TP 1
Linux : Commandes de base & utilitaires

I. Introduction : Rappel & révision

Nous prenons l'exemple du système d'exploitation Linux pour s'initier aux tâches d'administration système. Un des atouts de ce système est sa *facilité* d'administration puisque la majorité des fichiers de configuration sont des fichiers textes pouvant être modifiés directement en utilisant un simple éditeur. Bien sûre il faut connaître quel fichier modifier? et comment? pour changer tel ou tel paramètre du système. La plupart des systèmes dérivés d'Unix présentent plus au moins la même hiérarchie des répertoires. Dans le tableau suivant nous citons les principaux répertoires du système Linux avec une brève description de leur contenu.

Répertoire	***Sous répertoire***	***Contenu***
`/bin`	`/usr/bin`	Commande de base
	`/usr/local/bin`	Commandes supplémentaires ajoutées par l'administrateur
`/sbin`	`/usr/sbin`	Commandes d'administration
`/boot`		Contient les fichiers du noyau Linux
`/dev`		Contient les fichiers particuliers aux périphériques
`/etc`	`/etc/rc.d`	Contient les fichiers de configuration du système
	`/etc/init.d`	Sous-répertoire de démarrage des services sous Linux
`/home`		Contient les répertoires personnels des utilisateurs
`/lib`	`/usr/lib`	Bibliothèques de sous-programmes utilisées pour le développements
`/mnt`	`/mnt/cdrom`	Contient les répertoires des périphériques amovibles:CD/ disquette, USB
	`/mnt/floppy`	
	`/mnt/usb`	
`/proc`		Répertoire dédié aux processus
`/root`		Répertoire personnel de l'administrateur
`/tmp`		Les fichiers temporaires
`/usr`		Principal répertoire du système
	`/usr/include`	Sous répertoire des fichiers d'en-tête
	`/usr/share/man`	Sous répertoire de manuels Linux
	`/usr/local`	Logiciels installés par l'administrateur

Répertoire	*Sous répertoire*	*Contenu*

`/var`	`/var/log`
	`/var/spool`
	`/var/spool/mail`
	`/var/mail`
		Répertoire contenant la partie « variable » du système comme les traces d'activités du système; les boîtes aux lettres, etc.

Exercice 1 : Rappels des commandes de base

1. Rappeler les commandes de base de navigation dans le système de fichiers. Utiliser ces commandes afin d'explorer l'arborescence du système.
2. En utilisant la commande `cat`, créer un fichier `essai` contenant le texte « `ceci est un essai` »
3. Donner une commande qui permet d'afficher le contenu du fichier `essai`.
4. Afficher le nom du répertoire de travail
5. Sauvegarder la liste des fichiers contenus dans le répertoire de travail dans un fichier nommé `list.txt`
6. Donner **une commande** qui permet de créer dans `/tmp` un répertoire `moduleR3` qui contient un répertoire nommé `tp1`.
7. Copier le fichier `essai` dans le répertoire `/tmp/moduleR3/tp1`
8. Déplacer le fichier `list.txt` dans le répertoire `/tmp/moduleR3`
9. Effacer le fichier `/tmp/moduleR3/list.txt`.
10. Donner **une commande** qui permet d'effacer le contenu du `/tmp`. (y compris les sous-répertoires de /tmp)
11. Donner **une commande** qui retourne le contenu du répertoire « / » trié en ordre alphabétique inversée.
12. Donner **une commande** qui affiche les trois premiers lignes du fichier `/etc/passwd`. Donner une commande qui affiche l'avant dernière ligne de ce même fichier.
13. Que fait la commande `touch` ? et la commande `file` ?
14. Créer des fichiers vides (de tailles 0 octets) dans `/tmp`.
15. Donner **une commande** qui permet d'effacer tous les **fichiers** vides se trouvant dans `/tmp`.

Exercice 2. Commandes internes & externes

Il existe deux types de commandes systèmes : les commandes internes et les commandes externes. Les premières sont des commandes dont les programmes sont résident la mémoire de travail, les autres sont chargées en mémoire à leur appel. Une manière de les distinguer est d'utiliser la commande **`which`** suivie du nom de commande à tester. Si la commande retourne un chemin vers un fichier exécutable alors la commande est externe. Sinon (et si la commande existe) la commande est interne.

1. Rappeler le principe d'exécution de la commande **`which`.**
2. Quel est l'intérêt des commandes internes ?
3. Donner des exemples des deux types de commandes.

Les variables d'environnement

Les variables d'environnement sont utilisées par les *shells* afin de garder des informations utiles aux commandes et des logiciels utilisés durant la session du travail. Les shells étant des véritables langages de programmation l'utilisateur peut définir les variables qu'il veut. Nous rappelons ci-après les principales commandes de gestion de variables (en syntaxe *bash*).

opération	***Syntaxe***	***Exemples***

Affectation	`VAR=contenu`	`NOM=Dupont`
		`X='$PATH='$PATH`
		`Liste =`ls``
		`PC=`hostname: `$USER`

Affichage	`echo $VAR`	`echo $NOM` *affiche : Dupont*
exportation	`export VAR`	`export NOM`
destruction	`unset var`	`unset NOM`

Il faut bien respecter la syntaxe. Les espaces sont significatives. Ainsi l'instruction d'affectation suivante est fausse : `i = 1`. Il faut bien l'écrire `i=1`. Pour affecter une chaîne de caractère à une variable il faut entourer la chaîne par des apostrophes ou des double guillemets. Pour affecter à une variable le résultat d'un traitement (i.e. résultat d'une commande), on utilise les apostrophes inversés : `` ` ``.

