TP OS n°3

Sommaire

Les objectifs de ce tp sont d'être capable, en utilisant des commandes de base sous GNU/-Linux, de se déplacer dans une arborescence de fichiers et lister son contenu, d'obtenir des informations sur le système d'exploitation.

Introduction

Environnement de travail

Il vous faut ouvrir une **session** sur votre poste de travail. Vous pouvez utiliser soit le mode console soit l'interface graphique. Dans les deux cas, vous allez travailler **« en ligne de commande »**.

Pour ce TP, il vous faudra préalablement créer un répertoire de travail `tpos3` dans `$HOME/tpos` en tapant la commande suivante :

```
$ mkdir -p $HOME/tpos/tpos3
```

Pour se déplacer dans l'arborescence de travail, il faut taper la commande suivante :

```
$ cd $HOME/tpos/tpos3
```

Complétion

> « L'art de la saisie des commandes ultra rapide et sûre ».

À chaque fois que vous tapez une commande, Linux (avec le paquetage `bash-completion` installé) vous aide à compléter votre commande en appuyant sur la touche `Tab`. Tapez le début de votre commande et en appuyant sur la touche `Tab`, Linux vous la complète ou vous propose les différentes possibilités pour la compléter, à défaut vous aurez droit à un petit bip.

Le *shell* `bash` effectue la complétion en considérant successivement le texte comme une variable (s'il commence par $), un nom d'utilisateur (s'il commence par ~), un nom d'hôte (s'il commence par @), ou une commande (y compris les alias et les fonctions). Si rien ne fonctionne, il essaye la complétion en nom de fichier.

```
// Complétion des commandes :
$ c
Display all 132 possibilities? (y or n)
$ cd
cd        cd..      cdrdao    cdrecord  cdxa2mpeg

// Complétion des chemins et noms de fichiers :
$ cd /h complète la commande en cd /home/

// Complétion pour les options des commandes :
$ ls --a
--all       --almost-all --author

// Complétion pour les variables d'environnement :
$ $U
$UID  $UMASK_ROOT $UMASK_USER $USER
```


Système de fichiers

Un système de fichiers (*filesystem*) est une structure de données permettant de stocker les informations et de les organiser dans des fichiers sur ce que l'on appelle des mémoires secondaires ou de stockage (disque dur, disquette, CD-ROM, clé USB, etc.). Une telle gestion des fichiers permet de traiter, de conserver des quantités importantes de données ainsi que de les partager entre plusieurs programmes informatiques. Il offre à l'utilisateur une vue abstraite sur ses données et permet de les localiser à partir d'un chemin d'accès.

Il existe d'autres façons d'organiser les données, par exemple les bases de données.

Pour l'utilisateur, un système de fichiers est vu comme une arborescence : les fichiers sont regroupés dans des répertoires (concept utilisé par la plupart des systèmes d'exploitation). Ces répertoires contiennent soit des fichiers, soit d'autres répertoires. Il y a donc un répertoire racine et des sous-répertoires. Une telle organisation génère une hiérarchie de répertoires et de fichiers organisés en arbre.

Chemin d'accès

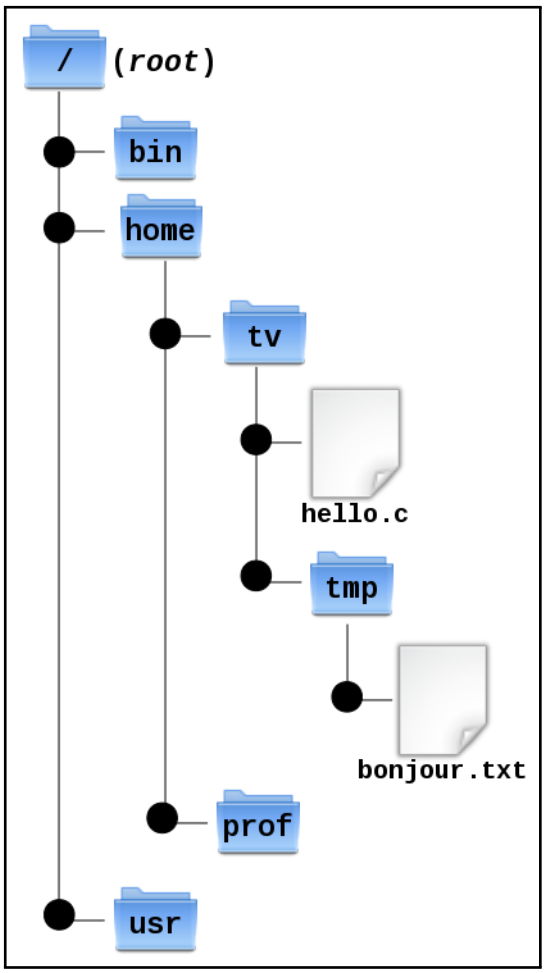
Le chemin d'accès d'un fichier ou d'un répertoire est une chaîne de caractères décrivant la position de ce fichier ou répertoire dans le système de fichiers.

