Les commandes fondamentales de Linux

REMARQUE : SOUS LINUX (comme sous tout système UNIX) LES MINUSCULES ET LES MAJUSCULES NE SONT PAS ÉQUIVALENTES.

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 /sbin	/usr/bin /usr/local/bin /usr/sbin	Commande de base Commandes supplémentaires ajoutées par l'administrateur Commandes d'administration
/boot		Contient les fichiers du noyau Linux
/dev		Contient les fichiers particuliers aux périphériques
/etc	/etc/rc.d /etc/init.d	Contient les fichiers de configuration du système 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 /mnt/floppy /mnt/usb	Contient les répertoires des périphériques amovibles: CD/ disquette, USB
/proc		Répertoire dédié aux processus
/root		Répertoire personnel de l'administrateur
/tmp		Les fichiers temporaires
/usr	 /usr/include /usr/share/man /usr/local	Principal répertoire du système Sous répertoire des fichiers d'entête Sous répertoire de manuels Linux Logiciels installés par l'administrateur
/var	/var/log /var/spool /var/spool/mail /var/mail	Répertoire contenant la partie « variable » du système co mme les traces d'activités du système; les boîtes aux lett res, etc.

II. Commandes de base & utilitaires

Qu'est-ce qu'un shell ?

Sous UNIX ou LINUX: C'est un interpréteur de commande (en mode texte) la partie du système d'exploitation utilisé comme interface avec l'utilisateur. Il ressemble le plus souvent à ceci :

[root@NomMachine /root]$

Se déplacer dans les répertoires (Change Directory)

cd .. : Remonte d'un niveau
cd / : Retourne à la racine

Lister les fichiers d'un répertoire: (List Sorted)

ls -l : Permet de lister les attributs des fichiers
ls -d: Affiche uniquement les répertoires
ls -a : Liste tous les fichiers du répertoire y compris les fichiers cachés.
ls -m : Affiche les fichiers en les séparant par une virgule.
ls -t : Affiche les fichiers par date.
ls -lu : Affiche les fichiers par date du dernier accès et indique la date.
ls -F : Affiche les fichiers par type
ls -S : Affiche les fichiers triés par ordre de taille décroissante.
ls -X : Affiche les fichiers par type d'extension
ls -r : Affiche les fichiers en ordre alphabétique inverse
ls -alR / : Affiche tous les fichiers d'un système
ls -alR |grep doc : Affiche tous les fichiers contenant doc

Copier un fichier ou un répertoire: (copy)

cp : Demande s'il peut écraser le nom de fichier : répondre par Oui(y) ou Non (n)
cp -i : Avertit de l'existence d'un fichier du même nom et demande s'il peut ou non le remplacer.
cp -l : Permet de faire un lien en "dur" entre le fichier source et sa copie
cp -s : Permet de faire un lien "symbolique" entre le fichier source et sa copie
cp –p : Permet lors de la copie de préserver toutes les informations concernant le fichier.
cp -r : Permet de copier de manière récursive l'ensemble d'un répertoire et de ses sous répertoires
cp -b : Permet comme l'option -i de s'assurer que la copie n'écrase pas un fichier existant : le fichier écrasé est sauvegardé, seul le nom du fichier est modifié et cp ajoute un tilde(**~**) à la fin du nom de fichier

Supprimer des fichiers et répertoires: (remove & remove directory)

rm -d : Permet de supprimer un répertoire qu'il soit plein ou non
rm -r : Permet de supprimer un répertoire et ses sous répertoires
rm -f : Permet de supprimer les fichiers protégés en écriture et répertoires sans confirmation
rmdir : Supprime un répertoire
rmdir -p rep1/rep2/rep : Supprime le répertoire et ses sous répertoires associés

Créer des répertoires: (make directory)

mkdir : Crée un répertoire
mkdir -p rep1/rep2/rep3 :Crée un répertoire et ses sous répertoires associés

