Introduction aux commandes Linux.

- A.Belcaid

Table de matière

- Introduction au Shell.
- Utilisation du manuel.
- Connaitre le contenu d'un dossier.

Pourqu'oi utiliser le shell

I currently work on a legacy system for a company. The system is really old - and although I was hired as a programmer, my job is pretty much glorified data entry. To summarise, I get a bunch of requirements, which is literally just lots of data for each month on spreadsheets and I have to configure the system to make it work, which is basically just writing a whole bunch of SQL scripts.

It’s not quite as simple as that, because whoever wrote the system originally really wrote it backwards, and in fact, the analysts who create the spreadsheets actually spend a fair bit of time verifying my work because the process is so tedious that it’s easy to make a mistake.

As you can guess, it is pretty much the most boring job ever. However, it’s a full time job with decent pay, and I work remotely so I can stay home with my son.

So I’ve been doing it for about 18 months and in that time, I’ve basically figured out all the traps to the point where I’ve actually **written a program** which for the past 6 months has been just doing the whole thing for me. So what used to **take the last guy like a month**, now **takes maybe 10 minutes** to clean the spreadsheet and run it through the program.

Remarques

Avant de commencer cette section des commandes bash, nous devons mentionner quelque **remarques** et **règles** a suivre:

1. Dans les systèmes Linux tout est `case sensitive`. Ce qui veut dire qu'on différencie entre **mylfile** et **Myfile**.
2. La notion de `corbeille` n'existe pas, tout fichier supprimé est perdu a jamais.
3. On doit pratiquer ces commandes dans un système de test pour éviter d'**endommager** votre système.

man

La commande `man` est utilisee pour obtenir l'aide d'une commande.

```
NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about  the FILEs (the current directory by default).
       Sort entries alphabetically if none of -cftuvSUX nor --sort  is  speci-
       fied.

       Mandatory  arguments  to  long  options are mandatory for short options
```

man

La commande `man` affiche le **manuel** d'utilisation d'une commande. Il est un peu dificile de naviguer ces pages.

Essayer la commande

```
man ls
```

Key	Fonction	Key	Fonction
Up	passer ligne prec	Down	ligne suivante
page Up	page prec	page up	page suivante
Q	quitter	/[string]	recherche

tldr

Il existe un programme `tldr` beaucoup plus simple a utiliser pour se familiariser avec une nouvelle commande.

- Installation:

```
sudo apt install tldr
```

- Utilisation:

```
tldr ls
```

> Essayer cette commande pour voir comment peut on afficher les fichiers **cachés**.

Whatis

Une autre commande plus simple pour obtenir une **courte description** sur une commande est:

```
whatis ls
ls (1p)              - list directory contents
ls (1)               - list directory contents
```

On peut passer plusieurs arguments a cette commande:

```
whatis ls who rm
ls (1)               - list directory contents
who (1)              - show who is logged on
rm (1)               - remove files or directories
```

> D'autre systèmes utilisent aussi la commande `apropos`.

ls

La commande `ls` liste le contenu d'un dossier.

- Usage:

```
ls [OPTIONS] [Directory]
```

- Exemple:

```
ls
shell01.md  shell01.pdf
```

> Et si on veut plus d'informations?

ls

Si on veut plus d'informations, on peut passer l'option `-l` (long)

```
ls -l
-rw-r--r-- 1 anass anass  4624 Mar  2 11:31 shell01.md
-rw-r--r-- 1 anass anass 69530 Mar  2 11:31 shell01.pdf
```

1. Les droits.
2. Nombre liens
3. utilisateur.
4. groupe.
5. Taille.
6. Date modification
7. nom

ls

On peut combiner plusieurs options

```
ls -a -l    # all files(including hidden) -l (long format)
ls -al      # Same as the previous command
```

- Utilisation classique:

commande	Information
`ls`	afficher les informations
`ls [Directory]`	Afficher les infos d'un dossier
`ls -l`	affichage long avec détails
`ls -la`	afficher les fichiers **cachés**
`ls -R`	affichage récursive

commande	Information
	lister seulement les dossiers

PWD

Une commande utile est `pwd` (*print working directory*) pour lister le *dossier courant*.