Chemins d'accès selon le système d'exploitation :

OS	Répertoire racine	Séparateur de répertoire
Système de type Unix/Linux	/	/
DOS et ses dérivés (OS/2 et Microsoft Windows)	$< lettredulecteur >: \backslash$	$\backslash$
Classic Mac OS	$< nomdudisque >:$	:

Remarque : les systèmes Unix/Linux disposent d'une arborescence unique.

On distingue deux types de chemins d'accès :
- le **chemin absolu** dont la référence est la **racine**. Sous UNIX/Linux, un chemin absolu commence toujours par `/`.
- le **chemin relatif** dont la référence est le **répertoire courant**.



Quel est le chemin d'accès à "`hello.c`" ?

→ Avec un chemin d'accès absolu : `/home/tv/hello.c`

→ Avec un chemin d'accès relatif : tout dépend de l'endroit où on exécute la commande, c'est à dire le répertoire de travail (ou répertoire courant). Pour cela, on peut utiliser deux références connus du système d'exploitation : le répertoire courant (noté `.`) ou le répertoire parent (noté `..`) :

– Supposons que le répertoire courant est `prof`, on pourra désigner `hello.c` par `../tv/hello.c`
– Supposons que le répertoire courant est `tv`, on pourra désigner `hello.c` par `./hello.c`

Quel est le chemin d'accès à "`bonjour.txt`" ?

→ Avec un chemin d'accès absolu : `/home/tv/tmp/bonjour.txt`

→ Avec un chemin d'accès relatif : tout dépend de l'endroit où on exécute la commande, c'est à dire le répertoire de travail (ou répertoire courant). Pour cela, on peut utiliser deux références connus du système d'exploitation : le répertoire courant (noté `.`) ou le répertoire parent (noté `..`) :

– Supposons que le répertoire courant est `prof`, on pourra désigner `bonjour.txt` par `../tv/tmp/bonjour.txt`
– Supposons que le répertoire courant est `tv`, on pourra désigner `bonjour.txt` par `./tmp/bonjour.txt`

Manipulations

Étape n°0 : commandes de base

Voici quelques commandes de base :

```
pwd   : affiche le chemin d'accès au répertoire courant
man   : le manuel
gcc   : compilateur (transforme un fichier source en un fichier exécutable)
vi, vim : éditeurs de texte
clear   : efface l'écran
echo  : affiche une ligne de texte
date  : affiche et modifie la date et l'heure
cal   : affiche le calendrier
bc    : calculatrice
cd    : permet de se déplacer dans une arborescence
ls    : liste le contenu d'un répertoire
ls -lR  : affiche une arborescence et ses caractéristiques
rm    : supprime un fichier
rm -rf  : force la destruction d'une arborescence (rmdir)
cp    : permet la copie de fichier (voir aussi cp -a)
mv    : déplace ou renomme une partie d'une arborescence
mkdir   : crée un répertoire dans une arborescence
du    : affiche une arborescence et sa taille (du -h)
df    : fournit la quantité d'espace occupé par les systèmes de fichiers (df -Th)
file  : affiche le type des fichiers
type  : indique le type pour une commande
locate  : localise un fichier
find  : recherche des fichiers sur le système
whereis : permet de trouver l'emplacement d'une commande
whatis  : donne une description d'une commande
which   : donne le chemin complet d'une commande
cat   : affiche et/ou concatène le(s) fichier(s) sur la sortie standard
od    : affiche le dump d'un fichier
wc    : compte les caractères, les mots et les lignes en entrée
more  : affiche à l'écran l'entrée standard (page par page)
less  : idem avec possibilité de retour en arrière
cut   : permet d'isoler des colonnes dans un fichier
expand  : transforme les tabulations en espace
head  : affiche les n première lignes
join  : joint les lignes de deux fichiers en fonction d'un champ commun
sort  : trie les lignes de texte en entrée
paste : concatène les lignes des fichiers
tail  : affiche les n dernières lignes d'un fichier
tac   : concatène les fichiers en inversant l'ordre des lignes
uniq  : élimine les doublons d'un fichier trié
rev   : inverse l'ordre des lignes d'un fichier
tr      : remplace ou efface des caractères
grep  : recherche des chaines de caractères dans des fichiers
```


Étape n°1 : arborescence de fichiers

Se déplacer dans l'arborescence :

```
$ cd $HOME/tpos/tpos3
```

Créer le répertoire `temp` :

```
$ mkdir temp
$ cd temp
```

Copier le fichier `/etc/passwd` dans le répertoire courant (désigné par un `.`) :

```
$ cp /etc/passwd .
```

