

# Les commandes Linux les plus utilisées,

Linux a une part de marché mondiale de 2,68 % sur les ordinateurs de bureau, mais plus de 90 % de tous les services d'infrastructure et d'hébergement dans le cloud fonctionnent avec ce système d'exploitation.

Linux est le système d'exploitation le plus utilisé par les développeurs professionnels, avec une part de marché impressionnante de 55,9 % (source stackOverflow 2020). Ce n'est pas une simple coïncidence. Linux est gratuit et open-source, offre une meilleure sécurité que ses concurrents et dispose d'une ligne de commande puissante qui rend les développeurs et les utilisateurs plus efficaces.

## Qu'est-ce qu'une commande Linux ?

Une commande Linux est un programme ou un utilitaire qui s'exécute en ligne de commande. Une ligne de commande est une interface qui accepte des lignes de texte et les traite en instructions pour votre ordinateur.

Toute interface utilisateur graphique (GUI) n'est qu'une abstraction des programmes en ligne de commande. Par exemple, lorsque vous fermez une fenêtre en cliquant sur le « X », une commande est exécutée derrière cette action.

Un **flag** est un moyen de passer des options à la commande que vous exécutez. La plupart des commandes Linux ont une page d'aide que l'on peut appeler avec le flag -h. La plupart du temps, les flags sont optionnels.

Un **argument** ou paramètre est l'entrée que nous donnons à une commande pour qu'elle puisse s'exécuter correctement. Dans la plupart des cas, l'argument est un chemin d'accès à un fichier, mais il peut s'agir de tout ce que vous saisissez dans le terminal.

Vous pouvez invoquer des flags en utilisant des tirets (-) et des doubles tirets (--), tandis que l'exécution des arguments dépend de l'ordre dans lequel vous les passez à la fonction.

→ Lancement d'un terminal : **Ctrl + Alt + T**

### 1. Commande **ls**

**ls** est probablement la première commande que tout utilisateur de Linux saisit dans son terminal. Elle vous permet de lister le contenu du répertoire que vous souhaitez (le répertoire courant par défaut), y compris les fichiers et autres répertoires imbriqués.

Elle possède de nombreuses options, il peut donc être utile d'obtenir de l'aide en utilisant l'option --help. Cette option renvoie toutes les options que vous pouvez utiliser avec ls.

Par exemple, pour coloriser la sortie de la commande ls, vous pouvez utiliser ce qui suit :

```
ls --color=auto
```

```

ludo@ludo-UM700:~$ ls
Bureau      Images      Musique     snap        Vidéos
Documents   Modèles     Public      Téléchargements

ludo@ludo-UM700:~$

ludo@ludo-UM700:~/Bureau$ ls --color=no
Dev todo Typos
ludo@ludo-UM700:~/Bureau$ ls --color=auto
Dev todo Typos
ludo@ludo-UM700:~/Bureau$

```

## 2. Commande **alias**

La commande `alias` vous permet de définir des alias temporaires dans votre session shell. En créant un alias, vous demandez à votre shell de remplacer un mot par une série de commandes.

Par exemple, pour que `ls` ait une couleur sans avoir à taper le flag `--color` à chaque fois, vous pouvez utiliser :

```
alias ls="ls --color=auto"
```

Comme vous pouvez le voir, la commande `alias` prend un paramètre de paire clé-valeur : `alias NAME="VALUE"`. Notez que la valeur doit être entre guillemets.

Si vous voulez lister tous les alias que vous avez dans votre session shell, vous pouvez exécuter la commande `alias` sans argument.

```

ludo@ludo-UM700:~/Bureau$ alias
alias alert='notify-send --urgency=low -i "$([ $? = 0 ] &
& echo terminal || echo error)" "$(history|tail -n1|sed -
e '\''s/^\s*[0-9]\+\s*//;s/[:;&]\s*alert$//'\''")'
alias egrep='egrep --color=auto'
alias fgrep='fgrep --color=auto'
alias grep='grep --color=auto'
alias l='ls -CF'
alias la='ls -A'
alias ll='ls -alF'
alias ls='ls --color=auto'
ludo@ludo-UM700:~/Bureau$

```

## 3. Commande **unalias**

Comme son nom l'indique, la commande `unalias` vise à supprimer un alias parmi les alias déjà définis. Pour supprimer l'alias précédent `ls`, vous pouvez utiliser :

```
unalias ls
```

#### 4. Commande **pwd**

La commande `pwd` signifie « print working directory » (afficher le répertoire de travail) et donne le chemin absolu du répertoire dans lequel vous vous trouvez. Par exemple, si votre nom d'utilisateur est « ludo » et que vous vous trouvez dans votre répertoire Documents, son chemin absolu sera le suivant : `/home/ludo/Documents`.

