Le DOS/Windows vers Linux HOWTO

Par Guido Gonzato, guido@ibogfs.cineca.it v1.2.2, 31 Octobre 1997

Ce HOWTO est dédié à tous les (bientôt ex- ?) utilisateurs de DOS et de Windows qui viennent de faire le grand saut et ont décidé de passer à Linux,le clone gratuit d'UNIX. Etant donné les similarités entre DOS et UNIX, le but de ce document est d'aider l'utilisateur à transposer sa connaissance de DOS et Windows à l'environnement Linux, pour devenir productif le plus rapidement possible. Adaptation française : Mark Hoebeke (Mark.Hoebeke@jouy.inra.fr).

Table des matières

1 Introduction

1.1 Est-ce Linux qu'il vous faut ?

Vous voulez passer de DOS/Windows à Linux ? Bonne idée, mais attention : cela peut ne pas être utile pour vous. A mon humble avis, il n'existe pas de "meilleur ordinateur" ou "meilleur système d'exploitation" : cela dépend de ce que l'on veut faire. Voilà pourquoi je ne crois pas que Linux soit la meilleure solution pour tout le monde, même s'il est techniquement supérieur à de nombreux systèmes d'exploitation du commerce. Vous tirerez un immense bénéfice de Linux si ce dont vous avez besoin, c'est de logiciels de programmation, de l'Internet, de TeX... de logiciels techniques en général, mais si vos besoins concernent d'abord des logiciels commerciaux, ou si vous ne vous sentez pas d'attaque pour apprendre et pour taper des commandes, passez votre chemin.

Linux n'est pas (pour l'instant) aussi facile à utiliser et à configurer que Windows ou le Mac, alors soyez prêt(e) à bidouiller pas mal. En dépit de ces avertissements, laissez-moi vous dire que je suis sûr à 100% que si vous faites partie du bon type d'utilisateur, vous trouverez en Linux votre Nirvana informatique. La balle est dans votre camp. Et rappelez-vous que, de toute façon, Linux et DOS/Windows peuvent coexister sur la même machine.

Pré-requis pour ce HOWTO : je supposerai que

- vous connaissez le commandes et les concepts de base de DOS;
- Linux, et le système X Window si possible, sont correctement installés sur votre PC;
- votre shell— l'équivalent du `COMMAND.COM`– est `bash`;
- vous comprenez que ce guide n'est qu'une introduction incomplète. Pour plus d'information, consultez le “Linux Installation and Getting Started” de Matt Welsh et/ou le “Linux User Guide” de Larry Greenfield (`sunsite.unc.edu:/pub/Linux/docs/LDP`).

  NdT. : ”Le Guide du Rootard”, une introduction approfondie à Linux en français est disponible sous divers format à l'URL `ftp://ftp.ibp.fr/pub/linux/french/docs/` .

Ce HOWTO remplace le vieux mini-HOWTO “From DOS to Linux - Quick !”. Remarquez également que, sauf indication contraire, toute l'information de ce travail concerne le vieux méchant DOS. Il y a une section Windows, mais, gardez à l'esprit que Windows et Linux sont totalement différents, au contraire de DOS qui est un parent éloigné d'UNIX.

1.2 Oui, c'est le bon choix. Dites-m'en plus.

Vous avez installé Linux et les programmes dont vous aviez besoin sur le PC. Vous vous êtes créé un compte (si ce n'est pas le cas, tapez `adduser` *sur le champ!*) et Linux tourne. Vous venez d'entrer votre nom et votre mot de passe, et, à présent vous fixez l'écran en pensant : “Et maintenant ?”

Ce n'est pas le moment de désespérer. Vous êtes tout près de faire les mêmes choses que celle que vous aviez l'habitude de faire sous DOS, et beaucoup plus. Si vous étiez sous DOS, au lieu d'être sous Linux, vous feriez quelques unes des tâches suivantes :

- exécuter des programmes et créer, copier, visualiser, effacer, renommer des fichiers;
- faire des CD, MD, RD et DIR sur vos répertoires;
- formater des disquettes et copier des fichiers à partir de/sur ceux-ci;
- vous occuper de vos `AUTOEXEC.BAT` et `CONFIG.SYS`;
- écrire vos propres fichiers .BAT et/ou programmes `QBasic` et/ou C/Pascal;
- le 1% restant;

Vous serez ravi de savoir que ces tâches peuvent être accomplies sous Linux de manière similaire que sous DOS. Sous DOS, l'utilisateur moyen utilise bien peu de la bonne centaine de commandes disponibles : la même chose, jusqu'à un certain point, reste valable pour Linux.

Quelques points à souligner avant de continuer :

- d'abord, comment en sortir. Pour quitter Linux : si vous voyez un écran en mode texte, appuyez sur CTRL-ALT-DEL, attendez que le système ait pris soin de ses entrailles et vous dise que tout est OK, éteignez alors le PC. Si vous travaillez sous le système X Window, appuyez d'abord sur CTRL-ALT-BACKSPACE, et ensuite sur CTRL-ALT-DEL. Ne *jamais* éteindre ou ré-initialiser le PC directement : cela peut endommager le système de fichiers;

- contrairement à DOS, Linux a des mécanismes de sécurité internes, dus à sa nature multi-utilisateurs. Des droits sont associés aux fichiers et aux répertoires, et certains d'entre eux ne peuvent donc pas être accessible par un utilisateur ordinaire; (cf. la section 2.3). Seul l'utilisateur dont le nom de login est "root" a tous les pouvoirs. (Ce gars là, c'est l'administrateur système. Si vous travaillez sur votre propre PC, c'est également vous qui serez root.) Au contraire, DOS vous laissera effacer tout le contenu de votre disque dur.
- vous êtes fortement encouragé(e) à expérimenter, jouer, essayer par vous-même : cela ne fera sûrement pas de mal. En case de besoin, voici ce que vous pouvez faire:
  - pour obtenir un peu d'aide sur les "commandes internes" du shell, tapez `help`;
  - pour avoir l'aide sur une commande, tapez `man commande` qui invoque la page de manuel ("man") relative à `commande`. Autrement, tapez `info commande` qui invoque la page d'informations relative à `commande` si elle est disponible. Info est un système de documentation basé sur l'hypertexte, pas toujours intuitif au départ. Enfin, vous pouvez essayer `apropos commande` ou `whatis commande` et taper 'q' pour quitter;
- une grande partie de la puissance et de la flexibilité d'UNIX provient des concepts simples de redirection et de "piping" bien plus puissants que sous DOS. Des commandes simples peuvent être chaînées les unes aux autres pour accomplir des tâches complexes. N'ayez pas peur d'utiliser ces commandes !
- conventions : <...> signifie que quelque chose doit être spécifié, alors que `[...]` désigne quelque chose d'optionnel. Par exemple :

  ```
  $ tar -tf <fichier.tar> [> fichier_de_redirection]
  ```

  `fichier.tar` doit être indiqué, mais la redirection vers `fichier_de_redirection` est optionnelle.
- à partir de maintenant, "LPM" signifie "veuillez vous reporter aux pages du manuel pour plus d'information".

1.3 Pour les impatients

Vous voulez vous faire la main ? Jetez un oeil à ceci :

```
DOS                     Linux                   Notes
------------------------------------------------------------------------------

BACKUP                  tar -Mcvf device dir/   completement different
CD dirname\             cd dirname/             presque la meme syntaxe
COPY file1 file2        cp file1 file2          idem
DEL file                rm file                 attention - pas de undelete
```

```
DELTREE dirname        rm -R dirname/         idem
DIR                    ls                     pas exactement la meme syntaxe
DIR file /S            find . -name file      completement different
EDIT file              vi file                je ne pense pas que vous
                                              aimerez
                       emacs file             ca c'est mieux
                       jstar file             un peu comme edit de dos
FORMAT                 fdformat,
                       mount, umount          une syntaxe assez differente
HELP command           man command            la meme philosophie
MD dirname             mkdir dirname/         presque la meme syntaxe
MOVE file1 file2       mv file1 file2         idem
NUL                    /dev/null              idem
PRINT file             lpr file               idem
PRN                    /dev/lp0,
                       /dev/lp1               idem
RD dirname             rmdir dirname/         presque la meme syntaxe
REN file1 file2        mv file1 file2         ne marche pas pour des fichiers
                                              mutliples
RESTORE                tar -Mxpvf device      une syntaxe differente
TYPE file              less file              beaucoup mieux
WIN                    startx                 completement differents !
```

Si vous avez besoin de plus que d'une table de commandes, veuillez vous reporter aux sections suivantes.