Lister le contenu du répertoire :

```
$ ls
```

*Remarque : le fichier **`passwd`** contient la liste des utilisateurs de la machine et le répertoire **`/etc`** contient l'ensemble des fichiers de configuration de la machine (ce sont tous des fichiers textes ASCII)*

Faire une copie de sauvegarde d'un fichier :

```
$ cp ./passwd ../passwd.bak
```

Renommer un fichier :

```
$ mv ./passwd ./listeUtilisateurs.txt
```

Visualiser le contenu d'un fichier texte ASCII :

```
$ more listeUtilisateurs.txt
$ cat listeUtilisateurs.txt
$ less listeUtilisateurs.txt
```

Rechercher un fichier dans son répertoire personnel :

```
$ find $HOME -name listeUtilisateurs.txt -print
$ find $HOME -name *.txt -print
```

*Remarque : l'étoile * est un caractère joker qui a la particularité de remplacer n'importe quel caractère autant de fois que nécessaire*

Effacer un fichier :

```
$ rm listeUtilisateurs.txt
```

*Remarque : l'option **`-f`** force la suppression (sans demander de confirmation) et celui-ci a été supprimé de manière définitive !*

Déplacer un répertoire :

```
$ mv $HOME/tpos/tpos3/temp $HOME
```

Déplacer un fichier :

```
$ mv $HOME/tpos/tpos3/passwd.bak $HOME
```

Effacer un répertoire :

```
$ rm -rf $HOME/tpos/tpos3
```

Remarque : Le répertoire (et tout son contenu avec l'option `-r`*) a été supprimé définitivement !*

Effacer un fichier :

```
$ rm $HOME/passwd.bak
```

Remarque : Il ne doit plus rester aucune trace (fichier ou répertoire) de cette manipulation sur votre système.

Étape n°2 : les commandes

Il existe plusieurs type de commandes :
– les commandes internes (au *shell*)
– les commandes externes (donc des programmes)
– les *alias* (voir plus loin)

Les commandes externes (donc des exécutables) sont généralement stockées dans un répertoire de nom `bin`. Il existe des exécutables dans :

– le répertoire `/sbin` contient les commandes pour *root*
– le répertoire `/bin` on trouve des commandes et des *shells*
– le répertoire `/usr/bin` est le répertoire de base des programmes

Voici quelques commandes sur les commandes :

```
// Localiser une commande :
$ which cat

// Localiser l'exécutable, le source et la page de manuel d'une commande :
$ whereis file

// Rechercher le nom du fichier qui sera exécuté à l'appel de commande :
$ type -p strings

// Déterminer le type d'une commande :
$ type echo
$ type cat
$ type ll
```

La commande `man` permet d'obtenir de l'aide sous Linux (`man man`). Elle reçoit en paramètre la commande (ou autre chose) sur laquelle on désire de l'aide. Pour quitter les pages d'aide, il faut appuyer sur la touche `q`. Les flèches permettent de descendre ou de remonter dans le texte, la touche `g` (ou la flèche `home`) permet d'aller au début du document, la touche `G` (ou la touche `fin`) à la fin du document, `b`

permet d'aller à la page suivante et la `barre d'espace` permet de revenir à la page précédente. Les touches `PgDn` et `PgUp` permettent-elles aussi de se déplacer par page. On peut faire des recherches en tapant `/` puis le mot recherché puis on navigue vers la prochaine occurrence avec la touche `n` (ou `N` pour rechercher en arrière).

```
// Obtenir de l'aide sur une commande :
$ man mkdir
$ help test
$ info pwd

// Rechercher de noms complets dans la base de données whatis :
$ whatis mkdir

// Rechercher des chaînes de caractères dans la base de données whatis :
$ apropos mkdir

// Lire la page d'introduction qui présente une section (ici la section 3 sur les Fonctions
   de bibliothèque) :
$ man 3 intro

// Obtenir de l'aide en précisant la section (ici la section 7 sur Le manuel d'
   administration) :
$ man 7 mailaddr
```

Remarque : Il existe d'autres sources d'informations qui sont les HowTo, les FAQ et les répertoires doc. Les HowTo sont des textes qui expliquent comment faire une installation, une configuration ... Les FAQ (Frequency Asked Questions ou Foire Aux Questions) sont des recueils de questions-réponses les plus fréquemment posées. Les documentations sont généralement archivées dans `/usr/share/doc/`.

Le terme *alias* signifie synonyme et permet créer de nouvelles commandes ou d'en redéfinir d'autres pour obtenir des comportements différents.

```
// Lister les alias actuellement définis :
$ alias

// Créer un alias :
$ alias mot=commande
$ alias dir='ls -alh'
$ dir

// Supprimer un alias :
$ unalias mot
$ unalias dir
```

Remarque : les alias nouvellement créés n'existent seulement pour cette session.

