If you are on a PC and want to use this list of SSH commands you will need a terminal emulator such as PuTTy to connect to your server.

What Is PuTTy?

Simply put:

PuTTy is an open source SSH client used to connect to a remote server. It's basically a terminal emulator for Windows based operating systems.

A terminal emulator? What is that?
To understand this you must first understand how Linux servers work.

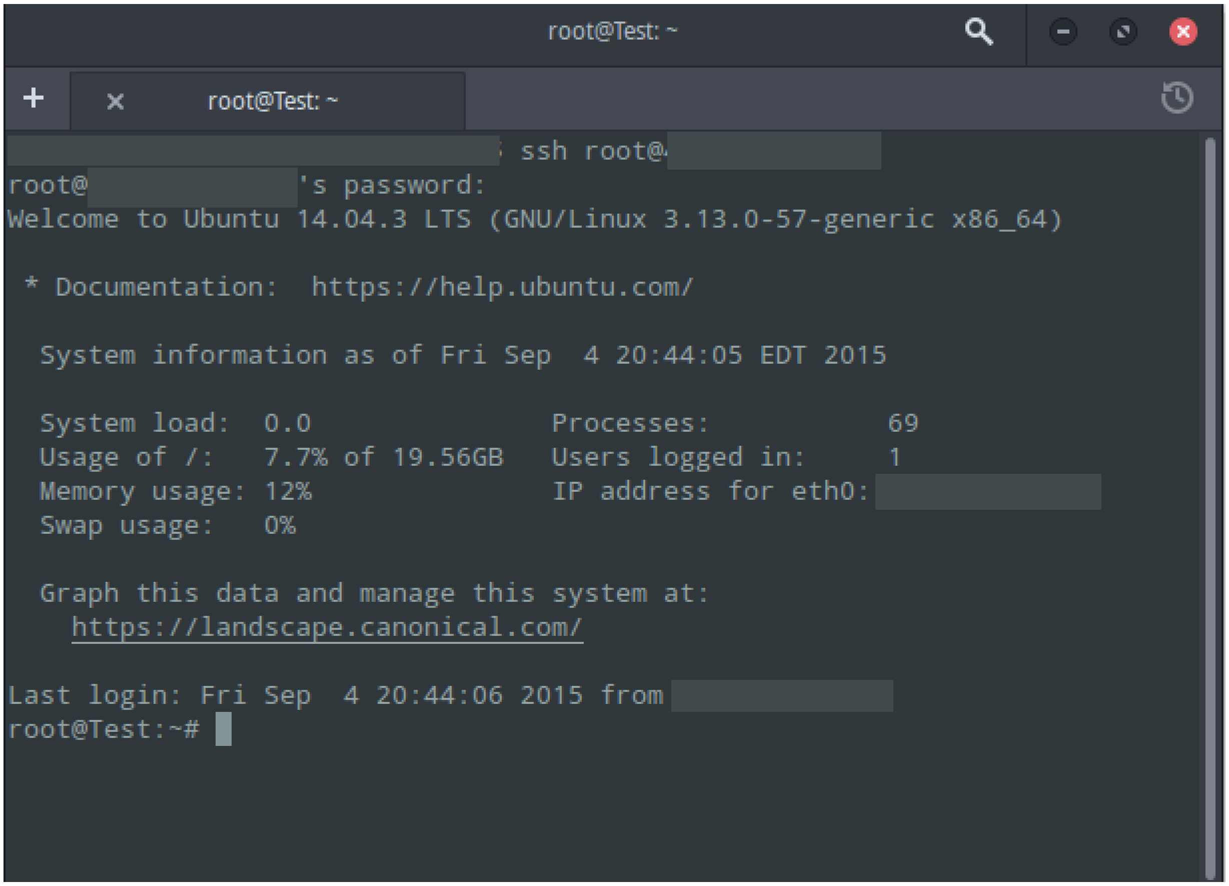
Say you have a server set up.

Now:

To connect to your server from your PC you can use PuTTy and type simple SSH commands to perform different basic actions such as creating folders, copying them and so on.

Linux already has a terminal.

So if you are running Linux as your desktop OS you don't need a ssh client because you can use the already build in terminal.



```
[redacted] ssh root@[redacted]
root@[redacted]'s password:
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-57-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

  System information as of Fri Sep  4 20:44:05 EDT 2015

  System load:  0.0               Processes:           69
  Usage of /:   7.7% of 19.56GB   Users logged in:     1
  Memory usage: 12%               IP address for eth0: [redacted]
  Swap usage:   0%

  Graph this data and manage this system at:
    https://landscape.canonical.com/

Last login: Fri Sep  4 20:44:06 2015 from [redacted]
root@Test:~#
```

Connecting to server using SSH from a Linux terminal

But if you are on a Windows machine, that' s where PuTTy comes in handy. It enables you to connect to your Linux server via SSH.

It sounds quite complicated but in reality it really isn't.

How To Use PuTTy To Login To Your Server

As I previously mentioned PuTTy is used to connect to your Linux server from a Windows based machine.

What you will need:

- A Linux based server
- PuTTy installed on your computer

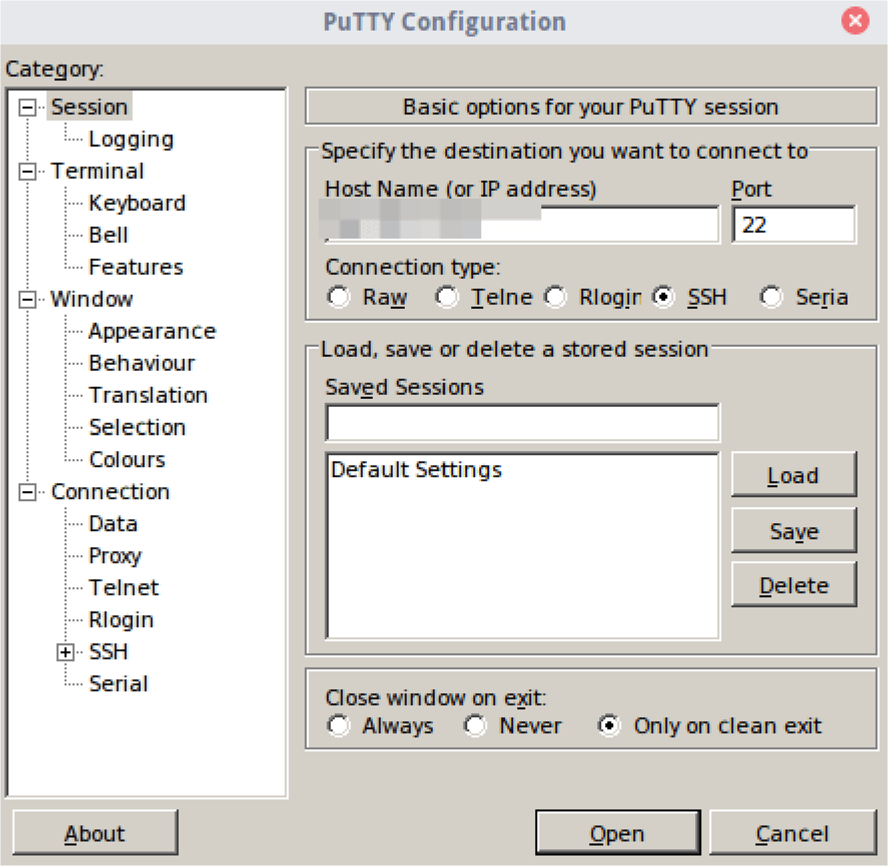
- The server’s IP address
- Username
- Password

Usually all the information you need is emailed to you by your cloud hosting provider automatically when you create a new server.

Once you have all the necessary information you can proceed connecting to your VPS using PuTTy.
Here’s how:

Download PuTTy and save it on your desktop. You don’t have to install it just click on it and it will run out of the box.

When you first open the program you will see something like this:


PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address) [redacted] Port 22

Connection type:
○ Raw ○ Telne ○ Rlogir ● SSH ○ Seria

Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit:
○ Always ○ Never ● Only on clean exit

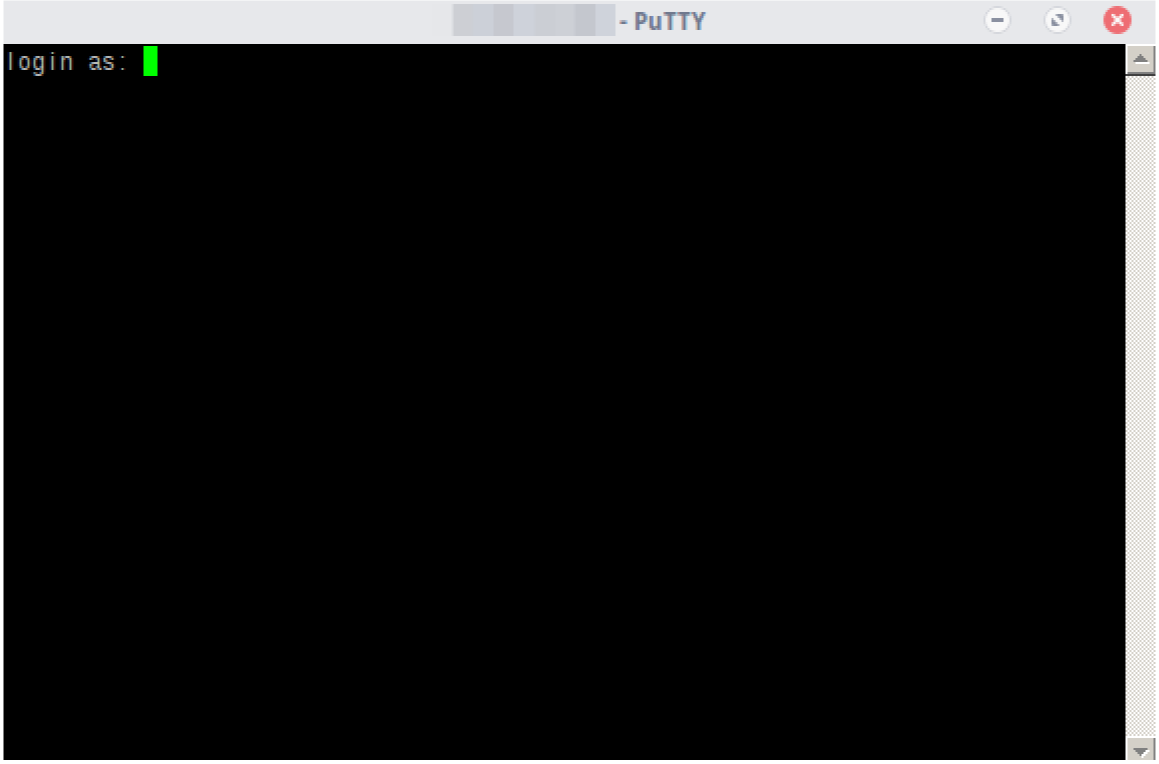
Out of all those options we are interested in well…just one and that's the Host Name field(the IP address).

