M2 CCI - Mise à niveau en Informatique
Commandes UNIX

Julie Bernauer, Thomas Tang

17 Septembre 2007

I. Introduction

Le système UNIX est un système d'exploitation multi-utilisateurs et multi-tâches : plusieurs personnes peuvent partager les ressources de la même machine et plusieurs programmes ou logiciels peuvent s'exécuter en même temps.
Le shell est un interpréteur de commandes : il permet à l'utilisateur de dialoguer avec le système. C'est le programme généralement exécuté lorsqu'un utilisateur se connecte. Il affiche un "prompt" et attend les commandes de l'utilisateur. Le shell est aussi un langage de programmation interprété puissant. Il offre à l'utilisateur un environnement composé d'un ensemble de variables et d'alias et un langage de commandes.

Il existe différents shells :

- sh : BourneShell (shell standard unix)
- ksh : Korn Shell
- csh : C Shell
- tcsh : extensiondeCShell
- bash : GNUBourne-AgainShell (shell par défaut des machines Linux)
- zsh : Z shell

Le shell par défaut sur les machines des salles de TP est bash, les autres shells sont aussi disponibles.

II. Syntaxe d'une commande UNIX

Lorsque vous êtes connectés, une console s'affiche à l'écran, avec un « prompt » : il attend de votre part une commande qu'il pourrait interpréter et exécuter. Le prompt est spécifique à la machine; en TD/TP, le « prompt » est du type:

tang@modigliani:~$

Chaque commande saisie au clavier doit être validée par un "return" (touche Entrée) pour être exécutée.

tang@modigliani:~$ commande -options <arguments>

Exemple :
tang@modigliani:~$ ls -al /home/tang

Remarque:
Les commandes doivent être saisies en minuscules.

Sous UNIX majuscules et minuscules ne sont pas équivalentes.

III. Commandes de Contrôle

exit : sortie (fin de session)
CTRL-D : sortie (équivalent du logout si on est au prompt)
CTRL-U : annulation de la ligne courante
CTRL-C : interruption d'un processus
CTRL-Z : suspension d'un processus (il n'est pas interrompu !)
CTRL-S et *CTRL-Q* : contrôle de flux (arrêt et reprise de l'édition)

IV. Aides en ligne

man <commande> ou *man <application>*
La commande "man" (pour manual) fournit des informations (description, options, syntaxe) sur une commande UNIX ou une application donnée.

Exemples :
man ls, man emacs, man chmod
man intro : dresse la liste des commandes accessibles par "man".

apropos <mot-clé> ou man –k <mot-clé>
Indique le nom des commandes et applications indexées par le mot-clé donné.

Exemple :
apropos image

V. Connexion et transfert

Un site du réseau Internet peut accéder à un autre site du réseau par les commandes telnet (connexion interactive) et ftp (transfert de fichiers). Les sites sont identifiés par des adresses IP et des noms.

nslookup <nom du site> : Retourne l'adresse IP du site.
Exemple : *nslookup www.ie2.u-psud.fr*

telnet <adresse hôte> : Etablit la connexion interactive avec un autre ordinateur sur le réseau Internet. Ce type de connexion tend à ne plus être utilisé en raison de sa vulnérabilité.
Exemple : *telnet c70.ie2.u-psud.fr*

ssh <adresse hôte> : Etablit la connexion interactive sécurisée avec un autre ordinateur sur le réseau Internet.
Exemple : *sshc70.ie2.u-psud.fr*

ftp< adresse hôte > : Etablit la connexion en transfert de fichiers avec un autre ordinateur sur le réseau Internet. Retourne un prompt.
Taper *help* pour avoir les commandes accessibles à ce niveau et *help <commande>* pour avoir le descriptif de chaque commande.

