Commandes unix/linux

Fichiers

`pwd`

`ls`

- `ls -a`
- `ls -l`
- `ls -ld`
- `ls -lh`
- `ls -lart` (tri par date de dernière modification)
- `ls -F` (etiquette : /=repertoire *=executable @=lien symbolique)

Fichiers

Répertoires . et ..

```
cd
      cd /
      cd bin
      cd ..
      cd /usr/local/
      cd -
      cd ~
      cd .
mkdir
rmdir
```

Fichiers

`cp`

- `cp -R` (recursif)
- `cp -f` (force)
- `cp -i` (demande confirmation avant ecrasement)
- `cp -p` (preserve owner, mod time...)

Fichiers

`rm`

`rm -f` (force)

`rm -i` (demande confirmation)

`rm -R` ou `rm -r` (recursif)

`rm -- -fichier.txt` ou `rm ./-fichier.txt` (supprimer le nom commence par "-")

`mv`

Fichiers

Les méta-caractères * ? et []

```
*
?
[abc]
[!abc]
[^abc]
```

Fichiers

chmod

```
 u   g   o
rwx rwx rwx
421 421 421
chmod 700 fichier.txt
chmod o+rwx fichier.txt
chmod g-r fichier.txt
chmod a-x fichier.txt
```

Fichiers

chown

```
chown user:group fichier.txt
chown -R
```

chgrp

```
chgrp group fichier.txt
chgrp -R
```

Fichiers

head

```
head -10 fichier.txt
head -n 10 fichier.txt
```

tail

```
tail -n 10 fichier.txt
tail -f fichier.txt
```

Fichiers

file
grep
sort

```
sort -n
sort -r
sort -nr
```

Contrôle de Tâches/Processus

& lancement en tâche de fond

Ex :

```
% mozilla

% mozilla&
%
```

; séparateur de commandes

| « pipeline », chaînage des E/S standard des processus

Ex :

```
% ls -la | grep toto | wc -l
```

Processus : ps

`ps -alx`

```
# ps alx
F   UID   PID  PPID PRI  NI    VSZ  RSS WCHAN  STAT TTY        TIME COMMAND
4     0     1     0   8   0   1356  496 select S    ?          0:04 init [5]
1     0     2     1   9   0      0    0 contex SW   ?          0:00 [keventd]
1     0     3     1   9   0      0    0 apm_ma SW   ?          0:00 [kapmd]
1     0     4     1  19  19      0    0 ksofti SWN  ?          0:00 [ksoftirqd_CPU0]
1     0     5     1   9   0      0    0 kswapd SW   ?          0:00 [kswapd]
1     0     6     1   9   0      0    0 bdflus SW   ?          0:00 [bdflush]
1     0     7     1   9   0      0    0 kupdat SW   ?          0:00 [kupdated]
1     0     8     1  -1 -20      0    0 md_thr SW<  ?          0:00 [mdrecoveryd]
1     0    12     1   9   0      0    0 ?      SW   ?          0:00 [kjournald]
5     0    99     1   9   0   1972 1248 devfsd S    ?          0:00 devfsd /dev
1     0   201     1   9   0      0    0 ?      SW   ?          0:00 [khubd]
```

Processus : ps

COMMAND	Commande associée.
CPU	Pourcentage de puissance de calcul utilisée.
MEM	Pourcentage de mémoire utilisée.
NI	Niveau de priorité. Plus il est élevé, moins le processus est prioritaire.
PAGEIN	Nombre de pages mémoire.
PID	Numéro d'identification du processus.
PPID	Numéro d'identification du père du processus.
PRI	Fréquence en HZ des activations possibles du processus.
RSS	Taille du programme en mémoire.
SHARE	Mémoire partagée.
VSZ	Taille de mémoire virtuelle (texte+données+pile).
START	Heure de lancement de la tâche.

Processus : ps

STAT	Information d'état :	R <-> actif, S <-> en sommeil, D <-> en sommeil non interrompable, T <-> arrêté, Z <-> zombie, W <-> sans page résidente N <-> si le processus est moins prioritaire.
SWAP	Mémoire secondaire (en Ko).	
TIME	Temps d'inactivité.	
TRS	Taille résidente de texte.	
TTY	Nom du terminal.	
UID	Numéro d'identification du propriétaire.	
USER	Nom du propriétaire.	
WCHAN	Nom de la fonction du noyau dont le processus attend une réponse.	