2 Les fichiers et les programmes

2.1 Les fichiers : notes préliminaires

Linux a un système de fichiers, ce qui signifie "la structure des répertoires et des fichiers qui s'y trouvent', très similaire à celui de DOS. Les fichiers ont des noms qui obéissent à des règles spéciales, sont disposés dans des répertoires, certains sont exécutables, et parmi ceux-ci la plupart ont des options. De plus, vous pouvez utiliser les caractères "joker", faire des redirections et des enchaînements de commandes. Il n'y a que quelques différences mineures :

- sous DOS, les noms de fichiers sont de la forme 8.3; ex. `SUFFITPA.TXT`. Sous Linux, on peut faire mieux. Si vous avez installé Linux en utilisant un système de fichiers de type ext2 ou umsdos, vous pouvez utiliser des noms de fichiers plus longs (jusqu'à 255 caractères), et qui comportent plus d'un point : par exemple `Ceci_est.un.TRES_long.nom.de.fichier`. Notez que j'ai utilisé aussi bien des majuscules que des minuscules : en fait...
- dans un nom de fichier ou une commande, les majuscules et les minuscules sont différenciées. Donc, `NOMDEFICHIER.tar.gz` et `nomdefichier.tar.gz` sont deux fichiers différents. `ls` est une commande, `LS` est une erreur;

– bien entendu, les utilisateurs de Windows 95 voudront utiliser les noms de fichiers longs sous Linux. Si un nom de fichier contient des espaces (ce qui est possible mais pas recommandé), vous devez entourer le fichier par des guillemets à chaque fois que vous y ferez référence. Par exemple:

```
$ # la commande qui suit cree un repertoire appelle "Mes vieux fichiers"
$ mkdir "Mes vieux fichiers"
$ ls
Mes vieux fichiers   bin   tmp
```

Certains caractères peuvent être utilisés, mais ne devraient pas l'être, comme : `!*$&`. Toutefois, je ne dirais pas comment.

– il n'y a pas d'extensions obligatoires pour les programmes comme .COM et .EXE, ou pour les fichiers batch comme .BAT. Les fichiers exécutables se terminent par une astérisque '*' lorsque vous faites la commande `ls -F`. Par exemple :

```
$ ls -F
lettre_a_Joe  cindy.jpg  cjpg*  Je_suis_un_repertoire/  mon_1er_script*  old~
```

Les fichiers `cjpg*` et `mon_1er_script*` sont exécutables— des “programmes”. Sous DOS, des fichiers de sauvegarde se terminent par .BAK, alors que sous Linux, ils se terminent par une tilde '~'. En outre, un fichier dont le nom commence par un point est considéré comme caché. Exemple : le fichier `.Je.suis.un.fichier.cache` ne figurera pas dans la sortie de la commande `ls`;

– Sous DOS, les options des programmes sont spécifiés avec `/option`, sous Linux il s'agit de `-option` ou `--option`. Exemple : `dir /s` devient `ls -R`. Remarquez que de nombreux programmes DOS, comme `PKZIP` ou `ARJ`, utilisent les options à la manière d'UNIX.

Vous pouvez maintenant vous rendre à la Section 2.4, mais, à votre place, je poursuivrais la lecture ici.

2.2 Les liens symboliques

UNIX dispose d'un type de fichier qui n'existe pas sous DOS : le lien symbolique. On peut le voir comme un pointeur sur un fichier ou sur un répertoire, et il peut être utilisé en lieu et place du fichier ou du répertoire sur lequel il pointe; cela ressemble aux raccourcis de Windows 95. Des exemples de liens symboliques sont `/usr/X11` qui pointe sur `/usr/X11R6`; `/dev/modem` qui pointe soit sur `/dev/cua0` ou `/dev/cua1`.

Pour créer un lien symbolique :

```
$ ln -s <fichier_ou_rep> <nom_du_lien>
```

Example :

```
$ ln -s /usr/doc/g77/DOC g77manual.txt
```

Maintenant, vous pouvez utiliser `g77manual.txt` au lieu de `/usr/doc/g77/DOC`. Les lignes apparaissent comme suit dans l'inventaire des répertoires:

```
$ ls -F
g77manual.txt@
$ ls -l
(diverses choses...)           g77manual.txt -> /usr/doc/g77/DOC
```

2.3 Droits et propriétés

Les fichiers et répertoires DOS ont les attributs suivants : A (archive), H (caché), R (lecture seule) et S (système). Seuls H et R ont un sens sous Linux : les fichiers cachés commencent par un point, et pour ce qui est de l'attribut R, lisez ce qui suit.

Sous UNIX, un fichier a des "droits" et un propriétaire qui appartient à un "groupe". Considérez cet exemple :

```
$ ls -l /bin/ls
-rwxr-xr-x  1  root  bin  27281 Aug 15 1995 /bin/ls*
```

Le premier champ contient les droits du fichier `/bin/ls`, qui appartient à root et au groupe bin. Si l'on ne tient pas compte du reste de l'information (il y a le livre de Matt pour cela), souvenez-vous que `-rwxr-xr-x` signifie (de gauche à droite) :

`-` est le type du fichier (- = fichier ordinaire, d = répertoire, l = lien etc...); `rwx` sont les droits pour le propriétaire du fichier (r=read=lecture, w=write=écriture, x=exécution); `r-x` sont les droits pour le groupe auquel appartient le propriétaire (lecture, exécution); (je n'aborderai pas le concept de groupe, tant que vous serez débutant(e), vous pouvez vous en passer ;-) `r-x` sont les droits pour tous les autres utilisateurs (lecture, exécution).

Ainsi, vous ne pouvez pas effacer le fichier `/bin/ls` à moins d'être root : vous n'avez pas les droits pour le faire. Pour changer les droits d'un fichier, la commande est :

```
$ chmod <quiXdroits> <fichier>
```

où 'qui' est `u` (user, c.à.d le propriétaire), `g` (groupe), `o` (other, les autres), 'X' est soit `+` ou `-`, 'droits' est `r` (lecture), `w` (écriture) ou `x` (exécution). NdT. : de plus, la lettre 'a' est un raccourci pour désigner à la fois 'u', 'g' et 'o'. Exemples :

```
$ chmod u+x fichier
```

ceci met les droits d'exécution pour le propriétaire du fichier.

```
$ chmod go-wx fichier
```

ceci enlève les droits d'écriture et d'exécution pour tout le monde sauf le propriétaire.

```
$ chmod ugo+rwx fichier
```

ceci donne les droits de lecture d'écriture et d'exécution à tout le monde.

```
# chmod +s fichier
```

ceci fabrique un fichier appelé "setuid" ou "suid" –un fichier que tout le monde peut exécuter avec les droits du propriétaire du fichier.

Un moyen plus court pour désigner les droits est d'utiliser des nombres : rwx-xr-x peut s'écrire 755 (chaque lettre correspond à un bit : --- vaut 0, --x vaut 1, -w- vaut 2, -wxvaut 3...). Cela paraît compliqué, mais avec un peu de pratique, vous comprendrez le concept. NdT. : la notation chiffrée des droits est faite en octal, donc 7 correspond au nombre binaire 111 où les 3 bits indiquant les permissions sont positionnés.

root, appelé également le super-utilisateur, peut changer les droits des fichiers quelque soit leur propriétaire. Il y aurait plus à en dire—LPM.

