Laboratoire de téléinformatique
Linux 1 (Commandes de base)

Objectif

Familiarisation du système d'exploitation Linux pour que vous vous sentiez à l'aise dans la suite des laboratoires puisqu'un grand nombre de manipulations vont s'effectuer sur ce système.

Introduction

Le laboratoire de téléinformatique comporte un certain nombre de PC qui sont équipés de 2 systèmes d'exploitation : Windows et Linux Redhat.

Les manipulations que nous proposons ici peuvent normalement être effectuées sur une autre distribution que RedHat (SuSE, Slackware, Debian,…)

La philosophie de Linux (vient du nom de son inventeur Linus Torvalds, un programmeur finlandais) vient du monde UNIX où il n'était pas coutume d'utiliser des fenêtres. Les commandes d'administration se font en ligne. Il y a une grande liberté pour créer de nouvelles commandes, contrairement au monde Windows. Ce système est complètement ouvert et les possibilités sont presque illimitées. Il faut en être conscient et être prudent quand on essaie de nouveaux « trucs ». Ce système d'exploitation s'adresse essentiellement à des professionnels qui savent ce qu'ils font. Au début, c'est un petit peu plus difficile mais ensuite nous verrons que nous pouvons accéder à toutes les ressources du système, ce qui nous facilite grandement la tâche dès que nous devons développer du code, des protocoles,…

Le but de ce laboratoire est comme nous l'avons dit, de se familiariser au système (édition et manipulation de fichiers, compilation,…), et non de développer de nouveaux outils, dans un premier temps. Par contre nous vous encourageons fortement à essayer d'autres commandes. Internet regorge de documentation à propos de Linux. L'étudiant curieux a largement de quoi assouvir son intérêt.

Comment commencer ?

Premièrement il faut vous mettre sur une station de travail (PC) et démarrer sous Linux (voir annexe pour voir comment entrer et quel PC choisir):

```
Login : labo
Password : labo
```

A ce stade il faut ouvrir une fenêtre de commande et nous pouvons commencer.

Tout d'abord on aime bien savoir où on se trouve. Il faut faire la commande suivante pour le savoir :

```
>pwd
```

Ensuite on aime bien savoir ce qu'il y a dans le répertoire dans lequel nous nous trouvons :

```
>ls -a
```

Le répertoire dans lequel vous vous trouvez est votre répertoire personnel (« home directory ») si vous avez un compte mais il s'agit d'un répertoire banalisé si vous avez accédé au système à l'aide d'un compte commun (comme dans notre cas avec « labo »). Remarque : faites attention à la casse. Le système est sensible aux majuscules/minuscules.

On peut changer de répertoire avec la commande `cd` :

```
>cd /etc
```

nous fait accéder au répertoire /etc. De façon générique la commande s'écrit

```
>cd <nom du répertoire>
```

Il y a néanmoins des simplifications possibles. Quand nous voulons remonter l'arborescence (par exemple aller de `/etc/hosts` à `/etc`), nous pouvons utiliser la commande

```
>cd ..
```

Depuis n'importe quel emplacement il est possible de revenir dans son répertoire de base (« home directory ») avec la commande

```
>cd ~
```

Question: Comment créer un répertoire (voir en annexe 1) ?

Maintenant nous savons comment nous déplacer dans tous les répertoires et lister le contenu des répertoires. Toutes les commandes ont un certain nombre d'options (par exemple `ls -a`, `ls -al`, `la -F` par rapport à `ls`). Si vous voulez connaître ces options facilement il suffit de consulter les « man pages » (vient de *manual* pages) avec la commande suivante :

```
>man ls
```

Et vous verrez apparaître une description de la commande avec tous les arguments possibles. C'est certainement la commande la plus utilisée. C'est une bonne habitude d'utiliser cette commande dès qu'on hésite.

