Aide-mémoire Linux

Éric Guichard

2009, 2013, 2015

Ce document rappelle les commandes les plus utiles de Linux. Il est «contextuel» : il accompagne divers cours proposés par Éric Guichard à l'Enssib. Il suppose d'avoir ouvert une session sous Linux et la connaissance du *terminal* (qui existe aussi sur Mac).

Voici une petite sélection (clicable) d'autres notes de cours en ligne :

— `http://www.linux-france.org/article/debutant/debutant-linux.html`
— `http://www-inf.enst.fr/~danzart/fiches/unix_abrege.html`
— `http://www.tuteurs.ens.fr/unix/`

1 Bases

1.1 Chemins d'accès, envoi de commandes

Pour Linux / Unix, l'articulation d'un chemin d'accès est le `/`. Le chemin d'accès (absolu) à un fichier, à un dossier, ou à une commande est donné par la succession des dossiers qui y conduisent, en partant de la racine (*root*) notée `/`.

Exemples :

— `/usr/bin/perl` signifie que le fichier-programme `perl` est dans le dossier `bin`, lui-même dans `usr`, qui se trouve en racine.
— `/home/dupont/pancarte.jpg` : le fichier `pancarte.jpg` est dans le dossier `dupont`, qui est dans `home`. Donc `dupont` correspond certainement à un nom d'utilisateur.

Une validation de commande s'obtient en appuyant sur la touche «retour-chariot».

1.2 Astuces et raccourcis propres au terminal

Ces points sont essentiels.

— `TAB` (touche tabulation) : complétion d'un nom de fichier, de dossier ou d'une commande. Exemple : vous voulez connaître la taille du fichier nommé `fiCHiEravecUnnomabsolumenTInfernal`. Saisissez alors la commande `ls -l fiCH`, et appuyez ensuite sur la touche tabulation. Sauf situation imprévue (néanmoins prévisible), vous verrez apparaître à l'écran `ls -l fiCHiEravecUnnomabsolumenTInfernal`.
De façon générale, **n'écrivez jamais un nom de fichier ou de dossier vous-même**. Vous avez 8 chances sur 10 de faire une erreur. Laissez donc le système compléter ce que vous commencez à écrire.
— Flèches montantes et descendantes du clavier (↑ et ↓) : rappel des commandes antérieures. Très utile. Cf. aussi la commande `history`.
— `Control-c` (maintenez appuyée la touche de fonction Control [Ctrl], et appuyez une fois sur la touche c, puis relâchez les deux) : pour arrêter un processus en cours. Utile quand on est perdu.
— `Ctrl-d` : pour se déconnecter ou se sortir d'un mauvais pas.
— `q` (quitter) pour clore le `less` (et bien d'autres logiciels). De façon générale, quand vous êtes perdu/e, osez saisir un `q`, un `quit`, appuyer sur la touche Escape (echap), tenter un `bye`, etc. Les programmes informatiques ont été faits par des *humains*. Mettez-vous à la place des auteurs et non des machines, c'est plus efficace.

— **Copier/coller**. En général, ces opérations se réalisent avec les combinaisons `Ctrl-c` et `Ctrl-v`. Comme `Ctrl-c` sert **dans le terminal** à interrompre un processus, ce sont les combinaisons de touches `Ctrl-C` (`Ctrl-Maj-c`) et `Ctrl-V` qui sont à utiliser.
— Manuel (man) : la plupart des commandes et programmes que vous utilisez avec le terminal disposent d'une aide, obtenue en saisissant la formule `man lacommande`. Exemples : `man ls`, `man less`, `man mv`, `man chown`. Faites défiler le contenu de l'aide en appuyant sur la barre d'espacement (grande touche longue en bas du clavier) et quittez-la en tapant comme prévu… `q`.
Vous pouvez aussi rechercher dans quel contexte est utilisé un mot ou une commande : `man -k lacommande`. Cela peut être utile lorsque le nom de la commande n'est pas intuitif. Ex. : `man password` ne donne pas de réponse, car la commande permettant de changer de mot de passe ne s'intitule pas *password*. Mais un `man -k password` aide à comprendre que la commande recherchée est `passwd`.

