Le Terminal Linux

Le terminal (ou console, ou shell) permet de faire exécuter des commandes au système sans avoir besoin d'interface graphique. Cela est très pratique pour administrer un serveur, pour dépanner une machine à distance, pour faire des raccourcis vers des commandes, ou encore pour écrire des scripts. L'utilisation de terminal est souvent plus rapide, et il a l'avantage d'être universel : peu importe le système Linux ou l'environnement de bureau, les commandes sont identiques.
Le terminal est accessible dans l'interface graphique via un émulateur de terminal. On peut aussi accéder à plusieurs terminaux (TTY, TeleTYpe) en quittant l'interface graphique, à l'aide des touches [Ctrl]+[alt]+[F1] à [F6]. On retourne à l'interface graphique avec [Ctrl]+[alt]+[F7] ou [F8].
En utilisant l'émulateur de terminal, il est possible de lancer depuis celui-ci des applications graphiques, ce qui n'est pas possible dans un terminal complètement textuel comme un TTY.

```
<wizardman> t'ouvres ton shell et tu run apache
<nadriX> je compren pa je sui quebecois
<wizardman> hmm.
<wizardman> tu ouvre le coquillage et tu lance l'indien
```

1 L'invite de commande, le prompt

Dans un terminal, on écrit des commandes après l'invite de commande :

```
toto@pc:/etc $ top -Is 2
```

Dans l'ordre, nous avons le nom de l'utilisateur connecté, suivi d'un @, puis du nom de la machine. Après des doubles points il y a le répertoire courant, c'est à dire le répertoire dans lequel s'appliqueront les commandes que nous lançons. Le symbole $ qui suit nous indique que l'utilisateur a des droits restreints. Un # à la place de $ indiquerait que l'utilisateur a des droits d'administration. Après ce dernier symbole, nous pouvons écrire la commande que nous souhaitons lancer, suivi de la touche [Entrée].
La commande elle même se compose du nom du programme appelé (top), puis d'une suite d'options et d'arguments. Les options sont précédées d'un caractère - pour la version courte, ou deux caractères -- pour la version longue. Les options courtes peuvent être combinées, ainsi les deux commandes ci-dessous sont équivalentes :

```
ls -l -a
ls -la
```

Pour stopper une commande lancée : [Ctrl]+[C]

2 L'autocomplétion

L'autocomplétion est une fonctionnalité du terminal. Elle permet de compléter automatiquement les lignes de commandes. Cette fonctionnalité s'active après avoir commencé à taper une ligne de commande, puis en appuyant sur la touche [Tab]. Si plusieurs possibilités existent, la fonction va compléter la portion de la commande qui est sûre. En appuyant deux fois sur [Tab], on voit les choix de complétion disponibles.
Les éléments complétés peuvent être : des noms de programmes, des chemins, noms de fichiers, noms de paquets, certains attributs...

```
apt-get install apac[Tab]he2
cd Bur[Tab]eau
apt-get install libapache2-[Tab][Tab]
```

```
dpkg-re[Tab]configure
```

3 Historique

Les flèches [Haut] et [Bas] permettent de naviguer dans l'historique des commandes lancées. Ces commandes peuvent être éditées avant d'être à nouveau exécutées.
Le caractère "!" en début de ligne permet également l'utilisation de l'historique : le terminal recherchera dans son historique la dernière occurrence commençant par ce qu'il y a derrière "!" pour l'exécuter à nouveau (Attention : il ne demandera pas confirmation).
L'historique est enregistré à la fermeture du terminal dans le fichier ~/.bash_history

```
tail /var/log/syslog
service bind9 restart
!tai
```

4 Lancer plusieurs commandes d'affilées

Il est possible, sur une seule ligne, de mettre plusieurs commandes qui se lanceront à la suite : la première commande doit être finie avant que la suivante s'exécute. Cela se fait en séparant les commandes avec "&&". Il est alors intéressant dans certains cas de temporiser l'execution avec la commande sleep suivie d'un nombre de secondes.

```
apt-get update && apt-get upgrade && apt-get install screen
sleep 300 && shutdown -h now
```

