Chrplr's Linux tips

Release 10/04/2022

Christophe Pallier

Apr 26, 2024

CONTENTS:

Pdf version of this document
https://media.readthedocs.org/pdf/chrplr-linux-tips/latest/chrplr-linux-tips.pdf

Epub version
https://media.readthedocs.org/epub/chrplr-linux-tips/latest/chrplr-linux-tips.epub

This is a collection of Linux tips accumulated over the years.

Even if you are are a seasoned Linux user, you may still learn one thing or two. For example, do you know how to make bash unsensitive to filename case during TAB completion? Create bookmarks for directories so that you can jump (cd) directly into them? List all bash functions defined in the current environment? Parallelize `xargs`? Rename files using bash variable substitution? Search files that have been modified in the last 2 weeks? Encrypt a file on the command line with `gpg`? Wake up a remote computer which is powered off? Read on.

CHAPTER

ONE

INTERACTING WITH THE SHELL

When you type command lines, you are sending instructions to a program called the "shell". Unless otherwise indicated, the tips assume that you are using the *Bash* shell. You can check which shell is running with the command:

```
echo $SHELL
```

For a basic introduction to interacting with the shell, I recommend Learning the shell and The Linux command line by William Shott, or any of the following tutorials:

- UNIX Tutorial for Beginners
- Openclassrooms Initiation à Linux (in French)
- YouTube series on The Linux Command Line
- Classic Shell Scripting book

1.1 Some useful shortcuts

Bash relies on the readline library to interact with the user. Its behavior can be customized by options in `$HOME/.inputrc`. For example, to enable case-insensitive tab-completion:

```
echo 'set completion-ignore-case On' >> ~/.inputrc
```

1.1.1 Editing the current command line

- Ctrl-d : delete character under the cursor
- Ctrl-k : delete everything from the cursor till the end of the line
- Ctrl-u : delete everythin from the start if the line untl the cursor
- Alt-d : delete till the end of the current word
- Ctrl-a : move the cursor to the beginning of the line
- Ctrl-e : move the cursor to the end of the line
- Ctrl-b, Alt-b: move backward, by one char or by one word
- Ctrl-f, Alt-f: move forward, by one char or by one word

1.1.2 Navigating the history of command lines:

- Ctrl-r : interactive backward search in the history of command lines
- Ctrl-p : move to previous command line
- Ctrl-n : move to next command line

If you start a command line with a space, this line will *not* be saved in the history of commands.

At any time, you can clear the list of commands typed in the current session with `history -c`

The history of commands lines typed in the current shell is saved on the disk (that is, appended to the file pointed to by `$HISTFILE`) *only when you exit this shell*. But you can write them explicitly at any time using `history -w`

I recommend to set `$HISTSIZE` and `$HISTFILESIZE` to large numbers, e.g.:

```
export HISTSIZE=10000
export HISTFILESIZE=100000
```

in `.bashrc`

Check out `man readline` for many other keybindings and to customize the behavior of the command lines by setting variables in `~/.inputrc`

1.1.3 Job control

- Ctrl-c: Terminate (kill) the running job
- Ctrl-z: Stop the running job (you can then put in background with `bg`)
- Ctrl-d: Terminate the Shell and close the Terminal
- Ctrl-l: Clear the terminal
- Ctrl+s: Stop all output to the screen. This is particularly useful when running commands with a lot of long, verbose output, but you don't want to stop the command itself with Ctrl+C.
- Ctrl+q: Resume output to the screen after stopping it with Ctrl+S.

1.2 Commands

When you type something on a command line and press `Return`, you give an order to the shell. The first token is a command and the following tokens on the line are parameters, a.k.a. agurments.

There are different types of commands:

- *shell builtins*
- *user-defined functions*
- *aliases*
- *programs* (scripts or binary executable files), that is, executable files located somewhere on your file system.

The `type` command tells you the category of a command.

```
type echo  # shell buitin
help echo

type ll  # alias (typically defined in most linuxes)

type cp    # executable
cp --help

declare -F # list all functions
```

The vast majority of commands that you are going to type are programs (scripts or binary executables). The list of directories containing programs is stored in the environment variable `PATH`:

```
echo $PATH
```

Note that directory names are separated by the character `:`.

If you want to add a directory, say `/opt/bin`, to the PATH:

```
export PATH="/opt/bin:$PATH"
```

From now on, the shell will search for programs in `/opt/bin` before scanning the other directories listed in`PATH`. The program executed is the first encountered in the list. You can scan the list with:

```
which -a command
```

If you "screw up" the PATH, you will no longer have access to programs. In this situation, the best is to close the shell (by pressing `Ctrl-D`) and open a new one. You can test this situation typing just:

```
PATH=
```

If you want a modification of the `PATH` variable to be permanent, i.e. to be active each time you start a shell, add the `export PATH=...` line to the file `~/.profile`.

1.3 Creating scripts

If you happen to often type the same series of commands, it is a good idea to create a script, that is, basically, a text file gathering the sequence of commands to be executed. Then, you will just have to type the filename of this script to execute all the commands.

If it does not exist yet, create a `bin` directory in your home folder:

```
mkdir $HOME/bin.
```

Use a text editor to create a file `myscript` in this directory, and enter the following on the first line:

```
#! /bin/bash
```

Then type the series of commands (one per line) you want to be executed.

Save the file `myscript` and enter the commands:

```
chmod +x ~/bin/myscript
PATH="$HOME/bin:$PATH"
```

You can now type `myscript` on the command line to execute the series of commands.

To go further, you should learn how to use arguments to scripts.

Note that you write scripts in other languages than bash, e.g. python.

1.4 Startup scripts: .profile, .bashrc, .bash_profile

`~/.bash_profile`, `~/.profile`, `.bashrc` are scripts that are executed automatically when you start a shell. This allows you to set up your environment (e.g. the PATH, the Prompt, create aliases for common operations, ...)

There are four types of shells:

- login shells and non-login shells.
- interactive and non-interactive shells

If you connect to a remote computer with `ssh remote`, you get an *interactive login* shell.

If you execute a command on a remote computer with `ssh remote command`, the script or the command is executed in a *login non-interactive* shell.

If you are already log in, that is, you have open a session, and open a new terminal, you get an *interactive non-login* shell.

If you execute a bash script, it is launched in a *non-interactive, non-login* shell.

Login shells execute `~/.profile` and `~/.bash_profile`.

Non-login shells only execute `~/.bashrc`, not `~/.profile` nor `~/.bash_profile`

Anything that should be available to graphical applications OR to `sh` (or `bash` invoked as `sh`) MUST be in `~/.profile`, not `.bashrc` (If you launch a graphical application not from the terminal, it only knows about the environment that was created at login. In particular, it will not know about stuff in `.bashrc`) .

- **`~/.profile` has the stuff NOT specifically related to bash, such**
  as environment variables (PATH and friends)
- Anything that should be available only to login shells should go in `~/.profile`
- `~/.bashrc` has anything you'd want at an interactive command line (Command prompt, EDITOR variable, bash aliases)
- `~/.bashrc` must not print anything on the terminal. This could screw up sftp for example.
- `~/.bash_profile` should just load .profile and .bashrc (in that order)
- Make sure that `~/.bash_login` does *not* exist.

