Les commandes Linux

1. Généralités

L'interface utilisateur en mode texte s'appelle le SHELL. Il existe plusieurs SHELL. Le plus courant sous linux s'appelle BASH.

1.1. Forme d'une commande.

Une commande LINUX est de la forme : **`nom_de_la_commande [options] [arguments]`**

`nom_de_la_commande` est dans la plupart des cas en minuscules. Les crochets indiquent le caractère optionnel.
Les options sont en général précédées d'un ou deux " - " elles servent à modifier le comportement d'une commande. Les arguments sont les paramètres sur lesquels s'appliquent la commande.

Remarque importante : LINUX fait la distinction entre Majuscules et minuscules pour les commandes, noms de fichiers, répertoires, c'est parfois agaçant et source d'erreurs pour les débutants. Toto.txt, TOTO.txt et ToTo.txt sont des noms différents.

1.2. Saisie et exécution d'une commande simple.

A l'invite du Shell saisissez la commande. L'exécution est lancée après validation par la touche "Entrée". Avant validation vous pouvez corriger votre saisie en utilisant les touches "Flèche Gauche", "Flèche Droite", "Effacement Arrière" et "Suppr" pour modifier le texte.

```
Ex 1 : [root@linuxserveur /root]# ls
Ex 2 : [root@linuxserveur /root]# ls -l
Ex 3 : [root@linuxserveur /root]# rm -rf mes-docs
```

Dans l'exemple 1 ci-dessus la commande `ls` qui liste fichiers, répertoires, du répertoire courant est présentée dans sa forme la plus simple elle n'a pas de paramètre. Dans l'exemple suivant la même commande est présentée avec l'option longue (`l`) permettant d'obtenir plus d'informations. Dans le troisième exemple la commande `rm` de suppression est présentée avec deux options combinées `r` (récursive) et `f` (force) et elle prend en argument le répertoire à supprimer. Certaines commandes disposent de plusieurs dizaines d'options et en la matière il est quasi impossible d'être exhaustif. Il vaut mieux s'habituer à utiliser les différentes aides en ligne. Un premier niveau d'aide est constitué par l'option `-h` ou `--help` selon les commandes. Cette option vous donne la forme de la commande et un court commentaire sur chaque option.

1.3. Historique des commandes.

Les touches flèches haut et bas permettent de parcourir les dernières commandes de l'utilisateur, stockées dans le répertoire personnel dans le fichier `.bash_history`

1.4. Les noms de fichiers dans les commandes.

Il est possible d'utiliser des métacaractères (jokers) pour référencer des noms de fichiers. On trouve ainsi les métacaractères suivants :

* Une chaine quelconque de caractères.
? Un caractère quelconque.
[. . .] Un caractère quelconque parmi l'ensemble indiqué
[!. . .] Un caractère quelconque hors de l'ensemble indiqué
L'ensemble peut être une liste de caractères ou un intervalle (indiqué par "-").

Exemples :

```
[root@linuxserveur /dev]# ls sdd*
sdd sdd10 sdd12 sdd14 sdd2 sdd4 sdd6 sdd8
sdd1 sdd11 sdd13 sdd15 sdd3 sdd5 sdd7 sdd9

[root@linuxserveur /dev]# ls sdd?
sdd1 sdd2 sdd3 sdd4 sdd5 sdd6 sdd7 sdd8 sdd9

[root@linuxserveur /dev]# ls sdd[1-5]
sdd1 sdd2 sdd3 sdd4 sdd5
```

```
[root@linuxserveur /dev]# ls sdd[!1-5]
sdd6 sdd7 sdd8 sdd9

[root@linuxserveur /dev]# ls sdd[17]
sdd1 sdd7

[root@linuxserveur /dev]# ls sd[a-z][01]
sda1 sdb1 sdc1 sdd1 sde1 sdf1 sdg1
```

2. Quelques commandes usuelles.

Il existe des commandes internes au Shell (elles sont marquées dans la suite d'un astérisque) et des commandes externes. La commande `type` (voir ci-après) permet de savoir de quel type est une commande.

