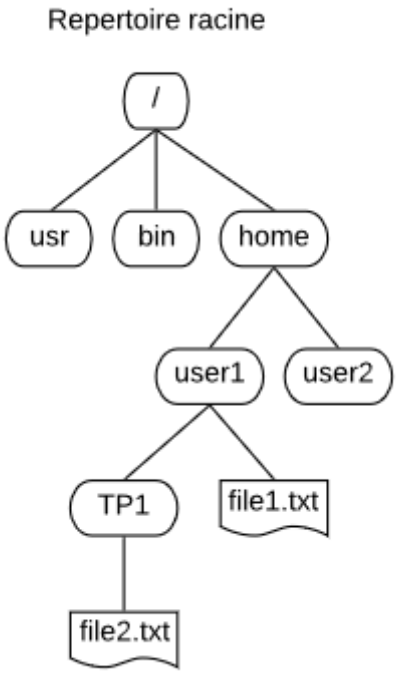
Commandes de bases Unix

Arborescence

Arbre

Les fichiers et les dossiers sont repérés par leur position dans l'arborescence.

Le fichier file1.txt est dans le répertoire user1 qui lui même est dans le répertoire home qui lui même est dans le répertoire racine.



Chemin absolu

Le chemin absolu d'un fichier ou d'un répertoire est son chemin d'accès depuis le répertoire racine.

Exemple : Pour le fichier file1.txt, le chemin absolu est /home/user1/file1.txt

Chemin relatif

Le chemin relatif d'un fichier ou d'un répertoire est son chemin d'accès depuis le répertoire courant, c'est à dire celui où l'on se place.

Exemple : Si l'on se place dans le répertoire home, pour le fichier file2.txt, le chemin relatif est user1/TP1/file2.txt

Si l'on se place dans user2, le chemin relatif pour le fichier file2.txt est ../user1/TP1/file2.txt.

Remarques :

. est le symbole pour le répertoire courant

.. est le symbole pour le répertoire parent (le niveau juste au dessus.

~ est le symbole pour le répertoire personne de l'utilisateur (/home/user1 si je suis user1).

commandes de base

pwd

pwd (Print Working Directory) : Afficher le nom du répertoire de travail en cours.

ls

ls (LiSt) : Afficher le contenu d'un répertoire.

options :

-a : Afficher tous les fichiers des répertoires, y compris les fichiers commençant par un `.'.

-l : En plus du nom, afficher d'autres informations sur les fichiers

cd

cd (Change Directory) : Changer de répertoire de travail

utilisation : cd *répertoire*

exemples : cd TP1

remarques :

- si l'on tape cd seul. On revient dans son répertoire utilisateur.
- si l'on tape cd .. , on remonte d'un niveau dans l'arborescence.

cp

cp(CoPy) : Copier des fichiers.

utilisation : cp *fichier chemin*

exemple : cp file1.txt /home/user1/TP1

mv

mv (MoVe) : Déplacer ou renommer des fichiers.

Si le dernier argument est le nom d'un répertoire existant, mv placera tous les autres fichiers à l'intérieur de ce répertoire, en conservant leurs noms. Sinon, s'il n'y a que deux fichiers indiqués, il renommera le premier pour remplacer le second.

utilisation : mv *fichier répertoire* ou mv *fichier fichier2*

exemples :mv file1.txt /home/user1/TP1 ou mv file4.txt file3.txt

rm

rm (ReMove) : Effacer des fichiers.

utilisation : rm *fichier*

exemple : rm file1.txt

mkdir

mkdir (MaKeDIRectory) : Créer des répertoires.

utilisation : mkdir *repertoire*

exemple : mkdir TP2

rmdir

rmdir (ReMoveDIRectory) : Supprimer des répertoires vides.

utilisation : rmdir *repertoire*

exemple : rmdir TP2

Joker

Un joker ou métacaractère (en anglais, wild card) est un type de caractère informatique utilisé lors de la recherche d'un mot ou d'une expression incomplète sur un réseau informatisé.

***** désigne une chaîne de caractères quelconques.

exemples:

ls b* liste tous les fichiers du répertoire courant commençant par b.

mv *.jpg répertoire_image déplace tous les fichiers du répertoire courant se terminant par .jpg dans le dossier répertoire_image.

rm *.bak efface tous les fichiers du répertoire courant se terminant par *.bak

Le manuel

Principe de base

Il est difficile de connaitre l'ensemble des commandes Unix, alors connaitre toutes les options existantes semble impossible. Heureusement, le manuel est là pour toi. Le manuel est une source très riche qui permet de s'informer et de se former. Il faut et il suffit de s'y référer chaque fois que nécessaire.

Tu as une question ? Va voir le "man" : il répondra à 95% de tes questions si tu prends la peine de le lire.

Pour les plus curieux, en résumé RTFM !