See:

- https://superuser.com/questions/789448/choosing-between-bashrc-profile-bash-profile-etc
- https://stackoverflow.com/questions/902946/about-bash-profile-bashrc-and-where-should-alias-be-written-in
- http://mywiki.wooledge.org/DotFiles

CHAPTER

TWO

JUMP DIRECTLY TO DIRECTORIES

2.1 Using symbolic links

You can use *ln -s* to create shortcuts to folders that you quickly need to access.

```
ln -s myproject_buried_in_some_very_deep_path_too_long_to_type  ~/myproject
```

After that, you will just need to type:

```
cd ~/myproject
```

2.2 Directory Bookmarks

If you are tired of typing intermediate directory names when changing directory, check out the *Directory Bookmarks functions for bash* described in this linux journal article about dirb.

Download https://raw.githubusercontent.com/icyfork/dirb/master/dirb.sh in your `$HOME` folder and add the following line to the file `$HOME/.bashrc`:

```
source $HOME/dirb.sh
```

Once installed, you can save bookmarks for specific directories (command `s`) and later jump into them directly (command `g`). Here are all the available operations:

```
s       Save a directory bookmark
g       go to a bookmark or named directory
p       push a bookmark/directory onto the dir stack
r       remove saved bookmark
d       display bookmarked directory path
sl      print the list of directory bookmarks
sl -l                   long list
sl -p                   path list
```

2.3 cd history

Another possibility is to use cdhist, a tool that replaces the `cd` command
by a new version that, when you type `cd --`, list the recently visited directories and let you select one. It is a python script, therefoere, to install it, you need to type:

```
pipx install cdhist
```

And add the following lines to `~/.bashrc`:

```
if type cdhist &>/dev/null; then
    . <(cdhist -i)
fi
```

CHAPTER

THREE

FIND FILES INTERACTIVELY

Install the fuzzy file finder `fzf`

    sudo apt install fzf

Then, the command `fzf` will let you explore the files in the working directory interactively

Later, we describe tools to find files (find, fdfind, grep, ack, ag).

3.1 Open a file from the command line

```
xdg-open file
```

CHAPTER

FOUR

KILL A PROGRAM THAT IS NO LONGER RESPONSIVE

It may happen that a program monopolizes most of the CPU, but does not longer respond to input. Such a program is crashed and should be "killed".

For applications running in a terminal, first try to press `Ctrl-C`.

If this does not work, or if the application is running in its own window but refusing to close, open a terminal and type:

```
pkill program_name
```

You can also use the command `ps -ef` to locate the application and note down the "process identification number" in the `PID` column. Then, type:

```
kill PID
```

(in place of PID, use the number associated to the process listed in 'ps' output). Check if the program was destroyed with the `ps` command; if not:

```
kill -9 PID
```

If the whole graphics system no longer responds, you can try to open a text mode terminal with `Ctrl-Alt-F1` or `Ctrl-Alt-F4`, log in and kill the programs that causes problem. Sometimes, the only solution is to kill `Xorg`, the display server).

It the keyboard does not repond anymore, before switching off the computer, you can try to connect from another computer on the same network using `ssh` and to kill the applications or do a proper shutdown (typing 'halt' on the command line).

CHAPTER

FIVE

REMOTE POWER OFF

Powering off is easy, just type:

```
sudo shutdown
```

You my want to specify a delay:

```
sudo shutdown --halt +1  # one minute delay
```

You can cancel the shutdown during the delay with:

```
  sudo shutdown -c

If you want to reboot the system::

  sudo shutdown -r now
```

CHAPTER

SIX

REMOTE POWER ON

If your workstation is switched off, but you can log to another linux computer on the same local area network, you might be able to power it on if you have authorized *Wake on lan (WOL)* in your station's BIOS parameters.

First, you need to know the MAC address of your computer's network interface (using `ip a` when the computer was on).

Say the MAC address is "c8:f7:50:bc:ea:f5", then the command:

```
wakeonlan c8:f7:50:bc:ea:f5
```

launched on the terminal of another computer will power on your computer.

See http://doc.ubuntu-fr.org/wakeonlan

CHAPTER

SEVEN

PRINTING

To get a list of available printers:

```
lpstat -p -d
```

To check the status of all printers:

```
lpstat -a
```

To print `file.pdf` (or more precisely to put in the printing queue) of the printer `printername`:

```
lpr -P printername file.pdf
```

To print two copies of a file

```
lpr -# 2 filename.pdf
```

To print 2 pages per side:

```
lpr -o number-up=2 -o sides=two-sides-long-edge filename.pdf
```

To remove a printing job:

```
lprm job-id
```

(`job-id` is the number reported by the `lpr` or `lpstat` commands).

If you use the same printer most of the time, you can create a script like the following in your `~/bin` directory:

```
#! /bin/sh
export PRINTER=my-beautiful-printer
lpr -P "$PRINTER" -o media=A4 "$*"
```

In case of printing problem, first Check that that the cups service is running:

```
systemctl  status cups.service
```

If you need to manage or add printers, open a browser on http://localhost:631

Check out Linux 101: Manage printers and printing for more information.

CHAPTER

EIGHT

ENCRYPT/DECRYPT FILES USING GPG

To use a one-time password:

To encrypt `file.txt`:

```
gpg --symmetric file.txt   # this will create file.txt.gpg
rm file.txt                # do not forget to remove the unencrypted file
```

To decrypt it:

```
gpg -o file.txt --decrypt file.txt.gpg
```

Note that it is also possible to use gpg to generate a private/public key pair to sign documents (see https://tutonics.com/2012/11/gpg-encryption-guide-part-1.html)

CHAPTER

NINE

CONFIGURE MULTIPLE DISPLAYS

Use the programs `xranrd` and `arandr`

```
arandr
xrandr --output eDP1 --rotate left
```

If you have a nvidia graphics card, you can also use `nvidia-settings`

CHAPTER

TEN

CONNECT TO REMOTE COMPUTERS USING SSH

A secure method to connect to a remote computer:

```
ssh computername
```

or, if your login id on the remote computer is different than the one on the local computer.

```
ssh login@computername
```

If you plan to launch graphical application on the remote computer, you need to add the `-X` option:

```
ssh -X login@computername
```

Note: you may need to run `xhost +` on the local (client) computer.

If you often connect to a computer, you can create an entry in `$HOME/.ssh/config`:

```
Host myserver
    Hostname gozilla.example.com
    User mickey
```

Then you will have just to type `ssh myserver` to log in.

