

40 Basic Linux Commands used Frequently

By Coding-Bootcamps.com

In this tutorial, we will show the very basic Linux commands with examples that are frequently used to get you more familiar with the Linux command line. To be an expert in Linux first step for a beginner would be to start learning the basic commands.

Command---options---arguments (on which command acts)

The command is followed by options (optional of course) and a list of arguments. The options can modify the behavior of a command. The arguments may be files or directories or some other data on which the command acts. Every command might not need arguments. Some commands work with or without them (e.g. 'ls' command). The options can be provided in two ways: full word options with -- (e.g. --help), or single letter options with - (e.g. -a -b -c or multiple options, -abc).

Syntax

The commands in Linux have the following syntax:

```
$command options arguments
```

Outline

- I. Linux Basic Commands
- II. Linux Filesystem commands
- III. Creating files and directories
- IV. Copy, move and remove commands
- V. Other file commands
- VI. Text Editors
- VII. Useful commands

I- Linux Basic Commands

Let's start with some simple commands.

1) pwd command

'pwd' command prints the **absolute path** to current working directory.

```
$ pwd  
/home/raghu
```

2) cal command

Displays the calendar of the current month.

```
$ cal
July 2012
Su Mo Tu We Th Fr Sa
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
```

'cal' will display calendar for the specified month and year.

```
$ cal 08 1991
August 1991
Su Mo Tu We Th Fr Sa
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31
```

3) echo command

This command will echo whatever you provide it.

```
$ echo "linuxide.com"
linuxide.com
```

The 'echo' command is used to display the values of a variable. One such variable is 'HOME'. To check the value of a variable precede the variable with a \$ sign.

```
$ echo $HOME
/home/raghu
```

4) date command

Displays current time and date.

```
$ date
Fri Jul 6 01:07:09 IST 2012
```

If you are interested only in time, you can use 'date +%T' (in hh:mm:ss):

```
$ date +%T
01:13:14
```

5) tty command

Displays current terminal.

```
$ tty
/dev/pts/0
```

6) whoami command

This command reveals the user who is currently logged in.

```
$ whoami
```

```
raghu
```

7) id command

This command prints user and groups (UID and GID) of the current user.

```
$ id
```

```
uid=1000(raghu) gid=1000(raghu)
groups=1000(raghu),4(adm),20(dialout),24(cdrom),46(plugdev),112(lpadmin),120(
admin),122(sambashare)
```

By default, information about the current user is displayed. If another username is provided as an argument, information about that user will be printed:

```
$ id root
```

```
uid=0(root) gid=0(root) groups=0(root)
```

8) clear command

This command clears the screen.

Help command

Nobody can remember all the commands. We can use help option from command like

9) help option

With almost every command, '--help' option shows usage summary for that command.

```
$ date --help
```

```
Usage: date [OPTION]... [+FORMAT] or: date [-u|--utc|--universal]
[MMDDhhmm[[CC]YY][.ss]] Display the current time in the given FORMAT, or set
the system date.
```

10) whatis command

This command gives a one line description about the command. It can be used as a quick reference for any command.

```
$ whatis date
```

```
date (1) - print or set the system date and time
```

```
$ whatis whatis
```

```
whatis (1) - display manual page descriptions
```

11) Manual Pages

'--help' option and 'whatis' command do not provide thorough information about the command. For more detailed information, Linux provides man pages and info pages. To see a command's manual page, man command is used.

```
$ man date
```

The man pages are properly documented pages. They have following sections:

NAME: The name and one line description of the command.

SYNOPSIS: The command syntax.

DESCRIPTION: Detailed description about what a command does.

OPTIONS: A list and description of all of the command's options.

EXAMPLES: Examples of command usage.

FILES: Any file associated with the command.

AUTHOR: Author of the man page

REPORTING BUGS: Link of website or mail-id where you can report any bug.

SEE ALSO: Any commands related to the command, for further reference.

