جامعة مولاي إسماعيل UNIVERSITÉ MOULAY ISMAÏL المدرسة العليا للأساتذة ÉCOLE NORMALE SUPÉRIEURE	**TP1**: SYSTEME D'EXPLOITATION 2 ➢ **Introduction au mode Linux** ➢ **Initiation à l'environnement GNU-Linux** ➢ **Quelques premières commandes Linux**	**(S5)** **PAR MR** : M.MOUKHAFI

I- Introduction

Un système GNU-**Linux** est un système d'exploitation répondant à un standard garantissant un ensemble de fonctionnalités utilisables de manière normalisée. Par exemple l'ensemble des commandes étudiées dans ce TP sur **Linux Ubuntu** restent valables sur le système Unix BSD (à la base d'OS X) ou sur le système Unix Solaris etc...

Cette séance est un 1er contact avec les commandes et schémas usuels d'une utilisation de système **linux** en ligne de commandes : nous envoyons des ordres et recevons des réponses en mode textuel dans une console ou terminal selon le vocabulaire Linux.

Que vous utilisiez les machines avec connexion à distance sur Linux, ou que vous utilisiez un Linux installé sur une machine personnelle, vous serez sur un Bureau avec icônes, lanceur d'applications, navigateur de fichier etc... Tout le confort moderne. C'est pratique mais ce n'est PAS ce qui nous intéresse car cette interface de haut niveau est très changeante (d'un système à l'autre, selon les versions, les options...) et ne constitue qu'un emballage superficiel du système : les commandes en terminal sont standardisées et présentent un intérêt pour des compétences à long terme.

II- Lancer votre session Linux-Ubuntu et ouvrir un terminal

Sur votre machine:

Accédez au menu **Applications**, dérouler → **Accessoires** → **Terminal**

Comme nous allons manœuvrer dans le système de fichiers (arborescence de répertoires) à partir du terminal, il peut être utile de vérifier ce qui se

passe depuis une interface plus intuitive : menu Raccourcis en haut du bureau → Poste de travail.

III- Le terminal et l'interpréteur de commande : shell

Le terminal ou la console est une interface textuelle qui permet à l'utilisateur d'entrer les commandes à faire exécuter par le système.

Une ligne de commande commence par le **nom** de la commande et est éventuellement suivie par des **options** ou **paramètres**. La commande est validée (l'action est déclenchée) par retour ligne (touche Entrée). Les lignes de commande ne sont pas directement transmises du terminal au système mais passent par l'interpréteur de commande ou **shell** qui est le véritable interlocuteur, le terminal n'étant qu'une boite de saisie et d'affichage de texte. Le **shell** vérifie la cohérence syntaxique de la ligne, gère le contexte des commandes (répertoire courant où a lieu l'action...) et lance effectivement l'exécutable associé à la commande. Dans le terminal vous devez voir une invite de commande ou prompt signalant que le shell est prêt à recevoir vos ordres. Les commandes s'écrivent directement après ce prompt. Vous validerez les commandes suivantes en vérifiant que le résultat décrit correspond bien à ce qui est indiqué.

1- QUI SUIS-JE et Où je suis dans l'arborescence ?

A- Taper **whoami :** ça renvoie le nom de votre l'utilisateur (le logging)

B- Taper **pwd** : ça indique le répertoire de travail le "répertoire courant"

Si vous lancer des commandes sur des fichiers ou répertoires sans préciser leurs emplacement, se sont les fichiers (et répertoires) de répertoire courant qui vont subir ces commandes **par default**

Sous GNU/Linux (aussi Unix) un répertoire (ou un fichier) est désigné par un chemin (path) qui est la séquence des répertoires qu'il faut traverser depuis la racine / pour y accéder en descendant l'arborescence.

echo : pour afficher un texte**,** une valeurs d'une variables

C- Taper **echo bonjour mes collègues**

$** : Quand on utilise le symbole **$ avant une variable c'est qu'on veut manipuler (afficher, affecter …) la valeur de cette variable

1.1 Exemple :

D- Taper dans l'invite de commande: **x=25**

E- Puis taper : **echo $x**
F- Taper aussi **: echo $HOME**

Cette dernière, nous renvoie l'arborescence du répertoire dédié à l'utilisateur (chaque utilisateur a son répertoire personnel). Au démarrage d'un nouveau terminal, le **HOME** porte la valeur répertoire courant.