> Ça peut sembler non utile, mais ça devient très puissant dans un script.

- Syntaxe:

```
pwd
/home/anass/teaching/ENSAS
```

- Une option utile pour sauter les **liens symboliques** est `-p` pour *Physical*.

```
pwd -P   # afficher les diques phyiques
```

CD

Si vous voulez changer votre dossier courant on utilise la `cd` (*chance directory*) qui vous permet de spécifier un autre dossier.

- **Syntaxe**:

```
cd [Directory]
```

- **Exemple**:

```
pwd
/home/anass/
cd documents
pwd
/home/anass/documents/
```

> Si vous voulez aller au dossier parent on utilise `..`

Tree

Si on peut obtenir toute l'**arborescence** d'un dossier, on utilise la commande `tree`

- **Syntaxe**

```
tree [OPTIONS] [Directory]
```

- **Exemple**

```
tree  -L 2 -d ENSAS
ENSAS
├── intro_linux
│   ├── GIIA24
│   └── resources
├── numericalAnalysis
│   └── homework1
├── operational_research
│   ├── homeworks
│   ├── presentations
│   └── resources
└── probability
```

Find

La commande `find` cherche un contenu dans un système de fichiers. Vous pouvez chercher des fichiers en utilisant plusieurs caractéristiques.

- La caractéristique la plus utilisée est le **nom de fichier**.
- On peut chercher aussi par:
  - taille
  - owner.
  - date de modification.
- **Syntaxe**:

```
find [PATH] [OPTION] [CRITERIA]
```

- **Exemple**

```
find / -name hosts
/etc/avahi/hosts
/etc/hosts
```

Find (exemples)

1. Chercher des fichiers d'un utilisateur:

```
find /var -user anass
/var/mail/anass
```

2. Chercher des répertoires:

```
find root_path -type d -name dossier
find root_path -type f -name fichier
```

3. Chercher certains tailles:

```
find root_path -size +500k  # superieur a 500 kilo
find root_path -size -10M   # inferieur a 10M
```

Locate

Une alternative plus rapide que `find` est `locate` pour afficher l'emplacement des fichiers et dossiers. Cette fonction crée une **base de données** d'indexation de votre système de fichier accélérer la recherche.

- **Syntaxe**:

```
locate [OPTIONS] [Directory / File]
```

- **Exemple**:

```
locate hosts # recherch dans la base de donnes
```

> la base de donnes de `locate` est mise a jour chaque jour via une automatisation **cron**.

Exemples locate

- **Exemple 1**:

```
locate *.md  #trouver tous les fichiers avec lextension md
```

- **Exemple 2**:

```
locate -n 4 *.py  # limiter le resultat a 4 entrees
```

- **Exemple 3**:

```
locate -i readMe.md  #ignorer la casse (majiscule miniscule)
```

- **Exemple 4**:

```
locate -c *.py       #afficher le nombre des resultat trouve
```

- **Exemple 5**:

Whereis

la commande `whereis` montre l'emplacement d'une commande et aussi celui de celui fichier d'**aide**.

- **Syntaxe**:

```
whereis [OPTIONS] [Commande/file]
```

- **Exemple**:

```
whereis ls
ls: /usr/bin/ls /usr/share/man/man1p/ls.1p.gz /usr/share/man/man1/ls.1.gz
```

> la commande `which` est similaire a whereis, cependant il affiche seulement le chemin de binary.

File

Le systeme d'exploitation **Windows** utilise les extensions (`.txt` , `.exe` ..)
pour identifier le type de fichiers.

Linux n'utilise pas ces extensions, si on veut obtenir des informations sur des fichiers, on utilise la commande `file` .

- **Syntaxe**:

```
file [OPTIONS] [FILE]
```

- **Exemples**:

```
file /bin/bash
/bin/bash: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically
 linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=6c75f9f0f273cf6
549f078b042c0a3f5a04f0357, for GNU/Linux 4.4.0, stripped
```

File

- **Exemple 2**:

```
file /home/anass/backup.tgz
backup.tgz : gzip compressed data, from Unix, last modified: Tue 19 2021
```

- Voici les different types qu'on peut obtenir:

Type	Description
Ascii Text Files	Fichier textes simple
Binary Files	Des programmes exécutables
Compressed files	Fichiers compressés
Device Files	Fichiers spéciaux des devices
Links	Liens (racourcci)

Stat

Si on veut obtenir des informations détaillés sur un fichier, on utilise la commande `stat`.