VI. Commandes utilitaires

passwd : Permet de changer de mot de passe.
Attention !!! Cette commande ne fonctionne pas sur les machines que vous utilisez

en TP car le système de gestion des mots de passe est centralisé. Toutefois, c'est cette commande qui est la commande standard UNIX/Linux.

who Affiche la liste des utilisateurs connectés.

date Affiche la date.

cal <mois> <année>
Affiche le calendrier du mois, de l'année spécifiée. Par défaut, année et mois courants.

echo <chaîne>
Retourne les arguments donnés. *echo* est notamment utile pour soumettre des données à un pipe, pour éditer le contenu de variables d'environnement, etc...
Exemples :
echo toto@yahoo.fr > .forward crée le fichier de ré-acheminement des mails
echo $USER

groups Affiche le(s) groupe(s) au(x)quel(s) appartient un utilisateur.

id <nom utilisateur>
Affiche l'uid et le(s) gid de l'utilisateur.

quota -v Retourne l'espace disque autorisé et utilisé (quota et nombre de fichiers) sur la totalité du compte. L'unité du quota est en ko, donc par exemple 10000 signifie 10 Mo. Cette commande ne fonctionne que si les comptes utilisateurs sont gérés avec des quotas.

du Retourne l'espace disque utilisé dans un répertoire donné (répertoire courant par défaut). L'unité est en demi-blocs (soit 512 caractères).

VII. Répertoires et fichiers

VII.1. Syntaxe des fichiers

Le système de fichiers (filesystem) est un arbre (organisation hiérarchique) dont les nœuds sont des répertoires (directories) et les feuilles, des fichiers. Le fichier (chemin/nom) peut être désigné :

- soit par un chemin absolu, qui commence par le caractère "/" (racine de l'arbre) suivie de la liste des nœuds (sous-répertoires, séparés par le caractère "/"), qu'il faut suivre depuis la racine pour atteindre le fichier
  Exemple : */home/tang/td1.c*
- soit par un chemin relatif : le fichier dans ce cas est désigné depuis le répertoire courant
  Exemple : *tang/td1.c* si on désigne le fichier précédent depuis */home*.

Le « *home directory* » est le répertoire d'accueil dans lequel on se trouve après avoir établi la connexion sur le compte.

Certains caractères ne sont pas autorisés dans le nom du fichier. Notamment, par exemple, le caractère espace, accepté dans les noms de fichiers sous Windows ou Mac, joue le rôle de séparateur sous UNIX (il sépare commande, options et paramètres les uns des autres). Pour

manipuler malgré tout un fichier qui contient ce type de caractère, le nom de ce fichier devra être encadré de guillemets, par exemple *touch "nom avec espace.txt"*

En règle générale, il est préférable de se limiter aux caractères alphabétiques, numériques ainsi que "-" et "_". Le fichier peut optionnellement contenir une extension. L'extension est une particule suffixe séparée de la racine du nom par un point. Elle vise généralement à renseigner sur la nature du fichier, par exemple *td1**.c***

Certains utilitaires sous UNIX (éditeurs de texte) conservent les deux dernières versions du fichier. Dans ce cas, si l'on modifie le contenu d'un fichier déjà existant, le nom de l'avant dernière version est "marqué" d'un signe particulier, tel que "~" ou "%", par exemple *td1.c~*

VII.2. Troncature

Sous UNIX, le caractère de troncature (caractère "joker") est le caractère "*" : il peut remplacer n'importe quel groupe de caractères d'un nom de fichier donné : il permet ainsi de simplifier la commande et/ou de généraliser une opération sur tout un ensemble de fichiers.

Exemple :
*ls *.txt* liste tous les fichiers .txt (fichiers texte)

VII.3. Droits d'accès aux fichiers

Dans tout système UNIX, les répertoires et fichiers ont des droits d'accès. Ceux-ci sont indiqués par la commande *ls –la*.