2.1. Quelques commandes de manipulation de répertoires.

pwd

La commande `pwd` affiche le nom absolu du répertoire courant, elle vous permet de connaître votre position dans l'arborescence.
syntaxe : `pwd`

```
[root@linuxserveur /]# cd /usr/X11/bin
[root@linuxserveur /]# pwd
/usr/X11/bin
```

ls

La commande `ls` liste le contenu de répertoires.
Syntaxe : `ls [-options] [répertoire1] [répertoire2]`
cette commande accepte un très grand nombre d'options pouvant être combinées.
`ls` liste les entrées du répertoire courant.
`ls -R` liste récursivement tous les répertoires.
`ls -a` liste toutes les entrées y compris les entrées cachées.
`ls -l` liste les entrées et affiche toutes les informations.
`ls -F` liste les entrées et ajoute / derrière les noms de répertoires.
`ls --color` liste les entrées avec une couleur différente suivant le type (fichier exécutable, texte, répertoire, liens ...)

cd (*)

La commande `cd` permet le positionnement sur un répertoire
syntaxe : `cd [répertoire]`

`cd /` positionnement à la racine
`cd ..` positionnement sur le répertoire parent
`cd /usr/X11R6/bin` positionnement depuis la racine sur le répertoire /usr/X11R6/bin .
`cd ../lib` positionnement à partir du répertoire parent sur le répertoire `/lib`.
`cd cd~` ou `cd $HOME` positionnement depuis n'importe où sur le répertoire personnel.

`cd -` positionnement sur le répertoire précédent.

mkdir

La commande `mkdir` permet la création de répertoires. Chaque répertoire contient deux entrées : le répertoire lui-même (.) et le répertoire parent (..).
syntaxe : `mkdir répertoire1 [répertoire2] [répertoireN]`

`mkdir perso` créé le répertoire perso dans le répertoire courant
`mkdir ../perso` créé le répertoire `perso` dans le répertoire parent
`mkdir /root/docs/perso` créé le répertoire `perso` dans le répertoire /root/docs

rmdir

`rmdir perso` supprime le répertoire `perso` du répertoire courant
`rmdir ../perso` supprime le répertoire `perso` du répertoire parent
`rmdir /root/docs/perso` supprime le répertoire `perso` du répertoire /root/docs

La commande `rmdir` permet la suppression de répertoires vides.
syntaxe : `rmdir répertoire1 [répertoire2] [répertoireN]`
N.B. pour supprimer un répertoire non vide voir la commande `rm`

2.2. Quelques commandes de manipulation de fichiers.

cat

La commande `cat` permet de visualiser le contenu d'un fichier texte.

Syntaxe : `cat fichier fichier1 [fichier2]`

Exemple d'affichage d'un fichier :

```
[root@linuxserveur /root]# cat addition2.java
class Addition2
{public static void main( String args[])
{int A=10;
A=A*2;
System.out.println("Le résultat est égal à : "+A);}
```

N.B. Pour savoir si un fichier peut être affiché voir la commande `file`.

more ou less

Ces commande permettent d'afficher un fichier page par page. On fait défiler le fichier ligne par ligne en appuyant sur la touche "`Entrée`", page par page avec "`Espace`", on revient en arrière avec "`b`" et on quitte l'affichage avec "`q`". Le pourcentage qui apparaît sur la dernière ligne indique la position actuelle par rapport à la totalité du fichier.

```
[root@linuxserveur linux_logo-2.12]# more README
/*-------------------------------------------------------------------------*\
LINUX_LOGO 2.12 -- Shows a Logo With some System Info - 30 October 1998
by Vince Weaver (weave@eng.umd.edu, http://www.glue.umd.edu/~weave )
http://www.vince.weaver.org/
SUPPORTS Linux (Intel, m68k, Alpha, Sparc) and some non-Linux OS's
\*-------------------------------------------------------------------------*/
. . .

The program itself has grown more "feature-full" as people around the
--More--(7%)
```