En ajoutant un seul symbole "&" entre deux commande, les commandes sont executées simultanément, sans que la deuxième ait besoin d'attendre la fin de la première. Cela est très utile dans les scripts, pour lancer plusieurs commandes simultanément, sans attendre que la première commande se termine. C'est ce qu'il faut faire par exemple dans le fichier /etc/rc.local, qui liste des commandes à exécuter au démarrage de la machine.

```
vlc & pluma
```

5 Les pipes - tubes

Un pipe permet d'envoyer la sortie standard d'une commande (ce qui est affiché à l'écran) vers l'entrée standard d'une autre commande. Il est représenté par le symbole | que l'on place entre les deux commandes. Dans l'exemple ci-dessous, on envoie la sortie de dmesg (messages du noyau) vers l'entrée de tail (afficher la fin).

```
dmesg | tail
```

6 Les commandes

Un certain nombre de commandes sont fournies de base par le système d'exploitation. Certaines commandes nécessitent l'installation de paquets supplémentaires.

6.1 man

La commande man sert à afficher le manuel. On l'appelle avec le nom du programme ou de la commande dont on veut afficher l'aide. C'est la manière la plus rapide pour savoir quelles options peuvent être utilisées avec la commande.

```
man top
```

On se déplace dans la page avec [pgup] et [pgdown] ou les flèches, et on quitte la manuel avec la touche [Q]. Par défaut, le manuel est en anglais. Il est possible d'arranger cela en installant les paquets manpages-fr et manpages-fr-extra.

6.2 Naviguer dans le système de fichier

6.2.1 Les chemins relatifs ou absolus

Le chemin décrit l'emplacement d'un fichier ou d'un dossier. Le chemin absolu est l'emplacement d'un fichier à partir de la racine du système (le point le plus bas, noté /). Un chemin absolu commence donc toujours par /. On aura par exemple le répertoire Bureau de toto à l'emplacement /home/toto/Bureau.
Un chemin relatif est un chemin exprimé par rapport à une référence autre que la racine. Cette référence peut être le répertoire courant, noté ".", le répertoire parent noté "..", ou le répertoire personnel de l'utilisateur noté "~". Ainsi, si toto veux créer un fichier "foo" sur son bureau (commande touch), il peut le faire de différentes façons :

```
toto@pc:~$ touch /home/toto/Bureau/foo
toto@pc:~$ touch ~/Bureau/foo
toto@pc:~$ touch ./Bureau/foo
toto@pc:~/Bureau$ touch foo
toto@pc:~/Bureau$ touch ./foo
toto@pc:~/Bureau/dossier$ touch ../foo
toto@pc:~/Images$ touch ../Bureau/foo
```

6.2.2 ls - list

Liste les fichiers du répertoire courant, ou du répertoire mentionné en argument. Les options utiles sont -l pour avoir plus de détails (droits et propriétaires) et -a pour afficher les fichiers cachés.

```
ls -la /
```

6.2.3 tree - arbre

tree affiche le contenu d'un répertoire sous forme d'arbre.

```
tree ~/Bureau
```

6.2.4 cd - change directory

Change le répertoire courant. Sans argument, c'est le répertoire personnel qui devient courant.

```
toto@pc:~$ cd Bureau
toto@pc:~/Bureau$ cd
toto@pc:~$
```

6.3 Lire un fichier

6.3.1 cat - concaténer

Affiche un ou plusieurs fichiers dans le terminal (sortie standard).

```
cat foot.txt bar.txt
```

6.3.2 less, more, most

Programmes interactifs affichant un fichier et permettant le défilement si celui-ci est trop long. On quitte le programme avec [Q]. Peut s'utiliser avec un nom de fichier en argument, ou bien avec l'entrée standard (pour afficher le résultat très long d'une commande).

```
less /etc/fstab
```

ifconfig -a | less

6.3.3 tail - queue

Affiche les dix dernières lignes d'un fichier. Pratique pour afficher la fin d'un fichier journal (log). Il peut s'utiliser avec un nom de fichier en argument, ou bien via l'entrée standard. On indique le

nombre de lignes désirées avec l'option -n.

```
tail /var/log/syslog
cat /var/log/syslog | tail -n 15
```