Exemple:

```
-rw-rw-r-- 1 boulange boulange  90667 2007-09-06 14:57 diff.jpg
-rw-rw-r-- 1 boulange boulange 467080 2007-09-06 14:11 fedora_screen.jpg
-rw-rw-r-- 1 boulange boulange  51352 2007-09-06 14:46 find.jpg
-rw-rw-r-- 1 boulange boulange 171254 2007-09-06 14:16 linux_commands_1.jpg
-rw-rw-r-- 1 boulange boulange 137313 2007-09-06 14:26 linux_commands_mv_cp.jpg
-rw-rw-r-- 1 boulange boulange  67712 2007-09-06 14:28 linux_commands_mv.jpg
-rw-rw-r-- 1 boulange boulange  95043 2007-09-06 14:14 man1.jpg
-rw-rw-r-- 1 boulange boulange 134833 2007-09-06 14:14 man2.jpg
-rw-rw-r-- 1 boulange boulange 177574 2007-09-06 15:13 ps_pipe.jpg
-rw-rw-r-- 1 boulange boulange 178050 2007-09-06 15:09 redirection_entree.jpg
-rw-rw-r-- 1 boulange boulange 271920 2007-09-06 15:04 redirection_sortie.jpg
-rw-rw-r-- 1 boulange boulange  34671 2007-09-06 14:13 shell.jpg
drwxrwxr-x 2 boulange boulange   4096 2007-09-06 15:08 Tests
```

VII.3.1. Catégories d'utilisateur et droits d'accès

Sous UNIX, on distingue trois catégories d'utilisateurs:

- u (**u**ser) : le propriétaire
- g (**g**roup) : le groupe
- o (**o**thers) : les autres
- a (**a**ll) : regroupe les 3 catégories

ainsi que trois types de droits, donnés sur 3 colonnes (rwx):

- r (**r**ead) : lecture
- w (**w**rite) : écriture
- x (e**x**ecute): exécution
- - : aucun droit

Pour un fichier, les droits sont exprimés par une chaîne de 10 caractères : *tuuugggooo*

- *t* est le type du fichier
    - o - : fichier ordinaire

  - d (**d**irectory) : répertoire
  - l (**l**ink) : lien symbolique
  - c ou b : fichier spécial
- *uuu* : droits du propriétaire
- *ggg* : droits du groupe
- *ooo* : droits des autres

VII.3.2. Commande *chmod*

Elle permet de changer les droits d'accès d'un fichier. Cela peut être fait de deux manières différentes.

chmod <utilisateurs> ±<droits> <fichier>

Exemples :

chmod u+r toto	donne le droit de lecture du fichier toto à l'utilisateur (vous-même).
chmod g+w toto	autorise une personne du même groupe que vous à lire le fichier.
chmod +x toto	autorise les autres à exécuter le fichier.
chmod o-w toto	interdit aux autres l'écriture du fichier.
chmod a+rw toto	autorise le propriétaire, le groupe et les autres à lire et écrire le fichier.

chmod <droits (octal)> <fichier>

Droits d'accès et correspondance en octal :

- --- : 0
- --x : 1
- -w- : 2
- -wx : 3
- r-- : 4
- r-x : 5
- rw- : 6
- rwx : 7

On peut ainsi définir avec *chmod* les droits en octal.

Exemples:

*chmod 600 **	attribuera rw------- à tous les fichiers
*chmod 644 **	attribuera rw-r--r-- à tous les fichiers
*chmod 750 **	attribuera rwxr-x--- à tous les fichiers

VII.4. Edition et manipulation de fichiers

pwd Affiche le chemin du répertoire courant.

ls Liste le nom des fichiers.
ls –l Edition du catalogue des fichiers du répertoire courant.
ls –a Edition de tous les fichiers du répertoire courant y compris les fichiers cachés commençant par un « . ».
ls –lt Edition du catalogue classé par date de la plus récente à la plus ancienne
ls -lR Lecture de la hiérarchie complète des fichiers

cd <répertoire>
Permet de se placer dans un répertoire donné.
Exemple : *cd /usr/bin* permet d'aller dans le répertoire */usr/bin*

cd ~ Retour au répertoire d'accueil (à la connexion de la session)
cd .. Retour au répertoire précédent (répertoire « parent »)
Le symbole "." représente le répertoire courant.
Le symbole ".." représente le répertoire parent.
Le symbole "~" représente le répertoire d'accueil (home directory).
Le "~" et le ".." peuvent être utilisés dans les chemins d'accès.

mkdir <répertoire> Crée un répertoire (Make directory).

rmdir <répertoire> Supprime un répertoire VIDE (Remove directory).