To have TAB completion on server names contained in `.ssh/config` , create a file `/etc/bash_completion.d/ssh` with the following content:

```
_ssh()
{
 local cur prev opts
 COMPREPLY=()
 cur="${COMP_WORDS[COMP_CWORD]}"
 prev="${COMP_WORDS[COMP_CWORD-1]}"
 opts=$(grep '^Host' ~/.ssh/config ~/.ssh/config.d/* 2>/dev/null | grep -v '[?*]' | cut -
↪d ' ' -f 2-)

 COMPREPLY=( $(compgen -W "$opts" -- ${cur}) )
 return 0
}
complete -F _ssh ssh
```

Note that:

- the client computer must have the ssh client (`sudo apt install openssh-client`)
- the remote computer must be running a `sshd` server (run `sudo apt install openssh-server` on it).

You can troubleshoot connection issues with

```
ssh -vv login@computer
```

10.1 Set up SSH

To avoid having to type your login password each time you use ssh or scp, you can setup SSH to use public and private keys to perform the authentification automagically.

First, you must generate keyfiles, once, on your local computer. To do so:

```
ssh-keygen
```

This generates, among other files, a public key stored in a file `~/.ssh/id_rsa.pub`). You now need to copy this key in the file `~/.ssh/authorized_keys` on the remote computer you want to connect to. This can be done with:

```
ssh-copy-id  login@remotecomputer
```

If you have left the passphrase empty, you can know use `ssh` or `scp` without entering your password. But so can do anyone who has access to your account on the local computer.

So you may prefer to use a passphrase. To avoid having to type it each time you log to the remote computer, copy the following lines in your `~/.bash_profile`:

```
eval `ssh-agent`
ssh-add < /dev/null
```

You will be prompted for the passphrase only once: when you login on the local computer (See the explanations about `ssh-agent` at http://mah.everybody.org/docs/ssh).

10.2 Execute commands on a remote computer, without login

```
ssh login@computername command
```

Beware the `~/.bashrc` script on the remote computer will *not* be executed because ssh launches a non-interactive, non-login shell. Thus the remote `PATH` may not be what you expect! (solution: set the `PATH` in `.profile`, not `.bashrc`)

10.3 Keep a remote session alive

Once connected on the remote computer, execute:

```
tmux
```

When you want to leave, press `Ctrl-b d`. The terminal is *detached* but not closed.

Next time you connect to this remote computer, to continue your work, you can access the session:

```
tmux a
```

See https://danielmiessler.com/study/tmux/ for a primer on tmux, or read the book *Tmux 2: Productive Mouse-Free Development* by Brian Hogan.

10.4 Copy files to or from a remote computer

```
scp -r localdir remotelogin@remotecomputer:remotedir

rsync -avh localdir/ remotelogin@remotecomputer:remotedir

tar  -cf - dir | ssh login@remotehost tar -xvf -
```

10.5 Mount a remote folder with sshfs

```
sshfs login@remotecomputer:path local_path
```

10.6 Set up X11 forwarding with ssh

To allow graphical applications running on the server to display their windows on the local computer, when using ssh:

From https://unix.stackexchange.com/questions/12755/how-to-forward-x-over-ssh-to-run-graphics-applications-remotely

X11 forwarding needs to be enabled on both the client side and the server side.

On the client side, the -X (capital X) option to ssh enables X11 forwarding, and you can make this the default (for all connections or for a specific conection) with `ForwardX11 yes` in `~/.ssh/config`.

On the server side, `X11Forwarding yes` must be specified in `/etc/ssh/sshd_config`. Note that the default is no forwarding (some distributions turn it on in their default `/etc/ssh/sshd_config`), and that the user cannot override this setting.

The `xauth` program must be installed on the server side. If there are any X11 programs there, it's very likely that `xauth` will be there. In the unlikely case `xauth` was installed in a nonstandard location, it can be called through `~/.ssh/rc` (on the server!).

Note that you do not need to set any environment variables on the server. `DISPLAY` and `XAUTHORITY` will automatically be set to their proper values. If you run `ssh` and `DISPLAY` is not set, it means `ssh` is not forwarding the X11 connection.

To confirm that `ssh` is forwarding X11, check for a line containing Requesting X11 forwarding in the `ssh -v -X` output. Note that the server won't reply either way, a security precaution of hiding details from potential attackers.

CHAPTER

ELEVEN

RUNNING JOBS IN PARALLEL

You cand launch commands in background by appending a & sign at the end of the command line:

```
gedit &
```

(Note if a command is already running, blocking the terminal, you use `Ctrl-Z` and then the built-in command `bg` to put the process in background)

You can use this to launch several processes in parallel.

Consider the following shell script `test.sh` that just print its arguments and sleeps for 5s:

```
#! /usr/bin/bash
echo "Args: $@"
sleep 5s
echo "Done"
```

Set the executre flag:

```
chmod +x test.sh
```

You can launch 5 instances of it with a `for loop`:

```
for i in $( seq 1 5 ); do (./test.sh $i &); done
```

If you want to , to avoid mangling the standard output, you can save the output of each process in a different file:

```
for i in $( seq 1 5 ); do (./test.sh $i >test_${i}.out.txt &); done
```

To run each command in a different terminal (you may need to install `xterm`):

```
for i in $( seq 1 5); do (xterm -hold -title "Process $i" -e ./test.sh $i &); done
```

The `xargs` and `parallel` commands also allows one to launch the same command in parallel with different parameters:

```
seq 1 5 | xargs -P 5 -n 1 ./test.sh

parallel ./test.sh ::: $( seq 1 5 )
```

To know more, read their documentations…

CHAPTER

TWELVE

GET INFORMATION ABOUT THE SYSTEM

12.1 Which computer am I currently working on?

To display the network node name (also called the `hostname`):

```
hostname
```

or

```
uname -n
```

12.2 What is my public IP address?

To know your public address on the Internet:

```
sudo apt install curl
curl ifconfig.me
```

To know your IP address *on the local area network*:

```
ip addr
```

(you must identify the physical interface (ethernet card or wifi card) and check for the `inet` line)

12.3 Check available space on local disks

```
df -hT -x squashfs -x tmpfs
```

I actually added the following in my `.bashrc`:

```
alias df="df -hT -x squashfs -x tmpfs"
```

if you need to make space you can search for large folders or files using:

```
ncdu
du -h | sort -hr | less
```

If there is a quota system that limits the amount of space you can use on your account, you can check how much is available:

```
quota -s
```

12.4 List available disk partitions

```
lsblk | grep -v loop   # excludes loop devices
blkid
```

12.5 List the processes currently running on the system

To list the processes currently running:

```
ps auf
ps axuf   # also show process no tied to a terminal
```

The most important columns are `TIME` and `RSS` which show the time used by process since it started and the amount of real memory it uses.

If you want to list just some programs, for example `matlab`, type

```
pgrep -a matlab
```

For a real-time display of processes, you can use `top` or `htop` but a more comprehensive too is `glances`:

```
glances
```

Not only does it display CPU and memory usage, but also DISK I/O and network I/O. You can sort processes, for example, by CPU usage, etc (Press `h` in glances to see the help). Glances is extremely useful to identify bottlenecks (see https://livebook.manning.com/book/linux-in-action/chapter-13/74)

You may have to install it with `pip install glances` or `sudo apt install glances`.