Avec Linux, on peut rediriger une sortie (ou une entrée) vers d'autres composants, vers des fichiers par exemple. Ces redirections sont utiles quand on a du texte. Nous pouvons créer un petit fichier en redirigeant la sortie d'une « man page » par exemple :

```
>man ls > man_ls.txt
```

Si nous observons maintenant le répertoire (avec ls) nous voyons qu'un nouveau fichier est apparu : `man_ls.txt`. Maintenant nous pouvons manipuler ce fichier. Copions-le avec la commande cp:

```
>cp man_ls.txt man_ls2.txt
```

Nous avons deux fichiers supplémentaires dans notre répertoire (vérifiez-le). La commande `cp` effectue la copie d'un fichier vers une destination spécifié :

```
>cp <source-file> <destination file>
```

Nous pouvons aussi détruire un fichier avec la commande `rm` :

```
>rm man_ls2.txt
```

Avec cette commande néanmoins il faut être **extrêment** prudent. Vous pouvez détruire une arborescence complète.

Question : Que fait `rm -rf *` ? Regardez dans les *man* pages.

Question : Que fait `mv ? tail ? head ?`

Nous voulons peut-être voir le contenu de notre fichier `man_ls.txt`. Comment faire ? Il y a plusieurs possibilités : vous pouvez utiliser more, cat, less avec la syntaxe suivante :

```
>more man_ls.txt
```

Vous pouvez presser sur `<return>` (pas-à-pas) ou `<espace>` (page par page) pour faire avancer. Si vous voulez quitter le fichier en cours de route, pressez simplement « q » (quit).

Toutes ces commandes (tâches) sont appelés des processus avec Unix. On peut également connecter la sortie d'un processus à l'entrée d'un autre. Cette connexion se fait avec le symbole « | » (*pipes*). On peut ainsi « chaîner » des commandes.

Une autre commande très pratique est la commande « grep » qui permet de chercher des bouts de chaînes de caractère dans des fichiers texte. On peut chercher le mot « list » dans notre fichier `man_ls.txt` :

```
>cat man_ls.txt | grep list
```

On peut aussi rechercher un fichier dans une arborescence (pas toujours évident à faire manuellement, surtout quand l'arborescence comporte de nombreux répertoires). On donne la racine de l'arborescence dans laquelle on veut chercher et une expression désignant le fichier qu'on désire rechercher. Si on désire rechercher notre fichier depuis la racine du système (tout en haut de l'arborescence), il faut faire la commande suivante :

```
>find / -name man_ls.txt -print
```

Exercice: Localisez les fichiers `include` (utilisez par exemple `stdio.h`).

Remarque: Vous pouvez aussi utiliser la commande `locate`.

Pour écrire un petit fichier, nous pouvons utiliser plusieurs éditeurs de texte. Le plus ancien, disponible sur tous les systèmes Unix est `vi`. Les utilisateurs Mac et Windows vont détester cet éditeur car il consiste en une suite de commandes difficiles à digérer. Néanmoins il reste le seul éditeur à pouvoir éditer de manière fiable un fichier à travers une connexion *telnet*. Toutes les commandes d'édition de `vi` se retrouvent dans le cadre du shell et permettent d'éditer les lignes de commande, ce qui s'avère très utile.

Nous allons maintenant approfondir un point très important, à savoir la manière dont on peut protéger l'accès aux fichiers et aux répertoires. L'administrateur a tous les droits sur tous les fichiers. Si on effectue la commande `ls -l`, nous obtenons par exemple :

```
>ls -l
total 67
drwxr-wr-x   2 root      root   2048 Aug 14 2001   bin
drwxr-xr-x   3 root      root   1024 Oct   8 10:15 boot
……
```

Que signifie les lettres de la première colonne (drwxr…)? En fait il s'agit des droits d'accès aux fichiers. On a trois groupes. En allant la gauche vers la droite, on a successivement les accès pour le propriétaire, les membres du groupe et les autres utilisateurs. Le fait que tous les caractères soient appondus n'arrange pas les choses mais on peut diviser la première ligne de la façon suivante : rwx r-w r-x. Dans ces trois groupes, on a trois lettres chaque fois. La première lettre r signifie « lecture », la deuxième (w) signifie « écriture » et la troisième (x) signifie « exécution ». Donc si on reprend l'exemple ci-dessus, on voit que « boot » est accessible en lecture, écriture et exécution pour le propriétaire (root), accessible en lecture et exécution pour les membres du groupe et accessible en exécution seulement pour les autres utilisateurs.

