by distributing swapping to /dev/sdb2

a. swapon /dev/sdb2

b. swapoff /dev/sda6

c. disable /dev/sda6 via /etc/fstab

**Create Swap based on File**

**Features:**

1. The ability to provision swap space based on a file, similar to pagefile.sys in Windows NT, etc., if you have no available disk space to partition.

2. Doesn't waste partitions

**Example:**

1. Create 512MB swap file

a. dd if=/dev/zero of=/home1/swapfile1 bs=1024 count=524288

b. mkswap /home1/swapfile1 - overlays swap file system

c. swapon /home1/swapfile1 - makes swap space available to the kernel

2. Ensure that when the system reboots, the swap file is made available to the kernel

a. nano /etc/fstab - /home1/swapfile1 swap swap defaults 0 0

3. Create 2GB swap file

a. dd if=/dev/zero of=/home1/swapfile2 count=2G

**Logical Volume Management (LVM)**

**Features:**

1. Ability to create volume sets and stripe sets

2. LVM masks the underlying physical technology (ATA,ATAPI,IDE,SCSI,SATA,PATA,etc.)

3. LVM represents storage using a hierarchy:

a. Volume groups

a1. Physical volumes (/dev/sda2, /dev/sdb2, etc.)

b. Logical Volumes

b1. File systems

3. LVM physical volumes can be of various sizes

4. Ability to resize volumes on the fly

**Note:** Volume groups join: physical volumes (PVs) and Logical Volumes (LVs)

**Steps to setup LVM:**

1. Create LVM partitions via fdisk or parted

a. fdisk /dev/sda, /dev/sdb, /dev/sdc

b. n

c. p

d. +10G

e. t - change to type '8e' (LVM)

f. w

g. partprobe /dev/sda

2. Create Physical Volumes using 'pvcreate'

a. pvcreate /dev/sda3 /dev/sdb3 /dev/sdc3

3. Create Volume Groups using 'vgcreate'

a. vgcreate volgroup001 /dev/sda3 /dev/sdb3 /dev/sdc3

**Note:** Volume groups can be segmented into multiple logical volumes

4. Create one or more Logical Volumes

a. lvcreate -L 10GB -n logvolvar1 volgroup001

b. lvcreate -L 10GB -n logvolusr1 volgroup001

5. Create File system on logical volume(s)

a. mke2fs -j /dev/volgroup001/logvolvar1

b. mke2fs -j /dev/volgroup001/logvolusr1

6. Mount logical volume

a. mkdir /var1

b. mount /dev/volgroup001/logvolvar1 /var1

c. mkdir /usr1

d. mount /dev/volgroup001/logvolusr1 /usr1

**Note:** Be certain to update: /etc/fstab so that volumes are mounted when the system reboots

**3-tiers of LVM display commands include:**

a. pvdisplay - physical volumes - represent raw LVM partitions

b. vgdisplay - volume groups - aggregate physical volumes

c. lvdisplay - logical volumes - file systems - mount here

**Rename of Logical Volume:**

1. lvrename volume\_group\_name old new - used to rename volumes

**Task:** Rename 'logvolvar1' to 'logvolopt1'

a. lvrename volgroup001 logvolvar1 logvolopt1

**Note:** LVM is updated immediately, even while volume is mounted

However, you must remount the logical volume to see the changes

b. umount /var1 && mount /dev/mapper/volgroup001-logvolopt1 /opt1

c. Update /etc/fstab

**Remove Logical Volume:**

**Example:** Remove 'logvolusr1' from the logical volume pool

a. umount /usr1

b. lvremove /dev/mapper/volgroup001-logvolusr1

c. use 'lvdisplay' to confirm removal

**Resize Logical Volume:**

**Example:** Grow (resize) 'logvolopt1' to 20GB

a. lvresize -L 20GB /dev/volgroup001/logvolopt1

b. lvdisplay - to confirm new size of logical volume

c. df -h - will still reveal the current size

d. Resize the file system to update the INODE table on the logical volume to account for the new storage in 'logvolopt1'

'resize2fs -f -p /dev/volgroup001/logvolopt1'