With -k option, a search through man pages can be performed. This searches for a pattern in the name and short description of a man page.

```
$ man -k gzip
gzip (1) - compress or expand files
lz (1) - gunzips and shows a listing of a gzip'd tar'd archive
tgz (1) - makes a gzip'd tar archive
uz (1) - gunzips and extracts a gzip'd tar'd archive
zforce (1) - force a '.gz' extension on all gzip files
```

12) Info pages

Info documents are sometimes more elaborate than the man pages. But for some commands, info pages are just the same as man pages. These are like web pages. Internal links are present within the info pages. These links are called nodes. Info pages can be navigated from one page to another through these nodes.

```
$ info date
```

II- Linux Filesystem commands

13) Changing Directories Command

```
$ cd [path-to-directory]
```

Change the current working directory to the directory provided as argument. If no argument is given to 'cd', it changes the directory to the user's home directory. The directory path can be an absolute path or relative to current directory. **The absolute path always starts with /.** The current directory can be checked with 'pwd' command (remember?):

```
$ pwd
/home/raghu
$ cd /usr/share/
$ pwd
/usr/share
$ cd doc
$ pwd
/usr/share/doc
```

In the first 'cd' command, absolute path (/usr/share) is used, and with second command, relative path (doc) is used.

14) Listing File And Directories Command

```
$ ls [files-or-directories]
```

List files and/or directories. If no argument is given, the contents of current directory are shown.

```
$ ls
example file1.txt file2.txt file3.txt
```

If a directory is given as an argument, files and directories in that directory are shown.

```
$ ls /usr
bin games include lib lib64 local sbin share src
```

'ls -l' displays a long listing of the files.

```
$ ls -l
total 4
drwxr-xr-x 2 raghu raghu 4096 2012-07-06 12:52 example
-rw-r--r-- 1 raghu raghu 0 2012-07-06 12:52 file1.txt

-rw-r--r-- 1 raghu raghu 0 2012-07-06 12:52 file2.txt
-rw-r--r-- 1 raghu raghu 0 2012-07-06 12:52 file3.txt
```

In this long listing, the first character is 'd' or '-'. It distinguishes between file types. **The entries with a '-' (dash) are regular files, and ones with 'd' are directories.**

The next 9 characters are permissions ('rwxr-xr-x' in first listing). The number following the permissions is the link count. Link count follows user and group owner. In the above example, the file owner is 'raghu' and group owner is 'raghu' as well. Next is the size of the file. And then time stamp before the name of file (or directory).

By default, hidden files or directories are not shown, to see hidden files as well, -a option is used.

Hidden files in Linux start with a period sign (.). Any file that starts with a period is hidden. So, to hide a file, you just need to rename it (and put a period before it).

```
$ ls -la odesk
total 16
drwxr-xr-x 4 raghu raghu 4096 2012-07-06 13:46 .
drwxr-xr-x 11 raghu raghu 4096 2012-07-06 13:15 ..
drwxr-xr-x 2 raghu raghu 4096 2012-07-06 12:52 example
-rw-r--r-- 1 raghu raghu 0 2012-07-06 12:52 file1.txt
-rw-r--r-- 1 raghu raghu 0 2012-07-06 12:52 file2.txt
-rw-r--r-- 1 raghu raghu 0 2012-07-06 12:52 file3.txt
drwxr-xr-x 2 raghu raghu 4096 2012-07-06 13:46 .hiddendir
-rw-r--r-- 1 raghu raghu 0 2012-07-06 13:46 .hiddenfile1.txt
-rw-r--r-- 1 raghu raghu 0 2012-07-06 13:46 .hiddenfile2.txt
```

If you want to see the properties of a directory instead of the files contained in it, use -d (with -l) option:

```
$ ls -ld odesk/
drwxr-xr-x 4 raghu raghu 4096 2012-07-06 13:46 odesk/
```

III- Creating files and directories

15) mkdir command

To create a directory, the 'mkdir' command is used.

```
$ mkdir example
$ ls -l
total 4
drwxr-xr-x 2 raghu raghu 4096 2012-07-06 14:09 example
```