cat <fichier> Affiche le contenu d'un fichier en mode déroulant.

more <fichier> Affiche le contenu d'un fichier en mode page.
Pour éditer la page suivante : barre espace

less <fichier> Affiche le contenu d'un fichier en mode page (équivalent de more).

head -n <fichier> Affiche les n premières lignes du fichier.

tail -n <fichier> Affiche les n dernières lignes du fichier.

wc -l -w -c <fichier> Retourne le nombre de lignes, de mots, de caractères dans le fichier indiqué.

sort <fichier> Trie le fichier donné .

cp <file1> <file2> Copie le contenu de <file1> dans <file2>, en détruisant <file2> s'il existait.

mv <file1> <file2> Renomme ou déplace <file1> en <file2>.
mv <file><dir> Déplace (et éventuellement renomme) <file> dans <dir>.

touch <fichier> Crée un fichier vide si <fichier> n'existe pas, modifie sa date d'accès sinon.

rm <fichier> Supprime le fichier.
rm -R <répertoire> Supprime un répertoire, ainsi que tous les sous répertoires et les fichiers qu'il contient (Remove Recursively).

grep <chaîne> Recherche une chaîne de caractères (pattern) dans un fichier donné et édite les lignes la contenant.

find<répertoire>-name<fichier>-print
Recherche le fichier dans toute l'arborescence (fichiers et sous répertoires) issue du répertoire indiqué.

cmp<file1><file2> Compare et affiche le numéro des lignes différentes entre file 1, 2.
diff<file1><file2> Affiche les lignes différentes entre deux fichiers.

VII.5. Compression et archivage

La compression d'un fichier vise à réduire la taille de celui-ci. Grossièrement, le taux de compression atteint est de 50 à 60% de la taille d'origine.

compress<fichier1>	Compression. Génère un fichier .Z
uncompress<fichier1>	Décompression d'un fichier .Z
gzip(resp.bzip2)<fichier1>	Compression. Génère un fichier .gz(resp. .bz2). Plus performant que compress.
gunzip(resp.bunzip2)<file1>	Décompression d'un fichier .gz (resp. .bz2)
zcat<fichier1>	Affiche le contenu d'un fichier .Z ou .gz ou .bz2
zless<fichier1>	Affiche le contenu d'un fichier .Z ou .gz ou .bz2
zgrep<fichier1>	Recherche une chaîne de caractères dans un fichier .Z ou .gz ou .bz2
tar cvf <tarfile> <dir>	Archive le répertoire en un fichier .tar unique.
tar xvf <tarfile>	Restaure un fichier .tar en fichiers d'origine.

VII.6. Editeurs de fichiers

VII.6.1. Editeur vi

L'éditeur vi est un peu complexe à utiliser au début (man vi pour tous les détails). Trois modes aux fonctionnalités différentes sont disponibles sous vi :

- mode commande (commandes par des caractères spéciaux)
- mode insertion (saisie du texte)
- mode ligne (commandes saisies en bas du fichier).

Mode INSERTION (invisible)

Touche	Fonction
i	insère avant le curseur
a	insère après le curseur
o	ouvre une nouvelle ligne en dessous
O	ouvre une nouvelle ligne au dessus
ESC	quitte le mode insertion et retourne au mode commande
ESC :	quitte le mode insertion et retourne au mode ligne

Mode LIGNE

Touche	Fonction
:n	positionne sur la ligne n
/chaîne	recherche une chaîne de caractères
:n,md	annule les lignes n à m
:w fichier	écrit dans un autre fichier
:wq	termine en sauvegardant le fichier
q !	quitte impérativement sans sauvegarder
:set nu	affiche les numéros de lignes
:set all	montre les options

Mode COMMANDE (invisible)