La commande **echo** demande un affichage immédiat des paramètres. Dans la commande suivante, les noms commençant par **$** sont des **variables du système**, qui sont des chaînes, on demande leur valeur.

G- echo C'est moi l'utilisateur $USER mon shell est le $SHELL

Cette dernière commande semble ne pas marcher : les guillemets simples utilisés en apostrophes encadrent **'est moi l'** comme une chaîne à utiliser telle quelle (strong quoting).

- On peut adopter une "quotation" plus faible pour garder la substitution des variables par leur contenu mais échapper les guillemets simples :
  H- echo " C'est moi l'utilisateur $USER mon shell est le $SHELL "

- On peut adopter une banalisation à ces guillemets simple en utilisant le back-slash « **** » pour éviter leur effet :
  I- echo C\'est moi l\'utilisateur $USER mon shell est le $SHELL

Le **shell** que nous utilisons est le **BASH**, un shell très répandu mais pas le seul à exister il y a :

- sh (Bourne shell)
- bash (Bourne again shell)
- csh (C shell)
- Tcsh (Tenex C shell)
- ksh Korn shell
- zsh Zero shell

1.2 Quelques commandes :

J- ls

Lister le contenu du répertoire courant

K- ls -l

(l comme Liste) Lister le contenu du répertoire courant, option longue = infos détaillées

La colonne de gauche indique pour chaque fichier ou dossier :

d pour "directory" → dossier

r pour "**read**" droit en lecture **w** pour "**write**" droit en écriture **x** pour "**eXecute**" droit en exécution

Le **1er** bloc **rwx** concerne le **propriétaire**,

Le **2ème** concerne le **groupe** (ici nous n'entrons pas dans les détails)

Le **3ème** concerne tout le **monde**

1.3 Exemple : **-rw-r--r--** Fichier (pas de **d**) lecture pour tous, écriture propriétaire, pas d'exécution

L- ls –a

Lister le contenu du répertoire courant, option tous (**all**) = afficher les fichiers cachés Les fichiers ou répertoires commençant par**. (point)** sont des fichiers de configuration, cachés par défaut.Les répertoires. **(Point)** et **..** (**Deux points**) sont des répertoires spéciaux qui représentent respectivement, le répertoire courant et le répertoire parent (au dessus du répertoire courant dans l'arborescence)

M- clear : sert au nettoyage de la console**.**

2- Aides (whatis pour what is this**, man** pour manual**) !**

N- whatis ls (cp, mv, mkdir cd, rm ….)
O- whatis whatis

La commande **whatis** donne un bref descriptif du rôle joué par la commande donnée en paramètre

P- man ls (cp, mv, mkdir cd ….)

Pour un descriptif plus complet on utilise le manuel, qui explique en détail toutes les options d'une commande. Vous pouvez retrouver **-a** et -l. Inutile de s'attarder (on ne va pas apprendre tout ça!) appuyer sur q pour quitter le manuel et revenir au shell.

Essayer

Q- man man (manuel du manuel) et jeter un œil à la 1ère page : on comprend pourquoi Unix est resté longtemps un système réservé aux professionnels et aux passionnés.

3- Changer de direction et aller dans d'autres répertoires !

Vous allez naviguer dans le système de fichier, ajouter des répertoires, créer/voir des fichiers...

R- ls ..
S- pwd

Affiche le contenu du répertoire parent du répertoire courant. Le répertoire courant est inchangé.

T- ls .. -l -a
U- pwd

Affiche la totalité du répertoire parent avec infos détaillées (on peut écrire-la au lieu de -l -a) Le répertoire courant est inchangé.

V- cd ..
W- pwd
X- ls

cd = Change Directory : Change le répertoire courant. Ici on atterrit dans le répertoire parent.

Y- cd /
Z- ls -la

Aller à la racine du système de fichier et voir en détail tout ce qui s'y trouve.