16) touch command

For creating an empty file, use the touch command.

```
$ touch file1 file2 file3
$ ls -l
total 4
drwxr-xr-x 2 raghu raghu 4096 2012-07-06 14:09 example
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file1
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file2
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file3
```

If a file already exists, touch will update its time stamp. There are a lot of other methods to create a new file, e.g. using a text editor like vi or gedit, or using redirection. Here is an example of creating a file using redirection:

```
$ ls -l /usr > usrlisting
$ ls -l
total 8
drwxr-xr-x 2 raghu raghu 4096 2012-07-06 14:09 example
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file1
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file2
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file3
-rw-r--r-- 1 raghu raghu 491 2012-07-06 14:23 usrlisting
```

A file named usrlisting is created in this example.

IV- Copy, move and remove commands

17) copy command

```
$cp source destination
```

Copy files and directories. If the source is a file, and the destination (file) name does not exist, then source is copied with new name i.e. with the name provided as the destination.

```
$ cp usrlisting listing_copy.txt
$ ls -l
total 12
drwxr-xr-x 2 raghu raghu 4096 2012-07-06 14:09 example
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file1
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file2
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file3
-rw-r--r-- 1 raghu raghu 491 2012-07-06 16:02 listing_copy.txt
-rw-r--r-- 1 raghu raghu 491 2012-07-06 14:23 usrlisting
```

If the destination is a directory, then the file is copied with its original name in that directory.

```
$ cp listing_copy.txt example/  
$ ls -l example/  
total 4  
-rw-r--r-- 1 raghu raghu 491 2012-07-06 16:07 listing_copy.txt
```

Multiple files can also be copied, but in that case, the last argument will be expected to be a directory where all the files are to be copied. And the rest of the arguments will be treated as file names.

```
$ cp file1 file2 example/  
$ ls -l example/  
total 4  
-rw-r--r-- 1 raghu raghu 0 2012-07-06 16:10 file1  
-rw-r--r-- 1 raghu raghu 0 2012-07-06 16:10 file2  
-rw-r--r-- 1 raghu raghu 491 2012-07-06 16:07 listing_copy.txt
```

If a directory is to be copied, then it must be copied recursively with the files contained in it. To copy a directory recursively, use `-r` option with `'cp'` command:

```
$ cp -r example /tmp/expertslogin/  
$ ls -l /tmp/expertslogin/  
total 4  
drwxr-xr-x 2 raghu raghu 4096 2012-07-06 16:12 example
```

18) move command

```
$ mv source destination
```

Move files or directories. The `'mv'` command works like `'cp'` command, except that the original file is removed. But, the `mv` command can be used to rename the files (or directories).

```
$ mv listing_copy.txt usrcopy  
$ ls -l  
total 12  
drwxr-xr-x 2 raghu raghu 4096 2012-07-06 16:10 example  
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file1  
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file2  
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file3  
-rw-r--r-- 1 raghu raghu 491 2012-07-06 16:02 usrcopy  
-rw-r--r-- 1 raghu raghu 491 2012-07-06 14:23 usrlisting
```

Here, `'listing_copy.txt'` is moved with the name `'usrcopy'` in the same directory (or you can say that it has been renamed).

19) To remove or Delete

```
$ rmdir
```

`'rmdir'` command removes any empty directories, but cannot delete a directory if a file is present in it. To use `'rmdir'` command, you must first remove all the files present in the directory you wish to remove (and possibly directories if any).

To remove files and directories

```
$ rm files|directories
```

A directory must be removed recursively with `-r` option.

```
$ rm file2  
$ rm -r example/  
$ ls -l  
total 8  
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file1  
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file3
```

```
-rw-r--r-- 1 raghu raghu 491 2012-07-06 16:02 usrcopy
-rw-r--r-- 1 raghu raghu 491 2012-07-06 14:23 usrlisting
```

Here, the file named 'file2' is removed first, and then the directory 'example' is removed recursively. This can be seen in the output of 'ls -l' command where these two are no longer present.