1.3 Astuces et conseils génériques

Ce point vaut pour la majorité des outils et renvoie à la culture informatique banale.
— Évitez de saisir les chiffres de votre **login** ou de votre mot de passe avec le pavé numérique : celui-ci n'envoie pas toujours des chiffres tant que votre session n'est pas effectivement ouverte (vos paramètres ne sont pas encore «lus» par le système).
— **Raccourcis clavier** (avec la touche Ctrl) : copier (`Crtl-c`, sauf dans le terminal), coller (`Crtl-v`), imprimer (`Ctrl-p`), rechercher `Ctrl-f`), etc.
— Valider/annuler une proposition énoncée dans une fenêtre : «retour-chariot»/«echap». Cela vous évite de répondre avec la souris.
— Enregistrer : **enregistrez aussi souvent que possible** (toutes les minutes) les documents sur lesquels vous travaillez (`Ctrl-s`) . **Enregistrez systématiquement vos programmes** avant de les compiler. Car c'est la version *inscrite* sur le disque dur qui sera utilisée.
— *Organisation des fenêtres* : vous allez vite travailler avec une dizaine de fenêtres à la fois ; **évitez que certaines prennent tout l'écran**, et ajustez-les en créneaux pour que leurs bords figurent un escalier. Ainsi, vous pourrez aisément sélectionner celle dont vous aurez besoin. En bas de l'écran, une représentation miniaturisée de ces fenêtres est souvent affichée.
— *Encodages* : ne vous inquiétez pas si, au début, les accents ne correspondent pas à ce que vous attendez. Ce point complexe peut dérouter les néophytes, il se règle assez aisément : le principe général est de choisir toujours l'option UTF-8. Consultez `http://www.tuteurs.ens.fr/faq/utf8.html` si vous êtes pressé/e d'en savoir plus.
— Causez *hacker* : pas d'accents ni d'espaces dans les noms de fichier ou de dossier !
— *Messages d'erreur* : n'hésitez pas à adresser toute réponse impolie de l'ordinateur en tant que requête à un moteur de recherche. Parfois, la réponse vous éclairera.
— Si avez le choix entre une documentation (souvent anglaise) et sa traduction française, préférez la première, même si vous en maîtrisez mal la langue. Les traductions sont souvent truffées d'erreurs (cas le moins tragique : un $ devient un S, etc.).

2 Commandes les plus utiles

2.1 Généralités

Comme annoncé, les commandes sont saisies dans un terminal et conclues par un passage à la ligne (le «retour-chariot» signale au système la fin de la commande).

Saisissez exactement les commandes comme indiqué : sans espace superflu, en tenant compte des majuscules et minuscules.

Une commande est souvent suivie d'options (précédées d'un tiret) et de paramètres. Le `man` précise tout cela.

Exemple : `ls -l *jpg` : affiche de façon détaillée (option `-l`) les descriptifs de tous les fichiers se terminant par `jpg` (paramètre `*jpg`).

2.2 La commande ls

Elle donne beaucoup d'informations sur la chose invoquée (fichier ou dossier). C'est une des commandes les plus utilisées.

Elle dispose de plusieurs variantes (options) :

— `ls -l` (moins l, comme elle) affiche le contenu du dossier courant (utile pour connaître les propriétaires des fichiers, les droits d'exécution...).
— `ls -l file` : même chose pour le fichier nommé `file`.
— `ls -la` (liste tous les fichiers, y compris les cachés [ceux dont le nom commence par un point]).
— `ls -lRt` (récursion + ordre temporel inverse).

2.3 La commande less

Elle affiche page d'écran par page d'écran le contenu d'un fichier. C'est le substitut de l'ancienne commande `more`. Elle est un logiciel à elle toute seule.

Exemple : `less file` affiche le début du fichier dont le nom est *file*.

Quand on est dans le *less*, les touches suivantes ont les effets suivants :

— `q` : quitter l'affichage (quitte l'application *less*) .
— *barre d'espacement* : descendre d'une page.
— *retour-chariot* : descendre d'une ligne.
— `k` : remonter d'une ligne.
— `g` : remonter en première page.
— `G` : descendre à la dernière page.
— `/expr` : recherche l'expression *expr* dans le document.

Les commandes `head` et `tail`, elles-mêmes accompagnées d'options, sont proches de la commande `less` dans ses usages les plus courants : elles permettent de visualiser le début et la fin d'un fichier.

2.4 La commande cd

Abrégé de *change directory*, elle permet de naviguer dans l'arborescence de l'ordinateur.