2.4 Traduction les commandes DOS vers Linux

A gauche, les commandes DOS; à droite, leurs équivalents Linux.

```
COPY :          cp
DEL :           rm
MOVE :          mv
REN :           mv
TYPE :          more, less, cat
```

Les opérateurs de redirection et d'enchaînement de commandes : <»>|

Les jokers : *?

nul : /dev/null

prn, lpt1 : /dev/lp0 ou /dev/lp1; lpr

- EXEMPLES -

```
DOS                                     Linux
---------------------------------------------------------------------

C:\GUIDO>COPY JOE.TXT JOE.DOC           $ cp joe.txt joe.doc
C:\GUIDO>COPY *.* TOTAL                 $ cat * > total
C:\GUIDO>COPY FRACTALS.DOC PRN          $ lpr fractals.doc
C:\GUIDO>DEL TEMP                       $ rm temp
C:\GUIDO>DEL *.BAK                      $ rm *~
C:\GUIDO>MOVE PAPER.TXT TMP\            $ mv paper.txt tmp/
C:\GUIDO>REN PAPER.TXT PAPER.ASC        $ mv paper.txt paper.asc
C:\GUIDO>PRINT LETTER.TXT               $ lpr letter.txt
C:\GUIDO>TYPE LETTER.TXT                $ more letter.txt
C:\GUIDO>TYPE LETTER.TXT                $ less letter.txt
C:\GUIDO>TYPE LETTER.TXT > NUL          $ cat letter.txt > /dev/null
```

n/a	`$ more *.txt *.asc`
n/a	`$ cat section*.txt | less`

Notes :

- * est plus malin sous Linux : * correspond à tous les fichiers, sauf les fichiers cachés; .* correspond à tous les fichiers cachés; *.* correspond seulement à ceux qui ont un '.' au milieu, suivi d'autres caractères; p*r correspond à la fois à 'par' et 'pour'; *c* correspond à la fois à 'secher' et 'bocal';
- lors de l'utilisation de `more`, appuyez sur ESPACE pour parcourir le fichier, 'q' ou CTRL-C pour quitter. `less` est plus intuitif et vous permet d'utiliser les flèches;
- il n'y a pas de `UNDELETE` alors *réflechissez-y à deux fois* avant d'effacer quoi que ce soit.
- en plus des `<` `>` `>>` du DOS, Linux connaît `2>` pour rediriger les messages d'erreur (stderr); de plus, `2>&1` redirige stderr vers stdout, alors que `1>&2` redirige stdout vers stderr; NdT.: stdout est "la sortie standard" (le lieu où sont envoyés tous les messages qui ne sont pas des messages d'erreur); stderr est la "sortie erreur standard" (lieu où sont envoyés tous les messages d'erreur).
- Linux a un autre caractère joker : le `[]`. Utilisation : `[abc]*` correspond à tous les fichiers commençant par a, b, c; `*[I-N123]` correspond aux fichiers se terminant par I,J,K,L,M,N,1,2,3;
- il n'y a pas de `RENAME` identique à celui du DOS; en fait, `mv` *.xxx *.yyy ne marchera pas. Vous pourriez essayer ce script simple; cf. la section 7.1 pour plus de détails.

```
#!/bin/sh
# ren: renomme plusieurs fichiers selon certaines regles

if [ $# -lt 3 ] ; then
  echo "usage: ren \"motif\" \"remplacement\" fichiers..."
  exit 1
fi

vieux=$1 ; nouveau=$2 ; shift ; shift

for file in $*
do
  new=`echo ${file} | sed s/${vieux}/${nouveau}/g`
  mv ${file} $vieux
done
```

  Attention: ce script ne se comporte pas comme le `REN` du DOS, puisqu'il utilise des "expressions régulières" que vous ne connaissez pas encore. En bref, si vous voulez simplement changer l'extension de certains fichiers, utilisez le comme suit: `ren "htm$" "html" *htm`. N'oubliez pas le signe `$`.

- utilisez `cp -i` et `mv -i` pour être averti(e) lorsqu'un fichier est sur le point d'être écrasé.
- utilisez `cp -i` et `mv -i` pour être averti lorsqu'un fichier risque d'être écrasé.

2.5 L'exécution de programmes : le multitâche et les sessions

Pour faire tourner un programme, entrez son nom comme vous le feriez sous DOS. Si le répertoire (section 3) où se trouve le programme est inclus dans le PATH (section 6.1), le programme démarrera. Attention : au contraire du DOS, sous Linux, un programme situé dans le répertoire courant ne sera pas lancé à moins que ce répertoire ne soit inclus dans le PATH. Solution : si `prog` est votre programme, tapez `./prog`.

Voici à quoi ressemble une ligne de commande typique:

```
$ commande -o1 -o2 ... -on par1 par2 ... parm < entree > sortie
```

où `-o1`, ..., `-on` sont les options de la commande, et `par1`, ..., `parm` ses paramètres. Vous pouvez spécifier plusieurs commandes sur la ligne de commande:

```
$ commande1 ; commande2 ; ... ; commanden
```

C'est à peu près tout en ce qui concerne l'exécution des programmes, mais on peut facilement franchir un pas de plus. L'une des raisons principales d'utiliser Linux est que c'est un système d'exploitation multitâche–il peut faire tourner plusieurs programmes (que nous appellerons processus à partir de maintenant) en même temps. Vous pouvez lancer des processus en arrière-plan et continuer à travailler sans aucune gêne. De plus, Linux vous permet d'avoir plusieurs sessions : c'est comme d'avoir plein d'ordinateurs à la fois pour travailler !

– Pour passer de la session 1..6 :

  ```
  $ ALT-F1 ... ALT-F6
  ```

– Pour commencer une nouvelle session sans quitter la session courante :

  ```
  $ su - <nom de login>
  ```

  Exemple :

  ```
  $ su - root
  ```

  Ceci est utile, par exemple, lorsque vous avez besoin de monter une disquette (section 4) : en temps normal, seul root peut le faire.

– Pour terminer une session :

  ```
  $ exit
  ```

  S'il y a des jobs arrêtés (lire plus loin), vous en serez averti(e).

– Pour lancer un processus en avant-plan :

  ```
  $ nomdeprogramme [-options] [parametres] [< entree] [> sortie]
  ```

– Pour lancer un processus en arrière-plan, ajoutez une éperluette '`&`' à la fin de la ligne de commande :

  ```
  $ nomdeprogramme [-options] [parametres] [< entree] [> sortie] &
  [1] 123
  ```

le shell identifie le processus à un numéro de job (ex. `[1]`; cf. ci-dessous), et à un PID (123 dans notre exemple).

- Pour voir combien il y a de processus :

```
$ ps -a
```

Ceci affichera une liste des processus en train de tourner.

- Pour tuer un processus :

```
$ kill <PID>
```

Vous pourrez avoir besoin de tuer un processus lorsque vous ne savez pas comment le quitter proprement... ;-). Parfois, un processus ne pourra être tué qu'en faisant ceci :

```
$ kill -15 <PID>
$ kill -9 <PID>
```

En plus de ceci, le shell vous permet d'arrêter ou de suspendre temporairement un processus, d'envoyer un processus en arrière-plan et d'amener un processus de l'arrière-plan vers le premier plan. Dans ce contexte, les processus sont appelés des "jobs".

- Pour voir combien il y a de jobs :

```
$ jobs
```

ici, les jobs sont identifiés par leur numéro de job, et non par leur PID.