That's where you will need to input your own server's IP address in order to connect to it.

Just enter the IP address making sure the connection type **SSH** is checked and port is 22.
Hit open.

You will be presented with a black screen like this:

I remember first when I saw this I freaked out but instinctively I knew I had to login first.

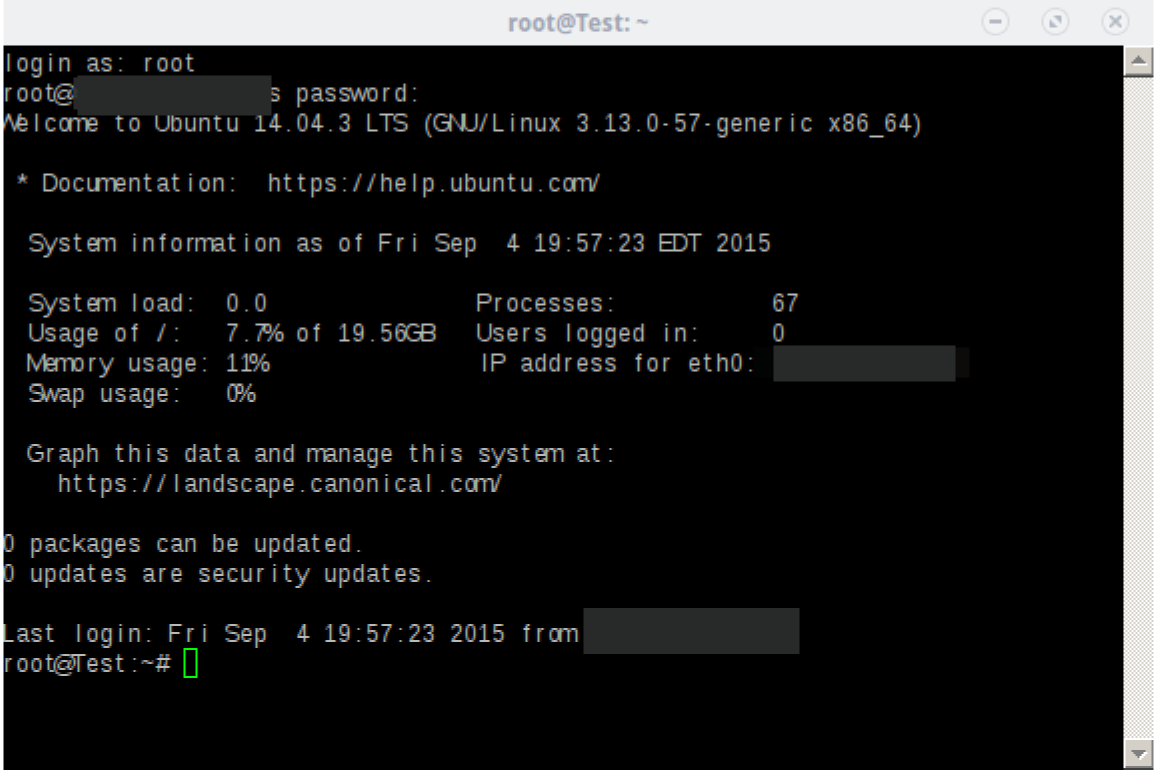


```
login as: 
```

Just enter your username and then your password.

To copy paste in PuTTy just right click and it will automatically paste.

Note: *Ubuntu does NOT show passwords not even asterisks so don't worry when you are typing your passwords and nothing shows on the screen.*
And tadaaa!



```
login as: root
root@[redacted]s password:
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-57-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

  System information as of Fri Sep  4 19:57:23 EDT 2015

  System load:  0.0               Processes:           67
  Usage of /:   7.7% of 19.56GB   Users logged in:     0
  Memory usage: 11%               IP address for eth0: [redacted]
  Swap usage:   0%

  Graph this data and manage this system at:
    https://landscape.canonical.com/

0 packages can be updated.
0 updates are security updates.

Last login: Fri Sep  4 19:57:23 2015 from [redacted]
root@Test:~#
```

You've successfully logged on to your server.

35 Useful SSH Commands

Now that you know how to use a SSH client we can take a look at a few useful ssh command lines and what they do.

I've compiled this list of SSH commands for anyone who struggles to managed their Linux servers.

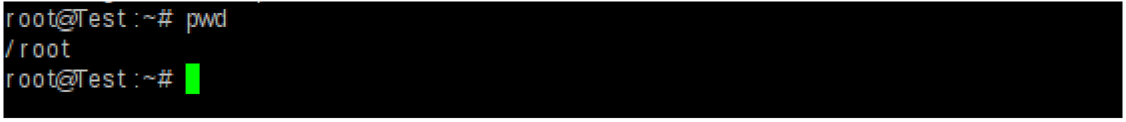
Basic Navigation

1.) How to find out where you are

pwd

Type **pwd** to see where on the server you are.

For example if I type **pwd** in PuTTy it returns **/root**



```
root@Test:~# pwd
/root
root@Test:~# 
```

2.) Navigate to...

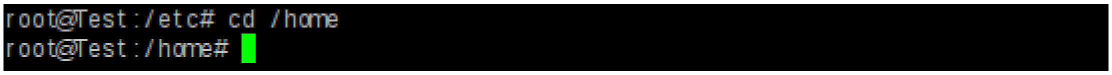
cd

The **cd** command (also known change directory) is used to navigate into a specified directory on your server.

Such as:

cd /home (moves you into the home folder)

Example:

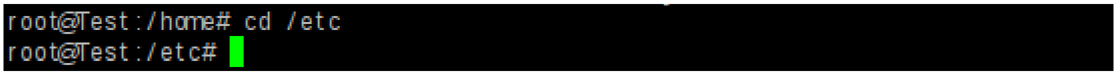


```
root@Test:/etc# cd /home
root@Test:/home# 
```

or

Say I want to navigate to the /etc directory. All I have to do is type cd /etc.

Example:

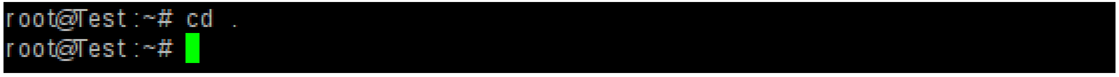


```
root@Test:/home# cd /etc
root@Test:/etc# 
```

3.) The same directory

cd .

The **cd .** command does basically nothing. Using this command you will remain in the same directory you were.



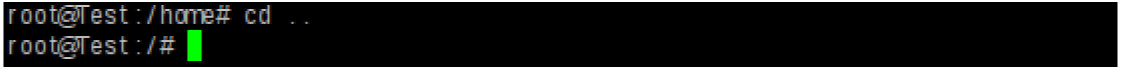
```
root@Test:~# cd .
root@Test:~#
```

4.) Move me up one directory

cd ..

Remember the previous command? Well I included it for a reason!

Add another dot(stop) to **cd .** and you will end up with **cd ..** which will move you up one directory.

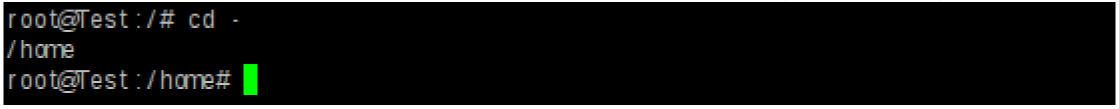


```
root@Test:/home# cd ..
root@Test:/#
```

5.) Go to the previous directory

cd –

The **cd –** command is used when you want to go to the previous directory.



```
root@Test:/# cd -
/home
root@Test:/home#
```

6.) Go to Home folder

cd ~

The **cd ~** commands is used when you want to go to the home directory on your server.

7.) Go to root

cd /

The **cd /** command is used when you want to jump to root.

List Directories and Contents

8.) List files

ls

usage: **ls [option] [file]**

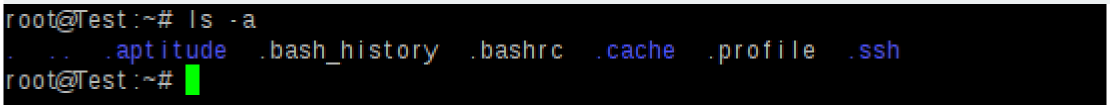
Using the ls command in a folder will display all it’s content.