Note: there exist other process monitoring tools: atop, btop, htop, nmon

12.6 Find the process that owns a file

Sometimes, it can useful to find the process that owns an open file:

```
lsof  filename
```

(See http://www.thegeekstuff.com/2012/08/lsof-command-examples/)

12.7 Get detailed information about your system

```
neofetch
```

```
sudo inxi -b
nvidia-smi  # if you have nvidia GPUs
```

To check how many CPU/cores are available on your machine:

```
lscpu -e
lscpu
```

To check the total amount of RAM installed on your computer and how much is currently being used by Linux:

```
free -h
```

Which Linux distribution is running:

```
inxi -b
lsb_release -a
```

Note: you may need to install the packages `inxi` and `lsb-core`:

```
# deb based linuxes: sudo apt install lsb-core
# rpm-based linuxes: yum install redhat-lsb-core
# redhat/fedora: dnf install redhat-lsb-core
```

Which version of the linux kernel is running:

```
uname -a
```

Note: for improved real-time performance, you need to see `PREEMPT_DYNAMIC`. You can also check the value of `CONFIG_HZ` with

```
grep ^CONFIG_HZ /boot/config-`uname -r`
```

If CONFIG_HZ is less than 1000, it may be worth installing a realtime kernel to reduce latencies and the probability of audio dropouts:

```
sudo apt install linux-lowlatency-hwe-22.04
```

12.8 Display detailed hardware information

```
lshw -short
hwinfo --short
lspci
inxi -b
```

12.9 Monitor temperatures

```
sudo apt install lm-sensors hddtemp
sudo sensors-detect
sensors
```

You can then install `psensor` to have a GUI monitoring the temperatures:

```
sudo apt install psensor
psensor
```

12.10 Monitor the performance of your computer

You can monitor your system with `glances`:

```
glances -t 5
```

or with `htop`:

```
htop -d 50 --sort-key PERCENT_CPU
htop -d 50 --sort-key M_RESIDENT
```

There are more specialized tools that focus on subsystems. For example, you can monitor the global activity of the CPUs with:

```
mpstat 5
```

To monitor the memory usage in real-time:

```
vmstat -S M 10
```

If any of the indicators `si` (`swap in`) or `so` (`swap out`) are high, your computer lacks memory and is using the swap (memory on disk).

You can check the file input/ouput volume and speed on the local drives:

```
iostat -x 2 5
iostat -h -d 10
```

Check the speed of your ethernet connection. Three tools are available:

```
mii-tool

ethtool

iperf
```

Or the general network performance:

```
netstat -i 10
```

Large `TX-ERR` or `RX-ERR` indicate a problem.

12.11 Check open listening ports

```
sudo netstat -tulpn
```

12.12 List all running services

```
systemctl -l -t service | less
```

12.13 Benchmark disk IO performance:

You can simply use *hdparm* and *dd*:

```
sudo hdparm -tv /dev/sdc1  # read test
dd if=/dev/zero of=/disk/temp oflag=direct bs=128k count=1G  # write test
```

(See https://linuxconfig.org/how-to-benchmark-disk-performance-on-linux)

For a more detailed analysis, install and run *fio*:

```
man fio

fio --name TEST --eta-newline=5s --filename=fio-tempfile.dat --rw=read --size=500m --io_
↪size=10g --blocksize=1024k --ioengine=libaio --fsync=10000 --iodepth=32 --direct=1 --
↪numjobs=1 --runtime=60 --group_reporting

fio --name TEST --eta-newline=5s --filename=fio-tempfile.dat --rw=write --size=500m --io_
↪size=10g --blocksize=1024k --ioengine=libaio --fsync=10000 --iodepth=32 --direct=1 --
↪numjobs=1 --runtime=60 --group_reporting
```

(from https://askubuntu.com/questions/87035/how-to-check-hard-disk-performance)

12.14 Benchmark 3D video performace

```
glmark2
```

12.15 Create a RAM disk

```
sudo mkdir -p /mnt/ramdisk
sudo mount -t tmpfs tmpfs /mnt/ramdisk -o size=1024M
sudo chown `whoami`:`whoami` /mnt/ramdisk
ls -al /mnt/ramdisk
```

12.16 Check power consumption

Two tools can be used to monitor power usage:

```
sudo powertop
powerstat
```

If you have a nvidia card:

```
nvidia-smi
```

12.17 Check open network connections

```
ss -tr
```

12.18 Perform a security check

```
sudo apt-get install -y lynis rkhunter clamav clamav-daemon -y

sudo lynis audit system
sudo rkhunter -c
```

CHAPTER

THIRTEEN

UPDATE FIRMWARE

```
systemctl start fwupd

# list devices that support firmware updates
fwupdmgr get-devices


# updating
fwupdmgr refresh
fwupdmgr get-updates
fwupdmgr update
```

CHAPTER

FOURTEEN

USERS

14.1 Who am I?

As far a the computer is concerned, the identity of the current user (its *user_id*), can be printed with:

```
whoami
```

Note that your login name and home directory are stored in the environment variables `LOGNAME` and `HOME`.

Each login is associated to a UserID (UID), an integer, and to a list of GroupIDs (GUID). You can list the information associate to the current login:

```
id
```

14.2 Check who is logged on the computer

To see who is currently logged on the system, use

```
who
```

or more simply:

```
w
```

If you are superuser, you can see a journal of the logins with the command:

```
sudo last
```

14.3 Who is that user?

To determine a person behind an user_id, use `finger`:

```
finger <user_id>
```

14.4 Change your identity

To temporally become `newuser`:

```
su - newuser
```

Of course, you will be prompted for newuser's password.

If you want to become `root`:

```
sudo -i
```

When you are done, type:

```
exit
```

14.5 Change your password

To change your password on the local system:

```
passwd
```

CHAPTER

FIFTEEN

GENERATE PASSWORDS

```
pwgen 10 --symbols
```

generates a 10 character long password with at least one special character

Run it online at https://pwgen.io/en/

15.1 Change the login shell

To change your login shell, e.g. from `/bin/csh` to `/bin/bash`:

```
chsh -s /bin/bash
```

15.2 Change group

Check which groups you belong to using `id`, then use

```
newgrp group
```

From now, the files and directories you create will belong to group `group`

To modify the group of already existing files in directory `dir`:

```
chgrp -R group dir
```

15.3 Change you UserID number

Each login is associated to a number called the `UID`. If for any reason you need to change your UID number, here is how to do it:

```
usermod -u <NEWUID> <LOGIN>
groupmod -g <NEWGID> <GROUP>
find / -user <OLDUID> -exec chown -h <NEWUID> {} \;
find / -group <OLDGID> -exec chgrp -h <NEWGID> {} \;
usermod -g <NEWGID> <LOGIN>
```

15.4 Grant a user the ability to run commands as root (sudo)

```
sudo usermod -aG sudo userlogin
```

Of course, you need to be in the list of sudoers yourself to be able to execute this command.