— `cd` : seule, elle vous ramène *chez vous*. Ce qui est bien utile quand vous ne savez plus où vous êtes.
— Vous pouvez aussi l'utiliser de deux façons : relative et absolue (explicite).

2.4.1 Usages relatifs

Vous naviguez pas à pas ou en utilisant des raccourcis :

— `cd ledossierquiestdessous` : permet d'aller dans un dossier au niveau exactement en dessous de celui où vous êtes. Exemple : vous êtes «chez vous» (dossier `/home/dupont`) et vous voulez aller dans le dossier `fichierslatex` qui est au niveau `/home/dupont/fichierslatex`. Un `cd fichierslatex` (abusez de la complétion !) vous amène au niveau `/home/dupont/fichierslatex`.
— Remonter d'un niveau : `cd ..`
Si vous étiez au niveau `/home/dupont`, vous arriverez au niveau `/home`.
— Jouer avec les commandes précédentes. Un `cd ../durand/sondossierlatex` vous mène à `/home/durand/sondossierlatex` si vous partez de `/home/dupont`.
— Le tilde (∼) vous permet d'arriver directement chez un utilisateur (si vous en avez le droit). Exemple : `cd ~durand/sondossierlatex` vous amène à l'endroit prévu, d'où que vous partiez.

2.4.2 Chemins d'accès «explicites»

Ils supposent que soient précisés, à partir de la racine, les endroits où vous voulez aller. Exemples :

- `cd /home/durand/sondossierlatex`
- `cd /home/dupont/fichierslatex`
- `cd /usr/bin/perl`
- `cd /`

Les deux navigations sont évidemment compatibles. Exemple :

```
cd ~durand/sondossierlatex/solutionsmagiques
```

2.5 En cas de panique

- `pwd` (où suis-je ?). Cette commande vous dit «où vous êtes» dans l'arborescence : «chez» vous, chez un collègue, dans une zone web, au sein des fichiers structurels de Linux…
- `cd` Comme indiqué précédemment, cette commande simple vous ramène dans votre dossier principal.
- Relisez les points précédents de cette documentation. N'oubliez pas que l'informatique, contrairement au langage, est *monosémantique*.

2.6 Quelques autres commandes utiles

- `cp` (copier). Syntaxe : `cp fichierdepart darrivee` (2 arguments). Exemples :
  - `cp ../durand/sondossierlatex/intro.tex ~/fichierslatex/`
    copie le fichier *intro.tex* de *durand* dans le dossier *fichierslatex* de *chez soi*.
  - `cp ../durand/sondossierlatex/intro.tex .`
    Le `.` signifie «là où je suis». On l'oublie souvent.
- `mkdir` (make directory). `mkdir folder` crée un dossier du nom de *folder* en dessous de l'endroit où on est.
  Voir aussi : `rmdir, rm -r, rm -rf, mv`
- `chmod` (change mode). Utile pour rendre exécutable un programme (ce qui est indispensable pour les «cgi») :
  `chmod a+x leprogramme.pl` (a=all, x=eXecute).
  Si vous désirez que vos collègues puissent lire vos fichiers ou dossiers :
  `chmod -R g+r *` (`-R`=récursivement dans les sous-dossiers, g=group, r=read, *=tout).
- Autres commandes d'usage courant : `cat, chown, cut, date, file, find, grep, kill, rmdir, sort, ssh, tar, touch, uniq, wc, where, which, whoami...`
- Lancement de programmes : `perl`, `gcc`, `pdflatex`, etc.
- De nombreux logiciels clicables peuvent aussi être lancés par le biais des commandes associées. Exemple : `firefox, gedit`
  Quand un tel logiciel risque d'être utilisé longtemps, il est *conseillé* de le lancer en tâche de fond (`&`) pour qu'il ne paralyse pas la fenêtre du terminal :
  `gedit &`
  `evince unfichier.pdf &` etc.

3 Récursivité et empilage de commandes

3.1 Récursivité

La plupart des commandes sont *récursives* : elles permettent de travailler non seulement sur un fichier ou dossier, mais sur tout le contenu d'un dossier.

Exemples :

- `grep -R expression *` (très utile)
- `ls -lRa`
- `chmod -R` etc.

3.2 Succession et résultats de commandes

C'est la force d'Unix/Linux : cet OS fonctionne sur le concept de brique élémentaire. Vous emboîtez l'une après l'autre pour obtenir le résultat de votre choix, que vous pouvez rediriger comme bon vous semble.

