Lionel **Dricot**

Ubuntu efficace

Préface de Mark **Shuttleworth**

Avec la contribution de Roland **Mas**
et Jean-Marie **Thomas**

Remerciements à Benoit **Caccinolo**, Gilles **Fabio**,
Joyce **Markoll**, Nicolas **Perriault**,
Thierry **Stoehr** et Patrick **Tonnerre**

ISBN : 2-212-12003-6, ISBN 13 : 978-2-212-12003-5

EYROLLES

En ligne de commande...

Sous un aspect austère qui fait parfois peur, la ligne de commande cache une puissance insoupçonnée. Vous êtes sans doute persuadé que la ligne de commande est réservée à l'élite, que taper des commandes incompréhensibles n'est pas à la portée du commun des mortels. Vous avez entièrement raison... Mais si vous êtes en train de lire ce chapitre, vous n'êtes déjà plus dans le commun des mortels alors, autant se lancer à fond. Bienvenue dans les entrailles d'Ubuntu !

SOMMAIRE

MOTS-CLÉS

- terminal
- shell
- ligne de commande
- chmod
- apt-get

Qu'est-ce qu'une console ?

À l'origine, les ordinateurs prenaient beaucoup de place et on s'y connectait depuis une station composée d'un écran et d'un clavier, le tout pouvant être très éloigné de l'ordinateur lui-même. Cette station était appelé terminal.

Depuis, le terme terminal est resté pour désigner une interface permettant de dialoguer directement avec l'ordinateur. Le terme console est un synonyme de terminal. Dans ce livre comme dans le langage courant (ou du moins courant chez les Linuxiens), vous entendrez parler de la même façon de terminal ou de console.

Vous pouvez lancer un terminal depuis le menu *Applications>Accessoires>Terminal*.

Figure 14–1
Le terminal par défaut sous Ubuntu

Le but du terminal est de dialoguer directement avec l'ordinateur, de lui donner directement les ordres sans passer par des icônes ou des boutons.

Le rectangle noir clignotant symbolise le début de la ligne où vous pourrez taper une commande. Ces commandes seront ensuite interprétées par l'ordinateur.

Comme vous pouvez le constater, du texte est déjà affiché. Il s'agit du *prompt*. Ce texte apparaîtra toujours au début de chaque commande et est invisible à l'ordinateur. Il ne sert qu'à vous donner certaines informations.

Par défaut, le prompt est du type :

```
ploum@ubuntu:~$
```

COMPRENDRE Les « vrais » terminaux

Pour être tout à fait exact, le programme que nous venons de lancer n'est pas un vrai terminal. En effet, un vrai terminal était un écran tout noir où scintillaient des lettres blanches. Il s'agit donc en fait d'un émulateur de terminal, un logiciel propre à GNOME qui se conduit comme un terminal.
Linux fournit cependant de vrais terminaux. Par défaut, sous Ubuntu, ils sont au nombre de 6.
Pressez simultanément les touches *Ctrl+Alt+F1*.
Votre écran devient tout noir et vous demande votre login : il s'agit là d'un vrai terminal, terminal dont le nom est tty1 (tty pour *text teletype*). En faisant *Ctrl+Alt+F2*, vous arrivez sur tty2 et ainsi de suite jusque tty6. Vous pouvez entrer votre login et votre password (attention, le password ne s'affiche pas, c'est entièrement normal), vous obtiendrez ensuite un écran similaire au terminal de GNOME.
Tout ce qui est expliqué ici s'applique aussi bien à un vrai terminal qu'à l'émulateur de GNOME. En fait, ils sont techniquement indifférenciables.
L'écran n° 7 lui est un peu particulier. Au lieu d'un terminal, c'est un logiciel appelé X.org qui l'occupe. Ce logiciel, anciennement appelé Xfree ou Xfree86 permet d'afficher des images à l'écran. C'est lui que Gnome utilise pour afficher une interface un peu plus sympathique que du texte sur fond noir.
Pour retrouver GNOME, la combinaison de touches est donc *Ctrl+Alt+F7*.
Pour résumer, on peut donc dire que votre ordinateur dispose de 7 écrans. Sur les 6 premiers sont lancés des terminaux. Dans le 7e est lancé X.org. Dans X.org lui-même est lancé l'environnement GNOME. Dans GNOME, vous disposez de 4 bureaux et dans un de ces bureaux vous avez à présent un terminal...

Le premier mot (`ploum`) est le nom de l'utilisateur connecté. Le second celui de l'ordinateur sur lequel l'utilisateur est connecté. Ainsi, le prompt actuel nous dit déjà que je suis l'utilisateur `ploum` connecté sur l'ordinateur ubuntu. C'est très important si, par exemple, on est connecté à plusieurs ordinateurs via le réseau.

La séquence après les deux points est le répertoire dans lequel se trouve actuellement le terminal. Par défaut, il s'agit du répertoire ~ qui signifie mon répertoire personnel. C'est équivalent de `/home/ploum`.

Enfin, le dernier symbole me renseigne sur le type d'utilisateur. $ signifie un utilisateur normal. Un symbole # signifie un utilisateur administrateur.

Par exemple, le prompt suivant :

```
root@marvin:/home/ploum#
```

m'informe que je suis l'utilisateur `root` sur l'ordinateur `marvin`. Je suis dans le répertoire `/home/ploum` (qui n'est pas mon répertoire personnel, sinon je verrais un ~) et je suis un administrateur.

COMPRENDRE Interpréteur de commandes

Techniquement, la commande n'est pas de suite envoyée à l'ordinateur. Elle est d'abord traduite par un programme appelé l'interpréteur de commandes, aussi appelé shell.
C'est aussi le programme qui se charge de vous afficher le prompt.
Par défaut, le shell utilisé est bash mais il en existe bien d'autres (comme zsh, csh, ksh...).
Mais du coup, la syntaxe des commandes ne sera pas toujours la même entre deux shells, ceux-ci interprétant les commandes de deux manières différentes.
Bash étant le plus courant et le plus utilisé, il est conseillé de le garder pour le moment. zsh est fort proche de bash et propose certaines fonctionnalités en plus qui font que beaucoup d'utilisateurs de bash passent à zsh après un certains temps. csh et ksh eux sont fort différents et sont généralement utilisés pour des raisons historiques.

Taper une commande

Après le prompt, il va falloir taper des commandes. C'est via les commandes que l'on communique avec l'ordinateur. C'est d'ailleurs pour cela que la console est souvent appelée « ligne de commande ».

Une commande a toujours la même structure : un mot sans espace ni caractères spéciaux (la commande principale) suivi d'un espace et suivi des options. Attention, la casse est importante et doit toujours être respectée : Linux considère une minuscule et une majuscule comme deux caractères différents.

Il est donc important de comprendre que votre premier mot sera primordial, il s'agit de la commande principale. Si il y a des options ou des choses qui suivent la commande, celles-ci seront **toujours** séparées par un espace de la commande principale. C'est comme ça que le shell reconnaît la commande.

Chaque commande est en fait un programme à part entière !

Par exemple, dans votre terminal, tapez la commande suivante :

```
firefox
```

Pour valider une commande, il faut appuyer sur la touche *Entrée*. Tant que la touche entrée n'est pas enfoncée, la commande n'est pas envoyée et l'ordinateur ne sait même pas qu'il y a une commande.

Immédiatement, le programme Firefox se lance ! Vous venez de comprendre le principe d'une commande. En fait, à chaque fois que vous cliquez sur une icône, celle-ci lance une commande avec laquelle elle est associée. Utiliser le terminal permet donc de passer outre l'icône en entrant directement la commande.

Remarquez que tant que Firefox est lancé, vous ne savez plus rien taper dans la ligne de commande. En effet, le terminal est bloqué par Firefox.

Fermez Firefox : un nouveau prompt apparaît vous permettant de taper une nouvelle commande.

Le programme Firefox permet de recevoir des paramètres. Par défaut, si on passe un paramètre à Firefox, celui le considère comme une adresse à ouvrir.

