Ubuntu Linux Commands Cheat Sheet

This document provides most of the basic linux commands that we use in day-to-day operations

Listing and Navigating

Command	Description
cd ..	go down a directory
cd /	go to lowest level
cd /var/www	go to absolute path
cd ~	go to logged in user's home
ls	list files
ls -la	list all files, permissions, and hidden too
pwd	print working directory

Manage Files and Folders

Command	Description
cp -R	copy directory from location to new location
cp	copy file from location to new
mkdir	create a directory
mv	move directory from location
mv	move file from location to new
rm -rf	remove a directory with
rmdir	remove an empty directory
touch	create an empty file

Users

Command	Description
Passwd <username>	change logged in users password
su - username	switch users
sudo su	switch to root
useradd -m -s /bin/bash username	Create User
usermod -a -G existing_group existing_user	Add User to Group
who	show all logged in users
whoami	show which user you are

Groups: Do not delete groups you don't know what they are used for, that's dangerous!

Command	Description
groups	see what groups current user belongs to
groupadd name	create a group
groupadd -g 900 name	create a group with
groupdel name	delete a group
useradd	add current user to a
usermod -aG	append any user to an
cat /etc/group	list all groups

Permissions

There are two ways to manage permissions, one is by text the other is by an octal value.

Change Mode

```
; Options: (O)wner (U)sers (G)roup or (A)ll
; File:    Owner: rwx, Group: rwx, User: rwx
```

Single File read/write permissions

```
chmod g+rw file
chmod og+rw file.txt
```

Change Ownership

```
chown user:group files_or_folder
chgrp group files_or_folder
```

Recursively:

```
chown -R user:group files_or_folder
chgrp -R group files_or_folder
chmod -R og+rw files_or_folder
chmod -R g+s files_or_folder
```

Preserve Group Permissions

A fantastic way to structure your users is within groups. A common example would be your `www-data` group. If I have a user `jesse`, I can add him with `sudo usermod -aG www-data jesse`.

After adding any users I would like, I want to have a folder where all the members of the `www-data` group can read/write a folder. If they are using git, I also want the permissions to stay the same, meaning if they pull the permissions will not change.

To accomplish this, here is an example:

```
sudo chown -R deploy:www-data /var/www
sudo chmod -R g+rws /var/www
```

Octal Permissions

You may have seen this a lot, you can use octal or decimal (begins with a 0) to do the same thing.

```
Permissions:
0 = None
1 = Execute (e)
2 = Write (w)
4 = Read (r)
```

- There are 3 Permission types (Read, Write, Execute), or 4 if you count "None".
- There are 3 Sets: Owner/User/Group (In that order)

- So if you did `chmod 700 file.txt` it would allow the user to Read, Write and Execute
    - Because `7` is the total of `4 + 2 + 1`

Octal Examples

```
chmod 600 file.txt – Owner Read, Write

chmod 660 file.txt – Owner Read, Write; User Read, Write

chmod 770 file.txt – Owner Read, Write, Execute

chmod 770 file.txt – Owner Read, Write, Execute; User Read, Write, Execute

chmod 666 file.txt – All Read, Write

chmod 777 file.txt – All Read, Write, Execute
```

VIM Editor

```
 #to install vim editor

$sudo apt-get install vim

i - insert mode, write into the file

:wq - save and exit

:q! - exit without saving the file


Press the below letters h , k, j, l
```

* k -> move up

* j -> move down

* h -> move right

* l -> move left

:set number # display the line numbers

OS Details

Get fundamental information about your OS with the following commands, you may have to run them as `sudo`, eg: `sudo lsb_release -a`

```
lsb_release

lsb_release -a

lsb_release -as     # Short Information

lsb_release --help
```

CPU Info

```
nproc               # How many Processing Units

cpuid               # Must install cpuid from terminal

cat /proc/cpuinfo   # Lots of info
```

Usage Info

```
free -h             # Human readable, or do --help for options

vmstat -s

cat /proc/meminfo   # Lots of info
```

Disk Space

```
df
df -B MB    (In Megabtyes, KB for Kilobytes, GB for Gigabytes)
```

System Processes

```
top
htop  # If you installed it
```

IP Address

Your IP is after `inet addr`. If you are connect via ethernet it's under `eth0 (Ethernet)` otherwise, wirelessly it is likely under `wlan0 (Wireless LAN)`.

```
ifconfig
ip
ip addr show
ip addr show wlan
ip addr show eth0
```

Services

Use the service command *(Requires sudo)*

```
service ssh status      (service status)
service --status-all    (all services status)
```