- Pour arrêter un processus qui tourne au premier plan (cela ne marchera pas toujours) :

```
$ CTRL-C
```

- Pour suspendre un processus qui tourne au premier plan (même remarque) :

```
$ CTRL-Z
```

- Pour envoyer un processus suspendu en arrière-plan (il devient alors un job) :

```
$ bg <job>
```

- Pour amener un job vers le premier plan :

```
$ fg <job>
```

- Pour tuer un job :

```
$ kill <%job>
```

ou <job> peut être 1, 2, 3, ... (NdT. : ce paramètre est optionnel s'il n'y a qu'un seul job.) A l'aide de ces commandes, vous pouvez formater un disque, comprimer un lot de fichiers, compiler un programme, décomprimer et archiver tout à la fois tout en gardant une invite à votre disposition. Essayez de faire ceci sous DOS ! Et essayez avec Windows, rien que pour voir la différence de performance.

2.6 L'exécution de programmes sur des ordinateurs distants

Pour faire tourner un programme sur une machine distante dont l'adresse IP est `grossebecane.distante.fr`, faites :

```
$ telnet grossebecane.distante.fr
```

Après vous être connecté, démarrez votre programme favori. Il va sans dire que vous devez avoir un compte sur la machine distante.

Si vous disposez de X11, vous pouvez même faire tourner une application X sur un ordinateur distant, avec affichage sur votre écran X. Supposons que la machine X distante soit `grossebecane.distante.fr` et que `boite.linux.locale` soit votre machine Linux. Pour faire tourner un programme X résidant sur `grossebecane.distante.fr` à partir de `boite.linux.locale`, faites ce qui suit :

- démarrez X11, lancez un `xterm` ou un émulateur de terminal équivalent, et tapez :

  ```
  $ xhost + grossebecane.distante.fr
  $ telnet grossebecane.distante.fr
  ```

- après vous être connecté, tapez :

  ```
  grossebecane:$ DISPLAY=boite.linux.locale:0.0
  grossebecane:$ nomdeprogramme &
  ```

  (au lieu de `DISPLAY...`, vous pourriez avoir à écrire `setenv DISPLAY boite.linux.locale:0.0`. Cela dépend du shell distant.)

Et voila! Maintenant, `nomdeprogramme` va démarrer sur `grossebecane.distante.fr` et s'affichera sur votre écran. Toutefois, n'essayez pas ceci sur une liaison PPP, celle-ci serait trop lente pour être vraiment utilisable.

3 L'utilisation des répertoires

3.1 Les répertoires : notions préliminaires

Nous avons vu les différences entre les fichiers sous DOS et Linux. Pour ce qui est des répertoires, sous DOS, le répertoire racine est sous Linux c'est /. De même, les répertoires imbriqués sont séparés par sous DOS, et par / sous Linux. Exemple de chemin d'accès à un fichier :

```
DOS :     C:\ARTICLES\GEOLOGIE\MI_EOC.TEX
Linux :   /home/guido/articles/geologie/mi_eocene.tex
```

Comme d'habitude, `..` est le répertoire père, et `.` est le répertoire courant. Rappelez-vous que le système ne vous permettra pas de faire des `cd`, `rd`, ou `md` n'importe où. Chaque utilisateur démarre de son propre répertoire, appelé 'home' qui lui est attribué par l'administrateur système; par exemple, sur mon PC, mon répertoire home est `/home/guido/`.

3.2 Les droits sur les répertoires

Les répertoires disposent également de droits. Ce que nous avons vu dans la section 2.3 est aussi valable pour les répertoires (utilisateur, groupe et autres). Pour un répertoire, `rx` signifie que vous pouvez faire `cd` vers ce répertoire, et `w` signifie que vous pouvez détruire un fichier dans ce répertoire (si les permissions du fichier vous le permettent, bien sûr), ou effacer le répertoire lui-même.

Par exemple, pour empêcher les utilisateurs de jeter un oeil sur `/home/guido/texte/`:

```
$ chmod o-rwx /home/guido/texte
```

3.3 Traduction des commandes DOS vers Linux

```
DIR :          ls, find, du
CD :           cd, pwd
MD :           mkdir
RD :           rmdir
DELTREE :      rm -R
MOVE :         mv
```

- EXEMPLES -

```
DOS                                     Linux
---------------------------------------------------------------------

C:\GUIDO>DIR                            $ ls
C:\GUIDO>DIR FILE.TXT                   $ ls file.txt
C:\GUIDO>DIR *.H *.C                    $ ls *.h *.c
C:\GUIDO>DIR/P                          $ ls | more
C:\GUIDO>DIR/A                          $ ls -l
C:\GUIDO>DIR *.TMP /S                   $ find / -name "*.tmp"
C:\GUIDO>CD                             $ pwd
        n/a - cf note                   $ cd
        idem                            $ cd ~
        idem                            $ cd ~/temp
C:\GUIDO>CD \AUTRE                      $ cd /autre
C:\GUIDO>CD ..\TEMP\POUBELLE            $ cd ../temp/poubelle
C:\GUIDO>MD NVXPROG                     $ mkdir nvxprogs
C:\GUIDO>MOVE PROG ..                   $ mv prog ..
C:\GUIDO>MD \PROGS\TURBO                $ mkdir /progs/turbo
C:\GUIDO>DELTREE TEMP\POUBELLE          $ rm -R temp/poubelle
C:\GUIDO>RD NVXPROGS                    $ rmdir nvxprogs
C:\GUIDO>RD \PROGS\TURBO                $ rmdir /progs/turbo
```

Notes :

1. pour utiliser `rmdir`, le répertoire à effacer doit être vide. Pour effacer un répertoire et tout son contenu,

utilisez `rm -R` (à vos risques et périls).

2. le caractère '~' est un raccourci pour le nom de votre répertoire home. Les commandes `cd` ou `cd ~` vous emmèneront dans votre répertoire home, où que vous soyez; la commande `cd ~/tmp` vous emmènera vers `/home/votre_rep_home/tmp`.

3. `cd -` "défait" le dernier `cd`.

4 Disquettes, disques durs consorts

4.1 Gestion des périphériques

Vous n'y avez jamais pensé, mais la commande DOS `FORMAT A:` fait beaucoup plus de travail qu'il n'y paraît. En fait, lorsque vous lancez la commande `FORMAT` cela va : 1) formater physiquement le disque, 2) créer le répertoire A: (= créer un système de fichiers); 3) rendre le disque accessible à l'utilisateur (= monter le disque).

Ces trois étapes sont distinguées sous Linux. Vous pouvez utiliser des disquettes au format MS-DOS, bien que d'autres formats, et des meilleurs, soient disponibles—le format MS-DOS ne vous laissera pas utiliser les noms de fichiers longs. Voici comment préparer un disque (vous aurez besoin de démarrer une session en tant que root) :

– Pour formater une disquette classique 1.44 Mo (A:) :

```
# fdformat /dev/fd0H1440
```

– Pour créer un système de fichiers :

```
# mkfs -t ext2 -c /dev/fd0H1440
```

pour créer un système de fichiers MS-DOS, utilisez `msdos` au lieu de `ext2`. Avant d'utiliser le disque, vous devez le monter.

– Pour monter le disque :

```
# mount -t ext2 /dev/fd0 /mnt
```

ou

```
# mount -t msdos /dev/fd0 /mnt
```

Maintenant, vous pouvez accéder aux fichiers de la disquette. Lorsque vous avez fini, et avant d'extraire la disquette vous *devez* le démonter.

– Pour démonter le disque :

```
# umount /mnt
```

A présent vous pouvez extraire la disquette. De toute évidence, vous ne devez faire `fdformat` et `mkfs` que sur des disquettes non formatées, et pas sur des disquettes déjà utilisées. Si vous voulez utiliser le lecteur B:, adressez `fd1H1440` et `fd1` au lieu de `fd0H1440` et `fd0` dans les exemples ci-dessus.