Tapez donc :

```
firefox http://ubuntu.com
```

Validez en appuyant sur *Entrée*. Magie ! Firefox s'ouvre directement sur le site demandé. Le programme a donc bien lu l'argument de la commande.

VOCABULAIRE Option et argument

Pour être exact, nous ne passons pas ici une option mais un argument.
La différence entre les deux est assez subtile : un argument fournit au programme les données qu'il va traiter (ici une page web). Une option fournit juste une indication sur la manière de traiter les données ou de s'exécuter.
L'adresse web est donc un argument.
L'option -v est elle un paramètre : elle indique à Firefox comment traiter les données (ici ne rien faire avec l'argument et afficher uniquement mon numéro de version).
Par tradition dans les systèmes Linux, on nomme les options avec un tiret devant (voire un double-tiret). Les arguments eux n'ont rien de particulier.
Cependant, ce n'est pas une règle absolue.
L'ensemble des options et des arguments forme les paramètres de la commande.

Figure 14–2
Firefox lancé depuis un terminal. La ligne de commande est bloquée.

Un autre exemple pour Firefox, est l'option `-v`. Cette option demande d'afficher simplement la version de Firefox sans lancer le programme.

Tapez la commande :

```
firefox -v
```

Le résultat s'affiche dans votre console :

```
Mozilla Firefox 1.5.0.3, Copyright (c) 1998 - 2006 mozilla.org
```

Un nouveau prompt vous indique que la commande est terminée, vous pouvez taper autre chose.

Afin de bien comprendre le fonctionnement des options, lancez le programme Xeyes :

```
xeyes
```

Xeyes est un simple petit programme qui suit votre souris des yeux. Il possède cependant des paramètres que l'on peut changer comme la couleur. Le paramètre `-fg` (pour *foreground*, avant plan) permet de choisir la couleur des pupilles.

Fermez Xeyes pour retrouver la console et lancez la commande :

```
xeyes -fg red
```

Comme vous pouvez le voir, le paramètre `-fg` a ici besoin d'une valeur. On donne cette valeur directement après le nom de l'option, suivi d'un espace. Ce n'est pas une règle absolue : certains programmes utilisent le

ASTUCE L'indispensable touche Tab

La touche *Tab* est la touche représentant deux flèches situées juste au dessus de *Caps Lock*.
Lorsque vous utiliserez un terminal, vous ne pourrez bien vite plus vous en passer.
La touche *Tab* permet en effet de compléter automatiquement une commande voire parfois les arguments et paramètres de la commande.
Faites l'expérience : tapez dans un terminal la commande `fir`.
Appuyez ensuite sur la touche *Tab*.
Comme il n'existe qu'une seul commande qui commence par le mot `fir`, votre terminal en déduit que vous voulez taper firefox et complète automatiquement !
Magique !
Par contre, il existe plusieurs commandes commençant par fi. Si vous appuyez sur *Tab*, il ne se passera donc rien. Insistez et appuyez une seconde fois : la liste de toutes les commandes commençant par fi est affichée !
Il vous suffit alors de taper la lettre suivante (ici r) puis une fois encore *Tab*. firefox est affiché, on peut valider avec la touche *Entrée*.
Cela fonctionne aussi avec les arguments : imaginons que vous souhaitiez ouvrir le fichier index.html dans votre répertoire personnel avec Firefox.
Tapez fir suivi de *Tab* puis espace puis i (la première lettre de index.html) puis *Tab*. Si index.html est le seul fichier de votre répertoire à commencer par la lettre i, il sera automatiquement complété !

LOCALISATION Pages de man en français

L'anglais est la langue de l'informatique. Si l'on désire plonger dans une utilisation plus poussée comme l'est la ligne de commande, une compréhension basique de l'anglais se révélera vite indispensable.
Cependant, les pages man sont disponibles en français. Pour cela, installez simplement le paquet manpages-fr avec Synaptic.

signe égal (ce qui donnerait `-fg=red`) et d'autres, plus rares, n'utilisent rien du tout (ce qui donnerait `-fgred`).

Xeyes dispose aussi des options `-outline` et `-center`. Essayez donc :

```
xeyes -fg red -outline blue -center green
```

ASTUCE Man man, RTFM !

Bien sûr il est impossible de retenir toutes les options de tous les programmes ! Cette aide est affichée par le programme man (pour manuel). Chaque programme dispose en fait d'une aide très précise. Le programme man prend en argument le nom d'un autre programme. Il affiche alors dans la console toutes les informations sur la manière d'utiliser ce programme. Essayez :

```
man firefox
man xeyes
```

Pour défiler dans la page, utilisez les touches *Flèche haut* et *Flèche bas*. Pour revenir à la console, appuyez sur la lettre *Q* (comme « Quit »). D'ailleurs, pour savoir comment fonctionne le man, il suffit logiquement de faire ....

```
man man
```

Il est considéré comme très grossier de poser, à propos d'une commande, une question dont la réponse se trouve dans le man. Par exemple, poser la question « Comment changer la couleur des yeux dans xeyes ? » recevra sans aucun doute la réponse : RTFM, sigle pour *Read The F* Manual*, « Lis Le Foutu Manuel ».

Premiers pas en console

Se déplacer et consulter les fichiers

Il est primordial de bien comprendre la notion de position dans le système de fichiers. En effet, votre terminal vous permet d'entrer des commandes mais, en plus, il est positionné dans un répertoire donné.

Comme nous l'avons vu, ce répertoire est indiqué dans le prompt. Par défaut, lorsqu'on lance un terminal, le répertoire dans lequel nous sommes positionné est le répertoire personnel, symbolisé par ~. Le chemin complet de votre répertoire personnel est `/home/login` où `login` est votre login personnel.

Dans notre exemple, le répertoire personnel est donc `/home/ploum`.

Sous Ubuntu, le système entier est contenu dans la racine (*root* en anglais) symbolisé par `/`.

Ainsi, dans la racine se trouve un répertoire nommé `home`. Dans ce répertoire `home` se trouve un répertoire appelé `ploum`. Le résultat est donc `/home/ploum`.

Pour savoir dans quel répertoire on se trouve, on utilise la commande `pwd` :

```
pwd
```

En résultat, la console vous affiche bel et bien le répertoire dans lequel vous vous trouvez :

```
ploum@ubuntu:~$ pwd
/home/ploum
```

`pwd` signifie en fait *Print Working Directory* (affiche le répertoire de travail).

Pour changer de répertoire, nous allons utiliser la commande `cd` (*change directory*). La commande `cd` prend un argument : le répertoire dans lequel on veut se déplacer.

POUR ALLER PLUS LOIN

📖 I. Hurbain, E. Dreyfus, *Mémento Unix/Linux*, Eyrolles 2006.

Essayons de nous déplacer dans le répertoire racine puis dans le répertoire `home` puis de revenir dans le répertoire personnel :

```
ploum@ubuntu:~$ cd /
ploum@ubuntu:/$ pwd
/
ploum@ubuntu:/$ cd /home
ploum@ubuntu:/home$ pwd
/home
ploum@ubuntu:/home$ cd /home/ploum/
ploum@ubuntu:~$ pwd
/home/ploum
```

Remarquons que l'autocomplétion avec la touche *Tab* est disponible pour l'argument. Il suffit donc de taper `cd /h` puis de presser la touche *Tab* pour voir apparaître `cd /home/`.

Remarquez qu'ici nous avons chaque fois décrit le répertoire en partant de la racine. Cela porte le nom de *chemin absolu*.

Le chemin d'un répertoire peut aussi être décrit en relatif. Au lieu de partir de la racine, on va partir du répertoire dans lequel on se trouve actuellement.

Pour décrire un chemin relatif, on ne commence donc pas par le symbole de la racine `/`.

Deux autres symboles sont disponibles :

- `.` : le point signifie le répertoire actuel. La commande `cd .` ne change donc pas le répertoire.
- `..` : deux points sont le symbole du répertoire parent. La commande `cd ..` remonte donc d'un niveau.

Dans votre répertoire personnel, vous disposez d'un répertoire appelé `Desktop`. Ce répertoire est le contenu affiché sur votre bureau.

