Lab - Utiliser les commandes Shell de base GNU/Linux

Objectifs

Apprendre à naviguer et à gérer des fichiers et des répertoires.

Contexte

Un shell sous Linux est une interface en ligne de commande qui permet à l'utilisateur d'interagir avec le système d'exploitation en utilisant des commandes textuelles. Il sert d'intermédiaire entre l'utilisateur et le noyau du système, permettant ainsi d'exécuter des programmes, de manipuler des fichiers, de gérer des processus et d'effectuer diverses tâches système. Il existe différentes implémentions Shells, les plus courantes sont : sh (Bourne shell), bash(Bourne again shell), csh (C Shell), ksh (Korn shell) Zsh…

Le shell **BASH**, développé par **GNU**(sous licence **GPL**) est le Shell par défaut sous le système GNU/Linux.

Les **commandes shell de base** sont souvent nommées par de courtes abréviations de leurs fonctions. Vous allez utiliser entre autres la commande ls pour répertorier les fichiers, la commande cp pour copier des fichiers, la commande mv pour déplacer ou renommer des fichiers,la commande rm pour supprimer des fichiers,la commande mkdir pour créer des répertoires, la commande rmdir pour supprimer des répertoires (s'ils sont vides) et la commande rm pour supprimer récursivement des répertoires contenant des fichiers.

Ressources requises

Ordinateur équipé d'Ubuntu Desktop LTS dans une machine virtuelle VirtualBox.

Ouvrez une fenêtre de terminal dans Ubuntu.

Connectez-vous à Ubuntu à l'aide de vos informations d'identification :

Cliquez sur l'icône du terminal pour ouvrir une fenêtre de terminal.

À l'invite de commande (shell), vous allez manipuler les commandes de base GNU/Linux .

Le prompt

Le prompt shell est l'invite de commande affichée par le shell pour indiquer à l'utilisateur qu'il est prêt à recevoir une commande.

Le prompt shell affiche généralement des informations telles que le nom de **l'utilisateur,** le nom de **l'hôte** (machine), le chemin du répertoire courant, etc.... Cela peut être personnalisé selon les préférences de l'utilisateur ou les configurations du système.

Dans de nombreux shells, le prompt est généralement suivi d'un symbole appelé "prompt de commande" ou "**invite de commande**", qui indique à l'utilisateur qu'il peut entrer une commande. Le dernier symbole le plus couramment utilisé est le signe dollar `$,` bien que cela puisse varier en fonction du shell et des configurations personnalisées.

Exemple de prompt shell :

```
user@hostname:~/directory$
```

Dans cet exemple :

- `user` est le nom de l'utilisateur actuel.
- `hostname` est le nom de l'hôte (le nom de l'ordinateur).
- `~/directory` est le chemin absolu du répertoire courant.
- `$` est le prompt de commande.

Dans de nombreux shells, le caractère `#` est utilisé pour indiquer le début d'un commentaire sur une ligne. Tout ce qui suit le caractère # sur cette ligne est ignoré par le shell lors de l'exécution des commandes. Les commentaires sont utiles pour ajouter des explications ou des annotations dans les commandes, sans affecter leur fonctionnement.

Syntaxe des commandes shell

La plupart des commandes Linux suivent un modèle de syntaxe simple:

```
commande<espace>[options…]<espace>[arguments…]
```

Vous commencerez par exécuter trois commandes: `pwd, cd et ls.` Ces commandes, dans leurs formes les plus simples, sont exécutées sans aucune option ni argument.

Navigation dans le système de fichiers :

Le répertoire personnel ou d' accueil est le répertoire dans lequel se trouve actuellement votre fenêtre de terminal Shell. Il s'agit également du répertoire courant.

La commande `pwd` affiche le répertoire où se trouve actuellement l'utilisateur dans le système de fichiers

```
utilisateur@localhost:~$ pwd
/home/utilisateur
utilisateur@localhost:~$
```

La sortie de la commande `pwd` (`/home/utilisateur` dans cet exemple) est appelée un **chemin**. La première barre oblique ou slash `/` représente le répertoire **racine(root)**, le répertoire **`home`** est un répertoire sous le répertoire **`racine`** et **`utilisateur`** est un répertoire sous le répertoire **`home`**.

Utilisez la commande `cd` (change directory) pour naviguer entre les répertoires.

```
utilisateur@localhost:~$cd      #Accéder au répertoire personnel de l'utilisateur
utilisateur@localhost:~$cd ..   #Accéder  au répertoire parent
utilisateur@localhost:~$cd /    # Accéder  au répertoire racine
```

Chemin absolu et chemins relatifs

Chemin absolu
pour accéder au repertoire `Documents` en utilisant le chemin absolu (à patir de la racine **/**)

```
utilisateur@localhost:~$cd /home/utilisateur/Documents #Remplacer utilisateur par votre identifiant (login)
```

Changer de répertoire avec **des chemins relatifs :**

```
utilisateur@localhost:~$cd Documents
```

Revenir au répertoire parent

```
utilisateur@localhost:~/Documents$cd ..
```

Lister le contenu des répertoires :

Utilisez la commande `ls` (list) pour afficher le contenu d'un répertoire.

```
ls          # Afficher le contenu du répertoire courant
ls -l       # Afficher le contenu en détaillé (liste longue)
ls -a       # Afficher tous les fichiers, y compris les fichiers cachés
ls /home/   # Afficher le contenu d'un répertoire spécifique
```

