TD 1 - Guide de survie sous Linux

13 septembre 2011

1 Introduction

Ce premier TD a pour objectif de vous familiariser avec l'utilisation d'un terminal sous Linux de manière à faciliter l'organisation et la gestion de répertoires et de fichiers dans le cadre des différents TDs sur machine au cours du semestre.

Avant toute chose, il est nécessaire d'apprendre à organiser son travail. Pour cela, créez des répertoires séparés pour chacun des enseignements (I11, I12, etc.) et prenez l'habitude, pour chacun de ces enseignements, de créer un nouveau sous-répertoire pour chaque nouveau TD (TD1, TD2, etc.).

La deuxième chose essentielle est d'apprendre à devenir autonome dans son utilisation de Linux. Pour cela, de nombreuses possibilités vous sont offertes pour obtenir de l'information sur les commandes et leurs options, *avant* d'aller interroger Internet pour obtenir la réponse. Ces différentes possibilités (man, info, apropos, help) sont décrites dans l'annexe 1 en fin d'énoncé.

Note1

La première chose à faire est d'ouvrir un terminal de commandes, c'est-à-dire une fenêtre dans laquelle vous pourrez taper des commandes à exécuter par le système. ***Nous n'utiliserons pas le gestionnaire de fichiers au cours de ce TD***.

Note2

Lorsque vous ouvrez la fenêtre du terminal, vous voyez apparaître un texte court qui se termine par le symbole `$`. Ce texte ressemble à ceci :

```
Last login: Sun Sep 19 04:41:29 on ttys001
getz-4:~ Guyom$
```

Le symbole `$` s'appelle l'*invite* de commande et signifie que le terminal est prêt à exécuter une commande. Dans les exemples ci-dessous, il ne faudra donc pas taper le symbole `$` précédant le nom de la commande à tester.

Note3

Pour chacune des commandes présentées dans la suite, vous pourrez obtenir une présentation complète de leur utilisation et de leurs options en tapant la commande suivante (sans taper le symbole `$` et en remplaçant `cmd` par le nom de la commande dont vous voulez connaître la syntaxe) :

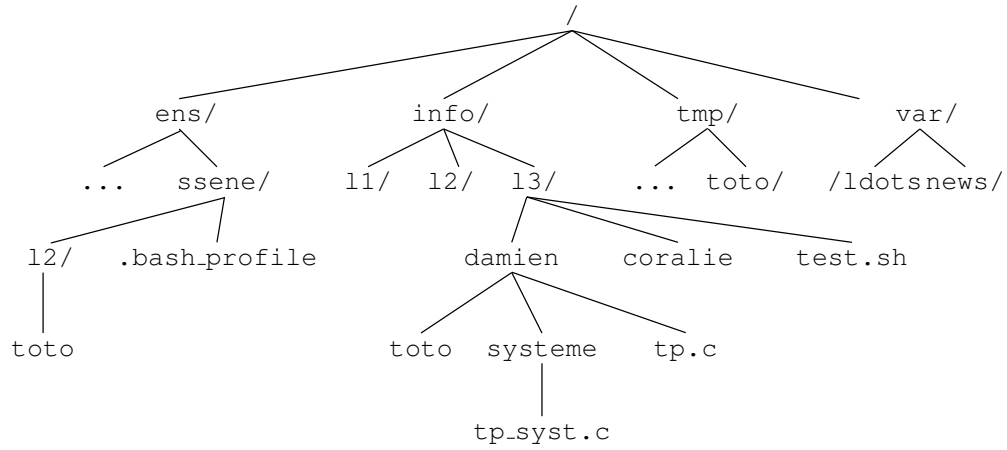
```
$ man cmd
```

2 Manipulation des répertoires et des fichiers

2.1 Généralités

Tous les systèmes d'exploitation récents (DOS, Windows, MacOS, Linux...) utilisent des systèmes de fichiers qui gèrent une hiérarchisation de leur contenu. Ces systèmes sont conçus autour de 2 entités de base: les fichiers qui contiennent des données et les répertoires qui contiennent des répertoires et des fichiers.

Les fichiers contiennent des informations comme du texte, du son, des exécutables... Les répertoires peuvent contenir des fichiers mais aussi d'autres répertoires. Lorsqu'un répertoire en contient d'autres, on commence à parler de hiérarchie entre eux, le répertoire qui contient un répertoire est appelé *répertoire parent*, celui ou ceux qui sont contenu sont généralement nommés *répertoires enfants*. La structure donnée par les répertoires est appelée l'*arborescence*. La figure ci-dessous montre un exemple d'arborescence.



