**YUM commands:**

Yum search <pkg name> : To search for a particular software or application

yum info <pkg name> : To get info about a package or software

yum install <pkg name> : Install a package from a repository to your system

yum update <pkg name>: Update one or all packages on your system

yum reinstall<pkg name> : Reinstall the current version of a package

remove: remove a package (and possibly dependencies) from your system

swap: Remove one package and install another

yum swap ftp lftp (Remove ftp package and install lftp package)

yum list installed <pkg name> :To check if a package is installed

yum repolist: Display enabled software repositories

yum deplist <pkg name> : Display dependencies for a package

history: View and use yum transactions

yum history list: List all yum install, update and erase actions

yum history info 3 : Show details of yum transaction 3

yum history undo 3: Undo the yum action from transaction 3

yum history redo 3: Redo the undone yum action from transaction 3

clean : Clear out cached package data

yum clean packages: Delete packages saved in cache

yum clean all: Clean out all packages and meta data from cache

**RPM commands:**

rpm -ivh <rpm-file> : Install the package

rpm -Uvh <rpm-file>: Upgrade package

rpm -ev <package> : Erase/remove/ an installed package rpm -ev mozilla-mail

rpm -ev --nodeps <package> : Erase/remove/ an installed package without checking for dependencies rpm -qa : Display list all installed packages

rpm -qi <package>: Display installed information along with package version and short description

rpm -qf </path/to/file>: Find out what package a file belongs to i.e. find what package owns the file

rpm -qc <package-name>: Display list of configuration file(s)for a package

rpm -qcf </path/to/file> : Display list of configuration files for a command

rpm -qa :last: Display list of all recently installed RPMs

rpm -qpR <.rpm-file>

rpm -qR <package> : Find out what dependencies a rpm file has

rpm :test :e links: to test the switch before applying, if warnings occur attention required else remove the links

**APT Commands:**

apt-get

sudo apt-get update : updates package list

sudo apt-get upgrade : installs new updated packages

sudo apt-get dist-upgrade : smart upgrade to new packages

sudo apt-get install <package name> : installs package

sudo apt-get check : check for broken packages

sudo apt-get autoremove : remove any orphaned packages

apt-cache

apt-cache search <string> : search name and description

apt-cache show <package> : all the information on a package

apt-cache showpkg <package> : all dependencies

apt-cache depends <package> : what it depends on

apt-cache rdepends <package> : what depends on it

apt-file

sudo apt-get install apt-file : needs to be installed

sudo apt-file update : sync with all repositories

apt-file search <string> : searches for string, local and remote

apt-file list <package> : list contents of package even if not installed

**DPKG Commands:**

dpkg -i <.deb package>: Install the package

dpkg -i <.deb package>

dpkg -R <Directory-name>: Install all packages recursively from directory

dpkg -r <package>: Remove/Delete an installed package except configuration files

dpkg -P <package>: Remove/Delete everything including configuration files

dpkg -l:List all installed packages, along with package version and short description

dpkg -l <package>: List individual installed packages, along with package version and short description

dpkg -L <package>: Find out files are provided by the installed package i.e. list where files were installed

dpkg -c <.Deb package>:List files provided (or owned) by the package i.e. List all files inside debian .deb package file, very useful to find where files would be installed

dpkg -S </path/to/file>:Find what package owns the file i.e. find out what package does file belong

dpkg -p <package>:Display details about package package group, version, maintainer, Architecture, display depends packages, description etc

dpkg -s <package> | grep Status:Find out if Debian package is installed or not (status)

**Command Line Section**

**File Commands:**

Linux/Unix is case sensitive

. represents current directory

.. represents one dir up

./hello.sh is same as giving the whole path as /home/user/hello.sh

ls : directory listing

ls –l: shows along with directory ownership, permissions and sizes

ls <dir> : lists the contents of specific directory

ls –R <dir>: lists along with contents of subdirectories under a specific directory

ls -al : formatted listing with hidden files

ls –Sl : sorts with file size being largest at the top

ls –t : sort by last time modified displaying the newest first

which <command> : to know where the command resides in

env: list environment variables

set : lists out all env variables in alphabetical order

**Quoting:**

echo “…” or echo $.... : prints the given string or env variable

echo “ current user is $LOGNAME” prints current user is user

echo ‘current user is $LOGNAME’ prints current user is $LOGNAME

echo “ Borrow $3.50 “ prints Borrow .50

echo “Borrow \$3.50” prints Borrow $3.50 ( \ disables any special character functionality)

ls <dir> does not take spaces in folder names so use ls ‘ dir name’ to take it literally or use ls dir \name/

