Linux Guide

Bash shell prompt:

# means you are using as superuser (root)

$ means you are a normal user

Command format

Command [option] [arguments]

Example:

```
$ ls -l /boot
```

Smart tricky commands

<tab>	: complete the commands with sufficient uniqueness
history	: display command history
alt	: filter through previously passed arguments
!<no. of previous command>	: recall previous command by number
!!	: execute the previous command
^r	: search command history with specific some text part
!$	: recall previous argument
.	: current directory
..	: parent directory of current directory
~	: current user home directory

Some useful commands to know user details

```
$whoami
$groups
$id
$grep <user_name> /etc/passwd
```

Gives user id’s and related user and group with directory information.

$grep <user_name> /etc/group

Gives group name, id and member details

$cp	: copy
$cp -r	: copy directory
$mv	: move file
$rm -r	: remove directory

Files

Absolute path: (/path)

Relative path: (path/name_location)

$ls file*

$ ls file{a..g}

$ date +%F

$ tail -1 /path/file   //show last line of file

Vim:

$vimtutor

:q	to exit
:help	
Press ‘I’	insert mode
Press ‘:’	command mode
:w path/file_name	save the file
yy	copy a line

<number>p	number times past a line
dw	delete word
x	deletes individual character
d$	deletes to end of the line
u	undo
:wq	save and quit in command mode
GG	save and quite in insert mode
o	take your file pointer at end of the lines
:q!	quit without saving
Esc	alter insert mode to command mode

Kill:

```
$ kill -signal_number p_id
```

Signal number-

‘1’ -> ‘HUP’ (hangup process or reinitialization without termination of process)

‘2’ -> ‘INT’ (keyboard interrupt, process can be blocked or handled. (Ctrl+c))

‘3’ -> ‘QUIT’ (keyboard quit, similar like ‘INT’ also produce process dump at termination. (Ctrl+\))

‘9’ -> ‘KILL’ (Kill, unblockable, causes abrupt program termination)

‘15’ -> ‘TERM’ (Terminate, default termination of program, allows self-cleanup)

‘18’ -> ‘CONT’ (Continue, always resume the process)

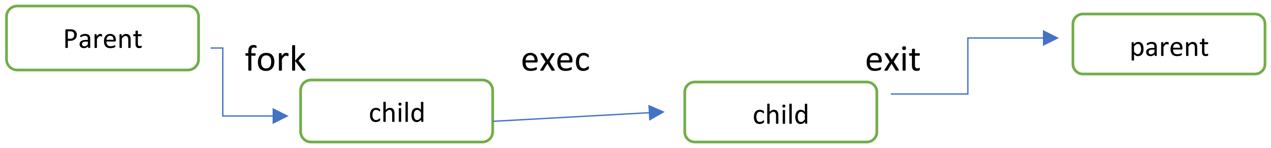
‘19’ -> ‘STOP’ (Stop, unbloackable, suspends the process)

‘20’ -> ‘TSTP’ (Keyboard stop (Ctrl+z))

//// funny commands

```
$ xeyes &                    // eye boles which follows your curser
```

Process:



Jobs:

$jobs //display running jobs

$ps j // display job information

$bg %job_no. // start a process in background

$fg %job_no. // bring a process to foreground

Permissions:

Sticky bit: ‘t’ sign in place of ‘x’, for other

# chmod o+t /path/file

Setgid bit: ‘s’ sign in place of ‘x’, for group owner

# chmod g+s /path/file

Setuid bit: ‘s’ sing in place of ‘x’, for user owner

# chmod u+s /path/file

#

$ ls -ld /path/file

Output gives drwxr-rw-r: user_name group_name time argument

First d – directory sign, rwx – user permission, rw – group permission, r – other permission.

Monitoring process activities

$uptime

Output provides load average: last_1min, last_5min, last_15min

Overload detection:

If (last_1min / total_no_cpu) > 1) then system is overloaded.

```
$ top       //running processes detailed info
```

Press ‘k’ and then press ‘enter’ will kill the highly %cpu process.

Press ‘q’ to close ‘top’ command.

```
$ps aux     // provide process details
```

Action:	**Required permissions**:
View the file contents	r
Change file contents	w
Execute file	rx
Change directory	x
List the directory content	rx
Create/delete file inside directory	w

Users:

```
# useradd user_name                    // add new user
# groupadd group_name                  // add new group
# usermod -aG group_name user_name          //add a user_name as a member of the  group_name
# groupmod -g new_id_no group_name          // assign a group id number to group_name
# groupdel group_name                 // delete group
```

/etc/passwd contains user information’s

```
# usermod -u new_user_id -c “new_user_argument” -s /bin/new_shell
user_name

//  it change user id, argument and default shell

# grep user_name /etc/passwd           // provide user details

# userdel user_name                    // delete user

# passwd user_name                     // setting up new user account password

# su -                                 // login as root user (super user)

# sudo -i                              // login as root

# logout                               // shortcut (Ctrl+d)
```

YUM:

```
$ yum info package

$ yum provides /path/package/package_config_file      //package information

# yum install -y package_name                          // install package

# yum remove package_name

# yum update package_name

# yum grouplist                                        // software package groups

# yum group info “group_name”                          // all info about software
group, in output ‘+’ package is part of this group and install by default with group
installation.

# yum group install -y “group_name”

# yum history                                          // provide all transactions
history with transaction number.

# yum history undo transaction_number                  // undo that specific
transaction
```

ssh:

$ ssh user_name@server_name // login to another server

$ ls .ssh // key storing files

$ ssh-keygen // create new ssh private (id_rsa) and public (id_rsa.pub) key in current server

$ ssh-copy-id user_name@server_name //store your public key file in another server machine.

File permissions

U – owning user

G – owning group

O – other

+/- - adding/taking_away to what is there already

= - setting permission irrespective of the current state

r – 4

w – 2

x – 1

$ chmod 764 /path/file_name // permission provided as rwx-rw-r

7 – rwx or 4+2+1

6 – rw or 4+2

4 – r or 4

# chown user_name /path/file //transferring ownership

# chgrp group_name /path/file // change owning group

# chown user_name:group_name /path/file // change user and group both