6.3.4 grep

Filtre le contenu d'un texte : il n'affiche que les lignes contenant le texte recherché. S'utilise généralement par son entrée standard (avec un pipe), mais on peut également lui fournir un nom de fichier, en plus du mot recherché..

```
lspci | grep VGA
grep dhcp /var/log/syslog
```

6.4 Écrire dans un fichier

Le symbole chevron > envoie la sortie standard d'une commande vers un fichier. Par exemple, pour enregistrer le résultat de la commande ifconfig:

```
ifconfig > config_pc.txt
```

Si le fichier n'existe pas, il est créé. Si il existe, il est écrasé (son contenu est remplacé).
Avec deux chevrons, on n'écrase plus le fichier, mais on ajoute à la fin de celui-ci :

```
lspci >> config_pc.txt
```

La commande echo écrit directement sur la sortie standard. On peut ainsi créer un fichier texte avec le contenu qu'on veut, en combinant echo et les chevrons.

```
echo "ne pas oublier le lait" >> liste_de_courses.txt
```

6.4.1 nano

nano est un éditeur de texte interactif. Une fois lancé, les fonctions de base, accessibles avec la touche [Ctrl], sont indiquées en bas de page.

```
nano mon_fichier.txt
```

6.5 Gérer les fichiers et dossiers

6.5.1 cp - copy (copier)

cp sert à copier un fichier ou un dossier. Pour copier un dossier, il faut l'utiliser avec l'option -R.

```
cp /chemin/fichier /autre_chemin/
cp /chemin/fichier /chemin/copie_de_mon_fichier
cp -R /chemin/dossier /autre_chemin/
```

6.5.2 mv - move (mouvoir)

mv est utilisé pour déplacer un fichier ou dossier, mais également pour renommer ceux-ci.

```
mv ancien_nom nouveau_nom
mv /chemin/dossier /autre_chemin/
```

6.5.3 mkdir - make directory

Pour créer un nouveau dossier

```
mkdir mon_nouveau_dossier
```

6.5.4 rmdir - remove directory

Pour supprimer un dossier vide. Il faut utiliser rm pour supprimer un dossier qui n'est pas vide.

```
rmdir mon_dossier
```

6.5.5 rm - remove

Pour supprimer un fichier ou un dossier. **Attention** : une mauvaise manipulation de cette commande peut détruire votre système d'exploitation et vos documents ! Il faut utiliser l'option -R pour

supprimer un dossier.

```
rm /chemin/fichier
rm -R /chemin/dossier
```

Pour effacer uniquement le contenu du dossier :

```
rm -R /chemin/dossier/*
```

6.5.6 touch - toucher

Utilisé pour créer rapidement un fichier vide.

```
touch nouveau_fichier
```

6.5.7 ln - link

Créer des liens entre différents fichiers. Par défaut, cela crée un lien matériel, c'est à dire qu'il rend un même fichier accessible à partir de différents chemins. Il faut supprimer tous les chemins menant au fichier pour qu'il soit vraiment effacé de la mémoire. À la différence d'une copie, faire plusieurs liens matériels d'un fichier ne demandera pas davantage de mémoire sur le disque, car les données ne figurent qu'une seule fois sur le disque.

```
ln /chemin/nom_de_fichier /autre_chemin/autre_nom_de_fichier
```

La commande ln est davantage utilisée pour faire des liens symboliques. Dans ce cas, le lien pointe vers le chemin que nous spécifions, et non directement vers les données elles-mêmes. Ainsi, si nous supprimons le fichier d'origine, le lien symbolique ne fonctionnera plus. Si on supprime le lien symbolique, le fichier ne sera pas altéré. Un lien symbolique est l'équivalent d'un raccourci sous Windows.