Almost every service has the following commands, some may have more like apache `graceful-restart`:

```
service servicename start
service servicename stop
service servicename restart
service servicename status
service servicename force-reload
```

System State

```
uname -a (get linux info)

top (See running processes/system status, I suggest installing `htop`)
top -u www-data
htop -u www-data

df          (display disk space in bytes, default)
df -h       (display disk space human readable)
df -Th      (display disk space with partitions)

free (see memory used)
free -g (in gigabytes)
```

Processes

```
ps -ef | more       (current running processes)
ps -efH | more      (current running processes in a tree)

ps -ef | grep vim   (find vim process id)
kill -9 <id>        (no brackets)
```

OS Shutdown

```
shutdown
reboot
shutdown -h now
shutdown -h +10     (shutdown 10 mins)
shutdown -r now     (reboot now)
```

Service Commands

Use the service command *(Requires sudo)*

```
service ssh status      (service status)
service --status-all    (all services status)
```

Almost every service has the following commands, some may have more like apache `graceful-restart`:

```
service servicename start
service servicename stop
service servicename restart
service servicename status
service servicename force-reload
```

Finding Files

Generally the following arguments are as follows:

- `-type f` file
- `-type d` directory
- `-iname` case insensistive (book.txt would the same as BOOK.TXT)
- `*` is a wildcard to find anything, usually you put it at the start or end of a filename.

```
find . -name tecmint.txt
find /home -name tecmint.txt
find /home -iname tecmint.txt                       (case ignore)
find / -type d -name Tecmint                        (directory)
find . -type f -perm 0777 -print (with perms)
find / -type f ! -perm 777 (find without)
find . -type f -name "tecmint.txt" -exec rm -f {} \; (find and remove a file)
```

```
find . -type f -name "*.txt" -exec rm -f {} \;       (find and remove
multiple)
find /tmp -type f -empty                             (Find empty files)
find /tmp -type d -empty                             (find empty directories)
find / -size +50M -size -100M (findby size)
```

Find in Files (GREP)

GREP means: Global Regular Expression Pattern (or Parser)

Some common GREP flags:

- `-r` is Recursive
- `-n` is Line Number
- `-w` Match the whole word
- `-l` is lowercase only
- `-c` supresses normal output and counts number of matching lines

```
grep "hello" file.txt (if in file)

grep "hello" files*  (if in many files)

grep -i "hello"  file.txt  (case insesitive)

grep -iw "is" file.txt (get full words, case insensitive)

grep "regex" file.txt
```

Reading Files

Without having to open a file you can simply read a part of it without `nano`, `pico`, vi, `or` vim`:

```
cat file.txt            (view file contents)

tail file.txt           (view end of file contents)

tail -n20 file.txt      (view top 20 lines)
```

```
tail -f filetxt         (follow a filename keep updating)

head file.txt           (view top of file contents)

head -n20 file.txt      (view top 20 lines)
```

SCP

Download from server to local

```
scp root@server.com:/path/to/file.txt file.txt
```

Upload from local to server

```
scp file.txt root@server.com:/path/to/file.txt
```

SSH

Connecting to a server

```
ssh name@server.com  (default port is 22)
ssh name@server.com -p 8000 (connect to specific port)
ssh name@server.com -i ~/.ssh/rsa_key.pub (connect with ssh key)
```

.SSH Permissions

These are safe permissions to use for SSH

```
chmod 700 ~/.ssh
chmod 644 ~/.ssh/id_rsa.pub
chmod 600 ~/.ssh/id_rsa

# Put your pubkeys (one per line) for SSH login
chmod 600 ~/.ssh/authorized_keys
```

Using the Config

You can also create a `~/.ssh/config` file and store entries such as:

```
Host aws
Hostname ec2-50-50-130-50.compute-1.amazonaws.com
Port 22
Identityfile ~/.ssh/id_rsa
User myusername

Host my-vps
Hostname 50.50.130.50
Port 22
User root
```

You can then simply type:

```
ssh aws
ssh my-vps
```

SSH to PEM

Sometimes you may need a `PEM` format SSH Key. You can easily add this alongside your other SSH keys.

```
openssl rsa -in ~/.ssh/keyname_rsa -outform pem > keyname_rsa.pem

chmod 700 keyname_rsa.pem
```

Sudo

```
Vi /etc/sudoers

username  ALL=(ALL) NOPASSWD:ALL
```

Software package Update(apt)

Install a package - $ `sudo apt install nmap`
Remove a Package `$sudo apt remove nmap`
`Update Ubuntu: $apt update`
`Upgrade Ubuntu: $apt upgrade`

`List all of packages - $dpkg -l`