2.2 Travail sur papier

2.2.1 Noms de fichiers

Tous les noms de fichiers sont soumis aux mêmes règles et doivent respecter les contraintes suivantes pour être compatibles avec d'anciens systèmes UNIX:

- un nom de fichier ne doit pas dépasser 255 caractères (normalement, pour être absolument portable sur tous les systèmes UNIX, un nom de fichier devrait contenir au maximum 14 caractères);
- un nom de fichier doit être composé de majuscules, minuscules et chiffres.
- un nom de fichier peut contenir, entre autres, le caractère *underscore* _, le point `.` ou encore les signes moins `-`, `+`.

1. Déterminez si les noms suivants sont des noms de fichiers valides ou litigieux (d'après les règles pré-citées) en écrivant les chaînes telles quelles dans un terminal derrière la commande `touch`: `.tcshr`, `CouCou`, `aujourd'hui`, `essai.tex`, `sujet-de_stage`, `Comment faire`, `"Comment faire"`, `test1&test2`, `$77fichier`, `33+11=44`.

2.2.2 Chemin d'accès absolu

1. À partir de l'exemple d'arborescence de données, donnez les chemins d'accès absolus aux fichiers ou aux répertoires suivants: `toto` (tous les chemins), `damien`, `news`, `tp_syst.c`.

2.2.3 Chemin d'accès relatif

1. Toujours à partir de l'arborescence, donnez les chemins d'accès relatifs aux fichiers ou aux répertoires suivants (abrégés par FD) par rapport au répertoire courant (abrégé par RC):
   - FD: `tp.c`; RC: `/info/l1`
   - FD: `toto` (dans `damien/`); RC: `/info/l3`
   - FD: `news`; RC: `/info/l3/coralie`
   - FD: `damien`; RC: `/info/l3/coralie`
   - FD: `toto tmp/`; RC: `/var/news`
   - FD: `ssene/`; RC: `/`
   - FD: `l3`; RC: `coralie`

2.2.4 Manipulation de fichiers

1. En utilisant les commandes `rm <fichier>` (suppression d'un fichier), `mv <source> <dest>` (déplacement de `source` vers `dest`), `cp <source> <dest>` (copie de `source` vers `dest`), donnez les lignes de commandes (en utilisant les chemins d'accès relatifs) qui permettent de:
   - copier le fichier `.bash_profile` dans le répertoire `coralie/` à partir de `/`.
   - déplacer `tp.c` dans `systeme/` à partir de `coralie/`.
   - renommer `tp.c` en `tp1.c` à partir de `ens/`.
   - effacer `toto` de `ssene/` à partir de `/info/`
   - échanger les deux fichiers `toto` de `damien/` et de `tmp/` à partir de `ens/`.

2.2.5 Manipulation de répertoires

1. En utilisant les commandes `mkdir <nom de répertoire>` (création), `rmdir <nom de répertoire>` (suppression) et `mv` vue précédemment, donnez les lignes commandes qui permettent de:
   - effacer le répertoire `/ssene/l2/` à partir de `/`.
   - créer un répertoire `programmation/` dans `damien/` à partir de `/info/`.
   - déplacer le répertoire `programmation` dans `coralie` à partir de `/`.
   - effacer le répertoire `damien/`.

2.3 Les commandes de base

2.3.1 Où suis-je ?

Lorsque l'on travaille dans le terminal, le système va interpréter les commandes en considérant que l'on se situe à un endroit précis dans l'arborescence des répertoires. Pour savoir où l'on se trouve dans cette arborescence on peut utiliser la commande `pwd` (*Path of Working Directory*) :

```
$ pwd
/home/user62/I11/TD1/
```

Exercice 2.1 *Testez la commande `pwd`. Comment s'appelle votre répertoire de base (*home directory*) ? Comment s'appelle le répertoire parent de votre répertoire de base ?*

2.3.2 Qu'est-ce qu'il y a ici ?

Pour visualiser le contenu d'un répertoire, la commande à utiliser est `ls` (*LiSt*).