Déplacer ou renommer un fichier: (move)

mv -b : Va effectuer une sauvegarde des fichiers avant de les déplacer
mv -i : Demande pour chaque fichier et chaque répertoire s'il peut ou non le déplacer

Les commandes de visualisation :

cat nom_fichier : visualisation du contenu d'un fichier
more nom_fichier : filtre de pagination

Archivage

gzip : Le compactage et le décompactage des fichiers au formatgz
gzip -gv backup.gz : Compresse le répertoire courant et crée le fichier backup.gz
gzip -d backup.gz : Décompresse backup.gz
tar -zcvf /usr/pluton /home : Sauvegarde le répertoire home vers pluton
tar -zxcf /usr/pluton /home : Restaure pluton dans le répertoire home

Commande d'administration système Linux

chmod : Placer les droits d'utilisation des fichiers
chown : Désigner l'utilisateur et le groupe propriétaire des fichiers
adduser : Ajouter un utilisateur
passwd : Spécifier ou modifier un mot de passe
chfn : Décrire un utilisateur
userdel : Supprimer un utilisateur

La gestion des processus

top : permet de suivre les ressources que le processus utilise
ps : permet de connaître les processus actifs à un moment donné
pstree : permet d'afficher les processus sous forme d'arborescence et donc de voir leurs interdépendances
kill : Permet de tuer un processus en court : syntaxe **kill** [option] PID. Pour tuer le processus, je peux d'abord faire un **ps** -ax pour connaître le numéro du PID et ensuite si par exemple le PID est 3600, je peux tuer la connexion en faisant :
[root@localhost/root]# **kill** 3600

Droits d'accès des fichiers sous Linux

Etant donné qu'Unix est système d'exploitation multi-utilisateurs, on distingue trois catégories d'utilisateurs :

1. **u** : propriétaire (**u**ser : celui qui a créé le fichier).
2. **g** : le **g**roupe (les autres utilisateurs appartenant au même groupe du propriétaire)
3. **o** : les autres, (**o**thers, le reste du monde, ni propriétaire du fichier, ni membre du même groupe que le propriétaire du fichier).

Ainsi, chaque fichier possédé trois types de droits:

- **r** : lecture (read)
- **w** : écriture (write)
- **x** : exécution (execute)
- **-** : aucun droit

Pour chaque utilisateur (u, g ou o) sont définies ces trois permissions

- le propriétaire (u) dispose ou non de la permission : r, w, x sur un fichier ;
- le membre du groupe (g) dispose ou non de la permission : r, w, x sur un fichier ;
- tous les autres utilisateurs (o) disposent ou non de la permission : r, w, x sur un fichier

- Les droits sont affichés (**ls –l**) par une série de 9 caractères, associé 3 par 3 (rwx rwx rwx) définissent les droits des 3 identités (u, g et o).
- Un dixième caractère indiquent les 4 types de fichier.
  - ✓ **-** : fichier ordinaire
  - ✓ **d** : répertoire (directory)
  - ✓ **l** : Lien symbolique (link)
  - ✓ **c** ou **b** : Spécial

ex : - rwx rwx rwx

chmod : La commande **chmod** (**ch**ange **mod**e, changer les permissions) permet de modifier les permissions sur un fichier. La commande chmod peut être utilisée de deux manières:

- soit en précisant les permissions de manière octale
  - ✓ r = **4** si actif ou 0 si inactif
  - ✓ w = **2** si actif ou 0 si inactif
  - ✓ x = **1** si actif ou 0 si inactif
- soit en ajoutant ou en retirant des permissions à une ou plusieurs catégories d'utilisateurs à l'aide des symboles r, w et x.

Comment puis-je en savoir plus sur les commandes ?

Toutes les commandes possèdent une "page de manuel" qui vous est livrée avec linux :
man cp : obtiendrez toute la documentation de **cp**. Pour quitter la page de manuel, vous pouvez appuyer à n'importe quel moment sur la touche "**q**".

III. 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 doubles guillemets. Pour affecter à une variable le résultat d'un traitement (i.e. résultat d'une commande), on utilise les apostrophes inversées : `. 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.