Tuer un processus

```
kill
       kill -l (liste des signaux)
       kill -HUP (hangup)
       kill -9
     1        HUP (hang up)
     2        INT (interrupt)
     3        QUIT (quit)
     6        ABRT (abort)
     9        KILL (non-catchable, non-ignorable kill)
```

pkill ou killall

pkill -u toto mozilla

Contrôle de processus

ctrl+c	(SIGINT = « kill -2 », arrêt du processus)
ctrl+z	(suspension d'exécution)
bg	(« background »)
fg	(« foreground »)

Redirections

> Redirection de la sortie standard (STDOUT)
>> idem, en « append »

Ex :

```
% echo « toto » > fichier.txt
% ls -l > resultat_ls.txt
```

Edition de texte

cat	affiche le contenu du fichier
more	idem, mais par page écran, avec navigation
less	version améliorée de « more »
vi	éditeur de texte en mode console
vim	« vi » amélioré

Edition de texte : vi

Commande	Touche(s)	Fin	
Insertion sous le curseur	**i**	**ESC**	
Insertion en début de ligne	I	ESC	
Ajout devant le curseur	a	ESC	
Ajout en fin de ligne	**A**	**ESC**	
Rajout d'une ligne sous le curseur	o	ESC	
Rajout d'une ligne au-dessus du curseur		O	ESC
Déplacement vers la droite	l		
Déplacement vers la gauche	h		
Déplacement vers le haut	j		
Déplacement vers le bas	k		
Déplacement de 4 lignes vers le bas	4k		
Placement à la ligne 23	**:23 puis**		**Return**
Effacement d'un caractère	**x**		
Effacement de 5 caractères	5x		
Effacement d'une ligne	**dd**		
Effacement de 3 lignes	3dd		

Edition de texte : vi (2)

Commande	Touche(s)	Fin
Sauvegarde d'un fichier	**:w puis**	**Return**
Arrêt de l'édition	**:q puis**	**Return**
Sauvegarde et arrêt	**:wq puis**	**Return**
Ajout du fichier truc	:r truc puis	Return
Forçage d'une commande	**!**	
Copie d'un bloc de 6 lignes	**6yy**	
Collage d'un bloc au-dessous du curseur		**p**
Collage d'un bloc au-dessus du curseur		**P**
Recherche de la chaîne toto	**/toto puis**	**Return**
Suite de la recherche vers la fin	**n**	
Suite de la recherche vers le début	**N**	

Utilisateurs

Notion de UserID (UID) et groupID (GID)

Correspondance UID <-> login : `/etc/passwd`
Correspondance GID <-> groupe : `/etc/group`

Utilisateurs(2)

who	liste les utilisateurs connectés au système
whoami	affiche le login courant
id	liste le login, groupe principal, secondaires

```
Ex :
$ id
uid=635(kai) gid=20(informat) groupes=20(informat)
```

Changer son mot de passe

```
$ passwd

$ yppasswd

# passwd toto
```

Utilisateurs (4) : last

last

last -n 10

last toto

last -n 10 toto

Utilisateurs (5) : su

Changer d’utilisateur (« devenir un utilisateur »)

su
su toto
su - root
su -

Utilisateurs (6) : finger

finger

finger toto

finger user@host

finger @host

```
% finger
Login       Name                TTY         Idle    When    Where
userA  utilisateur A           dtlocal       41 Sat 09:59  :11
toto utilisateur toto           pts/8           Sun 19:11  thor

% finger poutrain
Login name: poutrain                    In real life: Kai Poutrain
Directory: /home/poutrain               Shell: /bin/tcsh
On since Oct 16 19:11:10 on pts/8 from thor2
Mail last read Sun Oct 16 18:46:41 2005
Plan:
```

Système

hostname : nom de la machine

dmesg : affichage des derniers logs

uname

uname -a

```
# uname -a
Linux maverick 2.4.21-0.13mdk #1 Fri Mar 14 15:08:06 EST 2003
i686 unknown unknown GNU/Linux
```

uname -n identique à « hostname »

uptime : temps d'activité de la machine

```
# uptime
 19:20:42 up 3 days,  7:40,  1 user,  load average: 0.00, 0.00, 0.00
```