\ gives option to enter lengthy commands eg: ls \

**Basic:**

cd dir : change directory to dir

cd : change to home

cd - : return to last directory you were in

cd ~ : change to home dir of currently logged in user

pwd : show present working directory

mkdir dir : create a directory dir -p makes parent dir along with sub dir

rm file : delete file

rm -r dir : delete directory dir

rm -f file : force remove file

rm -rf dir : force remove directory dir \*

cp file1 file2 : copy file1 to file2

cp -r dir1 dir2 : copy dir1 to dir2; create dir2 if it doesn't exist

mv file1 file2 : rename or move file1 to file2, if file2 is an existing directory, moves file1 into directory file2

ln -s file link : create symbolic link link to file

touch file : create or update file

less : view a text file with arrow keys navigation

cat > file : places standard input into file

more file : output the contents of file

head file : output the first 10 lines of file

-n : gives n lines with head or tail

tail file : output the last 10 lines of file

tail -f file : output the contents of file as it grows, as new data is being written in the file, starting with the last 10 linescut: remove text from file and print specified fields to screen

-d “ ,” : specify delimiter to use

-f 6- filename : specify field to print …. Prints from fields 6 based on delim given

sort : sort content of file alphabetically based on first character in file

-n: sorts contents of file numerically

wc- word count, prints number of lines, words, characters in file

-l: print number of lines in file

-w : prints number of words in file

> : redirect output to new location, if output goes to file, replace the contents with output

>> : redirect output to new location, if output goes to file, append it to the existing contents

whoami : display your currently logged in user

su <username> : substitute user, change to another user account on the system

exit : leave a shell env that you are logged in

init 6 or reboot: legacy command for rebooting a system

init 0: legacy command to shut down a system

top : displays running processes on a system

uname or uname –s: displays the name of system kernel

uname –r: display the kernel release umber

uname –v: display the kernel build version

uname –m : display the machine type

uname –o : display the name of OS

uname –a : display all info uname can show

cat .bash\_history : displays the content of the hidden file .bash\_history which contains the log of commands entered at the bash prompt

HISTFILESIZE : env var thet specifies how many lines of history to keep

HISTCONTROL: env var that modifies Bash’s history behavior like ignore duplicates

history : prints out commands saved in .bash\_history with each command numbered

**Difference between login shell and non-login shell:**

**Login shell** is started after a successful login, using */bin/login*, by reading the */etc/passwd* file. Login shell is the first process that executes under our user ID when we log in to a session. For example, for [Bash](http://gnu.org/software/bash/) shell it will be ***-bash***.

When [Bash](http://gnu.org/software/bash/) is invoked as a Login shell;  
→*Login process* calls ***/etc/profile***  
→/etc/profile calls the scripts in ***/etc/profile.d/***  
→*Login process* calls ***~/.bash\_profile*** or ***~/.bash\_login*** or ***~/.profile*** (the first file found is used and others are ignored)  
→~/.bash\_profile calls ***~/.bashrc***  
→~/.bashrc calls ***/etc/bashrc***

**/etc/profile:** system wide environment and startup programs during a login shell

**/etc/profile.d/** : location of extra env setup scripts

**.bash\_profile** : used to set user specific shell env preference

**.bashrc** : non-login file that stores user specific functions and aliases

**/etc/bashrc** : system-wide functions and aliases

**.bash\_logout** : anything within it runs when the user logs out

A **Non login** shell is started by a program without a login. In this case, the program just passes the name of the shell executable. For example, for a [Bash](http://gnu.org/software/bash/) shell it will be simply ***bash***.

When bash is invoked as a Non login shell;  
→*Non-login process(shell)* calls ***~/.bashrc***  
→~/.bashrc calls ***/etc/bashrc***  
→/etc/bashrc calls the scripts in ***/etc/profile.d/***

A Non login shell can be recognized by the following procedure.  
Execute the below command in shell.

**#** echo $0

If the output is the name of our shell, does not prepend by a dash, then it is a Non login shell.  
For example ***bash***, ***su*** etc.

export <variable> : exports variable and value to other shells

**Globbing:**

\* Matches any character zero or more times, except for / eg: ls \*.txt or ls test\*

\*\* Matches any character zero or more times, including /

? Matches any character except for / one time eg: ls ????.txt matches 4 characters like fire.txt

[abc] Matches any characters inside the brackets. For example, [abc] would match the characters a, b or c, and nothing else.