Pour nous y déplacer, nous tapons donc, depuis le répertoire personnel :

```
cd Desktop
```

Pour revenir dans le répertoire personnel :

```
cd ..
```

Le tout peut être combiné. Ainsi, saurez-vous prévoir le résultat de la commande suivante ?

```
cd ~/Desktop/../../ploum/./Desktop/..
```

Une fois dans un répertoire, on va évidemment vouloir afficher le contenu d'un répertoire.

ASTUCE cd sans argument

Si l'on ne donne aucun argument à `cd`, ce dernier se déplace automatiquement dans le répertoire personnel.
Les commandes suivantes sont donc parfaitement équivalentes :

```
cd
cd ~
cd /home/ploum
```

HISTORIQUE cd.. en DOS

Les anciens utilisateurs du système DOS auront tendance à taper `cd..` pour remonter dans le répertoire parent.
Cependant, n'oublions pas que sous Linux il est nécessaire de séparer le nom de la commande et son argument par un espace. `cd..` renverra donc une erreur. Il faut s'habituer à taper `cd ..` à la place.

RÉPONSE Dans quel répertoire suis-je ?

La commande part du répertoire personnel, symbolisé par ~.
Ensuite, elle mène dans le sous-répertoire `Desktop`. À ce moment, nous serons donc dans `/home/ploum/Desktop`.
`../..` signifie que l'on va remonter deux fois. On remonte donc jusque dans `/home`.
Là, on repart dans le répertoire `ploum`.
Le point est tout à fait inutile ici et dit que l'on reste dans le même répertoire.
On rentre enfin dans le sous-répertoire `Desktop` dont on prend le parent.
En final, cette commande tordue se contente de nous ramener dans notre dossier personnel...
pwd peut d'ailleurs le confirmer. Ça valait bien la peine !

Le contenu d'un répertoire est obtenu via la commande `ls` (pour *listing*).

```
ploum@ubuntu:~$ ls
Desktop
```

La commande m'informe donc que je n'ai, dans mon répertoire personnel qu'un seul fichier : Desktop (il s'agit en fait d'un répertoire).

`ls` peut prendre en argument le dossier dont on veut afficher le contenu.

Ainsi la commande :

```
ls /home
```

est équivalente à la suite

```
cd /home
ls
cd /home/ploum
```

Bien sûr, le chemin peut-être indiqué en relatif :

```
ploum@ubuntu:~$ ls ..
ploum steve
```

Je remarque donc que le répertoire `home` contient deux fichiers : ploum, mon répertoire personnel et steve. Il y a de fortes chances qu'il s'agisse du répertoire personnel de l'utilisateur steve.

La commande `ls` dispose de plusieurs options (visibles via `man ls`). Les plus intéressantes sont :

- `ls -l` : affiche chaque fichier avec plein de détails (permissions, propriétaire, date de modification).
- `ls -a` : affiche aussi les fichiers cachés.

TECHNIQUE Les fichiers cachés

Sous Linux, les fichiers ou les répertoires dont le nom commence par un point « . » sont des fichiers cachés.
Cela permet de stocker facilement des fichiers et des données sans obstruer le répertoire de l'utilisateur. Ainsi, vos e-mails sont stockés par Evolution dans le répertoire `.evolution` situé dans votre répertoire personnel.
Faites donc l'expérience : créez un fichier et appelez-le `.machin`. Vous remarquerez que ce fichier ne s'affiche pas si vous faîtes `ls` ni via le traditionnel navigateur de fichier.
Par contre, `ls -a` l'affichera.
Dans le menu *Édition>Préférences* de votre navigateur de fichier se trouve aussi une option *Afficher les fichiers cachés*.

- `ls -R` : affiche récursivement. Ainsi, si le dossier dont on affiche le contenu contient un sous-dossier, le contenu du sous-dossier sera lui aussi affiché et ainsi de suite.

Les options peuvent être combinées :

```
ls -la /home/ploum
```

affichera en détail (option `-l`) le contenu du répertoire `/home/ploum` et ce y compris les fichiers cachés (option `-a`)

Opérations sur les fichiers

Nous allons à présent effectuer quelques opérations sur les fichiers.

La commande `touch` permet, entre autres, de créer un nouveau fichier vierge. Elle prend en argument le nom du fichier à créer.

```
ploum@ubuntu:~$ touch essai
ploum@ubuntu:~$ ls
Desktop essai
```

Nous avons donc bien créé un fichier appelé essai dans notre répertoire personnel. Créons-en un nommé `essai2` dans le répertoire `Desktop`.

```
ploum@ubuntu:~$ touch Desktop/essai2
ploum@ubuntu:~$ ls Desktop/
essai2
```

Le fichier `essai2` devrait d'ailleurs apparaître sur le bureau.

La commande `cp` permet de copier un fichier. Elle prend deux arguments : le fichier à copier suivi de la destination. Si la destination est un répertoire, le fichier sera copié dans ce répertoire.

Ainsi,

```
ploum@ubuntu:~$ cp essai copie
ploum@ubuntu:~$ ls
copie Desktop essai
```

crée une copie du fichier `essai` et l'appelle `copie`.

En revanche,

```
ploum@ubuntu:~$ cp essai Desktop/
ploum@ubuntu:~$ ls Desktop/
essai essai2
```

crée une copie du fichier dans le répertoire `Desktop` mais lui laisse du coup son nom original.

On peut bien sûr copier dans un autre répertoire et en même temps modifer le nom de la copie :

```
ploum@ubuntu:~$ cp essai Desktop/copie2
ploum@ubuntu:~$ ls Desktop/
copie2 essai essai2
```

La commande `mv` déplace au lieu de copier. Le déplacement et le renommage d'un fichier sont en fait exactement la même opération :

```
ploum@ubuntu:~$ mv copie Desktop/
ploum@ubuntu:~$ ls
Desktop essai
ploum@ubuntu:~$ mv essai nouveau
ploum@ubuntu:~$ ls
Desktop nouveau
```

Enfin, on veut évidemment parfois effacer un fichier. Ici, nous utilisons la commande `rm` (pour *remove*). La commande `rm` prend en argument le nom du fichier à effacer. On peut passer plusieurs fichiers à la fois :

```
ploum@ubuntu:~$ cd Desktop/
ploum@ubuntu:~/Desktop$ ls
copie copie2 essai essai2
ploum@ubuntu:~/Desktop$ rm copie2
ploum@ubuntu:~/Desktop$ rm essai2 copie essai
```

Nous avons donc effacé les fichiers créés dans le répertoire `Desktop`. Attention : les fichiers sont directement effacés sans corbeille ni confirmation ! Un fichier effacé avec `rm` est irrémédiablement perdu.

La commande `rm` peut prendre en option le paramètre `-i` qui va vous demander confirmation avant d'effacer un fichier. Il faut répondre par `y` (*yes*) ou `n` (*no*). Si votre système est en français, il acceptera les réponses `o` (*oui*) et `n` (*non*).

```
ploum@ubuntu:~/Desktop$ touch aze
ploum@ubuntu:~/Desktop$ rm -i aze
rm: détruire fichier régulier vide `aze`? y
```

Pour les répertoires, les commandes `touch` et `rm` ne fonctionnent pas. À la place on va utiliser `mkdir` (pour créer un répertoire) et `rmdir` (pour effacer un répertoire vide).

Afficher les fichiers

Il existe deux types de fichiers sur un ordinateur : des fichiers binaires et des fichiers de type texte.

ATTENTION rm -rf /

Attention, la commande `rm` peut se révéler très dangereuse avec les options `-f` (ne pas demander confirmation et forcer l'effacement) et `-r` (effacer de manière récursive : si on rencontre un répertoire, effacer tout son contenu puis le supprimer). Ainsi, la commande `rm -rf /` efface le répertoire / (la racine) et tout son contenu. Cela signifie l'effacement complet de tout votre système ! En une seule commande si lancée avec les droits d'administrateur ! On comprend donc la nécessité des backups. Un proverbe bien connu est justement : « Il existe deux types d'administrateurs sous Linux : ceux qui ont déjà tapé par erreur la commande `rm -rf /` et ceux qui vont bientôt le faire ».
La commande `rm -rf ~` est non moins dangereuse car elle efface tous vos fichiers personnels ! Relisez donc toujours deux fois une commande qui comporte `rm` avant d'appuyer sur *Entrée*.