Par défaut, une nouvelle variable n'est visible que du shell où elle a été créée. Elle est dite variable locale. l'exportation d'une variable la rend publique ; accessible aux autres logiciels. La commande *env* donne la liste des toutes les variables publiques.

Exercice 3 Variables d'environnement

1. Essayer les exemples d'affectation des variables donnés ci-haut et expliquer les résultats.
2. Dans un shell `bash` taper les commandes suivantes. Justifier les résultats obtenues de chaque commande.

```
>X1=3
>Y1=10
>Z1=4
>export Y1
>env |grep X1=
>echo $X1
>echo $x1
>env |grep Y1=
>unset Y1
>export X1
>bash
```

```
>env |grep X1=
>echo $Z1
>exit
>echo $Z1
```

3.. La variable `PS1` précise la valeur de l'invite de la ligne de commande. Changer l'invite pour qu'il devient de la forme: "login@nom de machine : "

II. L'éditeur vi

`vi` est le seul éditeur de texte présent sur tous les systèmes Unix. Ainsi maîtriser `vi` c'est s' assurer de pouvoir administrer n'importe quel système Unix. De plus et en dépit de son apparence brute cet éditeur permet d'enchaîner des commandes d'édition, de recherche et de remplacement de texte très performantes. `vi` est un éditeur *vidéo* mais pas graphique. Il gère les flèches de déplacement, le mode inverse vidéo, etc. n'oublier pas de renseigner la variable d'environnement **`TERM`** qui indique à l'éditeur le langage du terminal à utiliser (le plus souvent TERM prend la valeur **`vt100.`**)

L'éditeur **`vi`** a trois modes de fonctionnement :

1. *Mode de saisie ou insertion*. Dans ce mode l'utilisateur se limite à saisir du texte. Les commandes qui permettent d'y entrer sont par exemple *i* (pour insert) ou *a* (pour append). La frappe de la touche <*ESC*> permet de sortir de ce mode et entrer dans le mode de commande.
2. *Mode commande*. Dans ce mode l'utilisateur ne saisit que des commandes (souvent représentées par un caractère). C'est le mode utilisé par défaut à chaque nouvelle édition.
3. *Le mode EX*. C'est un mode de commande qui complète la première. Il permet de saisir des syntaxes plus longues et faire des actions plus complexes comme la substitution d'une chaîne de caractère par une autre dans tout le fichier. L'entrée à ce mode se fait en tapant le caractère « **:** » au niveau du mode de commande.

Nous résumons les principales commandes de vi dans les tableaux suivants :

Tableau 1: Mode de commande : Les déplacements

Commande	***Description***
h	Déplacement du curseur à gauche
l	Déplacement du curseur à droite
k	Déplacement du curseur en haut
j	Déplacement du curseur en bas
CTRL+F	Page suivante
CTRL+B	Page précédente
CTRL+D	Demi-page suivante
CTRL+U	Demi-page précédente
$	Déplacement enfin de ligne
0	déplacement en début de ligne

Commande	***Description***
w	mot suivant
b	mot précédent
e	fin du mot courant
G	Déplacement à la dernière ligne du fichier

Tableau 2: Mode de commande : Les insertions

Commande	***Description***
i ... <ESC>	Insère le texte saisi avant le caractère courant. A la fin du saisi appuyer sur la touche <ESC> pour revenir au mode de commande.
a ... <ESC>	Insérer le texte après le caractère courant
O .. <ESC>	Insérer des lignes avant la ligne courante
o .. <ESC>	Insérer des lignes après la ligne courante

Tableau 3: Mode de commande : effacement, remplacement & recherche

Commande	***Description***
x	effacer le caractère courant
dw	effacer le mot courant
dd	effacer la ligne courante
r	remplacer le caractère courant par un autre. Taper « r » puis le caractère de remplacement
/chaîne<CR>	Recherche de chaîne indiquée, en avant dans le fichier.
?chaîne<CR>	Recherche de la chaîne indiquée, en arrière dans le fichier
n	poursuivre la recherche
N	poursuivre la recherche en sens inverse.

Tableau 4 : Mode EX : édition

Commande	***Description***
*n*a	Ajout du texte après la ligne numéro *n*. a saisie doit être terminée par <ESC>
*n,m*c	remplacer les lignes de *n* à *m* par la nouvelle saisie. La saisie doit être terminée par <ESC>
*n,m*d	effacer les lignes de *n* à *m*.
1,$g/chaîne/d	effacer toutes les lignes contenant chaîne.

Commande	***Description***
w	sauvegarder le fichier
w!	forcer le sauvegarde
w *fichier*	enregistre le fichier sous le nom *fichier*
q	quitter l'éditeur
q!	forcer l'éditeur à quitter sans sauvegarder les modifications
wq	sauvegarder et quitter

Exercice 4 vi

1. Sauvegarder dans un fichier `liste.txt` le résultat de la commande ls -l /etc. nous allons éditer ce fichier en utilisant vi.
2. Effacer la première ligne de ce fichier
3. Insérer le texte : « Contenu de /etc » au débit du fichier et sauvegarder la modification.
4. Aller à la fin du fichier et ajouter le texte « fin de la liste »
5. Quitter l'éditeur sans sauvegarder les dernières modifications.
6. Éditer à nouveau le fichier et effacer toutes le lignes correspondant à des répertoires. sauvegarder les modifications.
7. Donner le diagramme d'états de l'éditeur vi.