Tout ce que vous aviez l'habitude de faire avec A: et B: peut maintenant être fait en utilisant `/mnt` à la place. Exemples :

```
DOS                                     Linux
---------------------------------------------------------------------

C:\GUIDO>DIR A:                         $ ls /mnt
C:\GUIDO>COPY A:*.*                     $ cp /mnt/* /docs/temp
C:\GUIDO>COPY *.ZIP A:                  $ cp *.zip /mnt
C:\GUIDO>A:                             $ cd /mnt
A:>_                                    /mnt/$ _
```

Si vous ne voulez pas vous embarrasser avec tous ces montages/démontages, utilisez `mtools`: il s'agit d'un ensemble de commandes parfaitement équivalentes à leurs homologues DOS, sauf qu'ils commencent par un 'm' : par ex., `mformat`, `mdir`, `mdel` etc.... Elles sont même capables de préserver les noms de fichiers longs, mais pas les droits. Utilisez ces commandes comme vous utiliseriez les commandes DOS sans vous poser d'autres questions.

Cela va sans dire, ce qui est vrai pour les disquettes vaut également pour d'autres périphériques; en l'occurrence, vous pouvez avoir envie de monter un autre disque dur, ou un lecteur de CD-ROM. Voici comment monter le CD-ROM :

```
# mount -t iso9660 /dev/cdrom /mnt
```

Ceci était la manière "officielle" de monter vos disques, mais il y a une astuce. Comme c'est un peu pénible de devoir être root pour monter une disquette ou un CD-ROM, voici comment faire pour permettre à tout utilisateur d'utiliser cette manière pour les monter :

– en tant que root, exécutez les commandes suivantes:

```
~# mkdir /mnt/a: ; mkdir /mnt/a ; mkdir /mnt/cdrom
~# chmod 777 /mnt/a* /mnt/cd*
~# # il faut s'assurer que le peripherique correspondant au CD-ROM est
~# # correct
~# chmod 666 /dev/hdb ; chmod 666 /dev/fd*
```

– ajoutez les lignes suivantes dans le fichier `/etc/fstab`:

```
/dev/cdrom      /mnt/cdrom  iso9660 ro,user,noauto          0       0
/dev/fd0        /mnt/a:     msdos   user,noauto             0       0
/dev/fd0        /mnt/a      ext2    user,noauto             0       0
```

A présent, pour monter une disquette au format DOS, au format ext2 et un CD-ROM :

```
$ mount /mnt/a:
$ mount /mnt/a
$ mount /mnt/cdrom
```

`/mnt/a`, `/mnt/a:`, et `/mnt/cdrom` peuvent maintenant être utilisés par chaque utilisateur. Rappelez -vous que d'autoriser n'importe qui de monter des disques de cette façon est un trou de sécurité béant, si cela vous préoccupe.

4.2 Les sauvegardes

Maintenant que vous savez comment manipuler les disquettes etc., voici quelques lignes montrant comment faire vos sauvegardes. Il y a plusieurs paquetages pour vous venir en aide, mais la moindre des choses que vous puissiez faire pour effectuer une sauvegarde multi-volumes est (en tant que root) :

```
# tar -M -cvf /dev/fd0H1440 /rep_a_sauvegarder
```

Assurez-vous d'avoir mis une disquette formatée dans le lecteur, et d'en avoir préparé plusieurs autres. Pour récupérer vos affaires, insérez la première disquette dans le lecteur et faites :

```
# tar -M -xpvf /dev/fd0H1440
```

5 Quid de Windows?

“L'équivalent” de Windows est le système graphique X11. Au contraire de Windows ou du Mac, X11 n'a pas été conçu pour être facile d'accès ou avoir une belle apparence, mais uniquement pour fournir des capacités graphiques aux stations de travail UNIX. Voici les différences principales:

- alors que Windows dispose du même “look and feel” dans le monde entier, ce n'est pas le cas pour X11: il est beaucoup plus configurable. L'apparence générale de X11 est donnée par un composant clé appelé le gestionnaire de fenêtres (window manager); il y en a tout un tas au choix. Le plus répandus sont `fvwm`, rudimentaire mais agréable et peu gourmand en mémoire, `fvwm2-95` et `The Next Level` qui donnent un goût de Windows-95 à X11, et beaucoup d'autres. Il y en a de vraiment esthétiques;
- votre gestionnaire de fenêtres peut être configure pour qu'une fenêtre se comporte comme dans, ahem, Windows: vous cliquez dessus et elle vient au premier plan. Une autre possibilité est de la faire venir au premier plan lorsque la souris s'y trouve. Cette fonctionnalité (“focus”) et beaucoup d'autres peuvent être modifiées en éditant un ou plusieurs fichiers de configuration. Reportez-vous à la documentation de votre gestionnaire de fenêtres;
- les applications X sont écrites à l'aide de bibliothèques spéciales (“widget sets”); comme il y en a plusieurs de disponible, les applications possèdent un look différent. Les plus élémentaires sont celles qui utilisent les composants (widgets) Athena (apparence 2-D; `xdvi`, `xman`); d'autres utilisent Motif

(`netscape`), d'autres encore utilisent Tcl/Tk, XForms, Qt etc... Certaines de ces bibliothèques, pas toutes, fournissent en gros le même "look and feel" que Windows;

- restons en là pour le look de X11, mais qu'en est-il de l'interactivité ? Malheureusement, toutes les applications se comportent différemment. Par exemple, si vous sélectionnez une ligne de texte à l'aide de la souris et que vous tapiez BACKSPACE, vous vous attendez à ce que la ligne disparaisse, n'est-ce pas ? Ceci ne marche pas pour les applications basées sur Athena, mais cela marche pour celles reposant sur Motif, Qt et Tcl/Tk;
- les barres de défilement, le changement de taille et l'iconification dépendent également du jeu de composants. Il y a trop de différences pour les mentionner toutes ici, voici donc quelques points particuliers. Dans des applications à base d'Athena, il est plus facile de déplacer les barres de défilement avec le bouton du milieu. Si vous n'avez pas de souris à trois boutons, essayez de maintenir les deux boutons enfoncés;
- les applications n'ont pas une icône par défaut, mais elles peuvent en avoir beaucoup. Cela dépend du gestionnaire de fenêtres. Le bureau s'appelle la "root window", et vous pouvez changer son apparence grâce à des applications comme `xsetroot` ou `xloadimage`;
- le presse-papiers ne peut contenir que du texte, et se comporte étrangement. Une fois que vous avez sélectionné du texte, il est copié automatiquement dans le presse-papiers: déplacez-vous ailleurs et appuyez sur le bouton central. Il existe une application, `xclipboard` qui fournit de multiples tampons pour le presse-papiers;
- le "drag and drop" est une option, uniquement disponible avec les applications X qui en tiennent compte.

Pour économiser de la mémoire, il vaut mieux utiliser des applications reposant sur les mêmes bibliothèques, mais ceci est difficile à mettre en pratique. Il existe un projet, appelé le K Desktop Environment qui a pour but de faire en sorte que X11 ait une apparence et un comportement aussi cohérents que Windows; il est encore en phase bêta précoce, mais, croyez-moi, il est impressionnant. Il fera rougir de honte l'interface Windows. Pointez votre navigateur sur `http://www.kde.org`.

6 Tailler le système à votre mesure

6.1 Les fichiers d'initialisation du système

Deux fichiers importants sous DOS sont `AUTOEXEC.BAT` et `CONFIG.SYS`, qui sont utilisés au démarrage pour initialiser le système, positionner quelques variables d'environnement comme PATH et FILES, et le cas échéant lancer un programme ou un fichier batch. Sous Linux, il y a plusieurs fichiers d'initialisation, dont certains que vous feriez mieux de ne pas modifier à moins de savoir exactement ce que vous faites. De toute façon, je vais vous dire lesquels sont les plus importants :

```
FICHIERS                        NOTES
```

```
/etc/inittab                    n'y touchez pas pour l'instant !
/etc/rc.d/*                     idem
```

Si tout ce dont vous avez besoin est de positionner votre PATH et d'autres variables d'environnement, ou de changer les messages de la connexion ou de lancer un programme après la connexion, regardez les fichiers suivants :

```
FILES                           NOTES

/etc/issue                      definit le message de pre-connexion
/etc/motd                       definit le message de post-connexion
/etc/profile                    definit PATH et d'autres variables etc.
/etc/bashrc                     definit les alias et les fonctions etc
                                (cf. ci-dessous)
/home/votre_home/.bashrc        definit vos alias + fonctions
/home/votre_home/.bash_profile  definit l'environnement +  lance vos progs
/home/votre_home/.profile       idem
```

Si ce dernier fichier existe (remarquez que c'est un fichier caché), il sera lu après la connexion et les commandes qu'il contient seront exécutées.