Les fichiers binaires sont des suites de 0 et 1 uniquement compréhensibles par un programme donné ou par l'ordinateur lui-même. Un fichier `.zip` ou un fichier musical `.ogg` est un fichier binaire. Sans un programme particulier, nous ne savons pas le lire et encore moins le modifier.

Cependant, beaucoup de fichiers sont de type texte : on peut les lire et les modifier très simplement.

Pour afficher un fichier texte dans votre console, il suffit d'utiliser la commande `cat` avec en argument le fichier à afficher.

Par exemple, nous savons à présent que Ubuntu est basé sur Debian. Il existe donc un fichier appelé `debian-version` qui informe sur quelle version de Debian est basée l'actuelle Ubuntu. Ce fichier se trouve dans le répertoire `etc`, répertoire situé à la racine :

```
ploum@ubuntu:~$ cat /etc/debian_version
testing/unstable
```

Ce fichier ne comporte donc qu'une seule ligne, ligne qui est affichée et qui comporte les mots « testing/unstable ».

Prenons le cas d'un long fichier. C'est par exemple le cas de la licence GPL située dans le répertoire `/usr/share/common-licenses` :

```
cat /usr/share/common-licenses/GPL
```

Ouhlala ! Quel blabla ! Le tout a défilé à la vitesse de l'éclair et nous n'avons pas eu le temps de tout lire.

Nous allons donc utiliser un lecteur de texte un peu plus évolué : less.

```
less /usr/share/common-licenses/GPL
```

Less affiche le texte complètement et vous pouvez vous déplacez en utilisant les touches *Flèche haut* et *Flèche bas*.

Si vous souhaitez trouver un mot particulier, tapez sur la touche / de votre clavier puis entrez le mot. Cherchons par exemple les occurrences du mot « GNU ». Remarquez que votre recherche s'affiche dans le bas de la fenêtre.

Appuyez sur *Entrée* pour valider la recherche. Toutes les occurrences du mot « GNU » sont surlignées. Pour passer à la suivante, appuyez sur la touche *n* (pour *next*). Vous pouvez de cette manière descendre jusqu'au dernier GNU, dans le bas de la page. Pour revenir à un GNU précédent, utilisez la touche *N*.

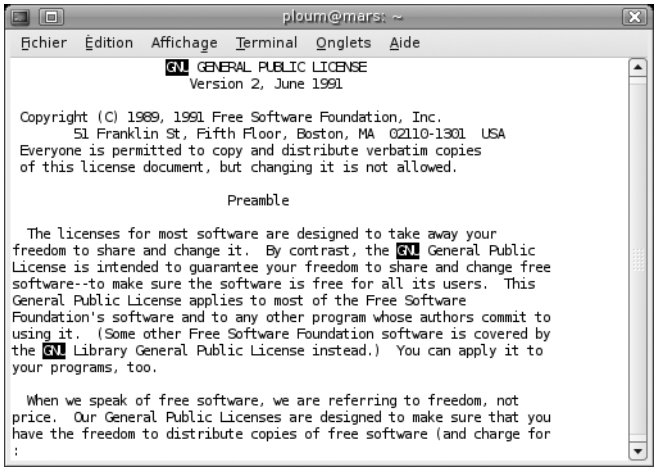

Figure 14–3
Une recherche sur les occurrences de GNU dans la GPL.

Enfin, pour quitter le lecteur de texte et revenir au prompt, appuyez sur *q* (*quit*).

CULTURE **more, less, outils GNU**

Toutes les commandes utilisées jusqu'à présent sont en fait des programmes à part entière. `cp`, `mv`, `ls` sont des programmes au même titre que Firefox ou Xeyes. Tout cet ensemble de petits outils fait généralement partie du projet GNU. On est tellement habitué à les utiliser et cela nous paraît tellement habituel qu'on en oublie tout le travail qui a été fourni pour concevoir ces outils.
Parfois, l'outil existait mais n'était pas un logiciel libre, cela l'empêchait donc de faire partie du projet GNU. Le lecteur de fichier historique s'appelait d'ailleurs « more » mais n'était pas libre. Le projet GNU a donc implémenté son propre lecteur de texte et, en forme de clin d'œil, l'a appelé « less » (*more or less* signifie plus ou moins en anglais).

Éditer du texte

Si l'on veut modifier du texte, il nous faut un éditeur de texte. L'éditeur de texte est le logiciel le plus important du Linuxien, son couteau suisse : un Linuxien sans éditeur de texte, c'est une voiture sans volant : on ne peut rien en faire (sauf aller tout droit) !

Il existe des éditeurs de texte graphiques (comme Gedit, accessible depuis le menu *Applications>Accessoires>Éditeur de texte*) et des éditeurs de texte en ligne de commande.

Commençons par lancer un éditeur de texte graphique, juste pour essayer :

```
gedit
```

Les éditeurs de texte suivent tous la même convention : si vous leur donnez en argument le nom d'un fichier, ils vont ouvrir ce fichier. Lancé sans argument, un éditeur de texte vous ouvrira alors un fichier vierge que vous devrez sauver.

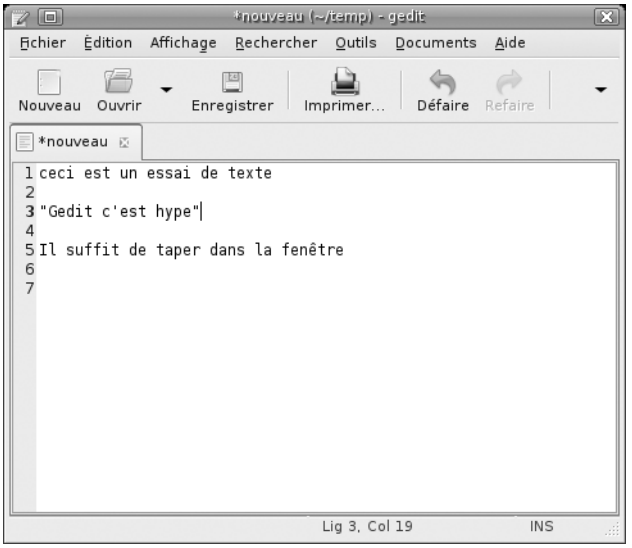

Figure 14–4
Gedit, éditeur de texte en mode graphique intégré à GNOME

Enregistrons le fichier sous le nom « texte.txt » et fermons Gedit. Passons à présent à un éditeur en mode texte. Un éditeur très simple est nano. Ouvrons le fichier texte.txt avec Nano.

```
nano texte.txt
```

Il suffit de taper pour ajouter du texte ou en supprimer.

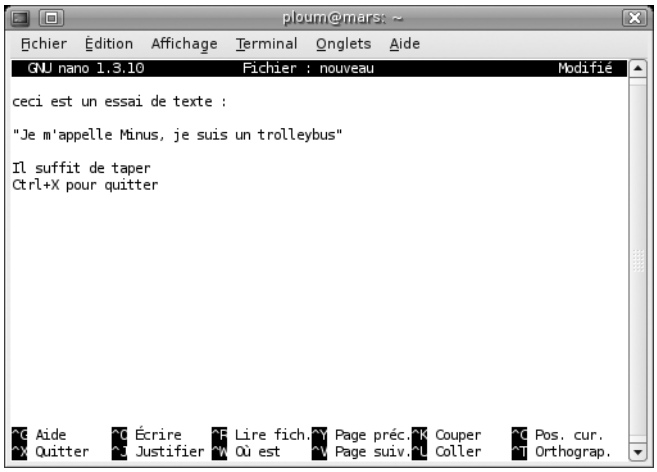

Figure 14–5
Nano, un éditeur simple en ligne de commande

Étant donné que la souris n'est pas disponible en ligne de commande, il va falloir utiliser les raccourcis clavier. La liste des raccourcis clavier se trouve dans le bas de votre écran. ^X signifie qu'il va falloir taper *Ctrl+X* pour quitter et retourner au prompt.