Such as: ls /home.

Will return all content of the folder /home.

9.) Show me all files in a directory

ls -a



```
root@Test:~# ls -a
.  ..  .aptitude  .bash_history  .bashrc  .cache  .profile  .ssh
root@Test:~#
```

10.) Show contents with file size

ls -h

If you wish to see a directory’s contents with file sizes just type **ls -h**

11.) How to list sub-directories recursively

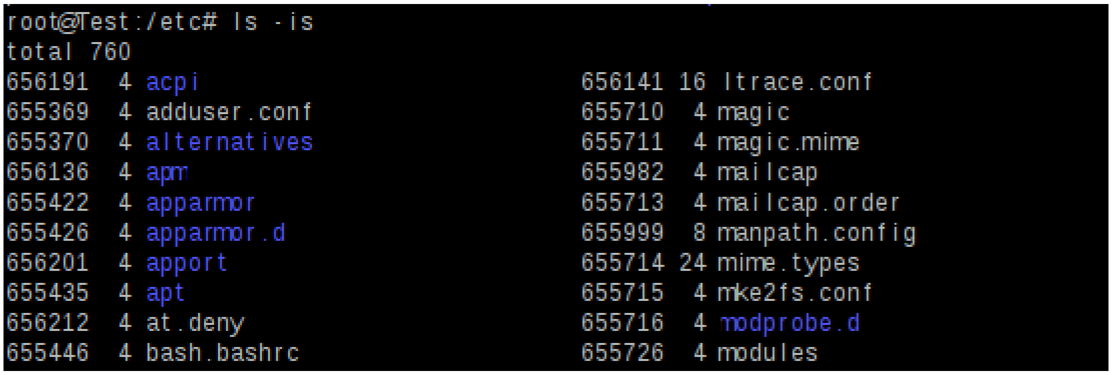
ls -r

The **ls -r** command is used to list sub-directories recursively.

12.) How to list files by file size

ls -is

Use this ssh command to list files by size.



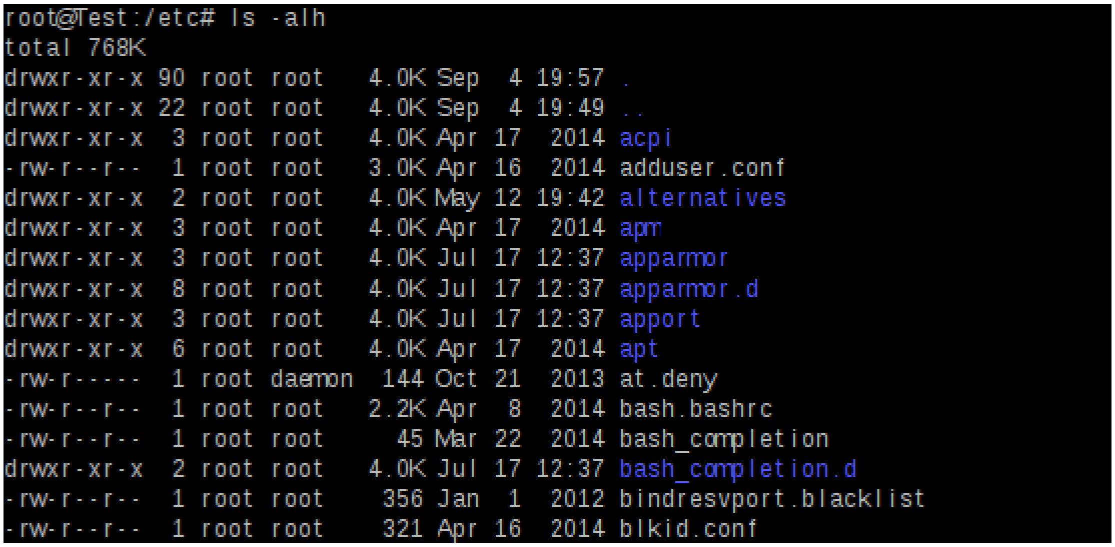
```
root@Test:/etc# ls -is
total 760
656191  4 acpi                          656141 16 ltrace.conf
655369  4 adduser.conf                  655710  4 magic
655370  4 alternatives                  655711  4 magic.mime
656136  4 apm                           655982  4 mailcap
655422  4 apparmor                      655713  4 mailcap.order
655426  4 apparmor.d                    655999  8 manpath.config
656201  4 apport                        655714 24 mime.types
655435  4 apt                           655715  4 mke2fs.conf
656212  4 at.deny                       655716  4 modprobe.d
655446  4 bash.bashrc                   655726  4 modules
```

13.) List all folders in directory with details

ls -alh

This one is very useful when you want to see more details about certain files.

For example you may want to know file permissions of a directory.



```
root@Test:/etc# ls -alh
total 768K
drwxr-xr-x 90 root root    4.0K Sep  4 19:57 .
drwxr-xr-x 22 root root    4.0K Sep  4 19:49 ..
drwxr-xr-x  3 root root    4.0K Apr 17  2014 acpi
-rw-r--r--  1 root root    3.0K Apr 16  2014 adduser.conf
drwxr-xr-x  2 root root    4.0K May 12 19:42 alternatives
drwxr-xr-x  3 root root    4.0K Apr 17  2014 apm
drwxr-xr-x  3 root root    4.0K Jul 17 12:37 apparmor
drwxr-xr-x  8 root root    4.0K Jul 17 12:37 apparmor.d
drwxr-xr-x  3 root root    4.0K Jul 17 12:37 apport
drwxr-xr-x  6 root root    4.0K Apr 17  2014 apt
-rw-r-----  1 root daemon  144 Oct 21  2013 at.deny
-rw-r--r--  1 root root    2.2K Apr  8  2014 bash.bashrc
-rw-r--r--  1 root root      45 Mar 22  2014 bash_completion
drwxr-xr-x  2 root root    4.0K Jul 17 12:37 bash_completion.d
-rw-r--r--  1 root root     356 Jan  1  2012 bindresvport.blacklist
-rw-r--r--  1 root root     321 Apr 16  2014 blkid.conf
```

Copying files

14.) Copying a file

cp

To copy a file just use the cp ssh command.

Example:

cp filename.php /home/filename.php

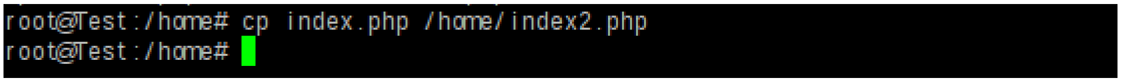
15.) Copy a folder with all files

cp -r

This one is used to copy the entire folder with all it’s contents.

16.) Copy and rename

cp filename.php /home/filename2.php



```
root@Test:/home# cp index.php /home/index2.php
root@Test:/home#
```

Moving files

17.) Moving a file

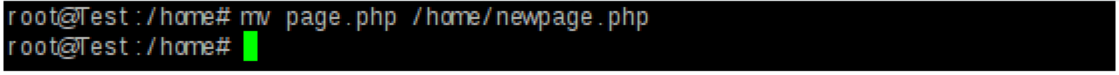
mv

Example:

mv page.php /home/page.php

18.) Move and rename

mv page.php /home/newpage.php



```
root@Test:/home# mv page.php /home/newpage.php
root@Test:/home#
```

19.) Move file up one directory

mv filename ..

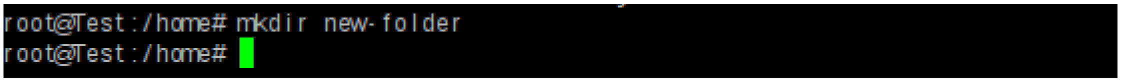
example: mv index.html/ ..

Creating files/folders

20.) Create a folder

mkdir

example: mkdir new-folder



```
root@Test:/home# mkdir new-folder
root@Test:/home# 
```

21.) Create a file

touch

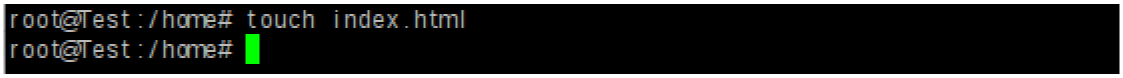
Use the touch command to create different files and file extensions (you can later edit them)

Example:

touch index.php

touch index.html

touch robots.txt



```
root@Test:/home# touch index.html
root@Test:/home# 
```

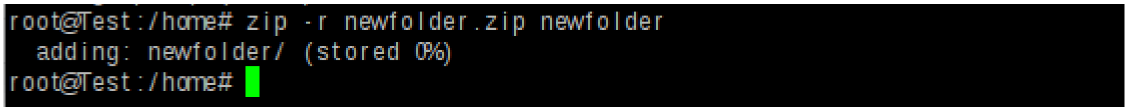
Compressing/Decompressing files

22.) Compressing folders

zip -r foldername.zip foldername

Example:

zip -r newfolder.zip newfolder



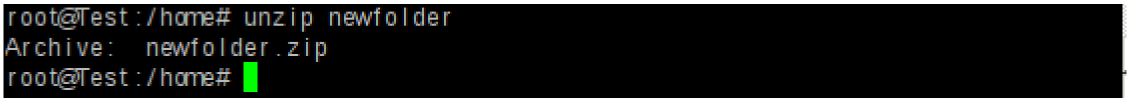
```
root@Test:/home# zip -r newfolder.zip newfolder
  adding: newfolder/ (stored 0%)
root@Test:/home# 
```

23.) Decompressing folders

unzip

Example:

unzip newfolder.zip



```
root@Test:/home# unzip newfolder
Archive:  newfolder.zip
root@Test:/home# 
```

24.) Compressing folders using tar -czvf

tar -czvf foldername.tar.gz foldername

Example:

tar -czvf wp-content.tar.gz wp-content

25.) Decompressing folders using tar -czvf

tar -xvf foldername.tar.gz

Example:

tar -xvf wp-content.tar.gz

Delete/Remove files

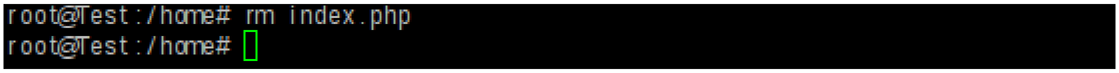
26.) Delete a file

rm

To delete a file on your server just use the **rm** command.