Pour l'utiliser, il suffit de saisir `pwd` dans le terminal :

```
ludo@ludo-UM700:~/Bureau$ pwd
/home/ludo/Bureau
ludo@ludo-UM700:~/Bureau$
```

#### 5. Commande **cd**

La commande `cd` est très populaire, tout comme `ls`. Elle signifie « Change Directory » et, comme son nom l'indique, vous fait passer au répertoire auquel vous essayez d'accéder.

Par exemple, si vous êtes dans votre répertoire utilisateur et que vous essayez d'accéder à l'un de ses sous-répertoires appelé Videos, vous pouvez y accéder en saisissant :

##### **cd Videos**

Vous pouvez également fournir le chemin absolu du répertoire :

##### **cd /home/ludo/Videos**

```
ludo@ludo-UM700:~/Bureau$ cd /home/ludo/Vidéos/
ludo@ludo-UM700:~/Vidéos$ cd ~
ludo@ludo-UM700:~$ cd Vidéos/
ludo@ludo-UM700:~/Vidéos$
```

Il existe quelques astuces avec la commande `cd` qui peuvent vous faire gagner beaucoup de temps lorsque vous jouez avec elle :

Aller dans le répertoire home : **cd**

Monter d'un niveau : **cd ..**

Retourner au répertoire précédent : **cd -**

```
ludo@ludo-UM700:~$ cd Vidéos
ludo@ludo-UM700:~/Vidéos$ cd
ludo@ludo-UM700:~$ cd -
/home/ludo/Vidéos
ludo@ludo-UM700:~/Vidéos$
ludo@ludo-UM700:~/Vidéos$ cd ..
ludo@ludo-UM700:~$
```

## 6. Commande **cp**

Il est facile de copier des fichiers et des répertoires directement dans le terminal Linux, il peut parfois remplacer les gestionnaires de fichiers conventionnels.

Pour utiliser la commande `cp`, il suffit de la saisir avec les fichiers source et destination :

**`cp file_to_copy.txt new_file.txt`**

Vous pouvez également copier des répertoires entiers en utilisant le flag récursif :

**`cp -r dir_to_copy/ new_copy_dir/`**

Rappelez-vous que sous Linux, les répertoires se terminent par une barre oblique ( / ).

## 7. Commande **rm**

Vous pouvez utiliser la commande `rm` pour supprimer des fichiers et des répertoires. Faites cependant attention lorsque vous l'utilisez, car il est très difficile (mais pas impossible) de récupérer des fichiers supprimés de cette façon.

Pour supprimer un fichier ordinaire, il faut saisir :

**`rm file_to_copy.txt`**

Si vous voulez supprimer un répertoire vide, vous pouvez utiliser le flag récursif ( `-r` ) :

**`rm -r dir_to_remove/`**

```
ludo@ludo-UM700:~$ ls
Bureau  Images  Musique  snap      toto
Documents  Modèles  Public  Téléchargements  Vidéos
ludo@ludo-UM700:~$ rm toto/
rm: impossible de supprimer 'toto/': est un dossier
ludo@ludo-UM700:~$ rm -r toto/
ludo@ludo-UM700:~$ ls
Bureau  Images  Musique  snap      Vidéos
Documents  Modèles  Public  Téléchargements
ludo@ludo-UM700:~$
```

```
ludo@ludo-UM700:~$ ls
Bureau  Images  Musique  snap      toto
Documents  Modèles  Public  Téléchargements  Vidéos
ludo@ludo-UM700:~$ ls toto/
ABox.svg      Browning.svg  StorageShelf.svg
BayonetBox.svg  Colt.svg      TypeTray.svg
ludo@ludo-UM700:~$ rm -r toto/
ludo@ludo-UM700:~$ ls
Bureau  Images  Musique  snap      Vidéos
Documents  Modèles  Public  Téléchargements
ludo@ludo-UM700:~$
```

**Attention** : Il n'y a pas de demande de confirmation...

## 8. Commande **mv**

Vous utilisez la commande **mv** pour déplacer (ou renommer) des fichiers et des répertoires dans votre système de fichiers.

Pour utiliser cette commande, il faut saisir son nom avec les fichiers source et destination :

```
mv source_file destination_folder/
```

```
mv list.txt commands/
```

Pour utiliser des chemins absolus, il faut utiliser :

```
mv /home/kinsta/BestMoviesOfAllTime ./
```

...où **./** est le répertoire dans lequel vous vous trouvez actuellement.

Vous pouvez également utiliser **mv** pour renommer des fichiers tout en les conservant dans le même répertoire :

```
mv old_file.txt new_file.txt
```

## 9. Commande **mkdir**

Pour créer des répertoires dans le shell, vous utilisez la commande **mkdir**. Il suffit de spécifier le nom du nouveau répertoire, de s'assurer qu'il n'existe pas, et le tour est joué.