Nano est donc relativement simple d'utilisation mais peu puissant. Il existe une multitude d'éditeurs de texte sous Linux, certains sont très puissants d'autres plus simples. Les deux grands concurrents sont Vim (parfois appelé Vi bien que Vim soit une version améliorée de Vi) et Emacs.

Autant le dire tout de suite, l'utilisation de ces deux éditeurs est assez complexe au premier abord. Ainsi, si vous lancez :

```
vim texte.txt
```

Vous ne pourrez pas éditer le texte ! Il faudra d'abord vous placer en mode insertion en appuyant sur « i ». Une fois le texte édité, il faut sortir du mode insertion en pressant la touche *Échap*. La commande pour quitter Vim et revenir au prompt en sauvant les modifications est `:wq`. Si vous ne souhaitez pas sauvez les modifications, il faut taper `:q!`.

Avouons-le, l'utilisation de Vim paraît cryptique au premier abord ! L'utilisation d'Emacs le sera tout autant et vous obligera à contorsionner vos doigts sur le clavier en tapant des combinaisons *Ctrl+C* suivi de *Ctrl+X*. À cela s'ajoute le fait que Emacs peut gérer vos e-mails, jouer aux échecs, jouer le rôle de psychanalyste et nourrir votre chien (encore expérimental).

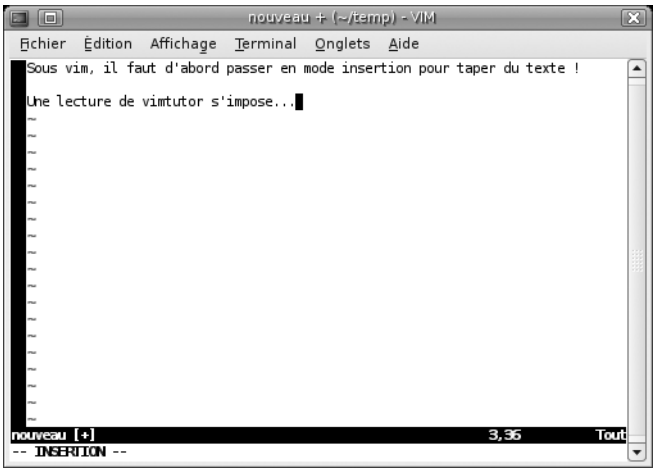

Figure 14–6
Vim, un éditeur très puissant

Maîtriser un éditeur de texte en console est donc tout un apprentissage.

CULTURE La guerre Vim/Emacs

Tout comme GNOME et KDE, la guerre Vim contre Emacs est une tradition du monde Linux. Il est de coutume de choisir un éditeur de texte, celui qu'on préfère, et d'en devenir un inconditionnel tout en lançant des blagues à l'encontre des personnes utilisant le second.
Cette bataille dégénère parfois en troll, longue discussion stérile et enflammée mais reste toujours bon enfant. Le troll Vim/Emacs est l'un des plus connus et des plus respectés, devançant largement le troll GNOME/KDE ou bien celui Ubuntu/Debian (bien que ce dernier fasse un retour en force).
Il n'est pas rare, lors d'une conférence, de voir organiser un match de football « utilisateurs de Vim contre utilisateurs d'Emacs ».
Cette sympathique rivalité est réellement un pilier historique de la culture Linux même si tout le monde sait à présent que Vim est le meilleur.
Jeu : saurez-vous deviner quel éditeur de texte utilise l'auteur de ces lignes ?

RESSOURCES Apprendre Vim et Emacs

Vim fournit un excellent tutoriel qui va vous permettre, en 20 minutes chrono, d'être fonctionnel et d'apprécier ce logiciel.
Dans un terminal, tapez simplement la commande :

```
vimtutor
```

Vimtutor va vous prendre en main et vous apprendre toutes les bases de Vim depuis le début.
Des tutoriels Vim sont également disponibles :

- http://people.via.ecp.fr/~alexis/formation-linux/vim.html
- http://sylvain.lhullier.org/guides/vi.html

Pour apprendre Emacs, il suffit de le lancer avec la commande :

```
emacs
```

(s'il n'est pas installé, installez le paquet du même nom).
Une fois dans l'éditeur, tapez les raccourcis : *Ctrl+H* puis *t*.
Un tutoriel avancé en français est également disponible à l'adresse :

- http://people.via.ecp.fr/~flo/2000/emacs-tut/

Administration en console

L'administrateur et root

Lorsque vous êtes connecté sur votre ordinateur, vous utilisez votre compte utilisateur. Par principe, un utilisateur ne peut pas modifier le

système. Il peut uniquement modifier ses fichiers et ses propres préférences.

Seul l'administrateur, habituellement appelé « root », peut modifier le système.

Dans beaucoup de distributions GNU/Linux, le compte `root` est un utilisateur complètement à part possédant son propre mot de passe et ses propres préférences. Son dossier personnel ne se trouve pas dans `/home/root` mais dans `/root` (cette précaution permet à l'administrateur d'avoir accès à son répertoire même dans le cas où le répertoire `/home` ne serait pas disponible).

Cependant, il est fortement déconseillé de lancer des applications sous le compte `root`. Ce dernier ne doit être utilisé que pour des tâches ponctuelles d'administration du système.

L'administrateur a donc généralement un compte normal, pour son travail quotidien.

La commande `su` (*super user*) permet de se connecter sous un autre nom d'utilisateur. Par exemple :

```
su steve
```

Me connectera sous l'utilisateur `steve` (pour peu qu'il existe sur mon système) après m'avoir demandé le mot de passe du compte. Pour quitter ce compte et redevenir moi-même, c'est très simple :

```
exit
```

Pour devenir administrateur, la logique voudrait donc qu'on tape simplement :

```
su root
```

En fait, cela équivaut à taper `su` sans argument.

Taper cette commande demande le mot de passe administrateur. Sous Ubuntu, ce mot de passe n'existe pas. Il n'est donc, par défaut, pas possible de se connecter sous le compte administrateur ! En effet, un compte administrateur c'est un mot de passe en plus, une complexité accrue et surtout la tentation de rester tout le temps connecté sur le compte administrateur. Pas de compte administrateur donc !

Mais comment faire ?

Un autre logiciel bien pratique existe : sudo. La commande `sudo` prend en fait en argument une commande et lance cette commande comme si l'utilisateur `root` l'avait lancée. Sudo va vous demander votre propre mot

de passe puis lancer la commande comme un administrateur. Lorsque Synaptic vous demande votre mot de passe au lancement, il s'agit en fait de Sudo !

Supposons par exemple que vous souhaitiez afficher le fichier de configuration de Sudo, justement. Ce fichier s'intitule `sudoers` et se trouve dans le répertoire `/etc`.

Un simple `cat` devrait suffire :

```
ploum@ubuntu:~$ cat /etc/sudoers
cat: /etc/sudoers: Permission non accordée
```

Notre utilisateur n'a pas le droit de lire la configuration de Sudo ! En effet, celle-ci fait partie du système et ne peut pas être lue par n'importe qui.

Nous devons donc la lire en tant qu'administrateur :

```
sudo cat /etc/sudoers
```

Sudo nous demande notre mot de passe (rien ne s'affiche quand on le tape, c'est normal) et, ensuite, `cat` affiche parfaitement le fichier demandé.

À chaque fois que nous voudrons taper une commande d'administration, il faudra donc taper `sudo` avant la commande.

Il arrive parfois que l'on souhaite effectuer beaucoup d'actions en mode administration. Plutôt que de taper `sudo` à chaque fois, nous allons lancer un shell root :

```
sudo -s
```

Le prompt qui s'affiche nous indique alors que nous sommes connecté en tant que `root` (notez le # à la fin du prompt):

```
root@ubuntu:~#
```

Attention ! Un shell en root peut être dangereux. N'y lancez que les commandes qui sont absolument nécessaires en root ! Fermez le shell dès qu'il n'est plus nécessaire (commande `exit`). Préférez utiliser systématiquement la commande `sudo`.

Quoiqu'il en soit, pas d'imprudences et bonne root...