Example:

rm index.php



```
root@Test:/home# rm index.php
root@Test:/home# 
```

27.) Delete all files from a directory

```
rm *
```

This command is used to delete all contents from a directory.

Example:

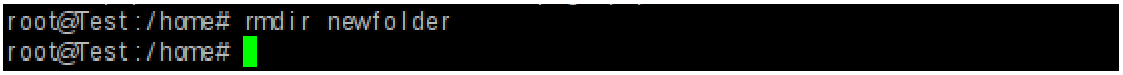
rm * foldername

28.) Delete a folder/directory

```
rmdir
```

Use this command to remove directories.

Here’s how: rmdir foldername



```
root@Test:/home# rmdir newfolder
root@Test:/home#
```

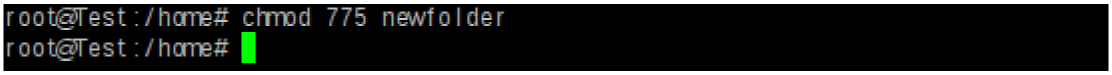
File permissions

29.) Change file permissions

```
chmod
```

Example:

chmod 775 newfolder



```
root@Test:/home# chmod 775 newfolder
root@Test:/home#
```

30.) Change permissions of folder and all files inside

```
chmod -r
```

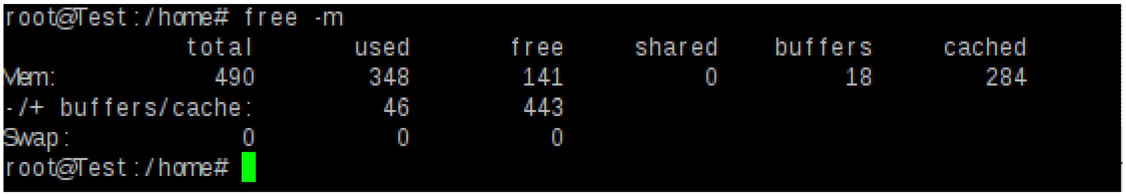
This command comes in handy when you want to change file permissions of an entire folder including it's contents.

Miscellaneous

31.) Memory usage

```
free -m
```

Great command to monitor memory usage!

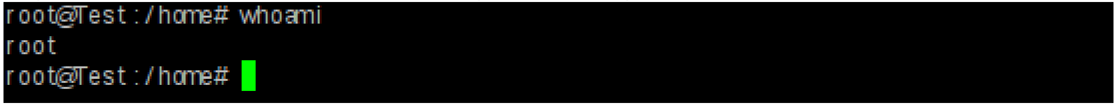


```
root@Test:/home# free -m
             total       used       free     shared    buffers     cached
Mem:           490        348        141          0         18        284
-/+ buffers/cache:         46        443
Swap:            0          0          0
root@Test:/home#
```

32.) What user am I?

```
whoami
```

If you don't know what user your are just use **whoami**

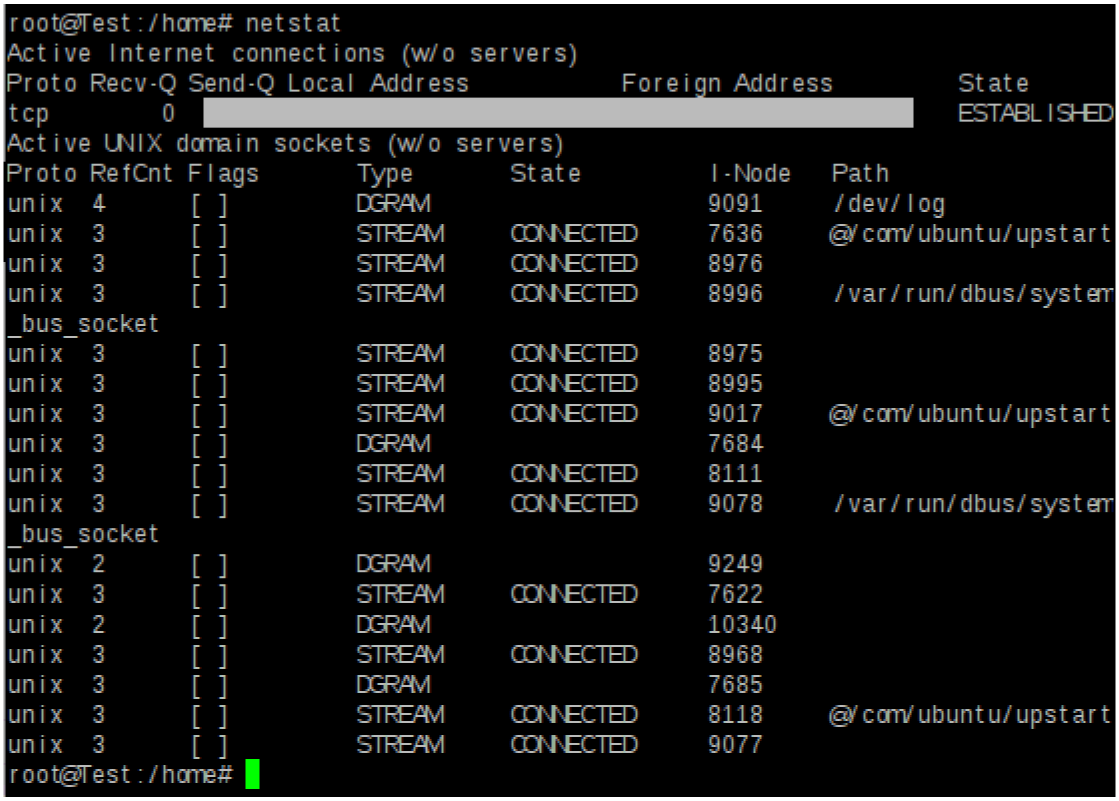


```
root@Test:/home# whoami
root
root@Test:/home#
```

33.) Show network connections

```
netstat
```

Displays current networks connections

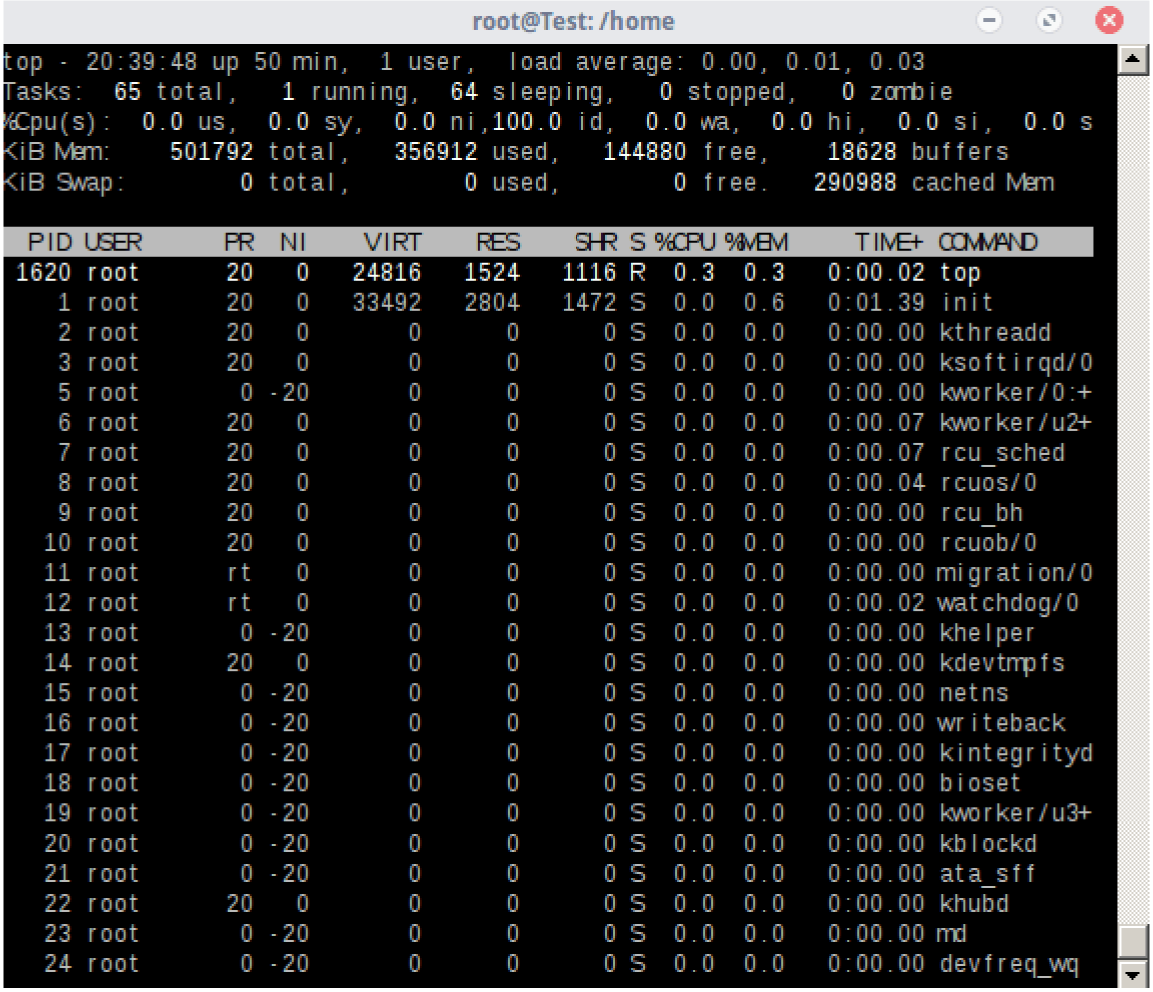


```
root@Test:/home# netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0  [redacted]                                            ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  4      [ ]         DGRAM                    9091     /dev/log
unix  3      [ ]         STREAM     CONNECTED     7636     @/com/ubuntu/upstart
unix  3      [ ]         STREAM     CONNECTED     8976
unix  3      [ ]         STREAM     CONNECTED     8996     /var/run/dbus/system
_bus_socket
unix  3      [ ]         STREAM     CONNECTED     8975
unix  3      [ ]         STREAM     CONNECTED     8995
unix  3      [ ]         STREAM     CONNECTED     9017     @/com/ubuntu/upstart
unix  3      [ ]         DGRAM                    7684
unix  3      [ ]         STREAM     CONNECTED     8111
unix  3      [ ]         STREAM     CONNECTED     9078     /var/run/dbus/system
_bus_socket
unix  2      [ ]         DGRAM                    9249
unix  3      [ ]         STREAM     CONNECTED     7622
unix  2      [ ]         DGRAM                    10340
unix  3      [ ]         STREAM     CONNECTED     8968
unix  3      [ ]         DGRAM                    7685
unix  3      [ ]         STREAM     CONNECTED     8118     @/com/ubuntu/upstart
unix  3      [ ]         STREAM     CONNECTED     9077
root@Test:/home#
```