**Note:** You may resize file systems online if the following are met:

1. 2.6x kernel series

2. MUST be formatted with ext3

**Example:** Shrink (resize) 'logvolopt1' to 15GB

a. lvresize -L 15GB /dev/volgroup001/logvolopt1

b. lvdisplay

c. df -h

d. resize2fs -f -p /dev/volgroup001/logvolopt1

**Note:** online shrinking is not supported

e. df -h

**Note:** Check disk utilization prior to shrinking to reduce the risk of losing data

**LVM GUI Utility:**

system-config-lvm

**RAID**

**Features:**

1. The ability to increase availability and reliability of data

**Example:**

1. Create a RAID-1 Device (/dev/md0..n)

a. fdisk /dev/sdb - to create usable raw partitions

b. partprobe /dev/sdb - to force a kernel update of the partition layout of the disk: /dev/sdb

b. mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sdb5 /dev/sdb6

c. cat /proc/mdstat - lists active RAID (md) information

d. mke2fs -j /dev/md0 - overlays a file system on the RAID device

e. mount /dev/md0 /raid1

f. update: /etc/fstab

**Note:** use 'mdadm --query /dev/md0' to get information about a RAID device

**Note:** You may create RAID volumes/devices on a single or on multiple disks

Ideally, your RAID volumes should span multiple physical disks to improve:

a. reliability

b. performance

c. availability

2. Remove the RAID-1 device

a. umount /dev/md0

b. mdadm --manage --stop /dev/md0

3. Create a RAID-5 Volume

a. fdisk /dev/sdb - to create a partition number 7

b. partprobe /dev/sdb - to update the kernel's view of the partition table

c. mdadm --create /dev/md0 --level=5 --raid-devices=3 /dev/sdb5 /dev/sdb6 /dev/sdb7

d. watch cat /proc/mdstat - refreshes every 2 seconds

e. Overlay a file system: mke2fs -j /dev/md0

f. mount /dev/md0 /raid5

g. Test I/O to RAID-5 device

h. Update: /etc/fstab

**RPM**

**Features:**

1. Provides package management

a. Query

b. Install

c. Uninstall

d. Upgrade

e. Verify

2. Auto-verifies packages using GPG, MD5, SHA1SUMs

3. Automatically reports on unresolved dependencies

**'rpm'**

**Query:**

1. rpm -qa - dumps all installed packages

2. rpm -qa | wc -l - this dumps all packages and provides a count

3. rpm -qa | grep -i nano

4. rpm -qi nano - dumps info. about the 'nano' package as it's recorded in the local RPM database

5. rpm -qf /usr/bin/nano - dumps package membership info. for the 'nano' file

6. rpm -qpi http://192.168.1.101/RH5/i386/Server/dhcp-3.0.5-7.el5.i386.rpm - dumps info. about the uninstalled 'dhcp' package, which resides on the repository

7. rpm -ql package\_name - returns all included files

**Verify:**

1. rpm -Va - verifies ALL packages on the system, returning info. only if there are discrepancies from the original installation

2. rpm -Vf /usr/bin/nano

**Example:** Change '/usr/bin/nano' then verify

SM5....T /usr/bin/nano

S(file size), M(mode or permissions), 5(MD5), T(mod time)

3. rpm -Vp nano

**Install (Does NOT overwrite previous package):**

**Note:** Use this method to install a new version of the kernel

1. rpm -ivh \*.rpm

2. rpm -ivh http://192.168.1.101/RH5/i386/Server/dhcp-3.0.5-7.el5.i386.rpm

**Upgrade (Installs or overwrites existing package):**

1. rpm -Uvh \*.rpm

2. rpm -Uvh http://192.168.1.101/RH5/i386/Server/dhcp-3.0.5-7.el5.i386.rpm

**Freshen (Updates an existing package):**

**Note:** Will NOT install the package, if it doesn't exist locally

1. rpm -Fvh \*.rpm - freshens the current version of a package

**Removal:**

1. rpm -ev \*.rpm - removes a package

**Note:** removal process considers dependencies and will complain if the removal will break 1 or more packages. To get around this, use '--nodeps' option with 'rpm -ev --nodeps \*.rpm'

2. rpm -ev gftp

**Package Management GUI:**

1. Add/Remove Software

2. system-config-packages

**YUM Configuration**

**Features:**

1. The ability to centralize packages (updates)

**Installation & Setup:**

1. Install 'createrepo\*rpm'

2. Setup directory structure

a. /srv/www/linuxsrv.com/RH5/yum

3. Run 'createrepo /srv/www/linuxsrv.com/RH5/yum'

4. Publish the yum repository using HTTP

5. Configure yum client to use HTTP to fetch the RPMs

a. /etc/yum.conf

a1. ###Included as our first repository on the SUSE box###

[0001]

name=linuxsrvsuse1

baseurl=http://192.168.1.101/RH5/yum

**Note:** Ensure that about 3GBs are available for the yum repository

tar -cjvf yum\_metadata.bz2 repodata

**Yum Usage:**

1. Search for packages

a. 'yum search gftp'

2. Install packages - Requires RedHat GPG Key for RPMs

rpm --import http://192.168.1.101/RH5/i386/RPM-GPG-KEY-redhat-release

a. 'yum -y install gftp'

b. 'yum -y install gftp dhcp' installs 2 packages

3. Remove Package

a. 'yum -y remove gftp'

**Cron - Scheduler**

**Features:**

1. Scheduler

2. Rules (Cron entries) are based on times:

a. minute (0-59)

b. hour (0-23)

c. day of the month (1-31)

d. month (1-12)

e. day of the week (Sun,Mon,Tue, etc. OR 0-7)

f. command to execute (shell, perl, php, etc.)