Droits et permissions

Les utilisateurs et les groupes

Comme vous l'avez vu, vous êtes un utilisateur du système. Créer un nouvel utilisateur, par exemple pour votre ami, se fait avec la commande :

```
sudo adduser steve
```

et supprimer un utilisateur :

```
sudo deluser steve
```

Les utilisateurs peuvent appartenir à des groupes (un utilisateur peut faire partie de plusieurs groupes). Pour créer un groupe :

```
sudo addgroup nom_du_groupe
```

et supprimer un groupe :

```
sudo delgroup nom_du_groupe
```

Pour ajouter un utilisateur à un groupe existant :

```
sudo adduser steve nom_du_groupe
```

Un fichier appartient toujours à la fois à un utilisateur et à un groupe. Lorsque vous entrez la commande `ls -l`, vous pouvez savoir à qui appartient les fichiers affichés.

```
ploum@ubuntu:~$ ls -l
total 164
drwxr-xr-x 2 ploum ploum   4096 2006-07-22 17:17 Desktop
-rw-r----- 1 ploum amis 110557 2006-07-22 22:24 gthumb1.png
```

Ainsi, on peut voir que le répertoire `Desktop` appartient à l'utilisateur `ploum` et au groupe `ploum` (en fait un utilisateur définit un groupe dont il est le seul membre).

Le fichier `gthumb1.png` appartient aussi à ploum mais au groupe `amis`.

Les droits sur un fichier

Vous remarquez que devant le propriétaire/groupe se trouve une ligne assez cryptique : il s'agit des droits de chacun sur le fichier.

Pour chaque fichier, nous allons définir trois types de droits : les droits du propriétaire du fichier, les droits du groupe et les droits du reste du monde.

Chaque droit consiste en trois permissions distinctes : `r` si le fichier peut être lu (*read*), `w` si le fichier peut-être modifié (*write*) et `x` si le fichier peut être exécuté (*execute*).

La permission d'exécution n'a de réel sens que pour un programme. Une image, par exemple, ne sera jamais exécutée. Sur un répertoire, la permission d'exécution décrit si l'utilisateur a le droit d'entrer dans le répertoire en question.

Analysons dons ces 10 caractères :

```
drwxr-xr-x
```

Le premier caractère nous informe du type de fichier. Ici `d` signifie répertoire (*directory*).

Les trois suivants concernent les droits du propriétaire : `rwx` signifie donc que le propriétaire peut lire un fichier contenu dans le répertoire, peut écrire un fichier dans ce répertoire et peut entrer dans le répertoire.

Les trois suivants sont les droits du groupe : `r-x`. On remarque que le `w` manque. Un membre du groupe ne pourra donc pas écrire de fichier dans ce répertoire. Il pourra cependant entrer dans le répertoire et lire les fichiers qu'il contient.

Enfin, les trois derniers concernent le reste du monde : les utilisateurs qui ne sont pas le propriétaire et qui ne sont pas membres du groupe propriétaire. Ici, le reste du monde a les mêmes droits que le groupe.

Prenons notre second fichier :

```
-rw-r-----
```

Le premier caractère (-) nous indique qu'il s'agit d'un fichier normal. Ensuite, on peut voir que le propriétaire a le droit de lire et modifier l'image mais pas de l'exécuter (ce qui prendrait du sens si le fichier contenait un programme). Le groupe amis peut seulement lire le fichier (`r--`) tandis que le reste du monde n'a pas du tout accès au fichier.

Modifier les droits d'un fichier

Pour changer le propriétaire ou le groupe d'un fichier, on utilise la commande `chown` (*CHange OWNer*) ou la commande `chgrp` :

- `chown ploum fichier` change le propriétaire du fichier à `ploum`.

- `chown ploum:amis fichier` change le propriétaire à `ploum` et le groupe à `amis`.
- `chown :amis fichier` change le groupe à `amis` (ne pas oublier les « : »).

Cette dernière commande est équivalente à :

```
chgrp amis fichier
```

Maintenant que le propriétaire et le groupe nous conviennent, nous pouvons fixer les droits. La commande utilisée est `chmod`.

Cette commande prend comme premier argument le droit de modifier et en second le fichier.

Pour construire votre arguments, vous devez utiliser une lettre parmi `u` (*user*, le propriétaire), `g` (*group*, le groupe), `o` (*other*, le reste du monde) et `a` (*all*, tous en même temps).

Si vous souhaitez ajouter un droit, faites suivre la lettre choisie du signe `+`. Si vous voulez supprimer, utilisez le signe `-`. Finissez l'argument en ajoutant les droits.

Ce n'est pas encore clair pour vous ?

Un exemple sera sans doute plus parlant :

- `chmod a-x fichier` va supprimer les droits d'exécution à tout le monde.
- `chmod g+rw fichier` va permettre aux membres du groupe de lire et écrire le fichier.
- `chmod o-r fichier` va supprimer au reste du monde la possibilité de lire le fichier.

TECHNIQUE Le fameux chmod 777

Il existe une autre manière de construire un argument à la commande `chmod`.
Cela consiste à donner à la permission `r` la valeur 4, à la permission `w` la valeur 2 et à la permission `x` la valeur 1.
On obtient donc un chiffre compris entre 0 (aucune permission, `---` dans `ls -l`) et 7 (toutes les permissions, `rwx`).
On calcule ce chiffre pour le propriétaire, le groupe et le reste du monde.
Le nombre final peut être passé en argument à `chmod`.
Ainsi, pour donner les droits de lecture-écriture au propriétaire, de lecture uniquement au groupe et aucun droit au reste du monde, on fera :

```
chmod 640 fichier
```

Si l'on veut donner tous les droits possibles à tout le monde, on tapera donc :

```
chmod 777 fichier
```

chmod 777 est donc, sous forme de blague, le symbole de l'ouverture extrême et de la sécurité nulle puisque tout le monde peut faire n'importe quoi.

À SAVOIR La sécurité sous Linux

Linux a la réputation d'être plus sécurisé et moins sensible aux virus que d'autres systèmes comme Microsoft Windows. Pourquoi une telle réputation et qu'en est-il exactement ?

REMARQUE **Remarques et recommandations sur les droits et la sécurité**

Root et les utilisateurs

Une majorité des virus et des problèmes sous Windows proviennent du fait que les utilisateurs lancent volontairement un programme malveillant, pensant ouvrir un document ou un fichier. Ce programme peut donc infecter le système et en prendre contrôle si l'utilisateur est administrateur. Or, sous Windows, la tentation est très grande de travailler en permanence sur le compte administrateur. En effet, il est relativement malaisé de faire certaines actions lorsqu'on n'est pas administrateur. Sous Linux, vous êtes toujours un simple utilisateur. Lorsque les droits de root se font sentir, sudo vous demandera votre mot de passe.
Il est donc primordial de réfléchir deux fois plutôt qu'une lorsque votre mot de passe est demandé : Avais-je explicitement demandé une action nécessitant les droits root ? Quelle action va effectuer le système si je rentre mon mot de passe ? L'utilisation de sudo est donc une protection supplémentaire mais n'est pas une barrière infranchissable : restez prudent !

Droits d'exécution

Comme nous l'avons vu, chaque fichier sur votre système possède des droits d'exécution. Ces droits sont aussi une garantie supplémentaire.
En effet, rien n'est plus facile que de faire un programme malveillant. Imaginons que je vous envoie un programme qui effectue simplement la commande :

```
rm -rf ~
```

Comme ce programme n'aura pas les droits root, il ne pourra affecter le système. Bonne nouvelle. Cependant, il effacera tous vos fichiers personnels si vous le lancez ! Cela nous fait une belle jambe de savoir que le système et les autres utilisateurs ne sont pas touchés.
Par défaut donc, un fichier reçu ou téléchargé n'aura pas les droits d'exécution, vous ne pourrez pas le lancer. Avant d'attribuer des droits d'exécution, soyez certain de l'origine du fichier.