Exemple—regardez ce `.bash_profile` :

```
# Je suis un commentaire
echo Environnement :
printenv | less   # equivalent de la commande SET sous DOS
alias d='ls -l'   # facile pour comprendre ce qu'est un alias
alias up='cd ..'
echo "Je vous rappelle que le path est "$PATH
echo "Aujourd hui est le `date`"  # utilise la sortie de la commande 'date'
echo "Bonjour, "$LOGNAME
# Ce qui suit est une "commande shell"
ctgz() # Liste le contenu d'une archive  .tar.gz .
{
  for file in $*
  do
    gzip -dc ${file} | tar tf -
  done
}
# fin du .bash_profile
```

PATH et LOGNAME, vous l'avez deviné, sont des variables d'environnement. Il y en a beaucoup d'autres avec lesquelles on peut s'amuser; par exemple, LPM, pour des applications comme `less` ou `bash`.

6.2 Les fichiers d'initialisation des programmes

Sous Linux, tout peut virtuellement être taillé selon vos besoins. La plupart des programmes a un ou plusieurs fichiers d'initialisation que vous pouvez manipuler, souvent sous forme de `.nomdeprogrammerc` dans votre répertoire home. Les premiers que vous voudrez modifier sont :

- `.inputrc` : utilisé par `bash` pour définir les affectations des touches du clavier.
- `.xinitrc` : utilisé par `startx` pour initialiser le système X Window.
- `.fvwmrc` : utilisé par le gestionnaire de fenêtres `fvwm`. Un exemple se trouve dans: `/usr/lib/X11/fvwm/system.fvwmrc` Ndt. : c'est ce dernier qui est pris par défaut lorsque le fichier `.fvwmrc` n'existe pas.
- `.Xdefaults` : utilisé par `rxvt`, un émulateur de terminal pour X, et d'autres programmes. NdT. : ce fichier permet de spécifier des "ressources" (couleur de fond, taille de fenêtre, police de caractères) pour de nombreuses applications graphiques.

Pour tous ceux-ci, et les autres que vous rencontrerez tôt ou tard, LPM.

7 Un peu de programmation

7.1 Les shell-scripts : des fichiers .BAT dopés aux stéroïdes

Si vous utilisiez les fichiers .BAT pour créer des raccourcis pour de longues lignes de commande (c'est ce que je faisais beaucoup moi-même), ce but peut être atteint en insérant des lignes alias appropriées (cf. les exemples ci-dessus) dans `profile` ou `.profile`. Mais si vos .BAT étaient plus compliqués, alors vous aimerez le langage de commandes que vous offre le shell : il est aussi puissant que le QBasic, sinon plus. Il a des variables, des structures comme while, for, case, if... then... else, et un tas d'autres caractéristiques : il peut constituer une bonne alternative à un "vrai" langage de programmation.

Pour écrire un script, l'équivalent d'un fichier .BAT sous DOS, tout ce que vous avez à faire est d'écrire un fichier ASCII standard contenant les instructions, de le sauver, et de le rendre exécutable avec la commande `chmod +x <fichierscript>`. Pour l'exécuter, tapez son nom.

Quelques mots de mise en garde. L'éditeur du système est appelé `vi`, et, d'après mon expérience, la plupart des nouveaux utilisateurs le trouve très difficile à utiliser. Je ne vais pas expliquer comment l'utiliser, parce que je ne l'aime pas et que je ne l'utilise pas, alors voilà. Lisez le "Linux installation..." de Math Welsh p. 109 (vous feriez mieux de mettre la main sur un autre éditeur comme `joe` ou `emacs` pour X.) Je me contenterai de dire que :

- pour insérer du texte, tapez 'i' puis votre texte;
- pour quitter `vi` sans sauver, tapez ESC puis :q!
- pour sauver et quitter, tapez ESC puis :wq

L'écriture de scripts sous `bash` est un sujet tellement vaste qu'il nécessiterait un livre à lui seul, et je ne creuserai pas le sujet plus en avant. Je vous donnerai juste un exemple de shell-script, duquel vous pouvez extraire quelques règles de base :

```
#!/bin/sh
# echantillon.sh
# je suis un commentaire
# ne changez pas la premiere ligne, elle doit se trouver la
echo "Ce systeme est : ‘uname -a‘" # utilise la sortie de la commande ’uname’
echo "Je m appelle $0" # variables predefinies
echo "Vous m avez donne les  $# parametres suivants : "$*
echo "Le premier parametre est : "$1
echo -n "Comment vous appelez-vous? " ; read votre_nom
echo notez la difference : "salut $votre_nom" # delimitation par "
echo notez la difference : ’salut $votre_nom’ # delimitation par ’
DIRS=0 ; FILES=0
for file in ‘ls .‘ ; do
  if [ -d ${file} ] ; then # si le fichier est un repertoire
    DIRS=‘expr $DIRS + 1‘  # DIRS = DIRS + 1
  elif [ -f ${file} ] ; then
    FILES=‘expr $FILES + 1‘
  fi
  case ${file} in
    *.gif|*jpg) echo "${file} : fichier graphique" ;;
    *.txt|*.tex) echo "${file} : fichier texte" ;;
    *.c|*.f|*.for) echo "${file} : fichier source" ;;
    *) echo "${file} : fichier generique" ;;
  esac
done
echo "il y a  ${DIRS} repertoires et ${FILES} fichiers"
ls | grep "ZxY--!!!WKW"
if [ $? != 0 ] ; then # code de sortie de la derniere commande
  echo "ZxY--!!!WKW not found"
fi
echo "Ca suffit... tapez ’man bash’ si vous voulez plus d infos."
```

7.2 E-C-iez par vous-même

Sous UNIX, le langage système est le C, que vous l'aimiez ou le détestiez. Des tonnes d'autres langages (FORTRAN, Pascal, Lisp, Basic, Perl, awk...) sont également disponibles. NdT. : bien sûr, une version de l'environnement Java est également disponible pour Linux.

En partant du principe que vous connaissez le C, voici quelques indications pour ceux d'entre vous qui ont été trop gâtés par le Turbo C++ ou l'un de ses frères du DOS. Le compilateur C de Linux est appelé `gcc` et n'a pas toutes les fioritures qui accompagnent généralement ses correspondants DOS : pas de IDE, d'aide en ligne, de debugger intégré etc... C'est juste un compilateur brut en mode ligne de commande, très puissant et efficace. Pour compiler votre hello.c standard, vous ferez :

```
$ gcc hello.c
```

ce qui va créer un fichier exécutable appelé a.out. Pour donner un nom différent à l'exécutable, faites :

```
$ gcc -o hola hello.c
```

Pour linker un programme avec une bibliothèque; ajoutez l'option -l<nomdebiblio>. Par exemple, pour linker avec la bibliothèque mathématique :

```
$ gcc -o mathprog mathprog.c -lm
```

(L'option -l<nomdebiblio> oblige gcc à linker la bibliothèque /usr/lib/lib<nomdebiblio>.a; donc, -lm renvoie à /usr/lib/libm.a.)