3. Wakes up every minute in search of programs to execute

4. Reads cron entries from multiple files

5. Maintains per-user and system-wide (/etc/crontab) schedules

/etc:

cron.d/

cron.deny - denies cron execution by user

cron.monthly/ - runs jobs monthly

cron.weekly/ - runs jobs weekly

cron.daily/ - runs jobs daily

cron.hourly/ - runs jobs hourly

crontab - contains system-wide schedules

**Note:** '\*' wildcard in a time column means to run for all values

Per-user Crontabs:

Stored in: /var/spool/cron

**Example:**

1. Create a cron entry for the user 'student1'

a. su student1

b. crontab -e

c. create an entry, minus the name of the user

**Note:** 'crontab -l' - enumerates per-user cron entries

System-wide Crontab:

Stored in: /etc/crontab

**Example:**

1. Create a cron entry in: /etc/crontab

**Note:** 'crontab -l -u username' - enumerates per-user cron entries

**SysLogD**

**Features:**

1. Handles logging

2. Unix Domain Sockets (/dev/log)

3. Internet Sockets (UDP:514)

4. Ability to log to local and remote targets

Implanted as 'sysklogd' package

Primary configuration file: /etc/syslog.conf

**Standard syslog.conf file contains:**

1. Rules

a.facilities -> applications/daemons/network device/etc.

b. levels -> Importance of message

Range: 0-7

7 = emergency (less information)

6 = alert

5 = critical

4 = error

3 = warning

2 = notice

1 = info

0 = debug (more information)

2. Targets

a. file - /var/log/messages

b. tty - /dev/console

c. remote hosts - @IP\_ADDR\_of\_REMOTE\_HOST

'\*' = catchall/wildcard to mean any facility or level

'.none' = exclusion rule

'man syslog.conf' to learn about the support facilities.levels

**Example:**

1. Enable UDP logging for remote Cisco gateway (192.168.1.1)

a. netstat -nul | grep 514 - reveals UDP:514 listener

b. nano /etc/sysconfig/syslog

b1. 'SYSLOGD\_OPTIONS="-r"'

c. restart syslog and confirm UDP:514 listener

c1. confirm using 'netstat -nul | grep 514'

d. Configure the router using facility 'local0' and level 'info'

e. configure /etc/syslog.conf to accept 'local0.info'

f. restart or reload 'syslog'

**Log Rotation**

**Features:**

1. Rotation of logs based on criteria

a. size

b. age (daily, weekly, monthly)

2. Compression

3. Maintain logs for a defined period

/etc/logrotate.conf - primary (global) config file for all logs

-can be overridden by context-sensitive files. i.e. apache

run 'man logrotate'

/etc/logrotate.d - directory for logs to be rotated

-httpd - used to rotate Apache logs

/var/log/httpd/\*log {

missingok

notifempty

sharedscripts

postrotate

/bin/kill -HUP `cat /var/run/httpd.pid 2>/dev/null` 2> /dev/null || true

endscript

}

**Example:** Setup rotation rule for Cisco log

1. Create entry in: /etc/logrotate.d based on /etc/logrotate.d/syslog

2. Modified the entry to rotate based on new criteria

3. Rotated using: 'logrotate /etc/logrotate.conf'

**Note:** Force using: 'logrotatate -f /etc/logrotate.conf'

**Common Network Utilities**

**Features:**

1. Useful for basic troubleshooting

**PING:**

**Features:**

1. ability to communicate with hosts using ICMP

a. PING sends ICMP echo-requests

b. PING expects to receive ICMP echo-replies

**Example:** PING some hosts and evaluate the output

1. ping localhost (127.0.0.1)