Système (2)

halt
telinit
reboot
shutdown
    shutdown -h now
    shutdown -r now

Gestion de paquetages

rpm

rpm -ihv <nom_package.rpm>
rpm -e <nom_package>

urpmi
urpme

Commandes

whatis

```
% whatis time
time(1)                 - time command execution
time(3)                 - get time of day
time(ntcl)              - Time the execution of a script
```

whereis : localise les programmes

```
% whereis time
/usr/bin/time
```

which : localise les commandes

```
% which time
time: shell built-in command.
```

awk

```
awk -Fc '{print $1}'
```

séparateur

`$0` : toute la ligne
`$n` : nième champ extrait
`"chaine"` : affiche la chaîne entre guillemets

Ex :

```
awk -F":" '{print $1 "/"$2}'
```

find

find *racine expression*

expression :

+n	: plus de « n »
-n	: moins de « n »
n	: exactement « n »
-group nom_groupe	: fichier appartenant au groupe « nom_groupe »
-amin n	: accédé il y a « n » minutes
-atime n	: accédé il y a « n » jours
-mmin n	: modifié il y a « n » minutes
-mtime n	: modifié il y a « n » jours
-name nom_fichier	: recherche un nom de fichier donné
-user uname	: fichiers appartenant à l'utilisateur uname

Find (2)

find *racine expression*

expression (suite):

-size n[bckw] :

b = blocks de 512 octets

c = octets

k = Ko

w = « word », mots de 2 octets

-type c : fichier de type « c » avec c =

b = fichier spécial de type block

c = fichier spécial de type caractère

f = fichier normal

d = répertoire

l = lien symbolique

…

Find (2)

find *racine expression*

expression (suite):

-exec *commande* ‘{}‘ \; : execute « commande »

-ok *commande* ‘{}‘ \; : execute « commande » après confirmation

Attention à l’espace avant le « \; » !!!

-a : « and »

-o : « or »

() : regroupe les expressions (attention il faut écrire \(et \) pour éviter l’interprétation des parenthèses par le shell et il faut mettre des espaces autour.

Exemples :

```
find / -name “*toto* -exec rm -rf ‘{}’\;
find /var -size +1000k -exec ls -lh '{}' \;
find . \( -mmin +1 -a -mmin -2 \) -exec ls -lh '{}' \;
```

Exercices

A l'aide de la commande « find » :
A- Trouvez où se situe le fichier « libc.so »
B- Trouvez tous les fichiers de l'utilisateur toto
Trouvez tous les fichiers que l'utilisateur toto :
C - a modifé depuis moins d'un quart d'heure
D - a modifié dans une période comprise entre -24 et -48 heures
E - a modifé dans une période comprise entre -5 et -10 minutes
F - a modifié soit il y a moins de 10 minutes, soit il y a plus d'une heure

G - Comment chercher tous les fichiers commençant par un «a» majuscule minuscule, suivi éventuellement de quelques lettres ou chiffres, puis par un chiffre entre 3 et 6 ?
H - Comment effacer les fichiers de la question G ?

Exercices

I - Copier les fichiers dont l'avant dernier caractère est un 4 ou 1
dans le répertoire **/tmp** en une seule commande.
J - Dans le fichier /etc/passwd, vous devez extraire la liste des logins, triez-les
dans l'ordre alphabétique inverse
K - Dans le fichier /etc/passwd, vous devez extraire la liste des shells utilisés,
dans la réponse chaque shell ne doit apparaître qu'une fois.
L - A l'aide des commandes « ps », « awk », « wc », comptez le nombre de processus
fils directs du processus « init »
M - Affichez la ligne de l'utilisateur toto du fichier /etc/passwd
N - Affichez quel est son GID
O - Dans le répertoire courant affichez tous les fichiers dont la taille
est supérieure à 100Ko, puis, ceux dont la taille est supérieure à 1Mo
P - Affichez uniquement leurs tailles
Q - Affichez (calculez) le nombre de fichiers dont la taille est supérieures à 100Ko,
le nombre de fichiers dont la taille est inférieure à 100Ko