V- Other file commands

20) file command

The file command determines the file type of a given file. For example:

```
$ file /etc/passwd
```

```
/etc/passwd: ASCII text
```

You can provide one or more than one file as an argument to the file command.

```
$ file td.c td.out ARP.java Screenshot.png StringTokenizer.class
idl.rar List.pdf
td.c: ASCII C program text, with CRLF line terminators
td.out: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically
linked (uses shared libs), for GNU/Linux 2.6.15, not stripped
ARP.java: ASCII Java program text, with CRLF line terminators
Screenshot.png: PNG image data, 1366 x 768, 8-bit/color RGB, non-interlaced
StringTokenizer.class: compiled Java class data, version 50.0 (Java 1.6)
idl.rar: RAR archive data, vld, os: Win32
List.pdf: PDF document, version 1.4
```

21) stat command

To check the status of a file. This provides more detailed information about a file than 'ls -l' output.

```
$ stat usrcopy
```

```
File: `usrcopy'
Size: 491 Blocks: 8 IO Block: 4096 regular file
Device: 808h/2056d Inode: 149452 Links: 1
Access: (0644/-rw-r--r--) Uid: ( 1000/ raghu) Gid: ( 1000/ raghu)
Access: 2012-07-06 16:07:06.413522009 +0530
Modify: 2012-07-06 16:02:30.204152386 +0530
Change: 2012-07-06 16:17:18.992559654 +0530
```

22) cat command

The 'cat' command is actually a concatenator but can be used to view the contents of a file.

```
$ cat /etc/passwd
```

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
```

23) paggers

The cat command lists file as a whole. But if the file is big enough to fit into one screen, then we will be able to see only the last page of the file. The commands 'less' and 'more' display files one page at a time.

So they are also called pagers. You can navigate through a file using arrow keys. To quit from a pager, hit 'q'.

24) head command

Displays the first few lines of a file. By default, the 'head' command displays the first 10 lines of a file. But with -n option, the number of lines to be viewed can be specified.

```
$ head /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
```

25) tail command

Similar to 'head'; the 'tail' command shows the last 10 lines by default, and -n option is available as well.

```
$ tail -n 4 /etc/passwd
raghu:x:1000:1000:Raghu Sharma,,,:/home/raghu:/bin/bash
sshd:x:113:65534:./var/run/ssh:/usr/sbin/nologin
dictd:x:114:123:Dictd Server,,,:/var/lib/dictd:/bin/false
mysql:x:115:124:MySQL Server,,,:/nonexistent:/bin/false
```

26) wc command

Word count

This command counts lines, words and letters of the input given to it.

```
$ wc /etc/passwd
```

```
35 57 1698 /etc/passwd
```

The /etc/passwd file has 35 lines, 57 words, and 1698 letters present in it.

27) grep command

The 'grep' command searches for a pattern in a file (or standard input). It supports regular expressions. It returns a line if it matches the pattern in that line. So, if we wish to find the lines containing the word 'nologin', we use 'grep' as follows:

```
$ grep nologin /etc/passwd
```

```
sshd:x:113:65534:./var/run/ssh:/usr/sbin/nologin
```

28) ln command

The ln command is used in linux to create links. Links are a kind of shortcuts to other files. The general form of command is:

```
$ ln TARGET LINK_NAME
```

There are two types of links, soft links and hard links. By default, hard links are created. If you want to create soft link, use -s option. In this example, both types of links are created for the file usrlisting.

```
$ ln usrlisting hard_link
$ ln -s usrlisting soft_link
$ ls -l
total 12
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file1
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file3
-rw-r--r-- 2 raghu raghu 491 2012-07-06 14:23 hard_link
lrwxrwxrwx 1 raghu raghu 10 2012-07-09 14:00 soft_link -> usrlisting
-rw-r--r-- 1 raghu raghu 491 2012-07-06 16:02 usrcopy
-rw-r--r-- 2 raghu raghu 491 2012-07-06 14:23 usrlisting
```

VI- Text Editors

29) Pico & Nano

'Pico' is a text editor in Linux. 'Nano' editor is inspired from 'pico'. They work almost the same. If the argument given as filename exists, then that file will be opened for editing in pico/nano. Otherwise, a new file with that name will be created. Let's create a new file named hello.txt:

```
$ pico hello.txt
GNU nano 2.2.6 File: hello.txt Modified

This file is edited with pico editor.