Exercice 2.2 *La commande* ls *prend de très nombreuses options. Testez la commande sans options puis avec les options `-l` et `-al`. Cherchez dans les pages de manuel la signification des options `-l` et `-a` (`-al` signifie que l'on veut à la fois l'option `-a` et l'option `-l`).*

Exercice 2.3 *En utilisant la commande `man` (voir annexe 1), trouvez les options à utiliser avec `ls` pour obtenir un affichage en couleurs, avec un `/` derrière les noms de répertoire.*

2.3.3 Mon premier répertoire

On peut maintenant créer un répertoire et vérifier qu'il existe. La commande pour créer un nouveau répertoire est `mkdir` (*MaKe DIRectory*) :

```
mkdir mon_repertoire
```

Exercice 2.4 *Créez un répertoire `I12` dans votre répertoire de base et vérifiez avec la commande `ls` qu'il a bien été créé.*

2.3.4 Zut, je me suis trompé

Zut, ce n'est pas le cours de I12 mais de I11. Pour supprimer un répertoire vide la commande est *rmdir* (*ReMoveDirectory*).

```
rmdir mon_repertoire
```

Exercice 2.5 *Supprimez le répertoire `I12` et créez-en un nouveau appelé `I11` dans votre répertoire de base.*

2.3.5 Où vais-je ?

Visualiser le contenu d'un répertoire est indispensable, mais il faut aussi pouvoir se déplacer dans la hiérarchie. La commande qui permet de changer de répertoire est `cd` (*Change Directory*). La commande s'utilise en la faisant suivre d'un espace puis du nom du répertoire dans lequel vous voulez vous déplacer. Si vous ne spécifiez aucun répertoire de destination, la commande vous ramène dans votre répertoire de base.

```
$ cd rep_destination
```

Exercice 2.6 *Créez un sous-répertoire `TD1` dans le répertoire `I11` que vous venez de créer, puis déplacez vous dans ce répertoire. Revenez ensuite dans le répertoire `I11`.*

2.3.6 Des répertoires étranges...

Lorsque vous avez listé le contenu d'un répertoire avec l'option `-a`, vous avez pu observer que la liste des fichiers commençaient par deux répertoires particuliers. Ces répertoires sont nommés `.` et `..` et sont présents dans tous les répertoires.

Exercice 2.7 *En utilisant les commandes vues précédemment, essayez de comprendre à quoi correspondent ces deux répertoires.*

2.4 Pour se simplifier la tâche

Rappeler une commande précédente

Lorsque l'on exécute régulièrement les mêmes commandes (typiquement, lorsqu'on programme, on passe son temps à compiler son programme et à l'exécuter), on n'a pas envie de retaper chaque commande en entier. Les commandes précédentes sont accessibles en tapant une ou plusieurs fois sur les flèches du haut et du bas (la flèche du haut permet de rappeler des commandes de plus en plus anciennes, la flèche du bas fait l'inverse).

L'auto-complétion

La touche tabulation (une flèche vers la droite avec une barre verticale au bout de la flèche, sur la gauche du clavier) permet de compléter automatiquement un nom de commande ou de fichier (on y fera référence par la suite par [TAB]). Pour cela, vous tapez la première lettre d'une commande (ou les 2 ou 3 premières) puis appuyez sur la touche [TAB]. Le système complètera le nom de commande s'il est unique ou proposera une liste de choix de commandes commençant par la lettre que vous avez tapée (certains systèmes signalent par un bip sonore que la saisie n'est pas unique, et qu'il ne peut pas compléter automatiquement). S'il y a plusieurs choix possibles, il faut appuyer à nouveau sur [TAB] pour voir les différents choix:

```
$ m[TAB][TAB]
Display all 216 possibilities? (y or n)  n
$ mo[TAB][TAB]
moc      moc-qt4  modprobe          mogrify  more   mountpoint  movemail  mozilla-firefox
moc-qt3  modinfo  module-assistant  montage  mount  mousepad    mozilla   mozilla-thunderbi
$ moz[TAB]
$ mozilla[TAB]
mozilla  mozilla-firefox  mozilla-thunderbird
```

L'autocomplétion est très utilisée notamment pour parcourir les répertoires et les fichiers.

Faire des alias des commandes

Taper toujours les mêmes arguments pour lancer une commande devient vite fastidieux. Pour gagner du temps il est possible de créer des *alias* des commandes :

```
$ alias l='ls -alFG'
$ l
drwxr-xr-x 5 lpoligny sydra  4096 2007-09-14 18:13 .
drwxr-xr-x 3 lpoligny sydra  4096 2007-09-10 15:01 ..
-rw-r--r-- 1 lpoligny sydra 33341 2007-09-14 15:54 enonce.pdf
drwxr-xr-x 7 lpoligny sydra  4096 2007-09-10 16:03 Exercice1
drwxr-xr-x 7 lpoligny sydra  4096 2007-09-11 07:06 Exercice2
drwxr-xr-x 7 lpoligny sydra  4096 2007-09-12 22:45 Exercice3
```

Pour rendre ces alias permanents, il faut les ajouter au fichier .bashrc à la racine de votre compte, et lancer la commande

```
$ source ~/.bashrc
```