Par exemple, pour créer un répertoire dans lequel vous conserverez toutes vos images, saisissez simplement

```
mkdir images/
```

Pour créer des sous-répertoires avec une simple commande, utilisez l'option parent ( **-p** ) :

```
mkdir -p music/1980/
```

## 10. Commande **man**

Une autre commande Linux essentielle est **man**. Elle affiche la page de manuel de n'importe quelle autre commande (si elle existe...).

Pour voir la page de manuel de la commande **mkdir**, saisissez :

```
man mkdir
```

Vous pouvez même vous référer à la page de manuel **man** :

```
man man
```

```
MAN(1)utilitaires de l'afficheur des pages de manueMAN(1)
NOM
    man - an interface to the system reference ma-
        nuals
SYNOPSIS
    anual page man(1) line 1 (press h for help or q to quit)
```

## 11. Commande **touch**

La commande touch vous permet de mettre à jour les temps d'accès et de modification des fichiers spécifiés.

Par exemple, j'ai un vieux fichier qui a été modifié pour la dernière fois le 12 juillet :

```
ludo@ludo-UM700:~/toto$ ls -lah
total 956K
drwxrwxr-x  2 ludo ludo 4,0K août  25 11:16 .
drwxrwxr-x  3 ludo ludo 4,0K août  25 11:14 ..
-rw-rw-r--  1 ludo ludo 948K juil.  12 18:37 Browning.svg
ludo@ludo-UM700:~/toto$
```

Pour changer sa date de modification à l'heure actuelle, nous devons utiliser l'option -m :

**touch -m Browning.svg**

La date correspond maintenant à la date actuelle (ici la date de la capture d'écran)

```
ludo@ludo-UM700:~/toto$ touch -m Browning.svg
ludo@ludo-UM700:~/toto$ ls -lah
total 956K
drwxrwxr-x  2 ludo ludo 4,0K août  25 11:16 .
drwxrwxr-x  3 ludo ludo 4,0K août  25 11:14 ..
-rw-rw-r--  1 ludo ludo 948K août  25 11:21 Browning.svg
ludo@ludo-UM700:~/toto$
```

La plupart du temps, touch n'est pas utilisé pour modifier les dates des fichiers, mais plutôt pour créer de nouveaux fichiers vides :

**touch new\_file\_name**

```

ludo@ludo-UM700:~/toto$ touch new_file.txt
ludo@ludo-UM700:~/toto$ ls
Browning.svg  new_file.txt
ludo@ludo-UM700:~/toto$ ls -lah
total 956K
drwxrwxr-x  2 ludo ludo 4,0K août  25 11:25 .
drwxr-xr-x 23 ludo ludo 4,0K août  25 11:14 ..
-rw-rw-r--  1 ludo ludo 948K août  25 11:21 Browning.svg
-rw-rw-r--  1 ludo ludo   0 août  25 11:25 new_file.txt
ludo@ludo-UM700:~/toto$

```

## 12. Commande **chmod**

La commande `chmod` vous permet de modifier rapidement le mode d'un fichier (permissions). Elle dispose d'un grand nombre d'options.

Les autorisations de base qu'un fichier peut avoir sont les suivantes :

- `r` (lire)
- `w` (écrire)
- `x` (exécuter)

L'un des cas d'utilisation les plus courants de `chmod` est de rendre un fichier exécutable par l'utilisateur. Pour ce faire, saisissez `chmod` et le flag `+x`, suivi du fichier dont vous voulez modifier les permissions :

### **chmod +x mon\_script**

Vous l'utilisez pour rendre les scripts exécutables, ce qui vous permet de les exécuter directement en utilisant la notation `./mon_script`.

## 13. Command `./`

La notation `./` n'est peut-être pas une commande en soi, mais elle mérite d'être mentionnée dans cette liste. Elle permet à votre shell d'exécuter un fichier exécutable avec n'importe quel interpréteur installé sur votre système directement depuis le terminal. Plus besoin de double-cliquer sur un fichier dans un gestionnaire graphique de fichiers !

Par exemple, avec cette commande, vous pouvez exécuter un script Python ou un programme disponible uniquement au format `.run`, comme XAMPP. Lorsque vous exécutez un exécutable, assurez-vous qu'il dispose des droits d'exécution (`x`), que vous pouvez modifier avec la commande `chmod`.