Il est possible de changer ces accès avec la commande `chmod`. On peut ajouter des droits (« + »), en retirer (« - ») ou en remplacer (« = ») pour le propriétaire (« u »), le groupe (« g »), les autres (« o »), ou tout le monde (« a »).

Question : Que signifie la commande suivante ?

```
>chmod a+r, a+w boot
```

On peut aussi donner une valeur en octal pour l'attribution des accès :

```
rwx=7, rw-=6, r-x=5, r--=4, -wx=3, -w-=2, --x=1, ---=0.
```

Question : Que signifie la commande suivante ?

```
>chmod 760 boot
```

La commande `umask` permet de définir des permissions par défaut. Tapez

```
>umask
```

et vous verrez quels sont les droits qui sont supprimés. Si vous obtenez 022 par exemple, ça signifie que vous ne retirez aucun droit au propriétaire, que vous retirez le droit à l'écriture aux autres.

Question : Que signifient les commandes suivantes ?

```
>umask 000
>umask 777
```

On peut aussi faire changer le propriétaire d'un fichier avec la commande `chown`. La syntaxe est la suivante :

```
>chown <nouveau propriétaire> fichier
```

Revenons à nos processus. Sur Linux, chaque processus a un identificateur numérique, le PID (process identifier). Ce numéro est important car il identifie de façon unique le processus. Nous pouvons faire la liste des processus actifs avec la commande `ps` (voir les « man pages » pour les options).

Question : Comment afficher tous les processus actifs ?

Nous pouvons supprimer des processus avec la commande `kill`. Il faut indiquer à la commande le PID du processus à supprimer. La syntaxe est la suivante : `>kill -9 PID`. Vous pouvez lancer un processus et le supprimer ensuite, par exemple un éditeur de texte donné par le système.

Faites toutefois attention quand vous supprimez des processus car vous pouvez faire des dégâts car le processus que vous supprimez ne va pas pouvoir terminer sa tâche ! On peut utiliser ceci quand on écrit un programme comportant une boucle infinie par exemple.

Question : Quel processus avez-vous créé et supprimé ?

Une autre notion dans le monde Unix concerne les « deamon » qui sont des processus qui fonctionnent en permanence. En principe, les « deamon » sont terminés par la lettre d. Ainsi ftpd représente le programme du serveur ftp (httpd pour le serveur http).

Question : Pourquoi les serveurs sont-ils des deamon ?

Pour faire de la sauvegarde, la vieille commande `tar` est toujours très utilisée. La syntaxe est la suivante :

```
>tar <options> <fichier spécial> <fichiers à sauvegarder>
```

Les options principales sont les suivantes : c (création d'une nouvelle archive), f (lorsque le support n'est pas le support pas défaut), v (archivage verbeux, affichage des fichiers traités), x (restauration d'une archive), z (compression).

Pour créer une archive nous pouvons faire par exemple :

```
>tar cvfz fichierarchive.tgz fichier1 fichier2 rep1
```

L'archive compressée (fichierarchive.tgz) comprendra donc les fichier 1 et 2 ainsi que tous les fichiers du répertoire 1. le nom de chaque fichier compressé apparaitra à l'écran.

```
>tar xvfz fichierarchive.tgz
```

Cette commande va décompresser et extraire tous les fichiers de l'archive compressée et crée automatiquement les sous-répertoires éventuellement nécessaires. Le nom de chaque fichier apparaît à l'écran.