Remarque : il existe des traductions du manuel, mais on échappe pas à la langue anglaise lorsqu'on veut faire de l'informatique, aussi est-il judicieux de lire le man en anglais, cela permet aussi d'enrichir son vocabulaire au fur et à mesure.

Utilisation du manuel

man : formater et afficher les pages de manuel en ligne

utilisation : man *commande*

exemple : man ls

Edition de fichier

Nano

Nano est un éditeur de texte accessible dans la console. Pour le lancer, il suffit de taper "nano file1.txt" pour ouvrir le fichier file1.txt. Certains raccourcis sont notés en bas de la fenêtre.

Le symbole ^ correspond à la touche contrôle. Le symbole M correspond à la touche alt.

^G : accéder à l'aide

^O : enregistrer le fichier

^X : quitter

SHIFT appuyé + flèches permet de sélectionner

^K : couper

M6 : copier

^U : coller

Remarques :

Nano gère la coloration syntaxique de nombreux langages et peut donc être utiliser pour écrire du code. D'ailleurs pour faciliter le codage, lorsqu'on tape "nano -l", les numéros de ligne apparaissent.

Il existe deux autres éditeurs de texte célèbres, ayant plus de fonctionnalités : emacs et vim ceux-ci sont beaucoup plus riches mais moins facile d'accès.

cat

cat : Concaténer des fichiers et les afficher sur la sortie standard.

utilisation : cat *fichier1* [*fichier2*] ... Affiche le fichier1 puis le fichier2 puis... sur la console.

less

less : Visualiser un fichier page par page sans le modifier.

utilisation : less *fichier.txt*

Utiliser les flèches pour se déplacer.

Taper *q* pour quitter

grep

grep : Afficher les lignes correspondant à un mot ou un motif donné.

utilisation : grep *chaîne de caractères fichier*

option:

-c : compte le nombre de lignes contenant la chaîne.

-n : donne le numéro de la ligne ou apparait la chaîne.

-l : affiche le nom des fichiers qui contiennent la chaîne.

exemples : grep bonjour file1.txt recherche le mot bonjour dans le fichier texte et affiche la ligne correspondante.

grep -l Fibonnaci *.py affiche le nom des fichiers du répertoire courant ayant une extension .py contenant le mot Fibonacci.

find

find : Rechercher des fichiers dans une hiérarchie de répertoires.

option : *-name*: Recherche d'un fichier par son nom

utilisation : find *répertoire* -name *fichier*

exemple : find . -name file1.txt

Recherche à partir du répertoire courant et dans toute l'arborescence en dessous un fichier nommé file1.txt

Redirection et tube

redirection sortie

Le flux d'informations qui provient d'une commande est généralement dirigé vers la console. On peut décider de le rediriger vers un fichier via l'opérateur **>** ou l'opérateur **>>**.

exemples :

ls -al > hist.txt fait une liste des fichier et répertoires et la met dans hist.txt

pwd >> hist.txt ajoute le résultat de la commande pwd à la suite dans hist.txt.

pwd > hist.txt écrase le fichier hist.txt et y place le résultat de pwd.

redirection entrée

La redirection de l'entrée est peu utilisée car la plupart des commandes acceptent un nom de fichier en argument.

Considérons le programme python salutation.py suivant :

```
nom = input("Quel est ton nom ?\n")
print("Bonjour ", nom,".")
```

Lorsqu'on lance ce programme, il attend que l'on rentre un nom.

Celui ci peut-être dans un fichier et servir d'entrée.

Imaginons que le fichier input.txt contiennent uniquement le mot "Nicolas"/

La commande "python salutation.py < input.txt" lancera le programme salutation et prendra en entrée le texte de input.txt et fonctionnera de la même manière que si l'utilisateur avait entrée "Nicolas".

Remarque: il existe aussi un opérateur **<<** , mais son rôle est différent.

tube

Le pipe "|" redirige la sortie d'une commande vers l'entrée d'une autre commande. On enchaine donc des commandes.

exemple : ls -al | grep -c txt

"ls -al" On liste les fichiers du répertoire courant.

"grep -c txt" Parmi les résultats de la commande précédente, on sélectionne les lignes qui contiennent le mot texte on les comptabilise.

Cette enchainement permet de savoir le nombre de fichiers contenant txt dans leur nom, dans le répertoire courant.

Archive tar et fichier compressé

tar

tar : utilitaire de gestion d'archive

options :

-c, --create : créer une nouvelle archive.

-v, --verbose : affiche la liste des fichiers traités.

-f, --file *fichier* : utilise le fichier *fichier*

-x, --extract : restaure les fichiers contenus dans une archive.

utilisation :

tar -cvf *fichier.tar* Ecrit toute l'arborescence courante sur fichier.tar.

tar -xvf *fichier.tar* Charge le contenu de l'archive fichier.tar dans le répertoire courant.

gzip

gzip,gunzip : Compacter ou décompacter des fichiers.

utilisation : gzip *fichier* , gunzip *fichier*

exemple : gzip file1.txt compacte le fichier et crée file1.txt.gz gunzip file1.txt.gz décompacte le fichier.

deux en un

Aller voir dans man tar l'option z.