3.2.1 Emboîtements de commandes

Le trait d'union entre deux commandes s'appelle le *pipe*. C'est le symbole `|`
On l'obtient avec les touches Alt_Gr-6 sur un PC, Alt-Maj-L sur un Mac.

Voici quelques exemples :

— `ls -l | less` : affiche page à page le contenu d'un dossier. Très utile s'il est volumineux.

— `cut -f2 file |sort |uniq -c |sort -rn` : compte les mots de la colonne 2 du fichier *file*, du plus fréquent au moins fréquent. Application : dénombrer les modalités d'un questionnaire sociologique. Détail des étapes :

1. On sélectionne la 2^e colonne du fichier *file* (`cut -f2 file`).
2. On trie le résultat (`sort`).
3. On repère les lignes identiques (`uniq`) et on les compte (option `-c`). Le `sort`précédent était nécessaire car `uniq` ne fonctionne que sur des fichiers triés.
4. On trie ce dernier résultat (`sort`) numériquement et non alphabétiquement (`-n`) du plus grand au plus petit (`-r` : reverse).

— `find . -iname \*utf8\* |grep -i exemple` pour rechercher dans l'arborescence, à partir de l'endroit où l'on est (.) tous les fichiers dont le nom contient la chaîne *utf8* (en majuscules ou minuscule) et la chaîne *exemple* sur la même ligne.

3.2.2 Usage du système de fichiers

Le résultat d'une commande (ou de plusieurs emboîtées) peut être redirigé vers des fichiers : `> file` (ou `>> file` si on veut éviter d'écraser le fichier *file* en cas de relance de la commande).

Exemple : `grep sub *pl | sort > mesroutines`

sélectionne toutes les lignes contenant le mot *sub* (qui définit une routine en Perl) dans les fichiers se terminant par *pl* (souvent des programmes Perl), les trie et copie le résultat dans le fichier *mesroutines*.

4 L'éditeur

Les logiciels dans lesquels nous pouvons *écrire* (saisir, modifier du texte) sont évidemment indispensables.

La majorité d'entre eux sont des *éditeurs*, gratuits, et fonctionnent avec des menus déroulants. La plupart d'entre eux sont multi plate-forme.

En voici une liste non ordonnée ni exhaustive :

`gedit, kwrite, kate, emacs, vi, vim, bbedit` (fabuleux, payant, sur Mac), `TeXnicCenter` (Windows), etc.

Pour travailler à partir du terminal : `joe`, très léger et très utile pour travailler à distance.

Pour finir, la commande qui vous permet de faire venir sur votre écran un logiciel Unix :

`ssh -Y login@machine.surlaquelle.jeveuxmeconnecter`

Ces premières pages devraient suffire pour une première prise en main de Linux. La page suivante est spécifique à Perl.

5 Unix et Perl

Il est conseillé de connaître des rudiments de Perl avant de lire cette partie, qui donne très succinctement quelques moyens de manipuler *Unix/Linux* via *Perl*.

5.1 Commande `system`

On peut appeler une commande Unix (en fait, un programme externe) à partir d'un programme Perl : `system ("la commande que je désire");`

Cet appel se produit au moment demandé et le programme Perl se poursuit une fois la commande externe achevée.

Cas classiques : `wget, ps2gif`, etc. Il est conseillé d'appeler ces commandes via leur chemin d'accès complet : `/bin/rm` mieux que `rm` !

5.2 Variables

Il arrive que l'on désire glisser dans une variable le résultat d'une commande Unix. En ce cas, on utilise des apostrophes inverses :

```
$ladatedelinstant= `date`;
```

5.3 Redirections

Il est utile de se rappeler la ligne suivante, fort utile pour travailler sur une série de fichiers :

```
open (L, ls |);
```

Le fichier L contient alors le résultat de la commande `ls`

Application : travailler sur tous les fichiers d'un dossier donné. Détail :

```
open (L, "ls | ");
while (<L>)
{chop;
$fichier=$_;
print "On va travailler sur le fichier $fichier \n";
open (F, $fichier);
while (<F>)
{traitement du fichier}
close (F);
}
close (L);
```

La suite de ce tutoriel vous sera donnée en cours...

Il vous est conseillé d'avoir toujours cet aide-mémoire *imprimé* sous la main lors de ce(s) cours.

Note : ce document a été écrit en LaTeX.