Le lien symbolique peut être relatif ou absolu. Dans le cas d'un lien absolu, si on déplace le lien, il fonctionnera encore. Le lien relatif, lui, risque de ne plus fonctionner.

```
ln -s /chemin/mon_fichier /autre_chemin/mon_lien  #lien absolu
ln -s ./dossier/fichier .  #lien relatif
```

6.6 Les droits, groupes, utilisateurs

6.6.1 chmod - change modus

Permet de changer les droits sur un fichier, pour le propriétaire du fichier (u - user), le groupe auquel appartient le fichier (g - group), ou les autres (o - others ou ôtres ;)). Les droits sont la lecture (r - read), l'écriture (w - write) et l'exécution (x - eXecute). On utilise le signe - pour retirer des droits, et le signe + pour en ajouter. L'option -R est utilisée pour appliquer les droits de façon récursive à tout un dossier (pour le dossier ET les fichiers qui sont à l'intérieur).
Donner les droits de lecture pour le propriétaire et le groupe au fichier :

```
chmod ug+r /chemin/fichier
```

Retirer les droits d'écriture pour le groupe et les autres au dossier récursivement :

```
chmod -R go-w /chemin/dossier
```

Attention : si on retire les droits d'execution à un répertoire, son contenu n'est plus accessible.

6.6.2 chown - change owner

La commande chown modifie le propriétaire et le groupe auquel appartient un fichier ou un dossier. L'option -R est à utiliser pour que ce soit récursif. Pour modifier le propriétaire, on stipule celui-ci en premier argument, suivi du nom de fichier auquel on applique la modification.

```
chown toto /chemin/mon_fichier
```

Pour modifier le propriétaire et le groupe, on sépare ceux-ci par un double-point :

```
chown toto:groupe /chemin/mon_fichier
```

Si on ne veux modifier que le groupe, on mentionnera celui-ci après des doubles points :

```
chown :groupe /chemin/mon_fichier
```

6.6.3 adduser

Ajouter un nouvel utilisateur au système, ou ajouter un utilisateur existant dans un groupe existant.

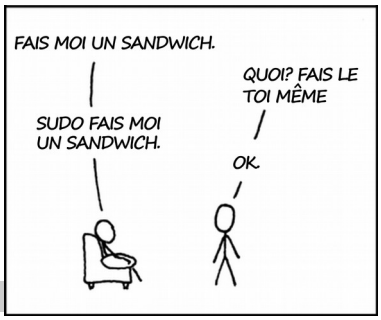
```
adduser jessie
adduser jessie partage
```

6.6.4 addgroup

Ajouter un nouveau groupe au système.

```
addgroup partage
```

6.6.5 sudo - super user do



On met **sudo** devant une commande pour l'exécuter avec les droits administrateur (avec des privilèges plus élevés), sans être connecté en tant qu'administrateur. On s'en sert pour exécuter des tâches d'administration système. Par défaut, seul l'utilisateur principal a le droit d'utiliser cette commande. Quand le mot de passe est demandé, il faut taper celui de l'utilisateur, à l'aveugle.

```
sudo nano /etc/rc.local
```

La commande **sudo** n'est pas installée et paramétrée sur toutes les distributions. Par exemple, **sudo** est utilisé sous Ubuntu et dérivés, mais généralement pas sous Debian.

6.6.6 su - substitute user

On utilise su pour se connecter avec un autre compte utilisateur. Si on ne précise pas de compte, c'est le compte administrateur root qui est choisi. Il faut entrer le mot de passe du compte choisi.
Sous Ubuntu et Linux Mint, par mesure de sécurité, l'administrateur n'a pas de mot de passe. Il faut alors utiliser **su** avec **sudo** et le mot de passe du comte principal.

```
toto@pc:~/$ sudo su
root@pc:~/home/toto# su jessie
jessie@pc:/home/toto$ exit
root@pc:/home/toto# exit
toto@pc:~/$
```

On revient au compte utilisé précédemment avec la commande **`exit`**.

6.7 Gestionnaire de paquets

Sous Linux, les logiciels sont disponibles sous forme de paquets. Un paquet est une archive compréssée contenant des données (exécutables, configurations, images...), avec des informations pour l'installation du paquet dans le système, et une liste de dépendances (un autre paquet doit-il être installé pour que celui-ci fonctionne ?). Pour installer un paquet, il faut passer par un logiciel appelé gestionnaire de paquets.
Il existe différents types de paquets : sous Debian et dérivées on utilise les paquets Deb, avec ces programmes :

6.7.1 apt-get

`apt-get` est un gestionnaire de paquets qui gère les dépendances, c'est à dire qu'il s'occupe lui-même de télécharger les paquets manquants lors de l'installation d'un logiciel. On utilise cet outil pour installer des logiciels et pour maintenir le système à jour.
Pour qu'**`apt-get`** sache quels paquets sont disponibles sur les serveurs :

```
apt-get update
```