34.) Monitor CPU, processes and memory

```
top
```

Type to to monitor CPU usage, processes and RAM.

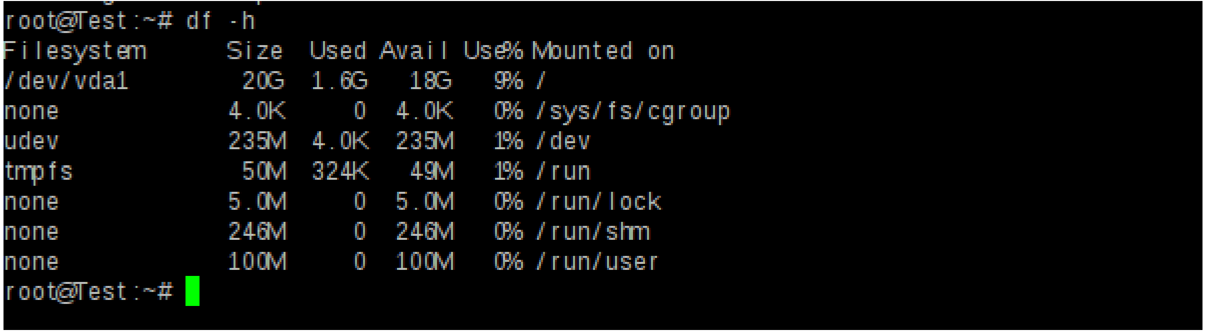


```
top - 20:39:48 up 50 min,  1 user,  load average: 0.00, 0.01, 0.03
Tasks:  65 total,   1 running,  64 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 s
KiB Mem:    501792 total,   356912 used,   144880 free,    18628 buffers
KiB Swap:        0 total,        0 used,        0 free.   290988 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
 1620 root      20   0   24816   1524   1116 R  0.3  0.3   0:00.02 top
    1 root      20   0   33492   2804   1472 S  0.0  0.6   0:01.39 init
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S  0.0  0.0   0:00.00 ksoftirqd/0
    5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:+
    6 root      20   0       0      0      0 S  0.0  0.0   0:00.07 kworker/u2+
    7 root      20   0       0      0      0 S  0.0  0.0   0:00.07 rcu_sched
    8 root      20   0       0      0      0 S  0.0  0.0   0:00.04 rcuos/0
    9 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcu_bh
   10 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcuob/0
   11 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 migration/0
   12 root      rt   0       0      0      0 S  0.0  0.0   0:00.02 watchdog/0
   13 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 khelper
   14 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kdevtmpfs
   15 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 netns
   16 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 writeback
   17 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kintegrityd
   18 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 bioset
   19 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/u3+
   20 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kblockd
   21 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 ata_sff
   22 root      20   0       0      0      0 S  0.0  0.0   0:00.00 khubd
   23 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 md
   24 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 devfreq_wq
```

35.) Display sever disk usage

```
df -h
```

If you want to see how much disk is used on your server just type **df -h**



```
root@Test:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/vda1        20G  1.6G   18G   9% /
none            4.0K     0  4.0K   0% /sys/fs/cgroup
udev            235M  4.0K  235M   1% /dev
tmpfs            50M  324K   49M   1% /run
none            5.0M     0  5.0M   0% /run/lock
none            246M     0  246M   0% /run/shm
none            100M     0  100M   0% /run/user
root@Test:~#
```

What about you?

****

20 TOP MOST USED & COMMON SSH COMMANDS

As promised, I will teach babies* to manage unmanaged server from scratch. It means I'll try to bring those babies from zero to hero being a Server Ninja. Hence, for my early posts I will firstly write all basic things for newbie server admins should know. In this page I list all necessary, most-used and common SSH commands useful to navigate through SSH client like Putty. I believe these commands are working on any Unix-based servers.

* *babies = newbies with no knowledge about Linux and server at all*

If you are really a newbie, then you should bookmark this page otherwise simply skip this post.

My next plan: I will post other articles of basic guides how to do things in your server via SSH. Browse it all here.

REQUIREMENTS TO SSH

- A working SSH client like Putty (Windows) or Terminal (Mac and Linux)
- A working Linux-based server with SSH enabled
- A cup of coffee if you wish
- A computer with stable Internet connection

SOME CONVENTIONS

1. Hit Enter or Return key on your keyboard after every each command / line
2. Each line is a single raw of command unless specified otherwise
3. Replace domain.com with your own domain name and TLD
4. Replace xxx.xxx.xxx.xxx with IP address of your own server
5. Replace example paths and file names according to your server information
6. If a path ends with slash "/" then its a directory

7. A directory: “/path/to/directory/” will always end with slash “/”
8. A file: “/path/to/file” won’t have slash “/” in the end
9. A file is not always having extension in the end of the file name

OK, so lets start with the most common commands to SSH you should know..

NAVIGATING

1. How to move into another directory

Use command below to change directory

```
cd [another directory]
```

example: move to directory “download”

```
cd download
```

2. How to go to home directory

```
cd ~
```

3. How go to the last directory you were in

```
cd -
```

4. How to go up a directory level

```
cd ..
```

5. How to show the full path of the current directory

Use this command to find out where are you currently in.

```
pwd
```

6. How to list files and/or directories in a directory

```
ls (just type ls and hit enter)
```

7. How to list all files and information

```
ls -al
```

8. How to list all files ending with certain extension

```
ls *.ext
```

example:

```
ls *.php
```

9. How to list all files and folders with detailed information including file size

```
ls -alh
```

10. How to quit and exit SSH client

```
exit
```

FILE MANAGEMENT

11. How to copy and rename file

Use this command to copy and rename a file

```
cp [filename] [new filename]
```

example: we’ll rename banner.jpg to banner728px.jpg

```
cp banner.jpg banner728px.jpg
```

example: we’ll cope banner.jpg to a folder called “ads”

```
cp banner.jpg ads/banner.jpg
```

example: copying and renaming at once

```
cp banner.jpg ads/banner728px.jpg
```

ps: original file will remain, it is just copied.

12. How to move and rename file

Use this command to move and rename file

```
mv [old filename] [new filename]
```

example: moving a file to another directory

```
mv banner.jpg ads/banner.jpg
```

example: moving a file to another directory and renaming it at once

```
mv banner.jpg ads/banner728px.jpg
```

ps: original file will be deleted as it was moved to another location

pps: you can also move a folder

example: moving “image” folder to “media” folder

```
mv image/ media
```

example: moving “image” folder to upper directory

```
mv image/ ..
```

13. How to delete / remove a file

```
rm [file name]
```

example:

```
rm banner.jpg
```

14. How to delete / remove all files in current directory

```
rm *
```

15. How to delete files with certain extension

```
rm *.extension
```

example: remove all files with .jpg extension

```
rm *.jpg
```

16. How to copy a folder with all files and folders in it

```
cp -r [directory] [new directory]
```

17. How to create new folder

```
mkdir [folder name]
```

example:

```
mkdir image
```

18. How to search for a file starting within current directory

```
find . -name [filename] -print
```

example: find a file called “banner.jpg” in current folder

```
find . -name banner.jpg -print
```

19. How to search for text within a file

```
grep [text] [filename]
```

example: find the word “sidebar” in file index.php

```
grep sidebar index.php
```