[^abc] To exclude characters, Matches anyone character except those in the list, case sensitive

[0-9] To match with numbers in the list

ls [Pp]\* Matches to anyfile starting with either P or p

Formatting Commands:

[user@hostname ~]$

user- logged in user name

hostname- machine name to identify it in a network of machines

~ - in the home directory

$ at the prompt – regular user

# at prompt –root user

Command – what to do?

Options- how to do it?

Argument- what to do it on?

locate: searches a local database of files and folders looking for items that match the search criteria

find: searches the files system for files that match the search criteria use globbing within ‘ ‘

whereis: locates binary,source and/or manual pages for a command

To Get Help

info <cmd>

man <section number> <command>: manual pages

whatis <command> or man –f <cmd> : lists summaries and related man pages based on search term

apropos <keyword> or man –k <cmd> : searches man pages for appearances of keyword

cd /usr/share/doc ---- absolute path given from first home dir

cd dir/dir1 ----- relative path from current dir

cd ../../.. takes 3 directories up from current location

**Archives and Compression:**

tar -czf <filenameof tar > <sourcefilename>

tar –xzf <tarfilename> ---- to extract

tar : manipulate archive files

-c: create a new archive

-z: pass the archive through gzip compression

-j : pass the archive through bzip2 compression

-f: filename of archive to create

-x : extract an archive

-t : to list the contents of archive

Zip: create a new compressed file

-r: create a compressed file of dir and its contents recursively

rm –r\*.tar : to remove the tar files in a dir

To list contents of a tar archive use: tar -tf archiveName.tar

To extract a tar archive use: tar -xf archiveName.tar

To extract a directory into a folder: tar -xf archiveName.tar -C ./directory

Filename.tgz or filename.tar.gz to name a file archive

zip –r <filename.zip> <filename>

unzip <filename.zip>

gzip \*.gz ---- create a gzip archive same as bzip2

gunzip ---- extract a gzip archive same as bunzip2

**Pipes and RegEx**:

grep <pattern> <filename>: show the lines in a file that match a given pattern

-i : perform a case –insensitive search

-v: return lines that do not contains the pattern

-r: perform a recursive search

| : pipe to sed output of one command as input to another command

Command1 | command 2

Regex:

^ : search at the beginning of a line eg : grep –i ‘^republic’

$ : search at the end of line

. : stands in for a single character

grep ‘ [abc]’ or grep ‘[^abc]’