Touche	Fonction
x	délétion de caractère
r	remplacement du caractère courant
dw	supprime le mot courant
dd	supprime la ligne courante
ndd	supprime n lignes
d0	efface jusqu'au début de ligne
dG	efface jusqu'en fin de ligne
u	annule la dernière fonction d'édition
.	répète la dernière fonction d'édition
^,$	pour se déplacer en début, en fin de ligne
G	position en fin de fichier (équivalent à :$)
nG	position sur la ligne n (équivalent à :n)
Y	sélectionne une ligne
nY	sélectionne n lignes
p	dépose la sélection après le curseur
P	dépose la sélection avant le curseur
sh	accès temporaire au shell (retour par exit)

VII.6.2. Editeur emacs

L'éditeur emacs a une utilisation plus intuitive que vi.
emacs <fichier> ouvre un fichier.
Les touches Ctrl et Esc, largement utilisées sous emacs, sont ci-dessous notées respectivement C- et E-.

Exemple :
C-k signifie "Maintenez la touche Ctrl enfoncée et appuyez sur la touche k".

Touches	Fonction
C-x puis C-c	Quitter emacs
C-x puis C-s	Sauvegarder sans quitter emacs

Déplacement du curseur	en avant	en arrière
caractère	C-b	C-f
mot	E-b	E-f
ligne	C-p	C-n
aller en début (fin) de ligne	C-a	C-e
aller en début (fin) de fichier	E-<	E->
écran précédent (suivant)	E-v	C-v

Effacement de caractères	Backspace ou Del ou C-d

.

Rechercher :	
en avant	C-s
en arrière	C-r
Arrêter la recherche en cours	C-g

Déplacer une partie de texte :

- sélectionner le début de la zone par C-espace
- placer le curseur à la fin de la zone à déplacer et effacer cette partie sélectionnée par C-w
- placer le curseur à l'endroit où l'on veut restituer le texte, et rappeler celui-ci par C-y

Aide en ligne : C-h ? puis initiale de la commande
Faire disparaître l'aide en ligne : C-x l Si on est bloqué : C-g

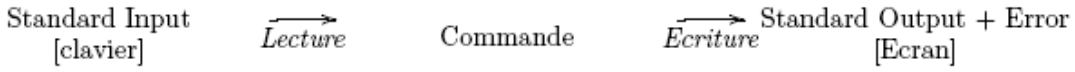
Effacer une ligne après le curseur : C-k.
Pour la rappeler à l'endroit où se trouve le curseur : C-y

Pour insérer un fichier : C-x i « nom du fichier »

VIII. Redirection, pipe

VIII.1. Redirection

De nombreuses commandes lisent leurs données (entrée=input) à partir de l'entrée standard (stdin), par défaut le clavier, et écrivent leurs résultats (sortie=output) dans la sortie standard (stdout) et les erreurs dans la sortie erreur standard (stderr), par défaut l'écran, selon le schéma:



Si l'on souhaite rediriger les entrées et sorties, la commande prendra la syntaxe suivante:

commande [-options] [arguments] < input-file > output file

avec les métacaractères de redirection suivants:

<	redirige l'entrée standard
>	redirige la sortie standard
>>	redirige et concatène la sortie standard
>&	redirige les sorties standard et erreur
>>&	redirige et concatène les sorties standard et erreur

Exemples :

- *who > names* édite les noms dans le fichier names
- *(pwd ; ls-l) > fichiers.out* écrit le nom du répertoire et le résultat de la commande ls dans fichier.out (le caractère ";" permet d'enchaîner des commandes)
- *mail –s "subject" cci@ie2.u-psud.fr < doc.txt* envoie le fichier doc.txt à cci@ie2.u-psud.fr
- *cat file1 file2 > file3* concatène deux fichiers (contenu de file2 après file1) dans un troisième
- *cat ligne > essai* écrit le contenu de ligne dans le fichier essai

VIII.2. pipe

Le caractère "|" (opérateur pipe) redirige la sortie standard (sdtout) d'une commande dans l'entrée standard (stdin) d'une autre commande. Plusieurs commandes peuvent être combinées ainsi.

commande1 | commande2

Exemples :

- *who | sort* trie et affiche les utilisateurs connectés
- *ls –lR | more* affiche page par page les noms de fichiers du répertoire
- *cat fichier | mail –s "sujet" cci@u-psud.fr* envoie le contenu du fichier à cci par mail
- *ls –lR | grep –i "apr10" > liste.txt* sélectionne les fichiers datés du 10 avril et les écrit dans le fichier liste.txt