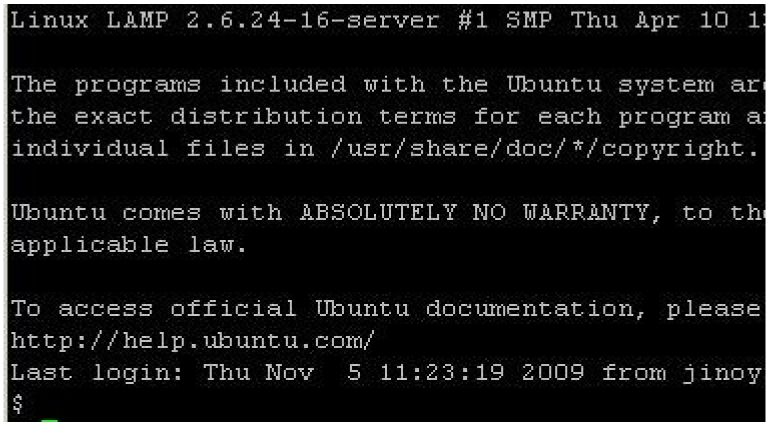
20. CHMOD – how to change file permission

```
chmod [permission type] [file/folder name]
```

example:

```
chmod 777 config.php
```

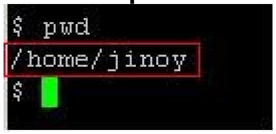
I assume you know how to login and logout from Linux system and one Linux system is available in your network. Ask your Linux administrator for your credentials like user name/password information. For connecting to Linux you need any Telnet/SSH client like PuTTY. If you don’t have one yet, download PuTTY from the following site.
http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html
OK, open your client, give host name/IP address and press Open button. First Linux asks for user name and then password. Give the information and you are now in command prompt.



```
Linux LAMP 2.6.24-16-server #1 SMP Thu Apr 10 1

The programs included with the Ubuntu system ar
the exact distribution terms for each program a
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to th
applicable law.

To access official Ubuntu documentation, please
http://help.ubuntu.com/
Last login: Thu Nov  5 11:23:19 2009 from jinoy
$
```

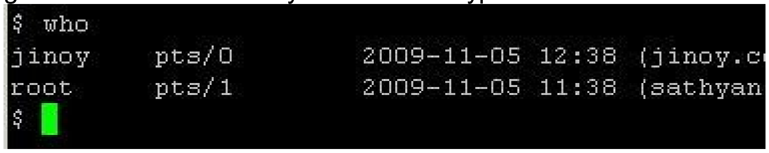
Now let us get into commands. Please note that everything in Linux is case-sensitive, so all commands should be in appropriate case.

- **pwd** – the command is called “print working directory” which is used to print the current directory. When you are logged in, you are taken into your home directory. When Linux administrator creates your user account, he specifies your home directory where you generally keep files, folders etc. Type the command **pwd** in command prompt and hit [Enter].



```
$ pwd
/home/jinoy
$
```

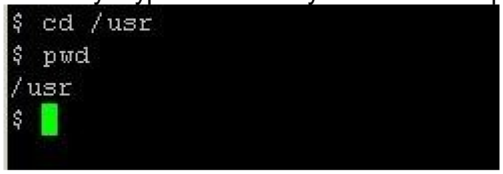
- **who** – this give you information regarding currently logged in users. Suppose you want to know how many users are connected to Linux system along with you. This command gives details like user id, logged in time, terminals to which users are connected etc. This has another variation **who am i** which gives information about your session. Type the command **who** in your command prompt and hit [Enter].



```
$ who
jinoy    pts/0        2009-11-05 12:38 (jinoy.c
root     pts/1        2009-11-05 11:38 (sathyan
$
```

First column is user id, second column is the terminal and third one is the logged in time. Now type **who am i**, hit [Enter] and verify the result.

- **cd** – short name of “change directory” which is used to switch to another directory from your current working directory. This command is used with the argument directory name to which we should change, **cd <directory name>**. If you type **cd** without any argument, you are changed to your home directory. Type **cd /usr** in your command prompt and hit enter. Then type **pwd** command and hit [Enter].



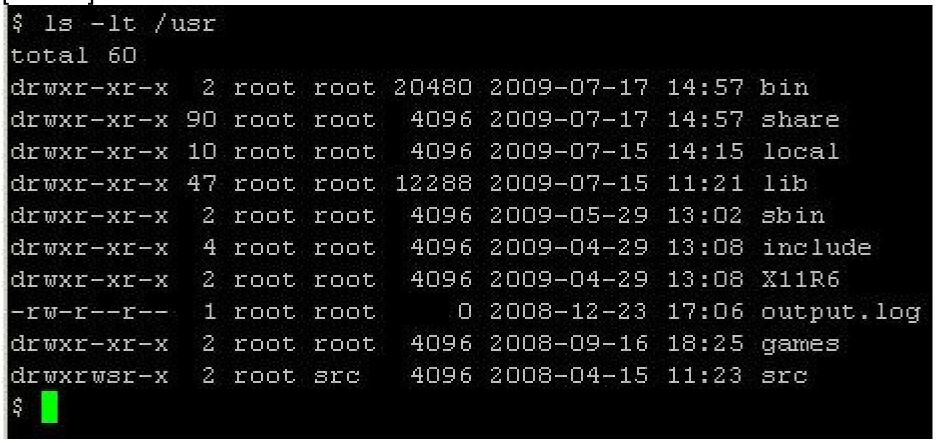
```
$ cd /usr
$ pwd
/usr
$
```

In the above diagram you could see that **/usr** is printed on executing **pwd** command. This means that we changed our working directory to **/usr** using **cd** command. Now just type **cd** and hit [Enter]. Validate the result using **pwd** command.

- **ls** – List command used to show files and directories. If you just type **ls**, it lists files and directories in the current working directory. Optionally you may give the directory name as argument to list content in that particular directory. You may add combination of additional arguments to get more features in the result set. Some arguments are given below.

- **-l** – this argument gives detailed listing of files and directories like owner of the file, size of the file, whether the file is a directory, last modified time etc. This is a common command everybody uses.
- **-R** – The argument -R is used for recursive purpose. This means it list all files in sub directories also. It searches all inner directories recursively and give results.
- **-a** – The argument -a lists all hidden files.
- **-t** – sort by modification time in descending order
- **directory name** – If you give directory name as argument, it lists content in that directory rather than listing current working directory.

Ok, now let us put everything into an example. Go to your command prompt, type **ls -lt /usr** and hit [Enter].



```
$ ls -lt /usr
total 60
drwxr-xr-x  2 root root 20480 2009-07-17 14:57 bin
drwxr-xr-x 90 root root  4096 2009-07-17 14:57 share
drwxr-xr-x 10 root root  4096 2009-07-15 14:15 local
drwxr-xr-x 47 root root 12288 2009-07-15 11:21 lib
drwxr-xr-x  2 root root  4096 2009-05-29 13:02 sbin
drwxr-xr-x  4 root root  4096 2009-04-29 13:08 include
drwxr-xr-x  2 root root  4096 2009-04-29 13:08 X11R6
-rw-r--r--  1 root root     0 2008-12-23 17:06 output.log
drwxr-xr-x  2 root root  4096 2008-09-16 18:25 games
drwxrwsr-x  2 root src   4096 2008-04-15 11:23 src
$
```

The above command gives detailed listing of files in the directory **/usr** in descending order of modified time. The first column is about file permissions. The first character shows whether it is a directory. For eg, in the above screenshot we could see that bin is a directory because the first character of file permission column is **d**. output.log is a file because the first character is **–**. Second column gives the number of links to the file. Third column shows the owner of the file and fourth column is about owner group. Fifth column shows file size in bytes and sixth column about last modified time. Final column shows the name of the file. Please note that in Linux everything is file. So directory is also a file in Linux system.

- **mkdir** – Used to create directories. You should give directory name as argument. Let us create a directory in your home directory. Go to your home directory using **cd** command. Type the following command and hit [Enter].

  mkdir testdir

  The above command creates testdir in your home directory. Verify the same using **ls** command.
- **touch** – This command is used to change the timestamp of the file. But one of the most import function of the command is to create an empty file if it do not exist. Go to your system directory, type the following command and hit [Enter].

  touch sample.txt

  It creates sample.txt file. Verify the same using **ls** command.
- **cat** – **cat** command is very useful for many purposes like creating files, adding and appending content to files etc. It also diplays the content of the file to standard output or another file. You may

concatenate multiple files and display the content. Let us try some examples. Type the following command and hit [Enter].

cat > sample.txt

It waits for the input which should be added to the file in the next line. Type the text "Hello how are you?" and hit [Enter]. Then press [Ctrl+d]. This add the text to sample.txt file. Now let us display the content of the file.

cat sample.txt

You get the message in screen. Ok, now rather than writing the content to standard output, let us write it into another file.

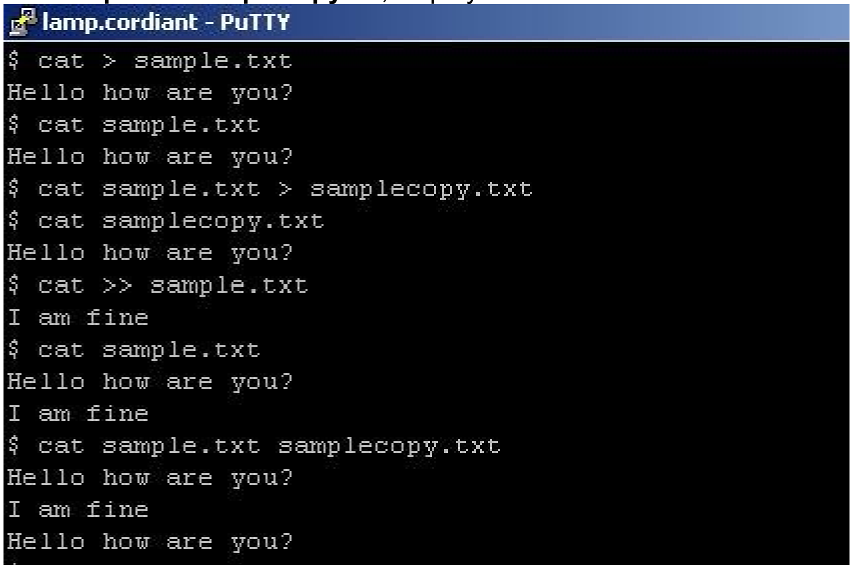
cat sample.txt > samplecopy.txt

The above command creates samplecopy.txt file if it do not exists already and add the content of the file sample.txt. Verify the result using **cat** command.

