TP1 : Initiation À Linux

Système D'exploitation

Samir Aknine, Antoine Gréa

☘ Compétences À Acquérir

- ✔ Savoir utiliser la ligne de commande Linux
- ✔ Comprendre le fonctionnement des tubes et redirections

À la connexion dans un terminal, l'utilisateur est mis en relation avec un **interpréteur de commandes** (interface de dialogue homme/machine), appelé *shell* en Unix.
Le shell choisi dans ce cours est le `bash` : **b**ourne **a**gain **sh**ell, apparu avec GNU/Linux.
Le shell possède un double rôle :

C'est d'abord un interpréteur de commandes exécutant la boucle infinie suivante :

1. Affichage de l'invite de commande ou prompt (« `$` ») d'attente de lecture au clavier.
2. Lecture d'une commande (validée par **↵ Entrée**).
3. *Analyse syntaxique* (découpage en mots).
4. Interprétation des *caractères spéciaux*.
5. Exécution de la commande et retour au début.

Le shell est aussi un langage de programmation gérant des variables.

Vous retrouverez les notions dont vous aurez besoin dans l'abrégé de système d'exploitation. Pensez à le consulter en priorité !

Exercice 1 Commandes UNIX

Les commandes UNIX sont documentées dans le manuel en ligne de comandes (`man`). Le manuel s'utilise sous la forme `$ man commande`. Pour faire une rechercher de texte dans le manuel, il faut taper `/texte` puis `↵Entrée`. Chaque pression sur la touche `N` ira ensuite à la prochaine occurrence du texte. Pour quitter utilisez la touche `Q`.Vous pouvez utiliser ce manuel pour répondre aux questions suivantes.

Question 1 Dossiers Et Liens

`cd` permet de changer de répertoire courant
`pwd` permet d'afficher le répertoire courant.

Créez l'arborescence suivante dans le dossier `/tmp`. Créez le lien **physique** entre les deux **fichiers** nommés `oui` et le lien **symbolique** entre `oui` et *non*.

```
$ tree /tmp/exo1/
/tmp/exo1/
├── des
│   └── dossiers
│       └── partout
│           └── mais
│               └── vraiment
│                   └── quoi
│                       └── oui
└── je
    └── repète
        ├── non -> exo1/des/dossiers/partout/mais/vraiment/quoi/oui
        └── oui => exo1/des/dossiers/partout/mais/vraiment/quoi/oui
8 directories, 3 files
```

Supprimer le premier `oui` en utilisant `rm`. Quel est le résultat ? Peut-on utiliser les fichiers dans le répertoire `repète` ?

Question 2 Fichiers

Quelle est la différence entre `cat` et `less` ?
Quelle est la différence entre `head` et `tail` ?
Comment afficher les lignes d'un fichier en les numérotant ?

Question 3 Utilisateurs Et Droits D'accès

Pour chaque fichier et répertoire, UNIX définis des droits d'accès RWX (Read, Write, eXecute) pour 3 ensembles d'utilisateurs :

Ces informations peuvent être connues par la commande `ls -l`. On peut modifier les droits d'un fichier avec la commande `chmod`.

3.1 Propriété Privée, Défense D'entrer

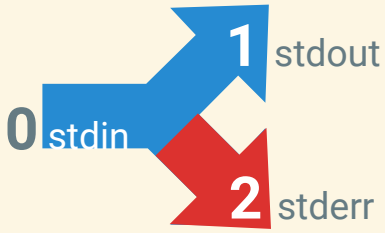
? Qui est le propriétaire du dossier `~`?
Quel en est le groupe propriétaire ?
Quels sont les droits du propriétaire, du groupe propriétaire, des autres utilisateurs ?

3.2 Tout Est Permis

? Modifier les droits pour que le fichier `/tmp/exo3` soit accessible en écriture pour les 3 types d'utilisateurs, et soit exécutable pour le propriétaire du fichier.

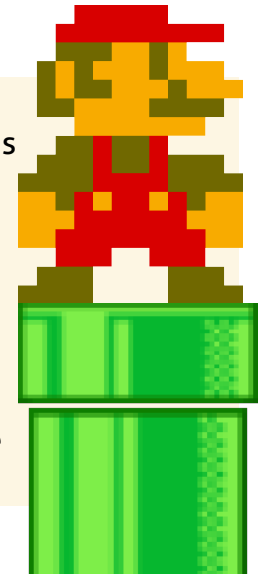
Question 4 Redirection Et Tube

i À la connexion, le shell dispose de trois flots de communication. L'association par défaut de ces flots est l'écran pour `stdout` et `stderr`, et le clavier pour `stdin`.



i Une redirection est une modification de l'une ou de l'autre de ces associations. Elle est valable uniquement le temps de la commande sur laquelle elle porte.
Ainsi, la redirection de la sortie standard avec `>` permet de récupérer le résultat dans un fichier : `commande > nom_fich`. La redirection de l'entrée standard se fait avec `<`.