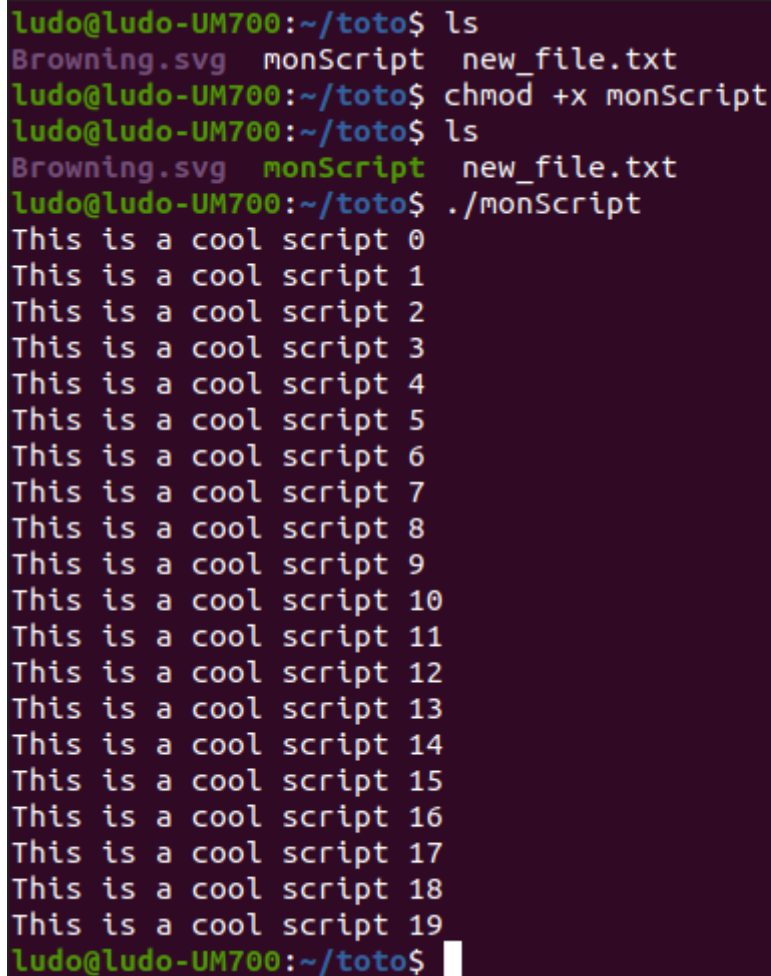
Voici un script Python simple et la façon dont nous l'exécuterions avec la notation `./` :

```
#!/usr/bin/python3
# filename: monScript
for i in range(20):
    print(f"This is a cool script {i}")
```

Voici comment convertir le script en un exécutable et l'exécuter :

**chmod +x monScript**

**./monScript**



```
ludo@ludo-UM700:~/toto$ ls
Browning.svg  monScript  new_file.txt
ludo@ludo-UM700:~/toto$ chmod +x monScript
ludo@ludo-UM700:~/toto$ ls
Browning.svg  monScript  new_file.txt
ludo@ludo-UM700:~/toto$ ./monScript
This is a cool script 0
This is a cool script 1
This is a cool script 2
This is a cool script 3
This is a cool script 4
This is a cool script 5
This is a cool script 6
This is a cool script 7
This is a cool script 8
This is a cool script 9
This is a cool script 10
This is a cool script 11
This is a cool script 12
This is a cool script 13
This is a cool script 14
This is a cool script 15
This is a cool script 16
This is a cool script 17
This is a cool script 18
This is a cool script 19
ludo@ludo-UM700:~/toto$
```

#### 14. Commande exit

La commande exit fait exactement ce que son nom suggère : Elle permet de mettre fin à une session du shell et, dans la plupart des cas, de fermer automatiquement le terminal que vous utilisez.



## 15. Commande **sudo**

Cette commande signifie « superuser do » et vous permet d’agir en tant que super-utilisateur ou utilisateur root pendant l’exécution d’une commande spécifique.

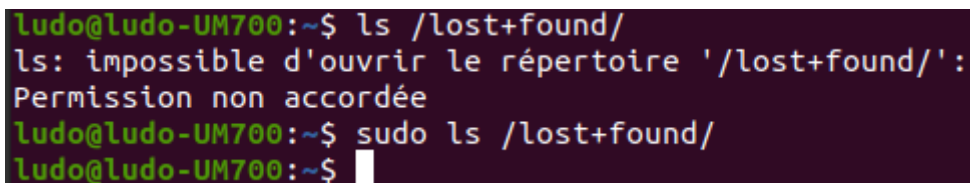
C’est ainsi que Linux se protège et empêche les utilisateurs de modifier accidentellement le système de fichiers de la machine ou d’installer des paquets inappropriés.

Sudo est couramment utilisé pour installer des logiciels ou pour modifier des fichiers en dehors du répertoire personnel de l’utilisateur :

```
sudo apt install gedit
```

```
sudo cd /root/
```

Il vous demandera le mot de passe de l’administrateur avant d’exécuter la commande que vous avez saisi.



```
ludo@ludo-UM700:~$ ls /lost+found/
ls: impossible d'ouvrir le répertoire '/lost+found/':
Permission non accordée
ludo@ludo-UM700:~$ sudo ls /lost+found/
ludo@ludo-UM700:~$
```

## 16. Commande **shutdown**

Comme vous pouvez le deviner, la commande shutdown permet d’éteindre votre machine. Cependant, elle peut également être utilisée pour l’arrêter et le redémarrer.