Jusque là, tout va bien. Mais lorsque votre programme est composé de plusieurs fichiers source, vous aurez besoin de faire appel à l'utilitaire make. Supposons que vous ayez écrit un analyseur d'expressions : son fichier source s'appelle parser.c et inclut (#include) deux fichiers d'en-têtes, parser.h et xy.h. Puis vous voulez utiliser les routines de parser.c dans un programme, disons, calc.c, qui inclut (#include) à son tour parser.h. Quel souk ! Comment devez vous faire pour compiler calc.c?

Vous aurez à écrire un fichier appelé makefile, qui apprend au compilateur les dépendances entre fichiers objet et fichiers source. Dans notre exemple :

```
# Voici le makefile, utilise pour compiler calc.c
# Pressez la touche <TAB> aux endroits indiques!

calc: calc.o parser.o
<TAB>gcc -o calc calc.o parser.o -lm
# calc depend de deux fichiers objet: calc.o et parser.o

calc.o: calc.c parser.h
<TAB>gcc -c calc.c
# calc.o depend de deux fichiers source

parser.o:  parser.c parser.h xy.h
<TAB>gcc -c parser.c
# parser.o depend de trois fichiers source

# fin du makefile.
```

Sauvez ce fichier sous le nom de makefile et tapez make pour compiler votre programme; sinon, sauvez le sous le nom calc.mak et tapez make -f calc.mak, et, bien sûr, LPM.

Vous pouvez invoquer un minimum d'aide à propos des fonctions C, qui sont traitées dans les pages man, section 3, par exemple,

```
$ man 3 printf
```

Il y a des tas de bibliothèques disponibles; parmi les premiers que vous voudrez utiliser il y a `ncurses`, qui s'occupe des effets du mode texte, et `svgalib` pour faire du graphique. Si vous vous sentez de taille à attaquer la programmation X, procurez-vous `XForms` (`bloch.phys.uwm.edu/pub/xforms`) et/ou l'une des nombreuses bibliothèques qui font de l'écriture de programmes utilisant X un jeu d'enfant. Jetez un oeil sur `http://www.xnet.com/ blatura/linapp6.html` .

De nombreux éditeurs peuvent fonctionner comme un IDE; `emacs` et `jed` par exemple, sont capables de surlignage syntaxique, d'indentation automatique etc... Autrement, récupérez le paquetage `rhide` à partir de `sunsite.unc.edu:/pub/Linux/devel/debuggers/`. Il s'agit d'un clone du IDE de Borland, et il y a des chances que vous l'aimiez.

8 Le 1% qui reste

8.1 L'utilisation de tar & zip

Sous UNIX, il y a deux applications très répandues pour archiver et comprimer les fichiers. `tar` est utilisé pour construire des archives—il agit comme `PKZIP` mais ne comprime pas, il archive seulement. Pour fabriquer une nouvelle archive :

```
$ tar -cvf <nom_d_archive.tar> <fichier> [fichier...]
```

Pour extraire des fichiers d'une archive :

```
$ tar -xpvf <nom_d_archive.tar> [fichier...]
```

Pour afficher le contenu d'une archive :

```
$ tar -tf <nom_d_archive.tar> | less
```

Vous pouvez comprimer les fichiers à l'aide de `compress`, qui est obsolète et ne devrait plus être utilisé, ou `gzip` :

```
$ compress <fichier>
$ gzip <fichier>
```

qui créent un fichier compressé avec l'extension .Z (`compress`) ou .gz (`gzip`). Ces programmes ne peuvent comprimer qu'un fichier à la fois. Pour décomprimer, utilisez :

```
$ compress -d <fichier.Z>
$ gzip -d <fichier.gz>
```

LPM.

Les utilitaires `unarj`, `zip` et `unzip` (compatibles avec PK??ZIP) sont également disponibles. Des fichiers avec l'extension `.tar.gz` ou `.tgz` (archivés avec `tar`, puis compressés avec `gzip`) sont aussi communs dans le monde UNIX que le sont les fichiers .ZIP sous DOS. Voici comment afficher la liste du contenu d'une archive `.tar.gz` :

```
$ gzip -dc <fichier.tar.gz> | tar tf - | less
```

```
$ tar -ztf <file.tar.gz> | less
```

8.2 L'installation d'applications

Tout d'abord : l'installation de paquetages est le travail de root. Certaines applications Linux sont distribuées sous forme d'archives `.tar.gz` ou `.tgz`, spécialement préparées pour être décompressées à partir de / à l'aide de la commande suivante :

```
# gzip -dc <fichier.tar.gz> | tar xvf -
```

NdT. : le tar, version GNU, permet de faire directement :

```
tar xzvf <fichier.tar.gz>
```

Les fichiers seront décompressés dans le bon répertoire qui sera crée au passage. Les utilisateurs de la distribution Slackware disposent d'un programme à l'utilisation facile: `pkgtool`; un autre outil est `rpm` qui est disponible dans toutes les distributions grâce à Red Hat.

La plupart des paquetages ne devrait pas être installé à partir de /; typiquement, l'archive contiendra un répertoire appelé `nomdupaquetage/` et tout un tas de fichiers et de sous-répertoires sous `nomdupaquetage`. Une bonne règle est d'installer ces paquetages à partir de `/usr/local`. Il existe également quelques paquetages distribués sous forme de fichiers source C ou C++ que vous devrez compiler pour créer les binaires. Dans la plupart des cas, tout ce que vous aurez à faire est de lancer `make`. Bien évidemment, vous aurez besoin du compilateur `gcc`.

8.3 Les astuces indispensables

La complétion des commandes : l'appui sur <TAB> lors de la saisie d'une commande complétera la ligne de commande pour vous. Exemple: vous n'avez pas besoin de taper `gcc ceci_est_un_nom_long`; l'entrée de `gcc cec<TAB>` suffira. (Si vous avez d'autres fichiers qui commencent par les mêmes caractères, il faut fournir assez de caractères pour résoudre toutes les ambiguïtés.)

Le défilement en arrière : l'appui sur SHIFT + PAG UP (la touche grise) vous permet de remonter de quelques pages, le nombre étant déterminé par la quantité de mémoire vidéo dont vous disposez.

Le nettoyage de l'écran : si par hasard, vous faites `more` ou `cat` sur un fichier binaire, votre écran peut se retrouver rempli de caractères quelconques. Pour y remédier, tapez `reset` ou cette séquence de caractères : `echo CTRL-V ESC c RETURN`.

Le collage de texte : en mode console, cf. ci-dessous; sous X, cliquez et déplacez la souris en maintenant enfoncé le bouton de gauche; pour sélectionner le texte dans une fenêtre `xterm`, cliquez alors sur le bouton du milieu (ou sur les deux boutons en même temps si vous avez une souris à deux boutons) pour coller. Il existe également `xclipboard` (hélas, uniquement pour le texte); ne vous laissez pas dérouter par son temps de réponse très long.

Utilisation de la souris : installez `gpm`, un pilote de souris pour le mode console. Cliquez et déplacez la souris en maintenant enfoncé le bouton de gauche; pour sélectionner le texte, cliquez ensuite sur le bouton de droite pour coller le texte sélectionné. Cela fonctionne d'une console à l'autre.

Les messages du noyau : jetez un oeil sur `/var/adm/messages` ou `/var/log/messages` en tant que root pour voir ce que le noyau a à vous dire, y compris les messages de démarrage. La commande `dmseg` est aussi très pratique.

8.4 Quelques programmes et des commandes utiles

Cette liste est bien sûr le reflet de mes préférences et de mes besoins personnels. En premier lieu, où les trouver. Puisque vous savez tous comment surfer sur le Net et comment utiliser `archie` et `ftp`, je me contenterai de vous donner trois des plus importantes adresses pour Linux : `sunsite.unc.edu`, `tsx-11.mit.edu`, et `nic.funet.fi`. Veuillez utiliser votre site miroir le plus proche.