Using `sudo` is better than using `su`, check out why at https://phoenixnap.com/kb/sudo-vs-su-differences

CHAPTER

SIXTEEN

FILES AND DIRECTORIES

16.1 Where am i?

To know the current working directory:

```
pwd
```

To change the current working directory:

```
cd subdirectory     # move down inside a subdirectory
cd ..               # move up in the hierarchy of directories
```

Note that you can always go back to your home directory by just typing `cd` (without argument).

16.2 List files and subdirectories

```
ls            # list (non hidden) files and subdirectories in the current working␣
↪directory
ls -A         # list all files (including hidden ones)
ls -1         # list in a single column
ls -l         # show detailed information (filesize, modification date,...)
ls -t         # sort by modification date (most recent first)
ls -Slt       # sort by size (largest first)


ls PATTERN
```

where PATTERN is a globbing pattern which can contain "wildcards" characters such as * or ?:

* | matches any string | |
? | matches any character | |
`my*` | filename starting with `my` | |
`my?` | filename of 3 characters stating with `my` | |
`*xyz*` | filename containing `xyz` | |
`*.tar*` | filenames finishing with `.tar` | |
`*{md,txt}` | filenames ending in either `md` or `txt` | |
`*.[ch]*` | filename ending in `.c` or `.h** | |

By default, `ls` only lists the files in the current working directory. To recursively visit the subdirectories:

```
ls -R
ls **/**.py
```

`**` will match the first-level subdirectories. With the option `shopt -s globstar`, subdirectories at all levels are visited.

To only display subdirectories:

```
ls -d */       # only directories
tree -d        # Recursively
tree -d -L 2   # limit depth to 2
```

16.3 Copy, rename, move or delete files

To copy a file inside the same directory, giving it name2:

```
cp file1 file2
```

To copy a file from the current directory to the existing directory `target_dir`:

```
cp file1 target_dir
```

To copy all the files from the current directory to another directory:

```
cp * target_dir
```

To do the same thing but showing a progress bar:

```
rsync --info=progress2 * target_dir
```

To rename a file:

```
mv file1 file2
```

To move a file to the existing directory `dir`:

```
mv file1 dir
```

To delete a file:

```
rm file
```

To avoid being asked for confirmation:

```
rm -f file
```

16.4 Create, copy, move or delete directories

To create a new directory:

```
mkdir -p newdir
```

To copy the directory `dir` inside the destination directory `destdir`:

```
cp -a dir destdir
```

(Note: the `-a` option does a recursive copy, that is, includes the subdirectories and preserves the attributes of files)

Alternatively, you can use `rsync`:

```
rsync -a --info=progress2 dir/ destdir
```

To move the whole directory `dir` inside the existing `destdir`:

```
mv dir1 destdir
```

To rename directory `dir` as `dir2`:

```
mv dir dir2
```

To delete the directory `dir` and all its content:

```
rm -rf dir
```

16.5 Rename files, replacing their name by their creation date

Here is a script that replaces filenames by creation date (this can be useful for a photo album)

```
#! /bin/bash

for fullfile in "$@";
do
  filename=$(basename "$fullfile")
  extension="${filename##*.}"
  filename="${filename%.*}"
mv -n "$fullfile" "$(date -r "$fullfile" +"%Y%m%d_%H%M%S").${extension}";
done
```

16.6 Check or modify the rights of access to a file or a directory

When you use `ls -l` to list the files in a directory, the first string of characters, made of `x`, `r`, `w`, `-`... specifies the *access rights* (Consult Understanding file permissions on Unix: a brief tutorial)

To allow everybody to read a file `aga` in the current directory:

```
chmod a+r aga
```

To allow everyone to enter a directory `mydir` and read its content:

```
chmod a+rx mydir
```

To make all subdirs and files readable by everyobody:

```
find -type d -exec chmod a+rx '{}' '+'
find -type f -exec chmod a+r  '{}' '+'
```

If, when using `ls -l`, there is a + sign is trailing the rights, it means that ACL (Access Control List), is set on the files or directories. The chmod command will not work: you must then use the `getfacl` and `setfacl` commands to list or modify the access/write rigths

16.7 Links

To avoid copying a file in several places on the same disk, it is a better idea to use a *hard link*:

```
ln existingname newname
```

Thus the same file can have several names (and be in several directories at the same time). Importantly, this only works if the directories are on the same filesystem.

To create a symbolic link (somewhat similar to a 'shortcut' in Windows):

```
ln -s filename newname
```

If you delete or move the file, the symbolic links will be 'dangling'.

To find and remove dangling links in a directory:

```
symlinks -rd directory
```

CHAPTER

SEVENTEEN

FIND FILES OR DIRECTORIES

The classic unix command to find files is, well, `find`.

We describe it below, but we first introduce a simpler and user-friendly alternatives: `fd`.

17.1 Using the `fd` command

Examples of usage:

```
fd statement    # search for files/directories containing the string "statement" in␣
↪their name
fd -t f statement    # restrict the search to files (not directories)
fd pdf ~/Downloads/ --changed-within 1hour
```

You can search for filenames matching a regular expression:

```
fd 'April.*docx$'
```

Features of fdfind:

- Regular expression (default) and glob-based patterns
- Very fast due to parallelized directory traversal
- Uses colors to highlight different file types (same as ls)
- Supports parallel command executio
- Smart case: the search is case-insensitive by default. It switches to
- case-sensitive if the pattern contains an uppercase character*.
- Ignores hidden directories and files, by default.
- Ignores patterns from your .gitignore, by default.

Note: You may need to install `fd` using `sudo apt install fd-find` or from https://github.com/sharkdp/fd, and define `alias fd=fdfind`.

17.2 Using the ag command

Another must know user-friendly search tool is `ag` which allows to spot text files containing a given string or regular expression:

```
ag --python "import numpy"    # search python files that import numpy
```

Note: To install `ag` under Ubuntu: `sudo apt install silversearcher-ag`.

17.3 Using the classic unix `find` command

`find` is the classic command, which is complex but powerful. THe basic syntax is:

```
find -name pattern
```

where `pattern` can be a string, or a glob pattern (not a regular expression):

```
find -iname 'filename.txt'
find -iname '*.doc'
```

The last command will list all `*.doc` files in the current directory and its subdirectories. The depth of subdirectories to visit can be limited:

```
find -maxdepth 2 -name '*.doc'
```

If you prefer regular expressions to glob patterns, use the option `-regex` instead of `-name`:

```
find -regex '.*.txt'
```

With `-o` you can specify an 'or'. For example, to search for for files with extension `nii` or `img`:

```
find \( -name '*.nii' -o -name '*.img' \)    # files ending in .nii or .img
```

With `!`, you can negate a search:

```
find ! -name '*.nii'   # all files except those ending in .nii
```

You can specify a time-range:

```
find -mtime 0  # find the files created or modified in the last 24hours
find -mtime +30 -mtime -60  # find files modified in the last 30-60 days
find -newermt 20171101 ! -newermt 20171201 -name '*.pdf' -ls  # find pdf files modified␣
↪between two dates
```

You can specify that you only search for, e.g., directories, using the `-type` argument:

```
find -type d # list all subdirectorectries
find -type d -mtime -10  # find the directories created or modified in the last 10 days:
```

You can find and delete all empty directories:

```
find . -type d -empty -print
find . -type d -empty -delete
```

You can filter on permissions

```
find -perm -o+x -ls -type f  # list all file with the execute flag set on 'others'
```

You can also execute a command on each file:

```
find -name '*~' -exec rm '{}' '+'  # delete all files '*~'
find -name '*.py' -exec mv -t path '{}' '+'  # move all py files to path
find -name '*.txt' -print0 | xargs -0 grep -l Alice   # show files
```

Note that `xargs` can be parallelized with the -P option:

```
find -name '*.nii' -o '*.img' -print0 | xargs -0 -P 10 gzip  # gzip all image files
```

Consult `info find` and `info xargs` for more information.