Les processus

définition

Un processus est une instance de programme en cours d'exécution.

ps

ps : Afficher l'état des processus en cours.

options :

-a : (autres) présente également les processus des autres utilisateurs.

-u : (utilisateur) présente le nom de l'utilisateur et l'heure de lancement.

-x : affiche les processus qui n'ont pas de terminal de contrôle.

remarques : Lorsqu'on lance cette commande, on voit apparaitre différentes informations.

L'une d'entre elle est le PID, c'est l'identificateur unique du processus.

Une autre est STAT, c'est l'état du processus

- R (running or runnable) exécuté ou prêt à être exécuté
- S (sleeping) endormi,
- D sommeil ininterruptible
- T (traced) arrêté ou suivi
- Z (zombie).

kill

kill : Envoyer un signal à un processus.

utilisation

kill *pid* : Envoie un signal au processus identifié par *PID* lui demandant de mettre ses données en ordre avant de se terminer lui-même.

kill -9 *pid* : Envoie un signal au processus identifié par *PID* afin de provoquer sa fin immédiate.

Les droits des fichiers

principe des droits

A chaque fichier est associer des droits suivant le type d'utilisateur.

Il y a 3 types d'utilisateurs.

- u : user le propriétaire du fichier
- g : groupe le groupe d'utilisateurs du fichier
- o : other les autres

Il y a 3 types de permissions

- r : read permission en lecture
- w : write permission en écriture
- x : execution permission en exécution

Ainsi lorsqu'on fait un "ls -al" et que l'on voit apparaitre à coté du fichier

```
-rwxr-x--x
```

le premier tiret indique un fichier, ce serait un d pour un répertoire.

le rwx indique les droits du propriétaires : il a accès au fichier en lecture, écriture et exécution

le r_x indique les droits du groupe : il a accès en lecture et en exécution au fichier.

le --x indique les droits des autres : ils ont accès en exécution au fichier.

chmod

chmod : Modifier les autorisations d'accès à un fichier.

utilisation : chmod *mode fichier*

le mode se décompose en 3

type d'utilisateur : u,g,o, ou a (all)

ajout, suppression : + ou -

droits accordés : r,w,x

exemples :

chmod o+x file1 : ajout des droits exécution aux autres.

chmod g-w file1 : suppression des droits d'écriture au groupe

chmod a+rwx file1 : ajout de tous les droits à tous les types d'utilisateurs.

remarques :

il existe une notation octale avec trois chiffres des droits.

1er chiffre user

2ème chiffre groupe

3ème chiffre other

1 droit en lecture

2 droit en écriture

4 droit en exécution

et ou on peut combiner par addition les droits le nombre 3 (1+2) correspond donc a des accès en lecture et en écriture

chmod 751 file1 s'interprète ainsi :

7 (1+2+4) l'utilisateur a tous les droits.

5 (1+4) le groupe a accès en lecture et en exécution.

1 les autres ont accès en lecture.

Il existe les commandes chown et chgroup pour changer les propriétaires et les groupes des fichiers.

Pour aller plus loin

alias

Les aliases sont des sortes de raccourcis pour les commandes fréquemment utilisées.

alias : Créer des alias de commandes

utilisation : alias *abréviation*='*commande*'

exemple : alias ll='ls -al'h

Fichier de configuration

Le shell (bash ou autre) est personnalisable via un fichier de configuration.

Pour BASH, il s'agit du fichier .bashrc.

Il se situe dans le dossier /home/$USER/.bashrc

Il est lu à chaque ouverture de console par l'utilisateur $USER.

On peut par exemple y ajouter la ligne alias ll='ls -al'.

Script shell

Un script shell est un fichier contenant une ou plusieurs commandes.

Il permet d'automatiser une série d'opérations.

```
#!/bin/bash
# script_simple.sh

echo "Bonjour" # affiche Bonjour
cd             # retourne dans l'espace de l'utilsateur
ls -al         # liste l'ensemble des fichiers et repertoire'
```

Pour lancer ce script, il suffit de taper dans le terminal : ./script_simple.sh

Les scripts peuvent être bien plus évolués, contenir des variables, des structures conditionnelles,... Bref, c'est un véritable langage.

En voici un exemple :

```
#!/bin/bash
# script_exemple.sh

echo "Voulez vous travaillez ? oui ? non ?"
read choix
if test $choix == 'oui'
 then
        spyder&
elif test $choix == 'non'
 then
        firefox&
 else
        echo "Je n'ai pas compris."
fi
```

Dans un programme Python

On peut appeler une commande à l'intérieur d'un programme python via la librairie os ou la librairie subprocess.

exemples :

```
import os
os.system('ls -al')
```

```
import os
os.system('ls -al')hh
```