- `at` vous permet de faire tourner des programmes à une date et une heure données;
- `awk` est un langage simple mais puissant pour manipuler les fichiers de données (et pas seulement ça). Par exemple, soit `data.dat` votre fichier de données à plusieurs champs,

  ```
  $ awk '$2 ~ "abc" {print $1, "\t", $4}' data.dat
  ```

  affiche les champs 1 et 4 de chaque ligne de `data.dat` dont le second champ contient "abc".
- `cron` est utile pour effectuer des tâches de façon périodique, à des dates et heures spécifiées.
- `delete-undelete` font ce qu'indique leur nom (NdT.: efface-"déseffface");
- `df` vous donne des information sur le(s)s disque(s)s montés;
- `dosemu` vous permet de faire tourner plusieurs programmes DOS (mas pas tous)— y compris Windows 3.x, avec un peu de bidouillage;
- `file` <`nomdefichier`> vous dit ce qu'est `nomdefichier` (texte ASCII, exécutable, archive, etc.);
- `find` (cf. également la section 3.3) est l'une des commandes les plus puissantes et les plus utiles. On l'utilise pour trouver des fichiers qui satisfont plusieurs critères, et pour effectuer des actions sur ceux-ci. Une utilisation courante de `find` est :

  ```
  $ find <repertoire> <expression>
  ```

  où <expression> comprend des critères de recherche et des actions. Exemples :

  ```
  $ find . -type l -exec ls -l {} \;
  ```

trouve tous les fichiers qui sont des liens symboliques et montre leur destination.

```
$ find / -name "*.old" -ok rm {} \;
```

trouve tous les fichiers correspondant au motif et les efface, en vous demandant la permission au préalable.

```
$ find . -perm +111
```

trouve tous les fichiers dont les droits correspondent à 111 (exécutables).

```
$ find . -user root
```

trouve tous les fichiers appartenant à root. Il y à énormément de possibilités—LPM.

- `gnuplot` est un programme brillant pour tracer des courbes de nature scientifique;
- `grep` trouve des motifs de texte dans des fichiers, par exemple :

  ```
  $ grep -l "geologie" *.tex
  ```

  affiche les fichiers *.tex qui contiennent le mot “geologie”. La variante `zgrep` marche sur des fichiers “gzippés” (NdT.: compressés avec gzip). LPM;
- `tcx` comprime les fichiers binaires tout en les maintenant exécutables.
- `joe` est un excellent éditeur. En le lançant par la commande `jstar` vous obtiendrez les mêmes raccourcis clavier que sous WordStar et ses descendants, y compris les éditeurs DOS et ceux des langages Turbo de chez Borland.
- `less` est probablement le meilleur “browser” (NdT.: outil de visualisation) de fichiers texte, et, lorsqu'il est bien configuré, vous permet également de visualiser des fichiers gzippés, “tarés” (NdT.: archivés avec tar)et zippés;
- `lpr` <fichier> imprime un fichier en arrière-plan. Pour vérifier l'état de la file d'impression, utilisez `lpq`; pour enlever un fichier de la file d'impression, utilisez `lprm`;
- `mc` est un gestionnaire de fichiers épatant;
- `pine` est un chouette programme pour le courrier électronique;
- `script <fichier_script>` copie dans `fichier_script` ce qui apparaît à l'écran jusqu'à ce que vous faites la commandes `exit`. Utile pour déboguer;
- `sudo` permet aux utilisateurs d'accomplir certaines tâches de root (ex. : formater et monter des disques; LPM);
- `uname -a` vous donne de l'information sur votre système;
- `zcat` et `zless` sont utiles pour visualiser des fichiers texte gzippés sans les “dé-gzipper” (NdT.: les décomprimer avec gunzip ou gzip -d). Exemples d'utilisation :

  ```
  $ zless fichiertexte.gz
  $ zcat fichiertexte.gz | lpr
  ```

- Les commandes suivantes se révèlent souvent très pratiques : `bc, cal, chsh, cmp, cut, fmt, head, hexdump, nl, passwd, printf, sort, split, strings, tac, tail, tee, touch, uniq, w, wall, wc, whereis, write, xargs, znew`. LPM.

8.5 Les extensions courantes et les programmes qui s'y rattachent

Vous pouvez rencontrer des douzaines d'extensions. A l'exception des plus exotiques (ex. polices de caractères etc...), voici une liste de qui fait quoi :

- `1 ... 8`: pages du manuel. Procurez-vous `man`.
- `arj`: archive fabriquée avec `arj`. `unarj` pour désarchiver.
- `dvi`: fichier produit en sortie de TeX (cf. ci-dessous). A visualiser avec `xdvi`, et à transformer en PostScript (.ps) avec `dvips`.
- `gif`: fichier graphique. Rapatriez `seejpeg` ou `xpaint`.
- `gz`: archive fabriquée avec `gzip`.
- `info`: fichier d'information (sorte d'alternative à aux pages man). Procurez-vous `info`.
- `jpg, jpeg`: fichiers graphiques. Procurez-vous `seejpeg`.
- `lsm`: fichier de type Linux Software Map. Il s'agit d'un fichier ASCII pur contenant la description d'un paquetage.
- `ps`: fichier PostScript. Pour le visualiser installez `gs` et, optionellement, `ghostview`.
- `rpm`: paquetage Red Hat. Vous pouvez l'installer sur n'importe que système à l'aide du gestionnaire de paquetages `rpm`.
- `tgz, .tar.gz`: archive fabriquée avec `tar` et compressée avec `gzip`.
- `tex`: fichier texte à soumettre a TeX, un programme puissant de composition de documents. Installez le paquetage `tex`, disponible dans de nombreuses distributions, mais méfiez-vous de NTeX, qui contient des polices de caractères incorrectes et fait partie de certaines versions de Slackware.
- `texi`: fichier texinfo, peut produire à la fois de fichiers TeX et des fichiers info (cf. `info`). Procurez-vous `texinfo`.
- `xbm, xpm, xwd`: fichiers graphiques. Procurez-vous `xpaint`.
- `Z`: archive fabriquée avec `compress`.
- `zip`: archive fabriquée avec `zip`. Procurez-vous `zip` et `unzip`.

9 La fin, pour l'instant du moins

Félicitations ! Vous connaissez maintenant un peu d'UNIX et êtes prêt(e) à vous mettre au travail. Rappelez-vous que votre connaissance du système est encore limitée, et que vous devrez pratiquer Linux un peu plus pour être à l'aise. Mais si tout ce que vous aviez à faire était de rapatrier un paquet d'applications et de commencer à travailler avec, je parie que ce que j'ai inclus ici vous a suffi.

Je suis sûr que vous aimerez utiliser Linux et continuerez à en apprendre plus à son sujet—tout le monde le fait. Je fais également le pari que vous ne retournerez plus jamais sous DOS ! J'espère que j'ai réussi à me faire comprendre et que j'ai bien servi mes 3 ou 4 lecteurs.

9.1 Copyright

9.2 Avertissement

"Le DOS/Windows vers Linux HOWTO" a été écrit par Guido Gonzato, `guido@ibogfs.cineca.it`. Un grand merci à Matt Welsh, l'auteur du "Linux Installation and Getting Started", à Ian Jackson, l'auteur du "Linux frequently asked questions with answers", à Giuseppe Zanetti, l'auteur du "Linux", à toutes les personnes qui m'ont envoyé des suggestions, et particulièrement à Linus Thorvalds et GNU qui nous ont donné Linux.

Ce document est fourni "en l'état". Je me suis astreint à le rédiger avec toute l'exactitude possible, mais l'utilisation que vous en faites est a vos risques et périls. Je ne saurais en aucune façon être tenu pour responsable des dommages résultant de l'utilisation de ce travail.

Vos réactions sont les bienvenues. N'hesitez pas à me contacter pour toute question, suggestion, lettre d'insultes etc...,

Vivez heureux et amusez-vous avec Linux, Guido `=8-)`