17.4 plocate

To accelerate file search, you can generate a database of all filenames on your filesystem. First of all, make sure you have installed `plocate`:

```
sudo apt install plocate
sudo updatedb
```

Enable an automatic update of the database:

```
sudo systemctl enable plocate-updatedb.timer
sudo systemctl start plocate-updatedb.timer
```

And then use the command:

```
plocate PATTERN
```

Note that the plocate will return all files where PATTERN matches any substring in the full pathname (including directories).

Read the manual:

```
man plocate
```

17.5 Search files by content

```
grep PATTERN file
```

where PATTERN is a regular expression (See `man grep`).

To search files recursively in subdirectories, you can combine `find` and `grep`:

```
find -type f -name "*.tex" -print0 | xargs -0 grep -n PATTERN
```

But this is complex! An interesting alternative is to use `ack` (https://beyondgrep.com/). By default, it does a recursive search and it can focus on certain file types.

```
ack --python -w TOKEN  # search only python file matching on word 'TOKEN'
```

To install `ack` under ubundu:

```
sudo apt install ack-grep
```

Another search tool is `ag` http://conqueringthecommandline.com/book/ack_ag:

```
sudo apt install silversearcher-ag
```

Tools like `grep`, `ack` and``ag`` are useful to search within text files but pretty useless for binary files. If you need to search within `.pdf` or `.doc` files, you first need to extract the textual content and then index it. Then, you will be able to search files by their content. To this end, you can install and use a tool like `recoll` (see http://www.lesbonscomptes.com/recoll/). One issue though it that the index can quickly grow very large.

CHAPTER

EIGHTEEN

COMPARE FILES OR DIRECTORIES

18.1 Compare two files

To list all the lines that differ between file1 and file2:

```
diff file1 file2
```

`meld` provides a nicer, graphical way to show the differences between two files or two directories.

```
meld file1 file2
```

When comparing text file, you may want to ignore changes in whitespaces (e.g. wrapping of paragraphs), then use `wdiff`.

```
wdiff file1.txt file.txt
```

To compare two latexfiles:

```
latexdiff file1.tex file2.tex
```

To create a patch listing the changes from version1 to version2:

```
diff -aur version1 version2 >dir2.diff
```

To apply the patch to version1 and generate version2:

```
patch -p1 <dir2.diff
```

18.2 Compare two directories

To compare two directories:

```
diff -r --brief dir1 dir2
```

`diff` compares the contents of the files. For large directory, this may be too slow. To run a faster comparison based on file sizes, you can use:

```
rsync --dry-run --recursive --size-only -i  source/ target/
```

18.3 Synchronize two directories bidirectionaly

```
unison
```

18.4 Backups

To back up my laptop, I use rsnapshot. I use an external harddrive with a large ext4 partition (~4 times the sizae of my laptop harddrive).

```
sudo apt install rsnaphost
```

Configuring rsnapshot essentially consists of editing `/etc/rsnapshot.conf` to specify where to save snapshots. In my case:

```
snapshot_root   /media/cp983411/WD_BLACK/rsnapshot/
```

Another nice backup utility, with a graphical interface, is:

```
backintime
```

It can be set up to automatically start so that you just have to plug your backup hardrive to performe a backup. Check out http://backintime.readthedocs.io.

CHAPTER

NINETEEN

WEB

19.1 Aspire pages from web sites

```
wget  URL
wget  --recursive --level 2  --no-cookies --page-requisites --convert-links URL

curl  address
```

19.2 Transfere files betwee computers

19.2.1 rsync

To send `folder` to a remote host:

> rsync -azv folder username@hostname:path

To reverse the direction of transfer, simply swith the two arguments.

An interesting option is `--delete` which makes the remote a mirror of the local.

19.2.2 netcat

See https://tutonics.com/2012/05/netcat-basics.html

19.2.3 scp

Copy remote folder locally:

```
scp -r username@hostname:path_to_folder .
```

Send local folder to remote host:

```
scp -r folder username@hostname:path
```

19.2.4 FTP

If you need to transfer files using the ftp protocol, you can use the following clients

```
ncftp
lftp
```

Transfer fil

CHAPTER

TWENTY

GIT

20.1 Use git to keep an history of your projects and collaborate

Another approach to synchronise dirs is to use git repositories.

Learn about git by reading https://git-scm.com/book/en/v2

See also git-annex

20.2 Create a copy of a local git repository on github.com

```
git push --mirror git@github.com:username/project.git
```

CHAPTER

TWENTYONE

DISABLE THE TOUCHPAD WHILE TYPING

```
killall syndaemon
syndaemon -i 1 -KRd
```

CHAPTER

TWENTYTWO

UNFREEZE THE MOUSE

```
sudo rmmod psmouse
sudo modprobe psmouse
```

CHAPTER

TWENTYTHREE

THE SYSTEM IS NOT RESPONDING

Try `Ctrl-Alt-F1` to open a terminal. From there, you might be able to do:

```
sudo shutdown now
```

Alternatively, press `Alt+PrintScr`, and, keeping this key pressed, type, slowly, `reisub`. This mysterious sequence is explained at https://linuxconfig.org/how-to-enable-all-sysrq-functions-on-linux#h6-the-sysrq-magic-key or https://en.wikipedia.org/wiki/Magic_SysRq_key

CHAPTER

TWENTYFOUR

CHANGE THE BRIGHTNESS OF THE DISPLAY

```
sudo brightlight -r     # read
sudo brightlight -i 10  # increase
sudo brightlight -d 10  # decrease
```

or

```
xbacklight -set 50
```

or

```
xrandr --output eDP1 --brightness 0.5
```

24.1 Lock the screen under X11

Assuming that `xscreensaver` is running in the background.

```
xscreensaver-command -lock
```

or:

```
i3lock -d 30 # if you use i3wm
```

24.2 Suspend to RAM

```
systemctl suspend
```

24.3 Suspend to disk

```
systemctl hibernate
```

Note: To hibernate on disk, the size of the swap partition must be larger than the RAM size.