Let us append the content. Type the following command and hit [Enter].

cat >> sample.txt

In the next line just add the text "I am fine", hit [Enter] and press [Ctrl+d]. Verify the content of the file sample.txt using **cat** command. You may combine multiple files and show the output. For eg, **cat sample.txt samplecopy.txt**, displays the content from both files. You may send it to another file also.



```
$ cat > sample.txt
Hello how are you?
$ cat sample.txt
Hello how are you?
$ cat sample.txt > samplecopy.txt
$ cat samplecopy.txt
Hello how are you?
$ cat >> sample.txt
I am fine
$ cat sample.txt
Hello how are you?
I am fine
$ cat sample.txt samplecopy.txt
Hello how are you?
I am fine
Hello how are you?
```

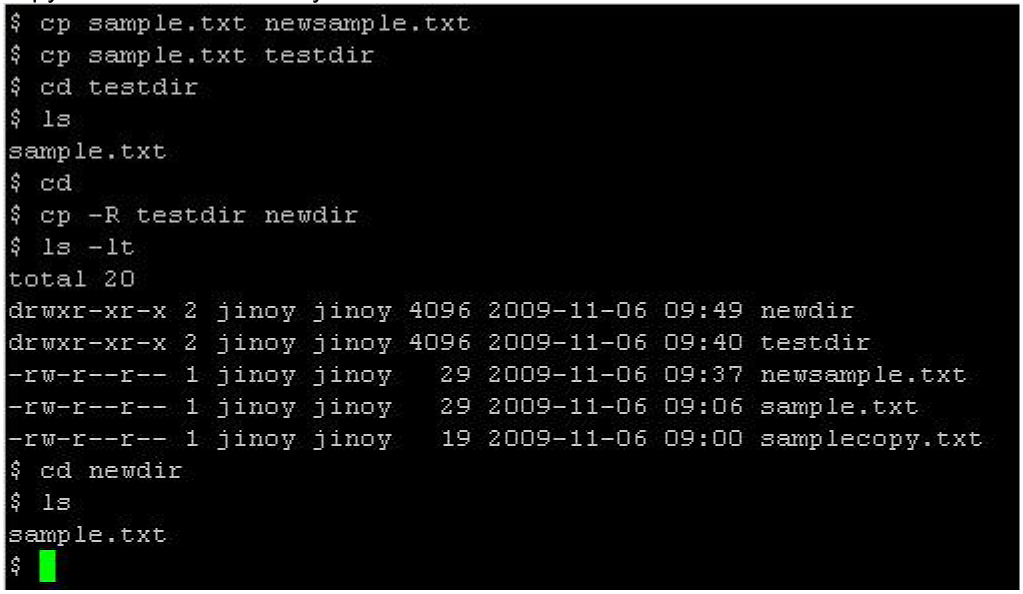
- **cp** – copy command is used to copy files and directories. You have to give source and destination of files and directories. For eg, when you copy file, you may give the destination as either file name or directory name to which the file should be copied. You may copy entire directory also to another location. Some sample examples are given below.

cp sample.txt newsample.txt, this copies the file sample.txt to newsample.txt. You have to give source file name as first argument followed by destination. Verify the result.

Now let us copy the file sample.txt to our testdir directory which we already created. Type the following command and hit [Enter].

cp sample.txt testdir, this copies the file to directory. Here we provide directory name as second argument. Verify the result using **cd** and **ls** commands. Now let us copy the directory testdir. Go to your home directory and type the following command. Hit [Enter].

cp -R testdir newdir, this command copies testdir and its content to another directory newdir. Verify the result. Check whether sample.txt file in testdir is also available in newdir directory. **-R** argument is used to copy directories recursively.



```
$ cp sample.txt newsample.txt
$ cp sample.txt testdir
$ cd testdir
$ ls
sample.txt
$ cd
$ cp -R testdir newdir
$ ls -lt
total 20
drwxr-xr-x 2 jinoy jinoy 4096 2009-11-06 09:49 newdir
drwxr-xr-x 2 jinoy jinoy 4096 2009-11-06 09:40 testdir
-rw-r--r-- 1 jinoy jinoy   29 2009-11-06 09:37 newsample.txt
-rw-r--r-- 1 jinoy jinoy   29 2009-11-06 09:06 sample.txt
-rw-r--r-- 1 jinoy jinoy   19 2009-11-06 09:00 samplecopy.txt
$ cd newdir
$ ls
sample.txt
$ █
```

- **mv** – move command is used to rename files and directories. It is also used to move files from source to destination. Let us try some examples. Go to your home directory and execute following commands.

mv sample.txt log.txt, this command rename sample.txt to log.txt. source file name is given as first argument.

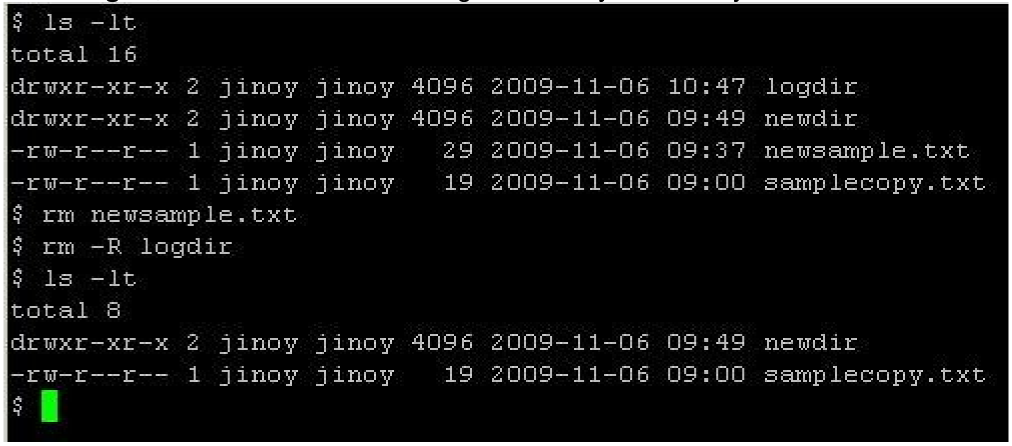
mv testdir logdir, this rename testdir directory to logdir directory. Please note that the content inside directory are not renamed.

mv log.txt logdir, this command moves log.txt file to logdir directory.

- **rm** – remove command is used to delete files and directories. Some examples are given below.

rm newsample.txt, deletes newsample.txt file. You can give multiple file names in single command delimited by space to delete multiple files in a stretch. For eg, **rm file1 file2** remove both files.

rm -R logdir, this command deletes logdir directory recursively.



```
$ ls -lt
total 16
drwxr-xr-x 2 jinoy jinoy 4096 2009-11-06 10:47 logdir
drwxr-xr-x 2 jinoy jinoy 4096 2009-11-06 09:49 newdir
-rw-r--r-- 1 jinoy jinoy   29 2009-11-06 09:37 newsample.txt
-rw-r--r-- 1 jinoy jinoy   19 2009-11-06 09:00 samplecopy.txt
$ rm newsample.txt
$ rm -R logdir
$ ls -lt
total 8
drwxr-xr-x 2 jinoy jinoy 4096 2009-11-06 09:49 newdir
-rw-r--r-- 1 jinoy jinoy   19 2009-11-06 09:00 samplecopy.txt
$ █
```

- **chmod** – this command is used to change file permissions. Every file or directory in Linux is associated with file permissions. There are three levels of permissions, read, edit and execute. You can't modify the file if you don’t have write permission. Similarly if you want to execute a shell script file, you need execute permission on it.

Users are categorized based on the following groups.

1. **Owner** – the owner of the file, basically if the file is created by you, you are the owner of the file.
2. **Owner Group** – the group in which the owner is a member. Owner group level permission allow other users who also belong to this group to share the same permission level.
3. **Others** – those users who do not belong to the above categories.

The command **ls -lt** gives details about file permissions in first column. For eg, in the above screen shot let us examine the permission of newdir directory. The first character **d** specifies that it is a directory. Next three characters is about owner permissions. **rwx** shows that owner has all rights. Next three characters are related to owner group file permission. Here the members of owner group have no write permission. Similarly last three characters are related to other members permission. Here also they don’t have write permission.

chmod could be used with different argument types, but I always like to use number arguments.

4 – read access

2 – write access

1 – execute access

You should add these integers according to the given permission. Let us try some examples.

chmod 777 samplecopy.txt, this command gives full rights to all three types of users. First number is for owner, second for owner group and third for others. Suppose you want to protect your file so that others are not allowed to read it. You may give the command as

chmod 700 samplecopy.txt

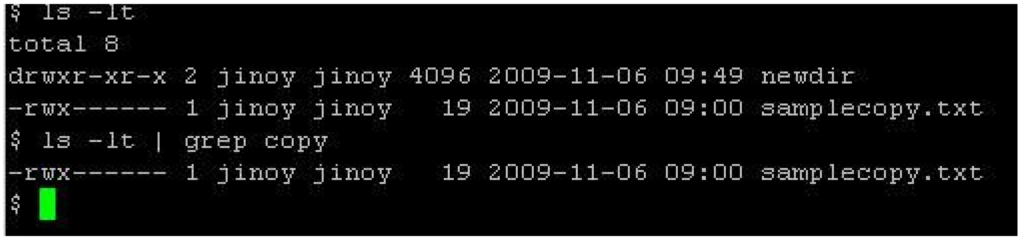
Similarly if you want to give just read access other than you, you may give command as,

chmod 744 samplecopy.txt

- **passwd –** this command is used to change the password of the current user. It asks for current password, once you supply it you may type the new password to change it.