\*: match zero or more of preceding characters or expression

nano <filename> uses ctrl + o to save file

ctrl + x to save and exit

ctrl + g to view help menu

ctrl + w to search for text in the file

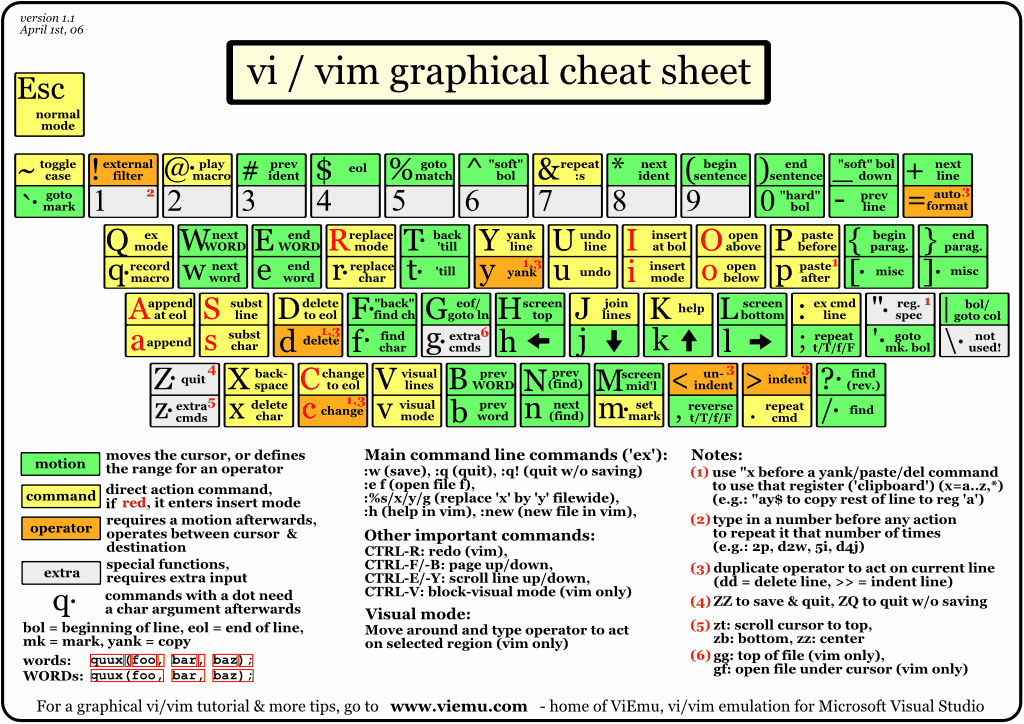
Alt + 6 to copy a line of text

Ctrl + u to paste

Ctrl +t to spell check

Ctrl + \ to search and replace text

Vi /vim



Shell script

Script: to perform a series a commands for more than once

First line is #!/bin/bash ------------ to tell the machine to use bash as interpreter

# ---------- comment line

To execute the script--------- ./daily.sh

Hardware:

cat /proc/cpuinfo : view the cpuinfo file to gather details about processor

free : view RAM statistics for the system

-m : show output in MB

-g : in GB

dmidecode : show details about motherboard, BIOS, processor and RAM

lsblk: view all block devices attached to the system

df : disk free space

-h : in human readable format

# Display your currently running processes

ps

# Display all the currently running processes on the system.

ps –ef

-u <username> : list the processes running on the system

-e: list all processes running on the system from all users

-H or --forest: list all processes with intended output, showing the hierarchy

-f: full format listing, including command arguments

# Display process information for processname

ps -ef | grep processname

# Display and manage the top processes

top

# Interactive process viewer (top alternative)

htop

# Kill process with process ID of pid

kill pid

# Kill all processes named processname

killall processname

# Start program in the background

program &

# Display stopped or background jobs

bg

# Brings the most recent background job to foreground

fg

# Brings job n to the foreground

fg n

pkill <pid> : to kill a process with pid

userdel <username> : to delete a user

IP address: unique id to identify a machine in a network

Subnet Mask(netmask address): more accurate descp of the network, the machine is attatched to

Gateway Address: IP of router which connects one network to other (usually internet), responsible for communication between hosts of different networks

DNS Server Address: To translate the name of a domain/machine to IP address

Local IP Address: 127.0.0.1

MAC (hardware) address: unique id for network interface adapter

$ip addr show : to view ip address

Gives output with link/loopback: localhost ip, eth0 inet : ip address, MAC with link/ether

$ifconfig gives eth0: ip address, loopback: localhost ip, netmask address, inet: ipv4 address, inet6: ipv6 address

$ip route show : view default route(gateway)

$host abc.com : Test DNS hostname resolution

Ping : test network connectivity

who: see who is logged into the system

w: see who is logged in with more details

id: view user and group id’s of a specified user

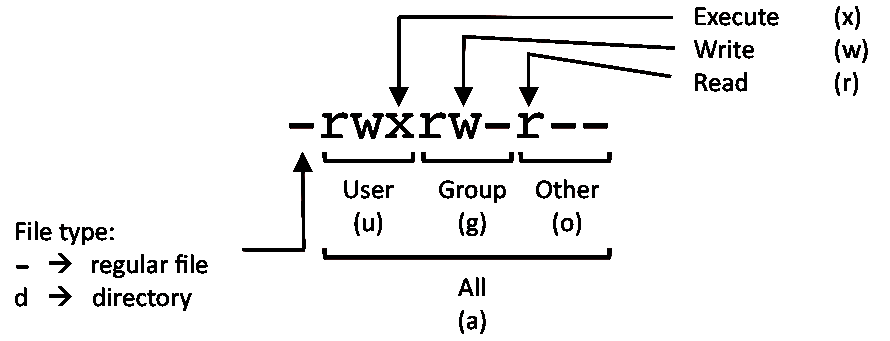
/etc/passwd file : primary config file for all users on a system

/etc/group: for groups on a system

groupadd <groupid> : add a new group to system

useradd or adduser : to add new user to system –m to create home dir

**File Permissions:**



U = User

G = Group

W = World

r = Read =4

w = write=2

x = execute=1

- = no access=0

chown: change ownership of a file or dir

chmod: change mode of file, effecting permissions