On peut ensuite demander à mettre à jour l'ensemble du système :

```
apt-get upgrade
```

Pour installer un logiciel ou un paquet, on utilise l'instruction install suivie du nom du paquet

```
apt-get install screen
```

Pour désinstaller un logiciel, on utilise l'instruction remove

```
apt-get remove screen
```

Avec **autoremove**, on désinstalle les paquets qui ne sont plus nécessaires (anciennes dépendances)

```
apt-get autoremove
```

Les paquets deb téléchargés sont conservés en cache sur le disque dur. Il est possible de vider ce cache pour récupérer de la mémoire, avec cette commande :

```
apt-get clean
```

6.7.2 apt-cache

apt-cache permet d'effectuer différentes actions sur la liste des paquets disponibles (le cache). On peut par exemple rechercher un paquet à partir d'un mot clé :

```
apt-cache search apache
```

Pour les autres possibilités, voir la page de manuel.

6.7.3 dpkg

dpkg est un gestionnaire de paquets qui ne gère pas les dépendances : il faut installer celles-ci soi même. On peut utiliser **dpkg** pour installer un paquet que l'on a préalablement téléchargé :

```
dpkg -i /chemin/nom_du_paquet.deb
```

On peut aussi utiliser **dpkg** pour supprimer un paquet installé :

```
dpkg -r nom_du_paquet
```

Voir le manuel pour plus d'options.

6.7.4 dpkg-reconfigure

Lors de l'installation d'un paquet, on peut vous poser plusieurs questions pour la configuration de celui-ci. **dpkg-reconfigure** permet de reconfigurer un paquet déjà installé, en ré-affichant ces questions.

```
dpkg-reconfigure isc-dhcp-server
```

6.8 Informations et outils système

6.8.1 ifconfig

Afficher la configuration réseau. On peut lui indiquer d'afficher également les interfaces non utilisées avec l'option -a. Pour afficher juste une interface, on donne son nom en argument.

```
ifconfig -a | less
ifconfig eth0
```

6.8.2 iwconfig

Pareil que **ifconfig**, mais réservé aux interfaces sans fil.

6.8.3 lspci - list PCI

Pour lister les composants de l'ordinateur connectés sur un port PCI

6.8.4 lsusb - list USB

Pour lister les composants de l'ordinateur connectés sur un port USB (certains composants internes, comme la webcam, sont également connectés en USB)

6.8.5 lshw - list HardWare

Pour lister tout le matériel de l'ordinateur. Avec l'option -X, le programme lance une interface graphique (si possible). L'option -html envoie le résultat sous format html, pratique pour enregistrer le résultat de la commande et ouvrir le fichier obtenu avec le navigateur web.
Pour que la commande accède à la totalité du matériel, il faut la lancer avec en tant qu"administrateur.

```
lshw | less
lshw -html > materiel.htm
```

6.8.6 dmesg - daemon messages

dmesg affiche les messages en provenance du noyau : erreurs, alertes ou simples informations. On l'utilise habituellement conjointement à **tail**, pour n'afficher que la fin.

```
dmesg | tail
```

6.8.7 df - disk free

Affiche le taux d'utilisation des différents systèmes de fichiers montés. Avec l'option -h (human) on obtient un résultat mieux adapté aux humains (Utilisation des multiples kilo, méga...).

6.8.8 du - disk usage

Affiche la quantité de mémoire prise par un fichier ou un dossier sur le disque dur. Pour améliorer la lisibilité, on utilise l'otion -h.

```
du -h /home/toto
```

6.8.9 free - libre

Affiche l'utilisation globale de la mémoire RAM. L'option -h améliore la lisibilité. Pour rafraîchir automatiquement l'affichage, on peut utiliser l'option -s avec un délai en secondes.

```
free -hs 2
```

6.8.10 kill, killall

kill est utilisé pour tuer un processus dont on connait le PID (cf top et ps).

```
kill 1234
```

Si le processus ne se termine pas, on peut forcer l'arrêt avec l'option -9

```
kill -9 1234
```