IX. Processus

Le système UNIX est un système d'exploitation multi-utilisateurs et multi-tâches. Le calculateur partage son temps entre tous les processus présents à un moment donné. Le "multi-tâches" est réalisé par l'élection d'un processus parmi d'autres, pour un temps déterminé. Le calculateur exécute pendant une tranche de temps les instructions de ce processus. L'aspect "multi-utilisateurs"est une extension du"multi-tâches", qui permet à plusieurs utilisateurs de faire exécuter leurs processus respectifs par le calculateur.

Processus du noyau:

- lorsque le calculateur est mis en service, le processus 1, nommé **init**, est créé : il est responsable des demandes de login sur chaque terminal.
- le **scheduler** alloue du temps CPU alternativement à tous les processus actifs du système.
- le **swapper** est activé lorsque plusieurs processus sont exécutés et que le noyau n'a plus de place en RAM (mémoire courante). Il possède le numéro 0. Il détermine, suivant des critères variés, le processus qui doit être déporté sur le disque.
- le **pagedaemon** est activé lorsqu'un processus requiert une page non présente en RAM.

Le processus 1 est l'ultime parent de tous les processus. Les processus peuvent engendrer d'autres processus (processus enfants). Chaque processus possède un numéro d'identification, le process id ou pid. Ce pid est attribué séquentiellement, débutant de 0, et est incrémenté à chaque création. Un utilisateur ne peut contrôler un processus s'il n'en est pas propriétaire.

ps	Affiche les informations sur les processus en cours
ps –fu login	Liste complète de tous les processus rattachés à l'utilisateur donné
ps –eaf	Liste complète de tous les processus (*man* => signification des champs retournés)
Ctrl-C	Termine le processus premier-plan courant
Ctrl-Z	Stoppe l'exécution du processus premier-plan courant (qui pourra être relancé par *bg* ou *fg*).

kill <n° pid> Arrête le processus désigné.
kill -9 <n° pid> peut être utilisé si on n'arrive pas à arrêter le processus avec kill.

<commande> & lance un programme en arrière plan

jobs Affiche les jobs (et leur numéro) stoppés et/ou passés en arrière-plan.

fg %<n° job> Ramène le job courant de l'arrière-plan au premier plan (foreground).
bg %<n° job> Relance un job stoppé et le place en arrière-plan (en tâche de fond, background).

X. Lancement d'un programme en différé

batch<programme> Lance un programme en batch (exécution immédiate, au moment où le système, load level system, le permettra).

at -f <programme>	Lance un programme en différé. Possibilité de spécifier le moment où le programme sera lancé.
atq	Jobs en attente, lancés avec la commande *at*.
cancel<n° job>	Supprime un job en attente.

XI. Utilisation du shell BASH

XI.1. Entrée de commandes

Quand vous tapez des commandes shell, vous avez accès à un mini-éditeur qui ressemble au DOSKEYS de MS-DOS. Voici quelques combinaisons de touches utiles (C- Ctrl et E- Esc).

Touche	Fonction
↑	Remonte d'une commande dans l'historique
↓	Descend d'une commande dans l'historique
←	Recule d'un caractère
→	Avance d'un caractère
E-f	Avance d'un mot
E-b	Recule d'un mot
C-a	Revient au début de la ligne
C-e	Avance à la fin de la ligne
C-d	Efface le caractère courant
Backspace	Efface le caractère précédent
E-d	Efface le mot courant
C-u	Efface du début de la ligne
E-k	Efface de la fin de la ligne
C-y	Récupère le dernier élément supprimé
E-.	Insère le dernier mot de la commande précédente
C-l	Efface l'écran, plaçant la ligne courante en haut
Tabulation	Essaie de compléter le mot courant, l'interprétant comme un nom de fichier, d'utilisateur, de variable, de machine, ou de commande déterminée par le contexte
E- ?	Liste les complétions possibles

L'une des touches d'édition les plus utiles, Tab, peut aussi être utilisée quand on tape une commande. Si vous tapez la première partie de la commande et appuyez sur Tab, le shell va tenter de compléter jusqu'à ce qu'il n'y ait plus qu'une solution.