Étape n°3 : informations diverses

Obtenir quelques informations sur le système d'exploitation et l'utilisation qui en est faite.

```
// Afficher des informations sur le système :
$ hostname
$ uname -a
$ cat /etc/release
$ cat /etc/version
$ cat /etc/issue

// Afficher des informations sur le matériel :
$ cat /proc/cpuinfo
$ cat /proc/cpuinfo | grep -e "^\(vendor_id\|model name\|cpu\)"
$ cat /proc/meminfo
$ cat /proc/mtrr
$ dmesg
$ df
$ lspci
$ lspcidrake
$ cat /etc/X11/xorg.conf

# lsusb
# lshw
# dmidecode
```

Rappel : toute commande précédée de l'invite `#` *nécessite les privilèges du super-utilisateur (*`root`*). Pour "passer" sous le compte super-utilisateur, on utilise la commande* `su` *pour ouvrir une session ou* `sudo` *pour exécuter une simple commande. Pour revenir sous son compte, on tape la commande* `exit`*.*

```
// Afficher des informations sur l'environnement de travail :
$ echo $SHELL
$ env

// Afficher des informations sur les utilisateurs :
$ whoami
$ who am i
$ logname
$ who
$ last
$ w
$ users
$ id
$ getent passwd

// Afficher des informations sur l'utilisation du système :
$ uptime
$ ps e
$ ps x
$ top   (q pour quitter)
$ free
$ vmstat
$ history
```



```
// Afficher des informations sur la date et l'heure :
$ date
$ date +%x
$ date +%X
$ cal
$ cal 2011
```

Questions de révision

L'idée de base des questions de révision est de vous donner une chance de voir si vous avez identifié et compris les points clés de ce TP.

Question 1. Un chemin absolu commence toujours par ?

Question 2. Un chemin relatif commence par ? Citer les deux possibilités.

Question 3. Donner le chemin absolu de votre répertoire racine personnel.

Question 4. Je suis dans le répertoire `/usr/bin/`, quel est le nom de mon répertoire parent désigné par `..` ?

Question 5. Je suis dans le répertoire `/usr/bin/`, quel est le nom du répertoire courant désigné par `.` ?

Question 6. Dans quel répertoire suis-je après l'exécution de ces commandes ? `$ cd / ; cd etc ; cd ../usr ; cd ./bin`

Question 7. Que contient la variable `$HOME` ?

Question 8. Quelle(s) est (ou sont) l'(es) erreur(s) dans cette commande ? `$ cd/ETC`

Question 9. Peut-on connaître la durée depuis laquelle le système fonctionne ?

Question 10. Qu'est-ce qu'un PID ?

Travail demandé

Exercice 1 : divers

Question 11. Donner l'option de la commande `ls` qui permet de lister une arborescence complète (fichiers, répertoires et sous-répertoires).

Question 12. Quelle est la commande qui permet d'afficher le répertoire de travail dans lequel je suis ?

Question 13. Pourquoi le fichier `.hello.txt` ne s'affiche-t-il pas lors d'un `ls -l` ?

Question 14. Quelle option de `ls` permettrait de l'afficher quand même ?

Question 15. Quelle est la version du noyau GNU/Linux de votre machine ?

Question 16. Quelle est la version de votre distribution Mandriva ?

Question 17. Bonus : Quelle est l'unité de mesure de la puissance du CPU au démarrage d'une machine sous Linux ?

Exercice 2 : exploitation évoluée des commandes

L'objectif de cet exercice est de s'initier au rôle d'administrateur.

L'administrateur d'un système n'a pas les mêmes besoins qu'un simple utilisateur. Il est souvent amené à :
– automatiser des traitements longs (par des scripts) et à
– exploiter les services offerts par le système (par les options des commandes)

La commande `find` permet d'effectuer des recherches approfondies dans une arborescence. Elle est souvent utilisée, compte tenu de la richesse de ses critères, en frontal d'une autre commande, pour procéder à la sélection de fichiers. On vous demande d'exploiter les « pouvoirs » de la commande `find` ! Il est donc indispensable de faire un : `$ man find`

Question 18. Donner la commande `find` qui permet de rechercher seulement les fichiers cachés dans votre répertoire personnel.

Question 19. Donner la commande `find` qui permet de rechercher les fichiers de taille >1000 Koctets dans votre répertoire personnel et afficher cette taille avec une unité adaptée (K, M ...).

Remarque : pour chaque résultat trouvé par ***`find`****, il faut lui demander d'exécuter la commande* ***`du -h`****. Le représentera chaque fichier trouvé : cela fait partie de la syntaxe de la commande find (ainsi que le \;).*

Question 20. Donner la commande `find` qui permet de rechercher les fichiers qui ont été modifiés depuis 24 heures dans votre répertoire personnel.