N.B. Pour savoir si un fichier peut être affiché voir la commande `file`.

file

La commande file permet d'obtenir la nature du ou des fichiers spécifiés.
syntaxe : `file fichier1 [fichier2] [fichierN]`

```
file x11.ps

x11.ps: PostScript document text conforming at level 2.0
file p*.* g??.c

p.ps:     PostScript document text conforming at level 2.0
gtk.c:    International language text
file *

progr.class:     compiled Java class data, version 45.3
Desktop:         directory
Mail:            directory
Printing-Usage:  PostScript document text conforming at level 2.0
addition2.class: compiled Java class data, version 45.3
addition2.java:  International language text
core:            ELF 32-bit LSB core file of 'fvwm' (signal 11), Intel 80386, version 1
documentation:   directory
essai.c:         C program text
guidex11.ps:     PostScript document text conforming at level 2.0
im.gif:          GIF image data, version 89a, 286 x 287,
lpbj:            ASCII text
mbox:            mail text
rep_gtk:         gzip compressed data, deflated, original filename, last modified: Tue Jun  2
21:22:18 1998, os: Unix
```

cp

La commande `cp` sert à copier des fichiers.
Syntaxe : `cp [options] [répertoire1/]fichier1 [répertoire2/]fichier2`
`cp [options] fichier1 [fichier2] ... [fichiern] répertoire`
avec la première syntaxe le `fichier1` du `répertoire1` est copié sous le nom `fichier2` dans le `répertoire2`. Notons que `répertoire1` et `2` peuvent être le même répertoire. Avec la seconde syntaxe les fichiers sont copiés dans le répertoire de destination à partir du répertoire en cours.

Exemples :
Dans l'exemple ci-dessous le fichier `profile` est copié du répertoire courant `/etc` vers le répertoire `/root`.

```
[root@linuxserveur /root]# cd /etc
[root@linuxserveur /etc]# cp profile /root
```

Dans l'exemple ci-dessous le fichier `profile` est copié du répertoire courant `/etc` vers le répertoire `/root` et renommé.

```
[root@linuxserveur /root]# cd /etc
[root@linuxserveur /etc]# cp profile /root/ex_de_profile
```

Dans l'exemple ci-dessous le fichier `profile` est copié du répertoire `/etc` vers le répertoire `/root` et renommé.

```
[root@linuxserveur /root]# cp /etc/profile /root/ex_de_profile
```

Dans l'exemple ci-dessous le répertoire `VG20` (fichiers et sous-répertoires) est copié du répertoire en cours vers le répertoire `/mnt/zip`.

```
[root@linuxserveur /etc]# cp -R VG20 /mnt/zip
```

N.B. si le ou les fichiers cible existent déjà, il seront écrasés purement et simplement.

mv

La commande `mv` sert soit à déplacer un fichier soit à le renommer.

Syntaxe : `mv [répertoire1/]fichier1 [répertoire2/]fichier2`
`mv fichier1 [fichier2] ... [fichiern] répertoire`

Dans la première forme si les répertoires sources et cibles sont les mêmes le fichier est simplement renommé. Dans l'exemple ci-dessous le fichier `monfichier1` est renommé en `monfichier2`.

```
root@linuxserveur /root]# ls
Mail documentation monfichier1 p.ps
root@linuxserveur /root]# cd /
root@linuxserveur /]# mv /root/monfichier1 /root/monfichier2
root@linuxserveur /]# cd /root
root@linuxserveur /]# ls
Mail documentation monfichier2 p.ps
```

Dans l'exemple suivant le fichier `monfichier1` est déplacé du répertoire `/root` vers le répertoire `/home/colling`

```
[root@linuxserveur /root]# cd /
[root@linuxserveur /]# mv /root/monfichier1 /home/colling
[root@linuxserveur /]# ls /home/colling
monfichier1
```

N.B. Si le ou les fichiers cibles existent déjà, il seront écrasés purement et simplement.