XI.2. *Filename Globbing* ou Utilisation des caractères "joker"

Avant de passer les arguments à une commande externe ou interpréter une commande, le shell scanne la ligne de commande et réalise une opération appelée *filename globbing*. Le *filename globbing* ressemble au traitement des caractères "joker" dans les commandes MS-DOS mais est plus sophistiqué. Voici quelques-uns des caractères utilisés appelés *filename metacharacters*.

Touche	Fonction
*	correspond à une chaîne de zéro ou plus de caractères
?	correspond à un caractère exactement
[abc...]	correspond à un des caractères spécifiés
[a-z]	correspond à un caractère de la plage spécifiée
[!abc]	correspond à un caractère autre que ceux spécifiés
[!a-z]	correspond à un caractère autre que ceux de la plage spécifiée
~	le répertoire d'accueil (home) de l'utilisateur courant
~ userid	le répertoire d'accueil (home) de l'utilisateur spécifié
~+ ou .	le répertoire de travail courant
~_	le répertoire de travail précédent
..	le répertoire de travail père

XI.3. Alias shell

Les alias shell simplifient l'utilisation de commandes en permettant d'établir des noms de commandes abrégés. Pour créer un alias il suffit de taper : *alias <name> = '<commande>'*

Exemple : *alias dir = 'ls-alrth'*

Par défaut, certains alias sont déjà spécifiés, pour les voir, tapez : *alias*. Pour enlever un alias, tapez *unalias <alias name>*
De cette façon, les alias ne seront effectifs que le temps d'une session à moins d'utiliser un script.

XI.4. Scripts shell

Un script shell est simplement un fichier contenant des commandes. En stockant des commandes dans un fichier, cela permet de pouvoir les exécuter plus simplement à nouveau encore et encore.

Exemple : Le fichier deleter.sh contient:

```
#!/bin/bash
echo -n Deleting the temporary files...
rm -f *.tmp
echo Done.
```

Pour exécuter ce script, qui efface les fichiers temporaires du répertoire courant, il suffit de taper : *bash deleter.sh* ou de rendre exécutable grâce à *chmod* et taper : *./deleter.sh*. Pour l'exécuter en ne tapant que : deleter.sh, il faut que le script soit dans le searchpath ou "chemin de recherche" qui fait partie des variables d'environnement.

XI.5. Variables d'environnement

Le shell est un langage de programmation à part entière qui permet de se référer à des "variables shell" ou "variables d'environnement" (*environment variables*). Pour assigner une valeur à une variable shell, on tape : *<nom de variable>=<valeur>*.

Les variables sont très utilisées sous UNIX. Pour afficher la liste des variables d'environnement (variables prédéfinies), on tape : *set*.

Voici la liste des variables d'environnement les plus importantes.

Variable	Fonction
DISPLAY	Le serveur graphique qui est (ou va être) utilisé
HOME	Le chemin absolu du répertoire d'accueil de l'utilisateur
HOSTNAME	Le nom de la machine
LOGNAME	Le nom de login de l'utilisateur
MAIL	Le chemin absolu du fichier de mail de l'utilisateur
PATH	Le *searchpath* ou chemin de recherche
TERM	Le type de terminal utilisé
USER	Le nom courant de l'utilisateur, peut différer du nom de login si l'utilisateur a utilisé la commande su.

Il est possible d'utiliser la variable d'une variable shell en précédant le nom de la variable par "*$*". Pour éviter la confusion avec le texte environnant, il est bien de mettre le nom de la variable entre crochets.
Exemple : *cd${HOME}/work*

Pour rendre accessible une variable shell à d'autres programmes, il faut l'exporter en tapant :

export<variable>

Pour enlever une variable, on tape : *unset <variable>*

XI.6. Le chemin de recherche ou *searchpath*

La variable shell spéciale PATH contient une série de chemins qui forment le chemin de recherche. A chaque fois qu'une commande est tapée, le shell recherche cette commande dans les chemins du PATH. Par exemple, supposons que le PATH ait la valeur suivante :

/usr/bin :/bin :/usr/local/bin :/usr/bin/X11 :/usr/X11R6/bin

Il est possible d'ajouter un nouveau chemin de recherche, par exemple /home/julie/executables en tapant:

PATH=${PATH} :/home/julie/executables

La commande which permet de savoir quel est le chemin d'une commande, par exemple : which wc renvoie /usr/bin/wc sur la plupart des machines Linux.