Failles de sécurité et virus

Même si l'utilisateur n'effectue aucune action, un ordinateur sous MS Windows a, une fois connecté à Internet, une espérance de vie d'une dizaine de secondes.
En cause, les failles de sécurité qui permettent d'exécuter à distance un programme sur un ordinateur.
Tout système possède ses failles et Linux ne fait pas exception. Cependant, Linux a gagné une certaine renommée par la rapidité avec laquelle elles sont corrigées.
Un autre aspect qui entre en jeu est la diversité qui compose le parc informatique Linux. Comme il existe plusieurs distributions GNU/Linux, que chacun est libre de personnaliser son système, d'utiliser le lecteur de courrier qu'il souhaite, il est extrêmement improbable d'arriver à trouver une faille de sécurité qui touche tout le monde en même temps.
C'est pourquoi on compare parfois le monopole de MS Windows à la culture intensive de pommes de terre en Irlande au 19ème siècle, les virus représentant le mildiou qui a anéanti toutes les récoltes et provoqué une famine considérable.

Quand Linux sera plus utilisé, il sera aussi envahi de virus

Les paragraphes précédents nous permettent de dire que la probabilité est plus faible que Linux connaisse un sort aussi terrible que celui de MS Windows actuellement.
Nous n'avons cependant pas de boule de cristal et seul l'avenir nous le dira. Mais même si Linux devait, dans 10 ans, connaître le sort de Windows, pourquoi ne pas profiter de ces 10 années de répit que nous offre Linux ?
Certains scientifiques auraient réussi à créer des virus pour Linux mais on ne reporte encore pour le moment aucune infection de masse. L'utilisation d'un antivirus est donc à ce jour tout à fait inutile dans le cas d'un ordinateur personnel et d'ailleurs mal comprise par les Linuxiens acharnés qui ne comprennent pas qu'on puisse vouloir se défendre contre son propre ordinateur.
Malgré tout, la vigilance et les règles élémentaires de sécurité restent de mise comme partout : on n'exécute pas n'importe quel fichier ou n'importe quelle commande aperçue au hasard d'un site web douteux !
On ne le répétera jamais assez : contre l'utilisateur lui-même, il n'existe aucun antivirus.

Signature et confiance

Afin de garantir la sécurité et la provenance d'un fichier, d'un programme ou même d'un e-mail, on utilise un système de signatures électroniques comme GPG.
Ainsi, lorsque vous téléchargez des paquets avec Synaptic ou apt-get, la signature de ceux-ci est vérifiée par rapport à une liste de clés de signatures connues et de confiance. Un paquet sans signature valide et reconnue vous affichera un avertissement. À vous alors de prendre vos responsabilités.
Les signatures des développeurs Ubuntu sont installées automatiquement avec le paquet ubuntu-keyring. Vous pouvez bien entendu importer les signatures de vos amis ou d'autres personnes de confiance. Pour gérer votre trousseau de clé GPG, vous pouvez installer le paquet seahorse.

📖 W. Lucas, *PGP & GPG, Assurer la confidentialité de ses e-mails et fichiers*, Eyrolles 2006.

Remarque Confidentialité et chiffrement

Le droit de lecture vous permet de contrôler qui a accès à la lecture d'un fichier sur votre ordinateur. Cependant cette technique a ses limites : elle est inefficace contre l'utilisateur root ou contre toute personne ayant accès physique à la machine et pouvant par exemple consulter les fichiers depuis un CD Vif.
De même, ne perdez jamais de vue que lorsque vous envoyez un courriel, celui-ci est comme une carte postale : tous les intermédiaires peuvent le lire avant qu'il soit remis à destination.
Lorsque vous désirez stocker ou envoyer des données confidentielles, il sera donc utile de les chiffrer. Le chiffrement de données peut se faire avec le même logiciel que pour les signatures : GPG. Celui-ci est automatiquement intégré à Evolution et, via l'extension Enigmail, à Thunderbird.
Notons que Linux offre bien d'autres possibilités de chiffrement et de sécurité.

- http://openpgp.vie-privee.org/
- http://doc.ubuntu-fr.org/applications/gnupg
- http://doc.ubuntu-fr.org/applications/mozilla-thunderbird_enigmail
- http://matrix.samizdat.net/crypto/gpg_intro/

Gestion des logiciels

Plutôt que d'utiliser Synaptic, vous pouvez très simplement gérer vos logiciels depuis la ligne de commande.

Pour installer le paquet lynx, navigateur web en mode texte, il suffira de taper la commande :

```
sudo apt-get install lynx
```

Pour le supprimer, un simple :

```
sudo apt-get remove lynx
```

Ces deux commandes vont vous demander des confirmations, vous informant parfois que d'autres paquets doivent être installés/supprimés suite à votre commande. Il suffit de lire les informations affichées et de suivre les instructions.

En fait, pour donner une image intuitive, vous êtes plus ou moins en train de faire à la main ce que Synaptic réalise de manière invisible lorsque vous lui demandez d'installer un logiciel.

Si vous n'êtes pas sûr du nom du paquet, vous pouvez faire une recherche avec la commande `apt-cache search` :

```
apt-cache search lynx
```

Une liste de paquet avec une brève description s'affiche.

Pour consulter une description plus détaillée d'un paquet donné, la commande est `apt-cache show` :

```
apt-cache show lynx
```

Enfin, pour savoir si le paquet est déjà installé et pour savoir quelle version sont disponibles, utilisez `apt-cache policy` :

```
apt-cache policy lynx
```

Avec un peu d'expérience, vous constaterez vite que la ligne de commande est bien plus rapide que Synaptic. N'oubliez pas que `man apt-get` est une grande source d'informations.

Si vous souhaitez installer directement un paquet au format `.deb` que vous avez téléchargé, utilisez la commande `dpkg` avec l'option `-i` (pour *installation*) :

```
sudo dpkg -i paquet.deb
```

NOTE apt-cache sans sudo

Notez que la commande `apt-cache` et ses dérivés ne modifie jamais le système. Il s'agit d'un outil purement informatif. C'est pourquoi il n'est pas nécessaire de le préfixer par `sudo`, contrairement à `apt-get`.

Utilisation avancée de la console

Gestion du système

Sur un système Ubuntu, comme sur la majorité des systèmes de ces dernières décennies, plusieurs programmes peuvent tourner en même temps.

En fait, chaque programme est lancé dans ce qu'on appelle un *processus*. Il serait donc intéressant à un moment donné d'établir la liste des processus actuellement actifs sur le système afin de monitorer ou de diagnostiquer un problème. Notons qu'en pratique un programme peut lui-même lancer plusieurs processus.

La commande `ps` affiche les processus lancés par votre utilisateur depuis le terminal duquel vous appelez la commande `ps`.

Vous verrez donc quelque chose ressemblant à :

```
ploum@ubuntu:~$ ps
  PID TTY          TIME CMD
14236 pts/5    00:00:00 bash
14237 pts/5    00:00:00 ps
```

Nous voyons que deux processus actuellement actifs ont été lancés depuis le terminal : `bash`, notre shell et la commande `ps` elle-même.

Lançons le programme Xeyes dans la console. Mais si nous le lançons, nous ne pourrons plus avoir accès à la console, celle-ci sera bloquée ! L'astuce consiste à ajouter le symbole & à la fin d'une commande.

Le résultat de la commande `ps` contient à présent le programme Xeyes :

```
ploum@ubuntu:~$ ps
  PID TTY          TIME CMD
14236 pts/5    00:00:00 bash
14608 pts/5    00:00:00 xeyes
14660 pts/5    00:00:00 ps
```

Remarquons que chaque processus dispose d'un numéro unique. C'est très important et cela permet de l'identifier.

Bon, c'est très bien tout ça mais si je pouvais voir tous les processus de mon système, ce serait bien mieux. Pour cela, il suffit d'ajouter l'option `-e` à `ps` :

```
ps -e
```

Une autre commande d'intérêt non négligeable est `top`. Tapez `top` dans un terminal.