Elle introduit des information qui ne sont pas disponible pour `ls`.

- **Syntaxe**:

```
stat [OPTIONS] [File/Diretory]
```

- **Exemple**:

```
stat /etc/hosts
  File: /etc/hosts
  Size: 158             Blocks: 8          IO Block: 4096   regular file
Device: 259,2   Inode: 11927995    Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2022-02-01 22:06:39.519071280 +0100
Modify: 2022-02-01 22:11:02.199601374 +0100
Change: 2022-02-01 22:11:02.199601374 +0100
 Birth: 2022-02-01 22:06:39.519071280 +0100
```

Stat

Si on ajoute l'option **-f** on peut obtenir l'information sur un système de fichier Entier.

```
stat -f /

  File: "/"
    ID: f4047ddd1a99b588 Namelen: 255     Type: ext2/ext3
Block size: 4096       Fundamental block size: 4096
Blocks: Total: 59737173   Free: 27407910   Available: 24355230
Inodes: Total: 15245312   Free: 14188533
```

Date

La commande `date` est très utile pour afficher et **configurer** la date du système.

- **Syntaxe**:

```
date [OPTIONS] [TIME/DATE]
```

- **Exemple**:

```
date
Thu Mar  3 10:17:20 AM +01 2022
```

- **Changer la date**:

```
date -s "07/10/2021 11:30"
```

Calendar

Une autre commande utile pour la gestion de temps est `cal` qui affiche un simple calendrier de votre système.

- **Syntaxe**:

```
cal [OPTIONS] [MONTH] [YEAR]
```

- **Exemple1**:

```
cal
     March 2022
Su Mo Tu We Th Fr Sa
       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
```

- **Exemple 2**:

```
cal 5 2020
```

Calendar

Voici quelque utilisations classiques:

Commande	Description
`cal`	Calendrier du mois courant
`cal -m`	Afficher lundi comme premier jour
`cal [month] [year]`	calendrier pour un mois précis
`cal [year]`	Calendrier pour toute une année
`cal -m number`	Calendirer des prochains mois

History

Et si on veut utiliser une commandes complexe qu'on a **déjà** taper au système?.

La commande `history` peut nous aider dans cette tache puisqu'elle affiche les dernières commandes saisies.

- **Syntaxe**:

```
history [OPTIONS]
```

- **Exemple**:

```
history 5 #Affiche les dernieres commandes saisie.
668   man uptime
684   cat /etc/hosts
442   ls -a
443  ping google.com
422  uptime
```

History

> Avec cette commande, on peut exécuter la commande avec son `num` "

Par exemple si on execute

```
!422        #executer la commande indexee par 422
```

> Puisque cette commande montre des information cruciales sur le système, la part des systèmes supprime le fichier `$HOME/.*history` régulièrement pour éviter d'exposer ces informations "

On peut aussi utiliser une autre commande utile pour filtrer les résultats.

```
history | grep [pattern] # ne montrer que certains résultats.
```

Clear et logout

Parfois si notre shell est plain d'information, on peut **supprimer le contenu** par la commande `clear`.

```
clear
```

- On doit remarquer que souvent cette commande est liée au raccourci `<C-l>`

Si on veut quitter la session ouverte par l'utilisateur, on utilise la commande `logout`

- **Syntaxe**:

```
logout
```

> Certains systèmes possèdent le fichier `.logout` ou `.bash_logout` qui contient des commandes a exécuter quand vous déconnecter.

Logout et exit

> Une remarque pour `logout`, elle est souvent recommandée surtout pour les administrateur systèmes pour des raisons de sécurité. Il ne faut jamais garder une session ouverte qui pourra être exploitée.

Une dernière commande pour cette lecture est `exit` qui termine aussi la session du terminal ouverte.

- **Exemple**:

```
exit [code]
```

> Cette commande est souvent attachée au raccourci `<Ctrl-D>`