La commande killall a la même fonction, mais on peut lui fournir le nom du programme que l'on souhaite tuer. Il tuera ainsi touts les instances ouvertes dudit programme.

```
killall vlc
killall -9 firefox
```

6.8.11 service

Les services, ou démons, sont des programmes lancés au démarrage de l'ordinateur et tournant en tâche de fond. Ces programmes sont contrôlables avec la commande **service**, qui accepte les instructions status, start (démarrer), restart (redémarrer), et stop.

```
service apache2 restart
```

6.8.12 top, ps

top affiche une liste tronquée des processus en cours sur la machine, actualisé chaque seconde, et classés suivant leur utilisation du processeur. Cela peut être pratique pour voir quel programme bloque l'ordinateur, ou surveiller les programmes lancés. L'option -s modifie le délai de rafraichissement.

```
top -s 0.5
```

ps liste les processus lancés sans actualisation. L'option -e permet de voir également les processus des autre utilisateurs.

```
ps -e | less
```

6.9 Monter un système de fichier

La commande **mount** permet de monter des systèmes de fichiers. C'est à dire qu'elle rend accessible le contenu d'un support de stockage depuis l'arborescence du système.

```
ls /mnt
```

```
mount /dev/sdb1 /mnt
ls /mnt
```

La commande **mount** ci-dessus monte le périphérique /dev/sdb1 dans le répertoire /mnt.
Les périphériques de stockage sont représentés sous forme de fichiers dans le répertoire /dev. Leur nom commence par sd, suivi d'une lettre (numéro du support). En ajoutant encore un chiffre, on désigne le numéro de la partition sur le support. Si on n'a qu'un seul disque dur dans notre machine, celui-ci sera sda. Si on branche une clé USB, celle-ci sera notée sdb, et ainsi de suite pour les prochains périphériques.
Pour monter le bon périphérique, on peut trouver des information utiles en listant le contenu de certains répertoires spéciaux :

```
ls -l /dev/disk/by-id
ls -l /dev/disk/by-label/
ls -l /dev/disk/by-path/
ls -l /dev/disk/by-uuid/
```

Pour démonter un périphérique, on utilise la commande **umount** suivie du point de montage (/mnt) ou du nom du périphérique (/dev/sdb1).

```
umount /mnt
```

Pour lister tous les périphériques montés :

```
mount -l
```

6.10 Gérer les archives

6.10.1 zip, unzip

La commande **zip** sert à inclure des fichiers dans une archive zip. On lui fournit le nom du fichier zip, puis du nom des fichiers à inclure.

```
zip monfichierzip.zip *.txt
```

À l'inverse, **unzip** décompresse une archive zip dans le répertoire courant.

```
unzip mes_photos_de_vacances.zip
```

6.10.2 gzip, gunzip

gzip sert à compresser un unique fichier. C'est un format très utilisé sous Linux, car il est libre et plus performant que zip.

```
gzip archive.gz monfichier.txt
```

gunzip extrait le contenu de l'archive dans le répertoire courant.

```
gunzip archive.gz
```

6.10.3 bzip2 bunzip2

Le format bzip2 offre une compression supérieure à **gzip**, mais est un peu plus lent. Comme **gzip**, ce format ne permet de compresser qu'un fichier à la fois.

```
bzip2 monfichier.txt.bz2 monfichier.txt
```

6.10.4 tar

Le format tar permet de regrouper plusieurs fichiers en un seul. On peut ainsi, par la suite, les compresser avec **gzip** ou **bz2** : c'est pourquoi on rencontre souvent des fichier avec l'extension .tar.gz, .tgz, ou .tar.bz2.
Les options dans les exemples ci-dessous sont -c pour compacter, -x pour extraire, v pour afficher plus d'infos (verbeux), j pour la compression gzip et z pour la compression bzip2.