rm

`rm` permet de supprimer fichiers et répertoires. Parmi les options de cette commande deux sont intéressantes : `r` permet la suppression d'un répertoire même non vide et `i` demande confirmation avant suppression. Cette commande est très dangereuse, c'est pourquoi en général un alias est créé, forçant la demande de confirmation. Vérifiez qu'il est créé sur votre système en tapant `alias`, vous devriez voir apparaître `alias rm='rm -i'`. si cet alias n'existe pas, un bon conseil : créez-le. (voir la commande `alias`).
L'exemple suivant montre la suppression de deux fichiers et du répertoire `mes-docs`. A noter que si le répertoire `mes-docs` contient des sous répertoires ceux-ci sont également supprimés récursivement (d'où le `r`).

```
[root@linuxserveur /root]# ls fich*
fichier1  fichier2
[root@linuxserveur /root]# ls mes-docs
doc1  doc2
[root@linuxserveur /root]# rm fich*
rm: détruire `fichier1'? y
rm: détruire `fichier2'? y
[root@linuxserveur /root]# rm -r mes-docs
rm: aller dans le répertoire `mes-docs'? y
rm: détruire `mes-docs/doc2'? y
rm: détruire `mes-docs/doc1'? y
rm: détruire le répertoire `mes-docs'? y
```

N.B.1. Si un alias "`rm=rm-i`" a été créé il est tout de même possible de forcer la suppression avec l'option "`-f`". Ainsi "`rm -rf`" ne vous demandera aucune confirmation pour la suppression d'un répertoire même non vide.
N.B.2. Les alias sont définis dans les fichiers `/etc/bashrc` et `~/.bashrc`.

type (*)

La commande `type` vous permet de connaître la nature d'une commande, d'un mot clé ...

Syntaxe : `type [-all|-type|-path] nom [nom]`

```
[root@linuxserveur /root]# type -all ls rm if cd wish
ls is /bin/ls # fichier de la commande situé dans /bin
rm is aliased to `rm -i' # alias
rm is /bin/rm # fichier de la commande situé dans /bin
if is a shell keyword # mot-clé du langage de commandes
cd is a shell builtin # commande interne
wish is /usr/bin/wish  # fichier de la commande situé dans /usr/bin
```

Et juste pour le plaisir !

```
[root@linuxserveur /root]# type -type type
builtin #type  est donc une commande de type interne
```

Remarque: les commentaires après # ont été ajoutés, ils n'apparaissent pas à l'exécution de la commande.

2.3. Quelques commandes de connexion et d'administration.

exit (*)

Cette commande permet d'interrompre une session en cours, elle déconnecte l'utilisateur du système. La combinaison de touches "CTRL+D" joue le même rôle.

who et whoami

La commande `who` permet de savoir qui est connecté à votre machine. Avec l'option « `-q`» le nombre d'utilisateurs connectés est fourni :

```
[colling@linuxserveur /]$ who -q
root root colling
# users=3
```

sinon la commande fournit, entre autres, l'identité et le type de connexion. Les commentaires (#) ont été ajoutés.

```
[colling@linuxserveur /]$ who
root tty1 Mar 18 11:42 # root depuis une console.
root ttyp1 Mar 18 11:46 (:0.0) # root via xterm.
colling ttyp0 Mar 18 13:34 (172.28.47.2) # colling via telnet.
whoami affiche le nom d'utilisateur associé à  l'U-ID effectif en cours. C'est à dire vous donne
votre identité. Cette commande peut être très utile en phase d'essais, par exemple, lorsque vous
travaillez sur plusieurs comptes.
[colling@linuxserveur /]$ whoami
colling
```

Remarque : dans le cas ci-dessus le prompt (`[colling@linux /]`) donne l'identité de l'utilisateur en cours, dans ce cas la commande perd de son intérêt. Le prompt est défini dans le fichier /etc/profile par : PS1="[\u@\h \W]\\$ " où "\u" désigne le nom de l'utilisateur, "`\h`" le nom d'hôte, "`\W`" le nom de base du répertoire en cours, "`\$`" produit l'affichage de "#" dans le cas de root (uid=0) et $ dans les autres cas.