XI.7. Remarque

Le fichier *.bashrc* dans le home de chaque utilisateur est lu par *bash* à chaque session. Il permet à chaque utilisateur de modifier les variables que *bash* utilisera pendant sa session.

Le fichier d'initialisation du shell bash pour chaque utilisateur est le fichier $HOME/.bashrc.

XI.8. Shell et chaînes de caractères

Parfois, il arrive que le shell n'interprète pas les chaînes de caractères comme on le souhaiterait. Pour palier ce problème, on utilise des *quoted characters* qui permettent de préciser ce que l'on veut dire:

Caractère	Fonction

’	Les caractères à l’intérieur d’apostrophes (*single quotes*) sont interprétés littéralement ; les metacaractères qu’ils contiennent sont donc ignorés. Le shell ne remplace pas les noms des variables d’environnement par les valeurs correspondantes.
‘‘	Les caractères à l’intérieur de guillemets (*double quotes*) sont interprétés littéralement ; les metacaractères qu’ils contiennent sont ignorés cependant les noms des variables d’environnement sont remplacés par leur valeur.
‘	Les caractères à l’intérieur de contre-apostrophes (*backquotes*) sont interprétés comme une commande, que le shell exécute avant d’exécuter le reste de la ligne de commande. Le résultat de la commande remplace le texte original entre contre-apostrophes.
\	Le caractère suivant est interprété littéralement ; sa signification en tant que metacaractère est ignorée. Le contre-oblique (backslash) sert aussi comme caractère de continuation de ligne. Quand une ligne se termine par un contre-oblique, la ligne suivante est considérée comme faisant partie de la même ligne.

XII. Divers

Vous trouverez ci-dessous quelques noms de commandes/programmes utiles.

Courrier électronique	mail, mailx, elm, mutt, pine, emacs, netscape, mozilla
Navigation internet	lynx, netscape, mozilla
News	netscape, emacs, mozilla
Tracé de courbe	gnuplot, openoffice, octave
Calcul simple	bc, calc, xcalc
Statistiques	R
Calendrier	cal, emacs
Tableur	openoffice, emacs

Les outils de programmation seront vus dans les TD correspondants.

XIII. Pour aller plus loin

Sites WEB :

- Ressources documentaires du serveur Infobiogen dont ce polycopié est largement inspiré http://www.infobiogen.fr
- Formation GNU/Linux d’Alexis de Lattre http://people.via.ecp.fr/~alexis/formation-linux/formation-linux.html
- Le serveur Informatique et Enseignement de l’Université d’Orsay http://www.ie2.u-psud.fr

Livres :

- Le Système Unix de Steve Bourne (InterEditions ou Addison-Wesley)
- Learning Debian GNU/Linux de BillMcCarty (O'Reilly)
- Learning the VI Editor de Linda Lamb & Arnold Robbins (O'Reilly)
- Les bases de l'administration système d'Aelen Frisch (O'Reilly)
- Learning the BASH shell de Paul Newham (O'Reilly)
- Introduction à GNU emacs de Debra Cameron, Bill Rosenblatt, Jean-Baptiste Yunès (O'Reilly)

Distributions Unix/Linux

- Pour débuter avec Linux : http://www.knoppix.org
- Pour les plus confirmés : http://www.debian.org et http://www.gentoo.org
- Pour les débutants intéressés par la bioinformatique
  http://www.vigyaancd.org
  http://bioinformatics.org/vlinux/
  http://bioknoppix.hpcf.upr.edu/
  http://dirk.eddelbuettel.com/quantian/

Table des matières