24.4 Reboot

```
systemctl reboot
```

24.5 Shutdown

```
systemctl poweroff
```

CHAPTER

TWENTYFIVE

GRAPHICS

25.1 Manipulating Images

Make sure to have ImageMagick installed (e.g. `sudo apt install imagemagick` on a Debian-based system)

To get information about an image:

```
identify image.png
```

To display an image (gif, .jpg, .png, .tiff, eps, ...) use:

```
display file.gif
eog image.png
```

To convert from one format to another:

```
convert file.jpg file.png
```

To resize an image:

```
convert img.png -resize 66%  img_small.png
convert img.png -resize 400x400 img_400.png
```

To juxtapose several images:

```
montage -tile 4x4  *.png -geometry 1024x768 output.png
```

To superimpose images:

```
composite img1.png img2.png result.png
```

For more complex manipulations of bitmap image, I mostly use The Gimp

```
gimp file.jpg
```

25.2 Photography

To manipulate photographs, checkout:

- darktable
- Lightzone
- RawTherapee

25.3 Drawing

To draw on canvas (with pencils, brush, . . .)

- mypaint
- krita

25.4 Creating graphics

To edit vector graphics files, e.g. `.svg`:

```
inkscape
```

To create graphs:

```
dot
```

To plot data, I use `R` or `Python``:

```
import matplotlib.pyplot as plt
import numpy as np
```

25.5 Take a screenshot

To take a snapshot, that is, copy a portion of the screen into an image file, you can use ImageMagick's command `import`:

```
import file.png
```

You will then be able to select a rectangle on the screen with the mouse, which will be copied in `file.png`.

Other screenshot programs include `gnome-screenshot`, `ksnapshot`, `scrot`, `maim`. . . See https://wiki.archlinux.org/index.php/Screen_capture for a list.

CHAPTER

TWENTYSIX

MAKE A SCREENCAST

Voir http://www.linuxlinks.com/article/20090720142023520/Screencasting.html

Under i3, see https://github.com/synaptiko/.files/blob/4a6a549dfe0c22d19f38e32129b5c05de2bb6d34/i3/record-screen.sh

CHAPTER

TWENTYSEVEN

SOUND

Assuming that your Linux distribution is running the pulseaudio sound server — which can be checked with `pactl list` —, install `pavucontrol` to control the sound levels and which sound card each software is using.

27.1 Connect a MIDI instrument

Follow the instructions at http://tedfelix.com/linux/linux-midi.html. In a nutshell:

```
sudo apt install jackd2 jack-tools fluidsynth aconnectgui vmpk qjackctl qsynth  fluid-
↪soundfont-gm
```

1. To avoid potential latencies, you may want to install a kernel with the PREEMPT option:

   sudo apt-get install linux-lowlatency-hwe-20.04

2. Launch `qjackctl`, in the setup tab, set Frame/period to 128 to reduce latency, ans press 'start'
3. Use aconnectgui to connect your MIDI keyboard
4. Launch qsynth, add the soundfounds in setup and restart it.
5. In qjackctl, use connect and the patchbay.

CHAPTER

TWENTYEIGHT

MISCELLANEOUS

28.1 Access files on a data CD or on a floppy

With some Linux systems, you just insert the CD or the floppy and the content become available in the directory `/mnt/cdrom` or `/mnt/floppy`:

```
ls /mnt/cdrom
ls /mnt/floppy
```

If the floppy is not write-protected, you can create or copy files in /mnt/floppy just like in any ordinary folder.

Note that if you have several cdrom or floppy drives, they may have names cdrom1, cdrom2, floppy1,...

In some Linux systems, it is necessary to manually *mount* the cdrom or the floppy before accessing the files, and *umount* it before ejecting it. For the cdrom:

```
mount /mnt/cdrom
ls /mnt/cdrom
...
umount /mnt/cdrom
eject
```

For the floppy:

```
mount /mnt/floppy
ls /mnt/floppy
umount /mnt/floppy
```

If you get an error message like `mount: only root can do that`, ask the system administrator to grant you right to mount floppies by adding the `user` option the configuration file `/etc/fstab`. More information in the manual pages of `mount` and `fstab`:

```
man mount
man fstab
```

Concerning floppies, some systems have `mtools` installed (see `man mtools') which provide themdirandmcopycommands that emulate the old DOS commandsdirandcopy`. It is not necessary to mount the floppy to use them.

28.2 Format a floppy

To format the floppy with an ext2 filesystem, and mount it:

```
fdformat /dev/fd0
mkfs -t ext2 /dev/fd0
mount -t ext2 /dev/fd0 /mnt/floppy
```

This floppy can be read only on other linux systems. To be able to read it under Windows/DOS, you should use a DOS filesystem with mkdosfs in place of mkfs -t ext2:

```
mkdosfs /dev/fd0
```

28.3 Split a large file on several floppies

First compress the file, with gzip or bzip2 (see section*41*). If it still does not fit on a single floppy (1.4Mb), you can use the command split:

```
split -b1m file
```

This create a series of x?? files which you can copy on separate floppies.

To reassemble the files:

```
cat x* >file
```

28.4 Rip an audio CD

to extract all tracks from an audio CD:

```
cdparanoia -B
```

To just extract one track:

```
cdparanoia -w track_number file.wav
```

If you prefer GUI, you can open `konqueror`, and type '`audiocd:/`' in the address bar. This will show you the content of the CD, which you can copy somewhere else. Copying from the mp3 or ogg folders will do the automatic translations for you.

There are various programs with graphical interface which allow you to rip audio CD: `grip` and `kaudiocreator`, `rhythmbox`.

28.5 Convert from wav to mp3

I use lame:

```
lame file.wav file.mp3
```

28.6 Convert from wav to ogg vorbis

I use oggenc:

```
oggenc file.wav -o file.ogg
```

28.7 Rip an Audio cd into mp3 or oggenc

You could write a script calling `cdparanoia` then `lame` but there is a nifti command line tool, `abcde`, which queries music databases to find the tracks' song titles.

```
abcde -o mp3  # rip an audio cd track and converts into mp3
```

If you prefer a GUI, use asunder

28.8 Rip a DVD

Use (handbrake)[https://handbrake.fr/]

28.9 Create a data CD

1. Gather all the files you want to save in a given directory, e.g. `/tmp/mycd`
2. Create an iso image:

   ```
   mkisofs -o cd.iso -J -R /tmp/mycd
   ls -l cd.iso
   ```

   Check that the resulting file `cd.iso` file is not too large to fit on the CD; if it less than 650Mb, this should be ok.
3. Record on the cd (you must be root).

   You must know which is the device is associated to the CD writer drive.

   ```
   cdrecord -scanbus
   ```

   To determine the x,y,z scsi coordinates of your cd writer. If it does not appear listed, it may be because the ide-scsi parameter was no passed to the Linux kernel (See the HOWTO about CD Writing).

   To record, do:

   ```
   cdrecord dev=x,y,z -multi speed=0 -data cd.iso
   ```

28.10 Create an audio CD

To record on an audio CD all the `*.wav` files which are in the current directory:

```
cdrecord dev=x,y,z -pad speed=0 -audio *.wav
```

(x,y,z must be replaced by the numbers returned by cdrecord -scanbus)

28.11 Make backups

You can write backup scripts using `rsync` but it has already been done many time. I have used backintime, but borgbackup looks interesting.