Exécutez la commande `ls` à l'invite et appuyez sur Entrée; cette commande listera les fichiers et répertoires qui se trouvent dans le répertoire courant:

```
utilisateur@localhost:~$ ls

Desktop    Downloads  Pictures  Templates  Documents  Music
Public  Videos

utilisateur@localhost:~$
```

Il est généralement possible de combiner plusieurs options dans n'importe quel ordre. Exécutez la commande `ls` suivante avec deux options :

```
ls -al
```

Cette sortie de la commande ls a l'effet combiné de toutes les options spécifiées:

L'option `-a` Inclut les fichiers commençant par un point (fichiers cachés).

L'option `-l` Affiche les proporietés des fichiers (fournit le format de liste longue).

Certaines commandes utilisent des arguments qui ne nécessitent aucune option. Par exemple, exécutez la commande `touch` avec un nom de fichier. Cela va créer un **fichier vide** . Exécutez ensuite la commande `ls` pour confirmer qu'il a créé le fichier:

```
utilisateur@localhost:~$ touch lpicfile.txt

utilisateur@localhost:~$ ls

Desktop    Downloads  Picture s  Templates  lpicfile.txt

Documents  Music      Public    Videos

utilisateur@localhost:~$
```

De même, vous pouvez exécuter la commande `rm` avec le même nom de fichier comme argument pour supprimer ce fichier:

```
utilisateur@localhost:~$ rm lpicfile.txt

utilisateur@localhost:~$ ls
```

```
Desktop  Documents  Downloads  Music  Pictures  Public  Templates
Videos    admin@localhost:~$
```

d'autres commandes, comme la commande `cp`, nécessitent au moins deux arguments. La commande `cp` nécessite la copie du fichier source et le fichier de destination (où copier le fichier). Par exemple, exécutez ce qui suit pour copier le fichier `hosts` du répertoire `/etc` dans le répertoire actuel avec un nouveau nom de `myhosts`

```
utilisateur@localhost:~$ cp /etc/hosts myhosts
utilisateur@localhost:~$ ls
Desktop    Downloads  Pictures  Templates  myhosts  Documents
Music Public    Videos

utilisateur@localhost:~$
```

La commande `mkdir` vous permet de créer un répertoire.

Pour créer un répertoire nommé `testdir` dans le répertoire actuel et vérifier que le nouveau répertoire existe, exécutez les commandes suivantes:

```
utilisateur@localhost:~$mkdir testdir

utilisateur@localhost:~$ls –d testdir/
```

Pour créer plusieurs répertoires dans le répertoire actuel et vérifier qu'ils sont créés, exécutez les commandes suivantes:

```
utilisateur@localhost:~$mkdir dir1 dir2 dir3 dir4 dir5

utilisateur@localhost:~$ls –d dir?
```

La commande `rmdir` est utilisée pour supprimer les répertoires vides.

```
utilisateur@localhost:~$rmdir testdir

utilisateur@localhost:~$ls –d testdir/
```

La complétion

Le shell fournit une fonctionnalité conçue pour éviter les erreurs de frappe et économiser l'effort de frappe; cette fonctionnalité est appelée **complétion de commande**. L'achèvement de la commande vous permet de taper quelques caractères d'une commande (ou son argument de nom de fichier), puis d'appuyer sur la **touche Tab**. Le Shell completera automatiquement la commande (ou son argument de nom de fichier) pour vous.

Par exemple, tapez d'abord `ec`

```
utilisateur@localhost:~$ ec
```

Ensuite, appuyez sur la touche ⇥Tab, et le shell terminera automatiquement la commande `echo` pour vous (il place même un espace après la commande afin que vous puissiez immédiatement commencer à taper l'option ou l'argument):

```
utilisateur@localhost:~$ echo
```

Lorsque les caractères saisis sont insuffisants pour correspondre à une seule commande, tous les choix possibles s'affichent si vous appuyez deux fois sur la touche Tab

Exemple:

Tapppez `ca` puis apuyez deux fois sur la touche Tab comme suite :

```
utilisateur@localhost:~$  ca

cal                capsh              cat
cautious-launchercalendar           captoinfo          catchsegv
caller             case               catman
utilisateur@localhost:~$  cal

  April 2020

 Su Mo Tu We Th Fr Sa
```

```
1  2  3  4

5  6  7  8  9 10 11

12 13 14 15 16 17 18

19 20 21 22 23 24 25

  26 27 28 29 30

utilisateur@localhost:~$
```

Pour accéder aux pages de manuel d'une commande exécutez la commande man suivi par le nom de la commande `man commande`

Exemple

```
utilisateur@localhost:~$man ls
```

Voici un tableau de quelques options qui vont vous permettre de manipulation les pages du manuel :

Commande	Fonction
Return (ou Enter)	Descendre d'une ligne ;
espace	Descendre d'une page.
/terme	Recherche du mot terme.
n	trouver l'élément de recherche suivant.
1G	Aller au début.
G	Allez jusqu'au bout.
h	Afficher l'aide.
q	Quitter la page man

Lorsque vous voulez quitter la page de manuel appuyez sur la **touche q**