TECHNIQUE Lancer un programme en arrière plan

Nous avons vu que lorsqu'un programme est lancé depuis une console, celle-ci est bloquée jusqu'à la fin du programme en question.
Il faut donc indiquer que nous voulons que le programme se lance mais que la console reste accessible. On dit alors que le programme est lancé en arrière-plan, la console reprenant le premier plan.
Pour cela, il suffit simplement de suffixer n'importe quelle commande par le symbole &.
Attention cependant de ne pas le faire avec des commandes qui requièrent une interaction dans la console. Lancer `apt-get` ou `nano` avec le suffixe & est donc à éviter. De manière générale, le signe & servira pour les applications graphiques comme Xeyes ou Firefox.
Essayez donc :

```
xeyes &
```

Si jamais l'application est déjà lancée et que la console est bloquée, il vous reste la possibilité d'effectuer la combinaison *Ctrl+z* dans la console bloquée.
Cette commande va geler temporairement le programme et vous redonner accès à la console. Il suffit ensuite de taper la commande `bg` (pour *background*) pour dégeler l'application tout en gardant une console accessible.
Essayez donc :

```
xeyes
```

La console est bloquée, tapez *Ctrl+z*.
La console est disponible mais le programme Xeyes est à présent bloqué. Il ne suit plus votre souris des yeux. Tapez donc :

```
bg
```

Et voilà...

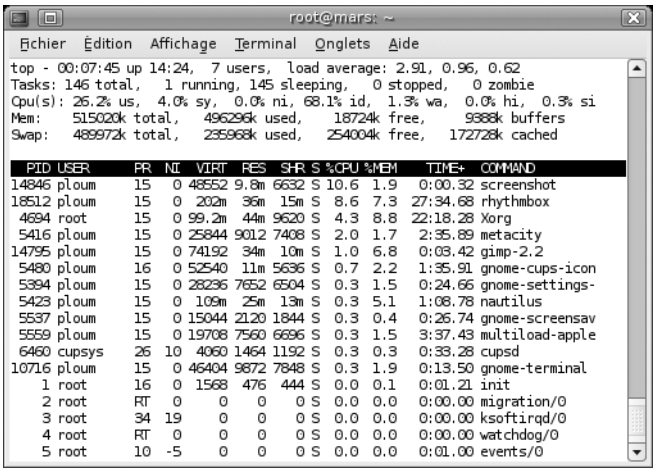

Figure 14–7
Top affiche les processus qui tournent

Top affiche les processus actuellement actifs par ordre décroissant de consommation de temps processeur. Les processus les plus gourmands sont listés en haut. On peut par exemple voir que le processus qui a effectué la capture d'écran prenait 10 % du temps processeur.

Si l'on appuie sur la touche *M* (*Shift+m*), le classement n'est plus en fonction du temps processeur mais par occupation de la mémoire RAM. Une bonne façon de voir quel programme ralentit tout votre système.

Pour quitter `top`, il suffit d'appuyer sur `q`.

Maintenant que nous avons pu explorer les processus, nous voulons pouvoir communiquer avec eux. Le plus généralement il s'agit de fermer un processus.

Comme vous l'avez constaté, chaque processus dispose d'un identifiant unique appelé PID.

Pour envoyer à un processus l'ordre de se fermer, il suffit d'appeler la commande `kill`. Le nom n'est-il pas magnifiquement explicite ? Imaginons que je souhaite interrompre le screenshot. Je vois que le PID est `14846`. Il me suffit donc de taper :

```
kill 14846
```

Parfois, je veux tuer tous les processus qui portent un nom donné. Dans ce cas, j'aurais employé la commande `killall` (tuer tout) :

```
killall screenshot
```

La commande `kill` envoie au processus l'ordre de se fermer. La commande `killall` envoie le même ordre mais à plusieurs processus si plusieurs portent le nom « screenshot ».

Cependant, il arrive qu'un processus n'obéisse plus aux ordres et soit complètement planté. Dans ce cas, on va devoir envoyer un ordre vraiment fort. En fait, on va tuer le processus sans même lui demander quoi que ce soit. Ce signal bien particulier porte le numéro 9 et doit être passé en option :

```
kill -9 14846
killall -9 screenshot
```

Attention ! Un `kill -9` est une action vraiment violente sur votre système qui peut résulter en des pertes de données. Ne l'utilisez pas à la légère !

TECHNIQUE Les signaux

Les processus peuvent réagir à différent signaux. Le programme kill sert à envoyer un signal choisi à un processus.
La liste des signaux disponibles est visibles avec la commande :

```
kill -l
```

Employé sans option, `kill` va envoyer le signal TERM au processus, lui demandant de se terminer.
L'option `-9` enverra elle le fameux signal KILL.
Le programme kill peut donc faire bien plus que simplement envoyer le signal KILL ! Cependant, les développeurs ont bien vite compris que dans 99 % des cas, ce serait le signal KILL que l'on voudrait envoyer et ils ont choisi le nom de la commande en conséquence.

Informations sur le matériel

La ligne de commande peut aussi être utile pour obtenir des informations sur votre système. Ainsi, la commande `lshw` va vous afficher une liste (très détaillée) du matériel contenu sur votre ordinateur.

De la même manière, la commande `lsusb` liste le matériel connecté via les portes USB et `lspci` le matériel disponible sur les ports PCI.

Avec un peu d'expérience, il vous sera facile de voir à quoi correspond chaque ligne et d'identifier le matériel de manière plus sûre qu'en se fiant à l'emballage !

Notons aussi le répertoire `/proc`, répertoire un peu particulier car il contient des informations propres au système. `/proc` n'est pas un répertoire classique et n'essayez pas d'écrire ou de modifier quoi que ce soit dedans.

Deux fichiers pourront vous intéresser dans ce répertoire : `cpuinfo`, qui contient des informations sur le processeur et `meminfo` qui contient les informations relatives à la mémoire vive.

Vous pouvez lire ces fichiers au choix avec `cat` ou avec `less`.

```
cat /proc/meminfo
cat /proc/cpuinfo
```

Aller plus loin

Il est très difficile, sans écrire un livre complet, de donner un aperçu de toute la puissance disponible grâce à la ligne de commande.

Que ce soit pour personnaliser son shell, pour rediriger et mixer les commandes grâce aux pipes et aux redirections ou écrire des scripts automatisant des tâches complexes, un terminal se révélera bien vite un outil indispensable.

La place manque, malheureusement, pour vous faire partager tout ce que nous aimons dans la ligne de commande. Cependant, nous vous recommandons chaudement la lecture de quelques liens.

1. http://people.via.ecp.fr/~alexis/formation-linux/shell.html
2. http://lea-linux.org/cached/index/Shell.html
3. http://www.linuxfrench.net/article.php3?id_article=983
4. http://lea-linux.org/cached/index/Dev-shell_script.html

Ce dernier lien est particulièrement bien réalisé et utile pour démarrer dans l'écriture de scripts.

La ligne de commande offre aussi tout une série de logiciels qui peuvent remplacer vos applications graphiques : mutt est un lecteur de courriels, lynx et links des navigateurs web, irssi un client IRC et bien d'autres que vous découvrirez sans doute au fil de vos pérégrinations. Installez-les avec Synaptic et essayez-les, cela vaut la peine !

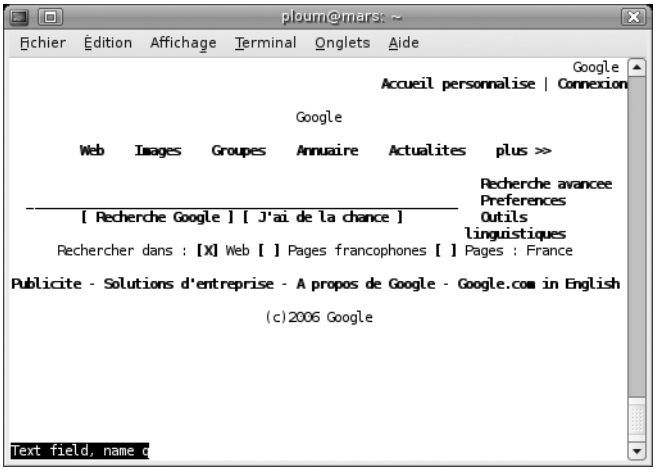

Figure 14–8
Le site Google affiché dans le navigateur links