28.12 Connect to a bluetooth device

```
sudo service bluetooth start
sudo service bluetooth status

rfkill list
rfkill unlock 0:


bluetoothctl
  power on
  devices
  scan on
  pair XXXXXXX
  connect XXXXXX
```

28.13 Convert doc or odt documents to pdf

```
libreoffice --headless --convert-to pdf *.odt
```

28.14 List the hosts in a NIS domain

If you are connected on a local network administrated by NIS (`yellow pages`), you can display the list of other computers on the network:

```
ypcat hosts
```

28.15 Mounting a Samba Share

Assuming you have a SAMBA server with IP 192.168.0.50

```
smbclient -L 192.168.0.50
sudo mount -t cifs //192.168.0.50/BACKUPS /mnt -o username=chrplr,file_mode=0777,dir_
↪mode=0777
```

28.16 Which shell is running?

When you enter commands on the command line in a terminal, the text you type is interpreted by a program called the 'shell'. There are different shells that speak different dialects. To determine the shell you are communicating with, type:

```
echo $SHELL
```

Note: this does not work well for subshells:

```
bash
echo $SHELL
csh
echo $SHELL
exit
exit
```

28.17 Get help. Find manuals

Many commands have associated `man pages`. To read the man page associated, for example, to the command `cp`:

```
man cp
```

Some commands also have manuals in the form of `info files`:

```
info gawk
```

On many linux systems, there is additional documentation in the `/usr/share/doc` folder. The HOWTOs can be especially helpful.

To browse them, install `dwww`:

```
sudo apt install dwww
sudo a2enmod cgi
sudo systemctl restart apache2
sudo  dwww-index++
```

Then:

```
dwww
```

28.18 Cut'n paste

Cutting & pasting under linux is not always straigtfoward. This is due to the fact that there are various systems of cut'n paste cohabitating.

To copy text, the following works with most applications:

- Click the left button and drag the cursor over the text to be copied.
- Click on the middle button to paste.

Note that this is very convenient: there no need to explicitly 'copy' the text.

If you use the window manager 'kde', there is a useful applet called 'klipper' located on the panel. Klipper keeps copies of the most recent clipboard contents. If a cut'n paste operation does not work, you may open klipper, select the relevant line, and retry to paste. It usually works.

If it does not work, then you can try the Cut/Copy/Paste functions from the applications' menus. Sometimes, it is necessary to save the region as a file in the first application, and insert this file in the second application.

28.19 set up tap to click in i3

```
sudo mkdir -p /etc/X11/xorg.conf.d && sudo tee <<'EOF' /etc/X11/xorg.conf.d/90-touchpad.
↪conf 1> /dev/null
Section "InputClass"
        Identifier "touchpad"
        MatchIsTouchpad "on"
        Driver "libinput"
        Option "Tapping" "on"
EndSection

EOF
```

Tip from https://cravencode.com/post/essentials/enable-tap-to-click-in-i3wm/

28.20 Mount a partition of a usb drive

Insert the USB drive, use `lsblk` or `dmesg` to find partitions, then use `pmount` or `udisksctl`:

```
lsblk
pmount /dev/sdb1
udisksctl mount -b /dev/sdb1
```

28.21 Check an SD card

```
sudo apt install f3
lsblk  # to find out which DEVICE the card is associated to
f3probe sudo ./f3probe --destructive --time-ops DEVICE
```

28.22 Setup an ethernet card to access the internet

You need to know IP, MASK, GATEWAY, DNS, HOSTNAME and DOMAIN:

```
ifconfig eth0 IP netmask MASK up
route add -net default gw GATEWAY netmask 0.0.0.0 eth0
hostname HOSTNAME
echo "domain DOMAIN" >/etc/resolv.conf
echo "nameserver DNS" >>/etc/resolv.conf
```

28.23 Changing/Editing network connection

```
nmtui  # text mode
nmcli  # text mode
unity-control-center
```

28.24 Install new software

If it come as a .tar.gz and contain a configure script

```
tar xzf package.tar.gz
cd package
./configure --prefix=$HOME & make & make install
```

This install the software in your home directory. To install it for every user, you need to omit the prefix option and be root when calling `make install`.

If you are on a apt-based system (Debian, Ubuntu):

```
sudo apt install packagename
```

If you have the .deb file:

```
sudo dpkg -i file.deb
```

If you are on a rpm-based linux system, to install an rpm file:

```
rpm -i package.rpm
```

To check if the package is correctly installed:

```
rpm -V package
```

To remove it:

```
rpm -e package
```

28.25 Check if a software package is installed

To check if, say, ghostscript is installed:

```
rpm -q ghostscript
```

You can get the list of all installed packages:

```
rpm -qa
```

28.26 Dynamic libraries

To run, some programs need to access functions in dynamic libraries. Dynamic libraries have the extension .so. They are located in /lib, /usr/lib, /usr/local/lib…

To list the libraries needed by a program:

```
ldd program
```

After adding new a new dynamic library, e.g. in /usr/local/lib, you must run, as superuser:

```
ldconfig -n /usr/local/lib
```

It is possible, as a user, to tell linux to search libraries in a particular place, using the LD_LIBRARY_PATH variable. For more information about how dynamic libraries are accessed, consult the manual of ld.so:

```
man ld.so
```

28.27 Command-line fun

```
sudo apt install cmatrix
cmatrix
```

28.28 Get back your sanity with a productive environment

The following works for me.

- Use a window manager that allows you that launch applications pinned on some workspace and to have the workspaces accessible by a fixed keystroke. The tiling window manager i3wm fits the bill.
- use Emacs/Spacemacs or vim as an editor
- Use Linux rather than Windows
- use anaconda3 for Python
- use git for projects

28.29 Common file types

e xtension	file type	application(s)
txt	text or ascii file	cat, less (view), vim, emacs (edit)
pdf	Adobe PDF	evince, okular (view, annotate), pdfarranger
ps, eps	postscript	gv (view) pstops (rearrange) ps2pdf (convert)
html, htm	web page	links, konqueror, mozilla (view) soffice (create)
png, jpg, gif…	graphic files	display (view) import (snapshot) convert (convert) gimp (manipulate)
doc, xls, ppt	Office document	soffice
sxc, sxi, sxw	OpenOffice document	soffice
tex	TeX and LaTeX documents	tex, latex, pdflatex (process)
dvi	Dvi documents	xdvi (view) dvips, dvipdf (convert to ps or pdf)
gz, Z, xz, bzip	Compressed file	gunzip, xz, unxz, zip, bunzip2, bzip2
tar	tar archive	tar tf (view) tar xf (extract) tar cf (create)
tar.gz	compressed archive	tar xzf (extract)
tar.bz2	Compressed tar archive	tar xjf
zip	zip archive	unzip -l (view) unzip (extract) zip (create)

CHAPTER

TWENTYNINE

SIMILAR RESOURCES:

- Linux Commands Cheat Sheet