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
Having made all the changes to the file, press 'ctrl+o' to write the changes to the file and 'ctrl+x' to exit from the editor. There are a lot of functions available with this editor. The help menu can be accessed with 'ctrl+g' keystrokes.
```

30) VI editor

The VI stands for Visual editor; another text editor in Linux. This is a standard editor in many Linux/Unix environments. This is the default editor that comes with many Linux distributions. It might be possible that it is the only text editor available with your distro.

You can open a file with vi for editing using the following:

```
$ vi hello.txt
```

The vi editor has 3 modes in which it performs its functions. The default is COMMAND mode, in which tasks like copy, paste, undo etc can be performed. You can change a mode from command mode only (and come back to it). The second mode is the INSERT mode, in which whatever key you type is treated as a character and will be loaded into the file buffer. To enter this mode, press 'i' when in command mode.

The final mode is EX mode or last line mode. The changes made in the buffer can be saved or discarded in this mode.

```
Hello world.
This file is edited using vi editor.
~
~
~

~
~
"hello.txt" 2 lines, 50 characters
```

VII- Useful commands

31) alias command

The 'alias' is another name for a command. If no argument is given, it shows current aliases. Aliases can be used for short names of commands. For example, you might use the clear command frequently. You can create an alias for it:

```
$ alias c="clear"
```

Next time you enter 'c' on command line, your screen will get clear. Current aliases can be checked with 'alias' command:

```
$ alias
alias alert='notify-send --urgency=low -i "${[ $? = 0 ] && echo terminal ||
echo error}" "${history|tail -n1|sed -e '\''s/^\s*[0-
9]\+\s*//;s/[\;|\&|]\s*alert$//'\''}"'
alias c='clear'
alias egrep='egrep --color=auto'
alias fgrep='fgrep --color=auto'
alias grep='grep --color=auto'
alias l='ls -CF'
alias la='ls -A'
alias ll='ls -alF'
alias ls='ls --color=auto'
```

32) w command

w command is used to check which users are logged in to the system, and what command they are executing at that particular time:

```
$ w
10:06:56 up 57 min, 3 users, load average: 0.04, 0.06, 0.09
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
root tty1 10:06 28.00s 1.02s 0.67s pager -s
raghu tty7 :0 09:19 57:33 1:22 0.20s gnome-session --session=classic-gnome
raghu pts/0 :0.0 09:34 0.00s 0.78s 0.00s w
```

It also shows the uptime, number of users logged in and load average of the system (in the first line of output above).

33) last command

Displays information about the users who logged in and out of the system. The output of the last command can be very large, so the following output has been filtered (through head) to display the top 10 lines only:

```
$ last | head
root tty1 Mon Jul 9 10:06 still logged in
root tty1 Mon Jul 9 10:06 - 10:06 (00:00)
raghu pts/1 :0.0 Mon Jul 9 10:05 - 10:06 (00:00)
raghu pts/0 :0.0 Mon Jul 9 09:34 still logged in
raghu tty7 :0 Mon Jul 9 09:19 still logged in
reboot system boot 2.6.38-13-generi Mon Jul 9 09:09 - 10:12 (01:02)
raghu tty7 :0 Sun Jul 8 23:36 - 00:30 (00:54)
reboot system boot 2.6.38-13-generi Sun Jul 8 23:36 - 00:30 (00:54)
raghu tty7 :0 Sun Jul 8 21:07 - down (01:06)
reboot system boot 2.6.38-13-generi Sun Jul 8 21:07 - 22:14 (01:07)
```

A similar command is 'lastb' that shows the last unsuccessful login attempts. But this command must be run as root otherwise you would get an error saying permission denied.