id la commande `id` permet de connaître les renseignements identifiant l'utilisateur.

```
[root@linuxserveur /root]# id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel)
[root@linuxserveur /root]# id colling
uid=500(colling) gid=100(users) groups=100(users)
```

passwd `passwd` permet à l'utilisateur de changer son mot de passe. Comme on peut le constater dans la séquence reproduite ci-dessous, Linux effectue un contrôle syntaxique du mot de passe.

```
[phil@linuxIG phil]$ passwd
Changing password for phil
(current) UNIX password:
New UNIX password:
BAD PASSWORD: it's WAY too short
New UNIX password:
BAD PASSWORD: is too similiar to the old one
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully
```

`passwd` permet à l'administrateur de changer le mot de passe d'un utilisateur.

```
[root@linuxIG /root]# passwd phil
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully
[root@linuxIG /root]#
```

Trois fichiers essentiels servent à stocker les informations sur les divers utilisateurs. Il s'agit des fichiers `/etc/passwd`, `/etc/shadow` et `/etc/group`. Ce sont des fichiers texte, toutes les informations y apparaissent en clair mais naturellement les mots de passe sont cryptés.

su `su` permet à tout utilisateur de passer en mode Super Utilisateur. Avec l'option "- " les fichiers profile sont pris en compte.

```
Password:
Last login: Wed Jan 27 14:45:49 from 172.28.47.2
[colling@linuxserveur colling]$ id
uid=500(colling) gid=100(users) groups=100(users)
[colling@linuxserveur colling]$ su -
Password:
[root@linuxserveur /root]# id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10 (wheel)
```

2.4. Quelques commandes d'information sur le matériel et le système.

df

`df` (**d**isk **f**ree) affiche l'espace libre sur les systèmes de fichiers montés.

```
[root@linuxserveur /root]# df
Filesystem         1024-blocks  Used Available Capacity Mounted on
/dev/hda1            3864732  751820  2912952     21%   /
/dev/fd0                1390      14     1304      1%   /mnt/disquette
```

free

`free` affiche les quantités de mémoire et de swap libres et utilisées, ainsi que la mémoire partagée et les buffers utilisés par le noyau.

```
[root@linuxserveur /root]# free
total used free shared buffers cached
Mem: 30824 29668 1156 30408 2480 10296
-/+ buffers/cache:      16892      13932
Swap:       128484          0     128484
```

uname

La commande `uname` fournit un certain nombre d'informations système.

syntaxe : `uname [options]`

`uname` ou `uname -s` affiche le nom du système d'exploitation.
`uname -r` affiche le numéro de version du système d'exploitation.
`uname -a` affiche les informations affichés par le commandes ci-dessus et d'autres.

```
[root@linuxserveur bin]# uname -sra
Linux linuxserveur.ac-nancy-metz.fr 2.0.36 #1 Tue Oct 13 22:17:11 EDT 1998 i586 unknown
```

A noter : le répertoire proc est constitué de fichiers contenant des informations sur le système. Ces fichiers sont visibles. Il y a par exemple `cpuinfo` qui contient des infos sur les processeurs, `meminfo` qui contient des infos sur la mémoire, `partitions` sur les disques et partitions. On peut visualiser par exemple ce dernier fichier par `cat /proc/partitions`.

2.4. Commandes d'arrêt du serveur.

halt ou shutdown -n now

Ces deux commandes sont équivalentes à un arrêt immédiat du système.
Une fois le système arrêté le message `System halted` s'affiche. C'est à ce moment que vous pouvez mettre hors tension, pas avant !
N.B. Ces commandes sont en principe réservées à l'administrateur.

reboot ou shutdown -r now

Ces deux commandes sont équivalentes à un redémarrage du système. Elles ne sont accessibles par défaut que par l'administrateur.
N.B : Vous pouvez également utiliser la combinaison de touches "CTRL+ALT+DEL", cette combinaison est, par défaut, accessible par tous. Le rôle de cette séquence de touches est défini dans le fichier `/etc/inittab` .