Pour mettre votre ordinateur hors tension immédiatement (la valeur par défaut est d’une minute), saisissez :

**shutdown now**

Vous pouvez également planifier l’extinction de votre système en format 24 heures :

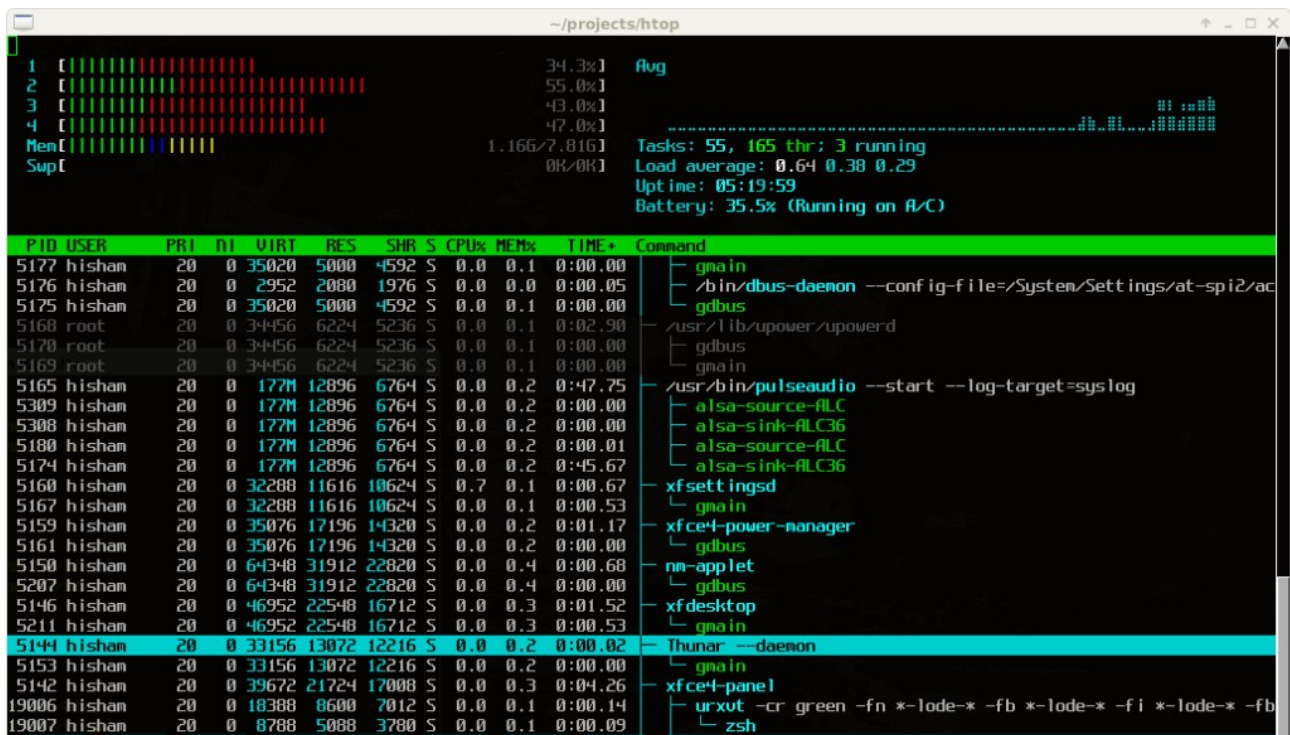
**shutdown 20:40**

Pour annuler un appel shutdown précédent, vous pouvez utiliser le flag -c :

**shutdown -c**

## 17. Commande **htop**

htop est un visualiseur de processus interactif qui vous permet de gérer les ressources de votre machine directement depuis le terminal. Dans la plupart des cas, il n’est pas installé par défaut.



## 18. Commande **unzip**

La commande `unzip` vous permet d'extraire le contenu d'un fichier `.zip` à partir du terminal. Encore une fois, ce paquet peut ne pas être installé par défaut, alors assurez-vous de l'installer avec votre gestionnaire de paquets.

Ici, nous décompressons un fichier `.zip` rempli d'images :

### **unzip images.zip**

## 19. Commandes **apt**, **yum**, **pacman**

Quelle que soit la distribution Linux que vous utilisez, il est probable que vous utilisiez des gestionnaires de paquets pour installer, mettre à jour et supprimer les logiciels que vous utilisez tous les jours.

Vous pouvez accéder à ces gestionnaires de paquets par la ligne de commande, et vous utiliserez l'un ou l'autre en fonction de la distribution utilisée par votre machine.