Si vous voulez personnaliser votre environnement, vous pouvez faire usage de la commande `env` (dépend du shell utilisé).

Question : Que voyez-vous apparaître quand vous tapez la commande `env` ?

Une variable est très intéressante pour nous : `PATH`. Elle dit au système où se trouvent les programmes exécutables que vous désirez exécuter. Si vous avez un exécutable dans un autre répertoire, non listé par cette variable, le système ne va pas le trouver. Pour faire exécuter votre programme il faudra donner le chemin complet, par exemple `/root/cours/demo`. Ou encore, si le fichier est dans le répertoire dans lequel vous vous trouvez : `./demo`.

En général, les variables d'environnement ne sont définies que pour la session en cours. Si nous voulons que la modification apportée à PATH soit définitive il faut modifier le fichier « `.bash_profile` » (ou analogue, lister avec `ls -l`) dans le « home directory » (root, robert,…). Examinez ce fichier.

Il y a encore beaucoup de choses que nous pouvons faire mais maintenant nous allons nous intéresser à l'écriture du scrits. Un script est un petit fichier texte dans lequel nous mettons une liste de commandes à effectuer et qui est exécutable. Il est écrit dans le langage Shell dans lequel un jeu de commandes est défini. Il existe par exemple diverses instructions de programmation (if, for,…) permettant d'écrire des petits programmes dans un langage proche de la syntaxe du langage C.

Exercice: Ecrire un petit script très simple (par exemple : `ps -ef`) et l'exécuter. Que faut-il faire pour qu'il soit exécuté ? Renommez ce fichier, est-il exécutable ? Que se passe-t-il ? Commentez. Pour ceci il faut utiliser un éditeur comme vi par exemple : vi <nom du fichier>, pressez « i » pour insérer du texte, tapez et quand vous avez fini : tapez :wq pour enregistrer le fichier et sortir de l'éditeur.

La commande `df` renseigne sur l'occupation des « file systems » montés. La commande `du` donne la taille en blocs de 512 octets d'un certain ensemble de répertoires ou de fichiers. Sans argument, `du` donne la taille de chaque sous-répertoire du répertoire courant, y compris ce répertoire.

Question : Quelle est l'utilisation de votre disque ?

La communication sous Linux

Avant d'examiner les fichiers de configuration d'interfaces, dressons la liste des fichiers de configuration primaires utilisés pour configurer le réseau. Le fait de comprendre le rôle joué par ces fichiers dans la mise en place de la pile réseau peut s'avérer utile lors de la personnalisation d'un système Red Hat Enterprise Linux

Les fichiers de configuration de réseau primaire sont les suivants :

/etc/hosts
L'objectif principal de ce fichier est de résoudre les noms d'hôtes n'ayant pu être résolus d'une autre façon. Il peut également être utilisé pour résoudre des noms d'hôtes sur de petits réseaux ne disposant pas de serveur DNS. Quel que soit le type de réseau utilisé par l'ordinateur, ce fichier doit contenir une ligne spécifiant l'adresse IP du périphérique de bouclage (loopback) (**127.0.0.1**) en tant que **localhost.localdomain**. Pour obtenir davantage d'informations, consultez la page de manuel de **hosts**.

/etc/resolv.conf
Ce fichier précise les adresses IP des serveurs DNS et le domaine de recherche. À moins d'être configuré autrement, les scripts d'initialisation du réseau sont contenus dans ce fichier. Pour

obtenir davantage d'informations sur ce fichier, consultez la page de manuel de **resolv.conf**.

/etc/sysconfig/networking/ifcfg-*<interface-name>*
Pour chaque interface réseau, il existe un script de configuration d'interfaces correspondant. Chacun de ces fichiers fournit des informations spécifiques à une interface réseau particulière.

Si nous tapons la commande `route`, toute la table de routage va s'afficher.

Question : Quelle est la table de routage de votre PC ? Quelles sont les options de la commande `route` ?