4.1 Plombier Numérique



? Créer un fichier `mario.txt` contenant le texte `"itsa me !"`.
Copier ce fichier dans un autre fichier nommé `luigi.txt` sans utiliser la commande `cp` mais en utilisant un tube.

i Un tube (ou pipe `|`) passe le résultat d'une commande à une autre sans passer par une redirection. L'objectif d'utiliser un tube est de faire agir une commande sur le résultat d'une autre sans fichiers intermédiaires. Le symbole « `|` » placé entre deux commandes redirige la sortie standard de la première sur l'entrée standard de la seconde.

4.2 Commandes Mystérieuses

Que fait la commande `wc -l /etc/passwd` ?
Quelle différence avec `cat /etc/passwd | wc -l` ?
Comment afficher les 3 dernières lignes d'un fichier en les numérotant ?

Le séparateur « `;` » permet d'enchaîner des commandes sans relation entre elles.
`(...)` permet de considérer les commandes incluses comme une seule pour un tube ou une redirection. Placée entre « `$(`» et «`)` » ou entre anti-quotes « `` ` `` », la commande à exécuter est remplacée par son résultat.

4.3 Interprétation

Que font les commandes suivantes :

```
$ (date ; who) > /tmp/qui

$ echo Je suis sous pwd

$ echo Je suis sous $(pwd)

$ echo Je suis sous '$(pwd)'
```

Quelle est la différence entre `"texte"` et `'texte'` ?

Question 5 Traitements Des Fichiers Texte

La commande `grep` recherche un mot dans un fichier et affiche les lignes dans lesquelles ce mot a été trouvé. Elle peut être utilisée de manière très simple :
`grep texte monfichier`

5.1 La Fin Justifie Les Moyens

Quelle commande permet de connaître le nombre de lignes où le mot `return` apparait dans le fichier `/etc/bash.bashrc` ?

La commande `find` permet de réaliser différents types de recherche sur le fichier : recherche par nom de fichier avec l'option `-name`, par propriétaire (`-user`), par taille (`-size`)...

+ Bonus

Tentez de découvrir comment fonctionne la commande suivante :

```
$ echo "cat" | sudo tee -a /dev/`w | grep tty | cut -d ' ' -f 3`
```

Question 6 Gestion Des Processus

i La commande `ps` permet d'obtenir la liste statique des processus qui tournent au moment où vous lancez la commande. Les quatre colonnes obtenues indiquent le numéro d'identification du processus (`PID`), le nom de la console depuis laquelle le processus a été lancé (`TTY`), la durée d'exécution du processus (`TIME`), le programme qui a généré ce processus (`CMD`).
La commande `ps` sans options indique uniquement les processus lancés par le même utilisateur dans la même console. L'option `-e` permet de lister tous les processus lancés par tous les utilisateurs sur toutes les consoles. L'option `-f` permet d'afficher plus de détails sur les processus (notamment l'`UID` ou user identifier). La commande `pstree` affiche les processus en cours d'exécution sous forme d'un arbre.

6.1 Je Suis Ton Père

? Quelle commande permet de connaître le nombre de processus appartenant au super-utilisateur (`UID` = root) ?
Quel est le processus père de tous les autres ?
Quel est le numéro de ce processus ?

Pour obtenir une liste dynamique (mise à jour régulièrement) vous pouvez utiliser la commande `top`. En utilisant cette commande, répondez aux questions suivantes :

6.2 Pourquoi Ça Rame ?

? Combien de processus sont présents actuellement sur votre ordinateur ?
Combien sont prêts (état running) ?
Combien sont bloqués (état sleeping) ?
Combien de processus zombie y-a-t-il ?

6.3 Stop

Lancez la commande `$ sh -c "man ls"`. Utilisez **Ctrl** + **Z** pour stopper le processus.

? Utilisez la commande `pstree` pour trouver l'instance de `man` que vous avez stoppé. Que remarquez-vous ?

La commande `kill` permet d'envoyer des signaux au processus.

? Quelle est la commande pour envoyer un signal `SIGINT` à ce processus ?