2. ping -c 3 localhost - sends 3 ICMP echo-requests

**Note:** 'ping localhost' performs name resolution using /etc/hosts

/etc/hosts stores static name-to-IP mappings

**Note:** 127.0.0.0/8 is fully-reserved to the loopback adapter of ALL IPv4 hosts

3. ping -c 3 192.168.1.99

4. ping -c 3 -i 3 192.168.1.99 - delays PINGs to 3 seconds apart

**Note:** PING defaults to a standard 1-second interval

**Note:** Firewall(s) may block ICMP traffic, causing PING to fail

**TELNET:**

**Features:**

1. Great for basic TCP port diagnosis

**Example:**

1. Connect to TCP ports on various hosts

a. telnet 192.168.1.101 22

b. telnet www.linuxsrv.com 80

**NETSTAT:**

**Features:**

1. Provides network connection information from /proc/net/\*

**Example:**

1. Return useful information for various protocols

a. netstat

b. netstat -a - returns all protocols/sockets

c. netstat -ntlp - returns all TCP LISTENERS without name resolution

d. netstat -nulp - returns all UDP LISTENERS without name resolution

**Note:** netstat uses /etc/services to translate ports to names

**Note:** 0.0.0.0:514 - this means that Syslog will accept traffic to any of the defined IP addresses/interfaces on the system

e. netstat -ntp - returns established connections (sockets)

f. netstat -rn - returns the routing table

**ARP:**

**Features:**

1. Resolves layer-2 (OSI model) MAC addresses to layer-3 IP addresses

**Example:**

1. Examine MAC addresses using: ifconfig and arp

a. ifconfig - returns our local MAC addresses

Link encap:Ethernet HWaddr 00:02:B3:98:41:08

b. arp -a - returns MAC to IP mappings

**Note:** When 2 TCP/IP hosts communicate, ARP is performed to translate the IP address (v6/v4) to a MAC address.

**Note:** If a one or more routers separate the communicating hosts, then the MAC address of the default router's (gateway's) interface is stored by each client

**IPv4 Configuration & Network Settings**

**Network Support:**

1. Boot system into a multi-user mode

2. /etc/modprobe.conf - contains alias and reference to module(s) to be loaded in order to provide networking

3. Linux decides if the interface is DHCP or static by viewing the contents of:

a. /etc/sysconfig/network - networking=yes|no, IPv6\_Support, Default Gateway, etc.

b. /etc/sysconfig/network-scripts/ifcfg-eth0 - contains ifup, ifdown, and ifcfg-\* scripts

c. /etc/init.d/network - main service

service network status - checks networking

system-config-network-\* - network interface configuration

**Note:** Either update your net configuration manually from the shell, or using the 'system-config-network\*' tools to avoid losing settings

/etc/resolv.conf - DNS configuration file

/etc/hosts - static list of hosts

**IPv4 Aliases:**

1. ifconfig eth0:1 192.168.1.11

2. ifconfig eth0:2 10.168.1.11

**Note:** To ensure that aliases persist do the following:

1. cp /etc/sysconfig/network-scripts/ifcfg-eth0 ./ifcfg-eth0:1

2. Modify ifcfg-eth0:1 to reflect aliased IP

**Note:** Aliases do NOT work with DHCP interfaces

ifconfig eth0:2 del 10.168.1.11 - removes the virtual interface

**IPv6 Config:**

**Features:**

1. Auto-configured by default gateway (router)

2. fe80:: - link-local address (loopback/local subnet address)

3. 2002:: - 6to4 address, that can be configured based on IPv4 embedded address, using HEX notation

ping6 -I eth0 fe80::

traceroute6 - used to trace routes on IPv6 networks

**Kernel Upgrade**

**Features:**

1. Provision of updated/patched kernel

**Example:**

1. Update the kernel

a. use 'uname -a' to reveal current version

b. use 'rpm -qa | grep -i kernel' - to reveal installed version

c. cat /etc/grub.conf -> /boot/grub/grub.conf - "" ""

2. Proper installation method is as follows:

a. 'rpm -ivh kernel\*rpm' - install a separate version

**Note:** Install the following kernel packages if necessary:

a. kernel-devel\* - if module compilation is necessary

b. kernel-headers\* - if recompilation is necessary

**Install:**

a. rpm -ivh kernel-2.6.18-53.el5.i686.rpm

**Note:** This will update GRUB (/boot/grub/grub.conf)

**Note:** Will also place the new kernel in the /boot file system

**Examine traces in:**

a. /boot

b. /boot/grub/grub.conf

3. Remove traces of former kernel using 'rpm -e [--nodeps]'

a. kernel-2.6.18-8.el5 - removes older version

b. kernel-headers-2.6.18-8.el5 - force remove ignoring dependencies 'rpm -e --nodeps kernel-headers-2.6.18-8.el5'

c. kernel-devel-2.6.18-8.el5

4. Install new 'kernel-headers' and 'kernel-devel' packages using YUM:

a. yum -y install kernel-headers

b. yum -y install kernel-devel

5. Confirm that the 3 'kernel-\*' packages are installed:

a. rpm -qa | grep kernel

**Note:** Removal of older kernel-\* packages cleans up:

a. /boot

b. /boot/grub/grub.conf (menu.lst)