- **grep** – this command is used to search for a particular pattern in a file or other input. This is commonly used with other commands like **ls**, **cat** etc as input using pipe symbol. One of the common use of the command is to search for a particular file in a directory. Suppose you have an image directory and you would like to know whether a particular image exists in directory. Then you may give command like this,

ls -lt | grep image.gif, this gives output row from detailed listing. For eg, just go through the following screenshot.



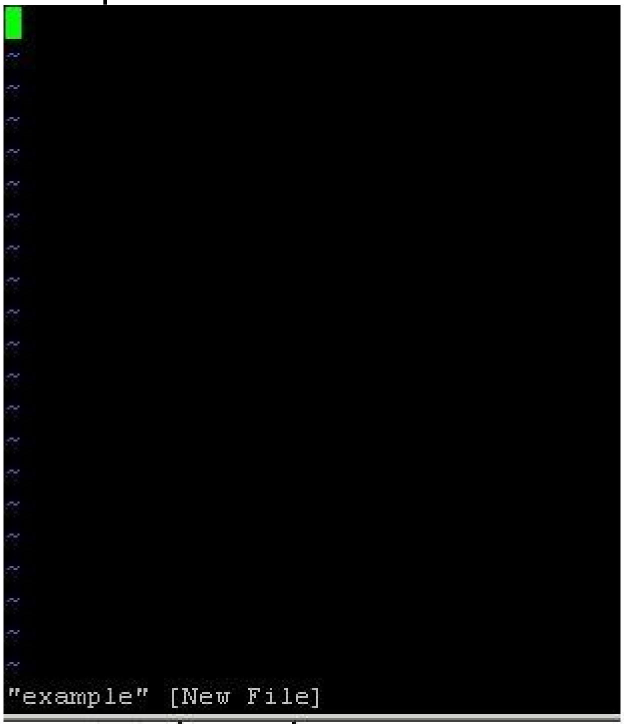
```
$ ls -lt
total 8
drwxr-xr-x 2 jinoy jinoy 4096 2009-11-06 09:49 newdir
-rwx------ 1 jinoy jinoy   19 2009-11-06 09:00 samplecopy.txt
$ ls -lt | grep copy
-rwx------ 1 jinoy jinoy   19 2009-11-06 09:00 samplecopy.txt
$
```

In the above example, we just want to search for a pattern “copy” which is found in samplecopy.txt file listing record.

- **vi** – **vi** is a text editor used to edit files which is a very import command in Linux. Some Linux versions like Ubuntu give an improved version of vi editor called **vim**. **vi** has three modes; insert mode, command mode and line mode.
- **Insert mode** – For making any changes in the file, you should be in insert mode.
- **Command mode** – When you open vi editor you will be in this mode. This mode is used to do many shortcut tasks like delete words, switch to insert mode, copy and paste operation etc. When you are in insert mode you may press [Esc] key to get into command mode.
- **Line mode** – this mode is used to issue commands for saving file, quitting editor etc. By pressing [**:**] key, you may get into this mode from command mode.

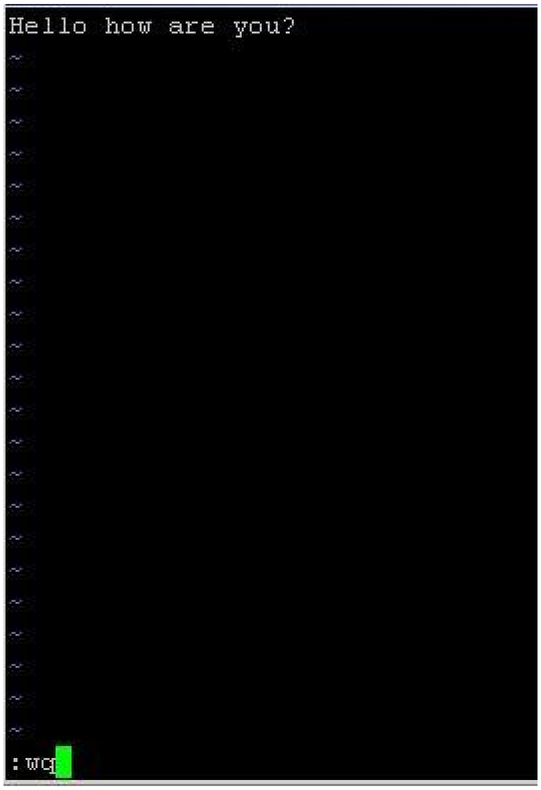
vi editor has large number of commands, you should be reasonably familiar with it in order to use Linux system. Let us do some examples here. Go to your home directory, type the following command and hit [Enter].

vi example



```
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"example" [New File]
```

The editor appears. Now we are in command prompt. If you want to get into insert mode press [i] key. Once you press [i] key, you may enter text into the editor. Just type "Hello how are you?" in the editor. Once you made modifications in the editor, you should go to command prompt by pressing [Esc] key. Now you are in command prompt. Now let us save the file. Save command is executed at line mode. Press [**:**] key. For saving the file, you have to give the command **wq** and hit enter. This command saves the file and quit the editor.



```
Hello how are you?
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq
```

Now verify the content of the file using **cat** command. Ok now let us examine some **vi** commands executed in command mode. In command mode, almost all commands do not require to press [Enter] key.

1. **i** – switch to insert mode, content is inserted just before the current cursor position.
2. **a** – switch to insert mode, text is appended just after the current cursor position.
3. **A** – switch to insert mode, start appending text after the end of the last character.
4. **x** – deletes single character in the current cursor position.
5. **dw** – deletes word in the current cursor position.
6. **dd** – delete current line.**u** – undo last change.
7. **h** – move cursor to left like left arrow key.
8. **l** – move cursor to right like right arrow key.
9. **j** – move cursor down to next line similar to down arrow key.
10. **k** – move cursor up to next line similar to up arrow key.

Some commands useful in line mode. Almost all commands in this mode are followed by pressing [Enter] key. You go to line mode from command mode by pressing [:] key.

1. **w** – save file, got back into command mode.
2. **wq** – save and quit editor.
3. **q** – exit editor.
4. **q!** – quit editor without saving last session changes.
5. **/[pattern]** – search for pattern in the file. It go back to command prompt and if any match is found, it keeps the cursor in the first match line.
6. **$** – move the cursor to the last line in the file.

- **man** – **man <command name>** is used to get help regarding a particular command in Liux. If you want to know more about a command you should use this command. For eg, **man ls** is used to get help for **list** command. Type the command in shell prompt and hit [Enter].

- **more** – Suppose you are making a detailed listing of large number of files in a directory. If you want to see the listing page by page, **more** command comes at your help. Just type the following command and hit [Enter].
  ls -lt /etc | more, here the listing of **ls** command is used as input for **more** command using pipe symbol. Here the number of records is showing in a page and waits for another key press. If you want to go to next page, you may press [Space] key .
- **tail** – **tail** command is used to print the last lines of input text. The default value of number of lines is 10. But you can limit the number of lines using **-n** or **–lines** argument. Go to your home directory and type,
  ls -lt | tail -n 1, this command prints the last line of listing. You may use the command to print the last portion of file also. For eg, **tail -n 1** example prints the last line of example file.
- **wc** – The word count command gives count about words, lines and characters in input text. **-l** argument gives number of lines, **-w** gives number of words, **-m** gives number of characters. For eg, **wc -w** example gives number of words in example file.

Navigating directories

`cd`

…change directory, method used for moving from one folder to another.

`cd foldername/foldername2`

…would take you to foldername2.

`cd ..`

…moves up one directory.

`cd /`

…moves you to root.

`pwd`

…prints to console your present working directory – where you currently are.

List contents

```
ls
```

...list contents of current directory.

```
ls -l
```

...shows long format inc. group owner, size, date modified, permissions.

```
ls -a
```

...shows ALL files inc. hidden files.

```
ls -R
```

...lists contents of sub directories recursively.

```
ls -la
```

...the above options can be used at the same time.

```
ls foldername/foldername
```

...list items in folder without actually moving to it.

Creating files and directories/folders

```
touch file.html
```

...use the touch command to create files.

```
rm file.html
```

...use the rm command to remove a file.

```
mkdir myfolder
```

...use the mkdir command to create a new directory/folder.

```
rmdir myfolder
```

…use the rmdir command to remove a directory/folder, folder must be empty.

```
mv folder1/file.html folder2/file.html
```

…use the mv command to move a file. Can also be used to rename a file.

Compressing and backing up files and folders

```
zip -r foo.zip foo/
```

…compress the folder ‘foo’ and all of its contents into a zip file called ‘foo.zip’.

```
zip foo.zip foo.html
```

…compress the file’foo.html’ into a zip file called ‘foo.zip’.

File and directory/folder permissions and ownership

```
chmod 755 file.html
```

…changes the file.html file permissions, same for a folder.

```
chmod -r 755 myfolder
```

…changes permissions for that folder and all folders and files inside it recursively.

Here are the chmod octal numeric values

700: only owner can read
755: everyone can read but not write
775: only group can read and write

770: no-one but group can read
666: everyone can read and write
1777: everyone can read,write,execute

```
chown user:myself file.html
```

…changes the ownership of file.html to the user called ‘myself’.