Pour inclure plusieurs fichiers dans une archive tar :

```
tar -cvf mon_archive.tar fichier1.txt fichier2.png
```

Ou pour inclure le contenu d'un dossier :

```
tar -cvf mon_archive.tar dossier/
```

On extrait l'archive comme ceci :

```
tar -xvf archive.tar -C dossier
```

Pour décompresser un fichier tgz ou tar.gz :

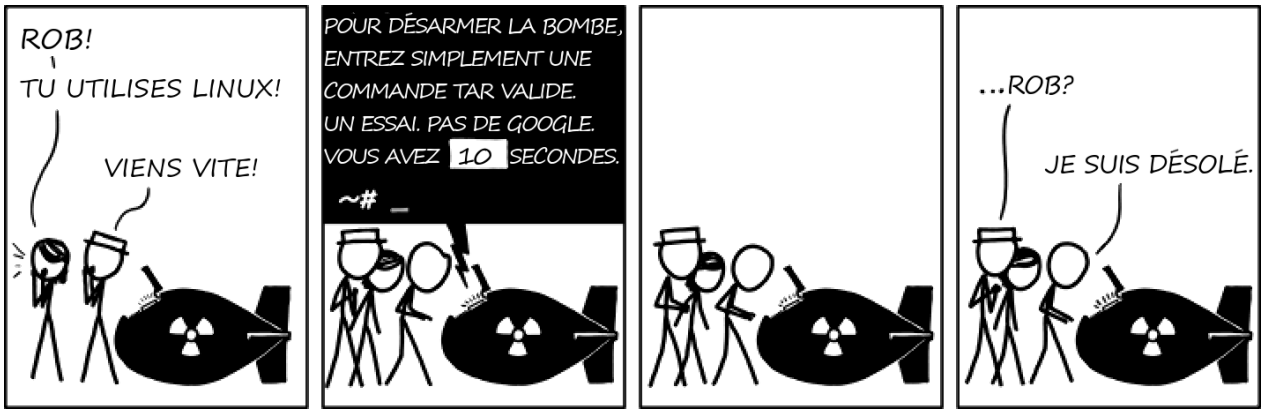
```
tar -xzvf fichier.tar.gz
```

Pour décompresser un fichier tar.bz2 :

```
tar -xjvf fichier.tar.bz2
```

Pour compresser des fichiers en tar.gz ou tar.bz2 :

```
tar -zcvf votre_archive.tar.gz votre_dossier/
tar -jcvf votre_archive.tar.bz2 votre_dossier/
```



6.11 Utilitaires divers

6.11.1 ssh - secure shell

ssh est un logiciel permettant d'accéder à distance à une machine. On l'utilise en spécifiant le nom d'utilisateur avec lequel on souhaite se connecter, et l'adresse de la machine de destination (nom d'hôte ou adresse ip).

```
ssh toto@pc.local
```

Avec l'option -p, on peut choisir un port différent pour la connexion (suivant la configuration du serveur). L'option -X permet de déporter l'affichage graphique, pour lancer des logiciels graphiques à distance. L'option -C active la compression des données, pour accélérer leur transmission à travers un réseau lent.

```
toto@pc:~$ ssh -XC jessie@1.2.3.4 -p 2222
jessie@ordi:~$ vlc
```

6.11.2 screen

screen est un multiplexeur de terminal, il permet de lancer plusieurs sessions dans un terminal. Ce programme a plusieurs intérêts, il permet :

- de récupérer une session même après avoir fermé le terminal, ou en cas de coupure réseau (ssh).
- de se connecter à plusieurs dans une même session screen
- de lancer plusieurs sessions screen

Pour lancer une session, il suffit de taper **screen**. L'option -S permet de donner un nom à la session, pour la retrouver plus facilement si on lance plusieurs sessions.

```
screen -S mise_a_jour
apt-get update && apt-get upgrade
[Ctrl]+[A] [D]
```

En appuyant sur les touches [Ctrl]+[A] [D], on se détache de la session screen. La session est toujours en route, et les programmes lancés à l'intérieur continuent à tourner. Pour se rattacher à la session, on utilise l'option -x, suivie éventuellement du nom de la session si plusieurs sont déjà lancées.

```
screen -x
screen -x mise_a_jour
```

Pour terminer la session, on peut utiliser le raccourci [Ctrl]+[A] [Q], ou bien entrer **`exit`** dans la session.

6.11.3 wget

On utilise **`wget`** pour télécharger un fichier depuis le web.

```
wget http://www.gnu.org/graphics/avatar-3d-baby-gnu.png
```

Table des matières