Les exemples suivants vont installer GIMP, un logiciel libre et gratuit généralement disponible dans la plupart des gestionnaires de paquets :

*Basé sur Debian (Ubuntu, Linux Mint)*

**sudo apt install gimp**

*Basé sur Red Hat (Fedora, CentOS)*

**sudo yum install gimp**

*Basé sur Arch (Manjaro, Arco Linux)*

**sudo pacman -S gimp**

## 20. Commande **echo**

La commande echo affiche un texte défini dans le terminal – c'est aussi simple que cela :

echo "Hello, World ;-)"

```
ludo@ludo-UM700:~$ echo "Hello, World ;-)"
Hello, World ;-)
```

Son utilisation principale est d'afficher les variables environnementales dans ces messages :

echo "Salut \$USER"

```
ludo@ludo-UM700:~$ echo "Salut $USER"
Salut ludo
```

## 21. Commande **cat**

Cat, abréviation de « concatenate », vous permet de créer, d'afficher et de concaténer des fichiers directement depuis le terminal. Il est principalement utilisé pour prévisualiser un fichier sans ouvrir un éditeur de texte :

**cat new\_file.txt**

```
ludo@ludo-UM700:~/toto$ cat new_file.txt
Hi mortuis soulless creaturas, imo monstra adventus v
ultus comedat cerebella viventium. Avium, canum, fuger
e ferae et infecti horrenda monstra. Ut fames cerebro
enim carnis, viscera et organa viventium. Qui tardius
moveri, sed in magna copia sint terribiles legionis.

Qui tardius moveri, sed in magna copia sint terribile
s legionis. Aenean a dolor vulnerum aperire accedunt,
mortui iam vivam. Putridi odores aere implent. The voo
doo sacerdos suscitatur mortuos comedere carnem. The hor
ror, monstra significant finem. Fit de nostra carne un
dead.

Panduntur portae inferi. Pestilentia est haec ambulab
at mortuos. Maleficia! Forsitan illud Apocalypsi, vel
malum poenae horrifying fecimus. The voodoo sacerdos s
uscitat mortuos comedere carnem. Avium, canum, fugere
ferae et infecti horrenda monstra. Zombies reversus ab
inferno, nam malum cerebro.
```

## 22. Commande **ps**

Avec **ps**, vous pouvez jeter un coup d'œil aux processus que votre session shell actuelle exécute. Elle affiche des informations utiles sur les programmes que vous exécutez, comme l'ID du processus, le TTY (téléscripteur), l'heure et le nom de la commande.

```
ludo@ludo-UM700:~/toto$ ps
  PID TTY          TIME CMD
 8759 pts/0        00:00:00 bash
 9931 pts/0        00:00:00 ps
ludo@ludo-UM700:~/toto$
```

## 23. Commande **kill**

C'est ennuyeux lorsqu'un programme ne répond pas, et que vous ne pouvez pas le fermer par quelque moyen que ce soit. Heureusement, la commande **kill** résout ce genre de problème.

En termes simples, **kill** envoie un signal **TERM** ou **kill** à un processus pour le terminer.

Vous pouvez stopper les processus en saisissant soit le PID (identifiant du processus), soit le nom binaire du programme :

**kill 533494**

**kill firefox**

Attention avec cette commande – avec **kill**, vous courez le risque d'effacer accidentellement le travail que vous avez effectué.

## 24. Commande **ping**

**ping** est l'utilitaire de terminal réseau le plus populaire, utilisé pour tester la connectivité du réseau. **ping** dispose d'une tonne d'options, mais dans la plupart des cas, vous l'utiliserez pour demander un domaine ou une adresse IP :

**ping google.com**

**ping 8.8.8.8**

Contrairement à Windows, la commande **ping** tournera jusqu'à ce que **CTRL + C** soient tapés.

## 25. Commande **vim**

**vim** est un éditeur de texte libre et open source utilisé depuis les années 90. Il vous permet de modifier des fichiers de texte brut en utilisant des raccourcis clavier efficaces.

Certains le considèrent comme difficile à utiliser – quitter Vim est l'une des questions les plus consultées sur internet – mais une fois que vous vous y êtes habitué, il devient votre meilleur allié en ligne de commande.

Pour lancer Vim, il suffit de saisir :

**vim**



## 27. Commande **passwd**

**passwd** vous permet de changer les mots de passe des comptes utilisateurs. Tout d'abord, elle vous invite à saisir votre mot de passe actuel, puis vous demande un nouveau mot de passe et une confirmation.

Le processus est similaire à tout autre changement de mot de passe que vous avez vu ailleurs, mais dans ce cas, cela se fait directement dans votre terminal :

### **passwd**

Faites attention en l'utilisant – vous ne voulez pas perdre votre mot de passe !

```
ludo@ludo-UM700:~$ passwd
Changing password for ludo.
Current password :
```

## 28. Command **which**

La commande **which** produit le chemin complet des commandes du shell. Si elle ne peut pas reconnaître la commande donnée, retournera une valeur vide.

Par exemple, nous pouvons l'utiliser pour vérifier le chemin binaire de Python et du navigateur web Brave :

```
ludo@ludo-UM700:~$ which python
ludo@ludo-UM700:~$ which python3
/usr/bin/python3
ludo@ludo-UM700:~$ which brave
/snap/bin/brave
ludo@ludo-UM700:~$
```

## 29. Commande **shred**

Si vous avez toujours voulu qu'un fichier soit presque impossible à récupérer, **shred** peut vous aider dans cette tâche. Cette commande remplace le contenu d'un fichier de manière répétée, et par conséquent, le fichier donné devient extrêmement difficile à récupérer.