```
$ lastb
raghu tty2 Mon Jul 9 10:16 - 10:16 (00:00)
UNKNOWN tty2 Mon Jul 9 10:15 - 10:15 (00:00)
ubuntu tty8 :1 Mon Jul 2 10:23 - 10:23 (00:00)

btmptmp begins Mon Jul 2 10:23:54 2012
```

34) du command

The du command determines disk usage of a file. If the argument given to it is a directory, then it will list disk usage of all the files and directories recursively under that directory:

```
$ du /etc/passwd
4 /etc/passwd
$ du hello/
52 hello/HelloApp
4 hello/orb.db/logs
20 hello/orb.db
108 hello/
```

35) df command

The df reports file system usage. For example:

```
$ df
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/sda7 10079084 7372872 2194212 78% /
none 1522384 768 1521616 1% /dev
none 1529012 252 1528760 1% /dev/shm
none 1529012 108 1528904 1% /var/run
none 1529012 4 1529008 1% /var/lock
/dev/sda8 5039616 3758824 1024792 79% /home
/dev/sda2 209715196 196519248 13195948 94% /media/Data
```

36) fdisk command

The fdisk is a tool for getting partition information, and for adding and removing partitions. The fdisk tool requires super user privileges. To list all the partitions of all the hard drives available:

```
$ fdisk -l

Disk /dev/sda: 320.1 GB, 320072933376 bytes
255 heads, 63 sectors/track, 38913 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x396f396f
```

```
Device Boot Start End Blocks Id System
/dev/sda1 1 2611 20971520 7 HPFS/NTFS
/dev/sda2 2611 28720 209715200 7 HPFS/NTFS
/dev/sda3 * 28720 38914 81882113 5 Extended
/dev/sda5 28720 33942 41943040 7 HPFS/NTFS
/dev/sda6 33942 34464 4194304 7 HPFS/NTFS
/dev/sda7 34464 35739 10240000 83 Linux
/dev/sda8 35739 36376 5120000 83 Linux
/dev/sda9 36376 36886 4096000 82 Linux swap / Solaris
/dev/sda10 36887 38276 11164672 83 Linux
/dev/sda11 38277 38914 5117952 83 Linux
```

The fdisk is an interactive tool to edit the partition table. It takes a device (hard disk) as an argument, whose partition table needs to be edited.

```
$ fdisk /dev/sda

WARNING: DOS-compatible mode is deprecated. It's strongly recommended to
switch off the mode (command 'c') and change display units to
sectors (command 'u').
```

```
Command (m for help): m
Command action
a toggle a bootable flag
b edit bsd disklabel
c toggle the dos compatibility flag
d delete a partition
l list known partition types
m print this menu
n add a new partition
o create a new empty DOS partition table
p print the partition table
q quit without saving changes
s create a new empty Sun disklabel
t change a partition's system id
u change display/entry units
v verify the partition table
w write table to disk and exit
x extra functionality (experts only)
```

Pressing 'm' at the fdisk prompt prints out the help shown above that lists all the commands available for fdisk. A new partition can be created with 'n' and an existing partition can be deleted with the 'd'

command. When you are done editing the partitions, press 'w' to write the changes to the disk, and finally, hit 'q' to quit from fdisk (q does not save changes).

37) netstat command

The 'netstat' is a command used to check the network statistics of the system. It will list the current network connections, routing table information, interface statistics, masquerade connections and a lot more information.

```
$ netstat | head
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address Foreign Address State
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags Type State I-Node Path
unix 13 [ ] DGRAM 8498 /dev/log
unix 2 [ ] DGRAM 6824 @/org/kernel/udev/udev
unix 3 [ ] STREAM CONNECTED 56738 /var/run/dbus/system_bus_socket
unix 3 [ ] STREAM CONNECTED 56113
unix 3 [ ] STREAM CONNECTED 29138
unix 3 [ ] STREAM CONNECTED 29137
```

38) history command

[History command](#) shows the commands you have entered on your terminal so far.

39) passwd command

To change your password with [passwd command](#).

40) shutdown -h now

Finally, you can [shutdown](#) your system using this command.