AA- cd bin
BB- ls

Aller dans le répertoire des commandes exécutables **bin** (binaries) et voir les commandes disponibles. Il y en a davantage à d'autres emplacements, voir par exemple dans le répertoire **/usr/bin : ls /usr/bin** (la commande **which** permet de savoir où se trouve une commande)

CC- cd ~ : Retour au répertoire HOME : le ~ représente votre répertoire personnel HOME
DD- Taper **pwd** pour s'assurer

On peut exécuter une commande sur un fichier ou répertoire existant dans un autre endroit (répertoire) à partir de notre emplacement : Par exemple si on veut lister le contenu de **/lib** on reste dans le HOME et on fait **ls /lib** directement :

A noter que les chemins qui commencent par **/** sont des chemins **absolus** (on part de la racine) alors que ceux qui ne commencent pas par **/** sont des chemins **relatifs** (on part du courant)

Par exemple **/tmp** se réfère au répertoire **tmp** à la racine du système tandis que **tmp** se réfère à un répertoire **tmp** supposé être dans le répertoire courant.

4- Historique des commandes et Complétion

Par utilisation de **[ctrl+p]** ou de **[echap +k]** ou aussi des **flèches haut** et **bas** on peut voir (et relancer, ou éditer et relancer) une commande précédente.

Essayer de lancer **echo Rebonjour les collègues** en retrouvant la commande initiale et en ne modifiant que le "Re"

History : renvoie la liste des commande que vous avez tapées c'est un historique numéroté de commandes.

history 10 : pour avoir seulement les **10** dernières...

Vous pouvez relancer une commande en tapant **!** Immédiatement suivi du numéro de commande de l'historique. Essayez avec une des commandes déjà entrée...

Quand on tape une commande qui nécessite d'indiquer un chemin un peu long il est possible de demander la **complétion** de nom en appuyant sur la touche [**Tab]ou deux fois sur la touche [echap]**

Nous allons voir (sans déplacements avec **cd**) les fichiers en-têtes standards du C à notre disposition dans le répertoire **/usr/include**

ls /usr/inc[Appuyer sur Tab] → on obtient la complétion **ls /usr/include/**

Donc on écrit le début d'un nom de répertoire (ou de fichier) et en appuyant sur tab on a la suite. Si il y a plusieurs noms correspondant **Tab** ne complète pas, **Tab** une 2ème fois pour avoir la liste...

ls /u[Tab] → ls /usr/ ajouter juste **s ls /usr/s[Tab][Tab]** → plusieurs réponses

compléter **ls /usr/sh[Tab]** → **ls /usr/share/** (liste de différents utilitaires du système)

L'efficacité du terminal/shell pour piloter un système tient à l'utilisation assidue de ces raccourcis.

Par exemple vous pouvez compléter les commandes : **his[Tab] → history**

5/ Le système de fichier (sous votre HOME) est votre terrain de jeu et le shell est votre bulldozer…

Il est temps de créer un espace de travail.

Vérifiez que vous êtes bien dans votre HOME avec **pwd**, refaire **cd ~** si nécessaire.

mkdir tp1

Make Directory : créer répertoire tp1 dans le répertoire courant (dans HOME donc)

Attention pour vos choix de noms de fichiers : utiliser des caractères spéciaux (autres que _ ou -) accents ou même seulement des espaces c'est chercher les ennuis. Il reste possible de "quoter" pour gérer des chemins avec espaces : cd **"Mon TP1 avec espaces"** par exemple mais ça complique.

cd tp1

mkdir test1

mkdir test2

cd test1

Vous pouvez vérifier qu'on a bien dans home un sous-répertoire tp1 qui lui même contient 2 sous-répertoires test1 et test2 en utilisant l'explorateur de fichiers (poste de travail).

6- Commandes de changement de nom et copie de fichiers.

Vérifier bien que le fichier fichier.txt existe dans le répertoire courant si non :

ls fichier.txt

Puis faire **: cp fichier.txt fichier_copie.txt**

Cette commande copie le contenu de dans le fichier fichier_copie.txt

Attention ! : Si le fichier fichier_copie.txt existe déjà il va s'écraser, si non il sera créer.

mv fichier.txt fichier_avec_autre_nom.txt

Le fichier **fichier.txt** change de nom et devient **fichier_avec_autre_nom.txt**

Attention ! : Si le fichier fichier_avec_autre_nom.txt existe déjà il va s'écraser