Voici un fichier avec peu de contenu :

```
ludo@ludo-UM700:~/toto$ cat file.txt
Un fichier que je veux détruire...
ludo@ludo-UM700:~/toto$
```

**shred** va remplacer le contenu du fichier par des caractères aléatoires.





### 31. Commande **tail**

Semblable à cat, tail affiche le contenu d'un fichier avec une réserve majeure : elle n'affiche que les dernières lignes. Par défaut, elle affiche les 10 dernières lignes, mais vous pouvez modifier ce nombre avec -n.

Par exemple, pour afficher les dernières lignes d'un grand fichier texte, vous utiliserez :

**tail new\_file.txt**

```
ludo@ludo-UM700:~/toto$ tail new_file.txt

Panduntur portae inferi. Pestilentia est haec ambula
bat mortuos. Maleficia! Forsitan illud Apocalypsi, ve
l malum poenae horrifying fecimus. The voodoo sacerdo
s suscitāt mortuos comedere carnem. Avium, canum, fug
ere ferae et infecti horrenda monstra. Zombies revers
us ab inferno, nam malum cerebro.

Hi mortuis soulless creaturas, imo monstra adventus
vultus comedit cerebella viventium. Avium, canum, fug
ere ferae et infecti horrenda monstra. Ut fames cereb
ro enim carnis, viscera et organa viventium. Qui tard
ius moveri, sed in magna copia sint terribiles legion
is.

Qui tardius moveri, sed in magna copia sint terribil
es legionis. Aenean a dolor vulnerum aperire accedunt
, mortui iam vivam. Putridi odores aere implent. The
voodoo sacerdos suscitāt mortuos comedere carnem. The
horror, monstra significant finem. Fit de nostra car
ne undead.

Panduntur portae inferi. Pestilentia est haec ambula
bat mortuos. Maleficia! Forsitan illud Apocalypsi, ve
l malum poenae horrifying fecimus. The voodoo sacerdo
s suscitāt mortuos comedere carnem. Avium, canum, fug
ere ferae et infecti horrenda monstra. Zombies revers
us ab inferno, nam malum cerebro.
```

### 32. Commande **head**

Celle-ci est complémentaire de la commande tail. head affiche les 10 premières lignes d'un fichier texte, mais vous pouvez définir le nombre de lignes que vous souhaitez afficher avec l'option -n



```
ludo@ludo-UM700:~/toto$ head new_file.txt -n 3
Hi mortuis soulless creaturas, imo monstra adventus
vultus comedat cerebella viventium. Avium, canum, fug
ere ferae et infecti horrenda monstra. Ut fames cereb
ro enim carnis, viscera et organa viventium. Qui tard
ius moveri, sed in magna copia sint terribiles legion
is.

Qui tardius moveri, sed in magna copia sint terribil
es legionis. Aenean a dolor vulnerum aperire accedunt
, mortui iam vivam. Putridi odores aere implent. The
voodoo sacerdos suscitatur mortuos comedere carnem. The
horror, monstra significant finem. Fit de nostra car
ne undead.
ludo@ludo-UM700:~/toto$
```

### 33. Commande **grep**

Grep est l'un des utilitaires les plus puissants pour travailler avec des fichiers texte. Il recherche les lignes qui correspondent à une expression régulière et les affiche :

**grep "Linux" new\_file.txt**

```
ludo@ludo-UM700:~/toto$ grep "linux" new_file.txt
ludo@ludo-UM700:~/toto$ grep "Linux" new_file.txt
Hi mortuis soulless creaturas, imo monstra adventus
vultus comedat cerebella viventium. Avium, canum, fug
ere ferae et infecti horrenda monstra. Ut fames cereb
ro enim carnis, viscera et organa viventium. Qui tard
ius moveri, sed in magna copia sint terribiles legion
is. Linux, canum, fugere ferae et infecti horrenda mo
nstra. Zombies reversus ab inferno.
ludo@ludo-UM700:~/toto$
```

Attention, la recherche est case sensitive.

Vous pouvez compter le nombre d'occurrences du terme en utilisant le flag -c :

**grep -c "Linux" new\_file.txt**

```
ludo@ludo-UM700:~/toto$ grep -c "Linux" new_file.txt
1
ludo@ludo-UM700:~/toto$
```

### 34. Commande **whoami**

La commande whoami (abréviation de « who am i ») affiche le nom d'utilisateur actuellement utilisé :

**whoami**

```
ludo@ludo-UM700:~/toto$ whoami
ludo
ludo@ludo-UM700:~/toto$
```

Vous obtiendriez le même résultat en utilisant `echo` et la variable environnementale `$USER` :

### 35. Commande **whatis**

`whatis` affiche une description d'une seule ligne de toute autre commande, ce qui en fait une référence utile :

#### **whatis python3**

```
ludo@ludo-UM700:~/toto$ whatis python3
python3 (1)          - an interpreted, interactive, object-oriented programming l...
ludo@ludo-UM700:~/toto$ whatis gimp
gimp : rien d'adéquat
ludo@ludo-UM700:~/toto$
```

### 36. Commande **wc**

`wc` signifie « Word Count », et comme son nom l'indique, il renvoie le nombre de mots dans un fichier texte :

#### **wc new\_file.txt**

```
ludo@ludo-UM700:~/toto$ wc new_file.txt
 19  363 2631 new_file.txt
ludo@ludo-UM700:~/toto$
```

Analysons le résultat de cette commande :

19 lignes

363 mots

Taille de 2634 octets

Nom du fichier `new_file.txt`

Si vous avez seulement besoin du nombre de mots, utilisez le flag `-w` :

### 37. Commande **uname**

`uname` (abréviation de « Unix name ») affiche les informations du système d'exploitation, ce qui est pratique pour connaître la version actuelle de Linux.

La plupart du temps, vous utiliserez l'option `-a` (`--all`), car la sortie par défaut n'est pas très utile :

#### **uname -a**


```
ludo@ludo-UM700:~/toto$ uname
Linux
ludo@ludo-UM700:~/toto$ uname -a
Linux ludo-UM700 5.15.0-46-generic #49~20.04.1-Ubuntu SMP Thu Aug 4 19:15:44 UTC 2022 x86_64 x86_64 x86_64 GNU/Linux
ludo@ludo-UM700:~/toto$
```

### 38. Commande neofetch

Neofetch est un outil CLI (interface de ligne de commande) qui affiche des informations sur votre système – comme la version du noyau, le shell et le matériel – à côté d’un logo ASCII de votre distribution Linux :

```
ludo@ludo-UM700:~/toto$ neofetch
      .-/+oosssso+/-.
      `:+ssssssssssssss++:`
      -+ssssssssssssssyyssss+-
      .ossssssssssssssdMMMMyssso.
      /ssssssssssshdmmNNmmyNMMMhsssss/
      +ssssssssshmydMMMMMMNddddyssssss+
      /ssssssshNMMMyhhyyyhNMMMNhssssss/
      .sssssssdMMMNhssssssshNMMMdssssss.
      +sssshhhyNMMNysssssssssyNMMMyssssss+
      ossyNMMMNyMMhssssssssshmmhssssssso
      ossyNMMMNyMMhssssssssshmmhssssssso
      +sssshhhyNMMNysssssssssyNMMMyssssss+
      .sssssssdMMMNhssssssshNMMMdssssss.
      /ssssssshNMMMyhhyyyhdNMMMNhssssss/
      +sssssssdmydMMMMMMNddddyssssss+
      /ssssssssshdmmNNNmyNMMMhssssss/
      .ossssssssssssssdMMMMyssso.
      -+ssssssssssssssyyssss+-
      `:+ssssssssssssss++:`
      .-/+oosssso+/-.

ludo@ludo-UM700
-----
OS: Ubuntu 20.04.4 LTS x86_64
Host: UM700
Kernel: 5.15.0-46-generic
Uptime: 58 mins
Packages: 2109 (dpkg), 18 (snap)
Shell: bash 5.0.17
Resolution: 1920x1080, 1920x1080
DE: GNOME
WM: Mutter
WM Theme: Adwaita
Theme: Yaru-dark [GTK2/3]
Icons: Yaru [GTK2/3]
Terminal: gnome-terminal
CPU: AMD Ryzen 7 3750H with Radeon Vega
GPU: AMD ATI 04:00.0 Picasso
Memory: 2339MiB / 13934MiB
```



```
ludo@ludo-UM700:~/toto$
```

Dans la plupart des machines, cette commande n’est pas disponible par défaut, assurez-vous donc de l’installer d’abord avec votre gestionnaire de paquets.

### 39. Commande find

La commande find recherche des fichiers dans une hiérarchie de répertoires en se basant sur une expression regex. Pour l’utiliser, suivez la syntaxe ci-dessous :

```
find [flags] [path] -name [expression]
```

Pour rechercher un fichier nommé new\_file.txt dans le répertoire actuel, saisissez ceci :

```
find ./ -name new_file.txt
```

```
ludo@ludo-UM700:~/toto$ find ./ -name new_file.txt
./new_file.txt
ludo@ludo-UM700:~/toto$
```

#### 40. Commande wget

wget (World Wide Web get) est un utilitaire permettant de récupérer du contenu sur Internet. Il possède l'une des plus grandes collections de flags.

Voici comment télécharger ce module à partir d'un dépôt GitHub :

**wget**

**<https://github.com/LudoFormation/commandesLinux/blob/main/01%20CommandesLinux.pdf>**