Informatique tronc commun TP 1

28 août 2013

Note : après la séance, vous devez rédiger un compte-rendu de TP et l'envoyer au format électronique à votre enseignant.

Le seul format accepté pour l'envoi d'un texte de compte-rendu est le format PDF.

1 Démontage d'un PC

Cette partie est à faire en binôme, vous ne rendrez donc qu'un compte-rendu pour deux.

Avant tout, notez bien qu'on ne démonte pas les ordinateurs de la salle de TP mais ceux fournis par l'enseignant dans ce but !

Deuxième chose : on ne touche pas l'intérieur du PC, ou alors le moins possible et en ayant pris soin au préalable de se décharger de son électricité statique en touchant la carcasse du PC.

Chaque binôme dispose d'un PC. Le but du jeu est de l'ouvrir, d'identifier les différents composants (où est le disque dur ? la carte mère ? le processeur ? la mémoire vive ?), puis de le refermer, le tout sans casse.

N'oubliez pas de prendre des photos de l'intérieur pour votre compte-rendu.

Q1 Vous préciserez sur le compte-rendu les différents éléments que vous avez identifiés, de préférence à partir de photos.

2 Shell et terminal

Cette partie est à faire de façon individuelle.

Démarrez un PC avec la distribution Slitaz qui vous est fournie sur CD.

2.1 Fichiers

Sous Unix (dont la distribution GNU/Linux est un représentant) les fichiers sont organisés hiérarchiquement en une arborescence unique de répertoires. La racine de cette arborescence, c'est-à-dire le répertoire supérieur de la hiérarchie contenant tous les fichiers auxquels à accès le système, est noté `/`. Ses sous-répertoires directs (de l'ordre de la dizaine ou quelques dizaines de répertoires), comme `home`, `media`, ... sont notés `/home`, `/media`, ...

Note : il existe une unique arborescence même si physiquement, les fichiers sont stockés dans plusieurs emplacement distincts. Ici, les données de l'arborescence sont stockées en mémoire vive (donc non préservées à l'extinction de la machine) **sauf** les sous-répertoires de `/media` qui peuvent présenter les données du disque dur de la machine, celles du CD, ainsi que celle d'une éventuelle clé USB connectée à la machine. **On s'abstiendra donc dans un premier temps de tenter de modifier quoi que ce soit sous le répertoire `/media` sous peine de risquer de causer des dégâts.**

Lancez l'explorateur de fichiers (icône « Mes Documents »). Il est initialement dans un répertoire (familièrement appelé « Mes Documents ») et dont le nom pour le système (on appelle cela le *nom absolu* du répertoire) est en fait `/home/tux`, autrement dit, il s'agit du sous-répertoire `tux` de `/home`. Ce nom est affiché par l'explorateur de fichiers.

Vous pouvez vous déplacer dans les répertoires avec l'explorateur.

Q2 Quel sont les noms des sous-répertoires directs de `/home/tux` ? Quel sont leurs noms absolus ?

Se rendre dans le répertoire `/media`.

Q3 Quels sont les sous-répertoires contenus dans ce répertoire ?

Dans la colonne de gauche de l'explorateur de fichiers, vous pouvez voir notamment la liste des dispositifs de stockage de fichiers associés. Par un clic droit sur l'un d'eux, vous pouvez choisir ou bien de le « monter » dans l'arborescence, ou bien de le « démonter » ou de l'« éjecter ». **Q4 Quel effet cela a t-il sur le contenu visible dans les sous-répertoires de `/media` ?**

On peut obtenir des informations sur un répertoire ou un fichier : cliquer sur le fichier ou le répertoire en question avec le bouton droit, choisir l'article de menu « Propriétés », une boîte d'information contenant deux onglets s'ouvre alors.

Q5 Quelles sont les informations disponibles sur les répertoires ? Sur les fichiers ?

Vérifiez que vous pouvez créer un nouveau répertoire `TP1` dans `/home/tux` et un fichier `texte.txt` dans `TP1` (créer un fichier texte vide à partir de l'explorateur de fichier, puis double-cliquer dessus pour l'ouvrir avec l'*éditeur de texte* leafpad).

Q6 Essayez de copier votre fichier `/home/tux/TP1/texte.txt` dans le répertoire `/home/tux`, puis dans `/home`. Que se passe t-il ? Pourquoi ?

Q7 Essayez de copier le fichier `/bin/sh` puis `/etc/shadow` dans `/home/tux`. Cela fonctionne t-il ? Pourquoi ?

L'explorateur de fichiers permet de renommer un fichier existant. **Q8 Essayez de renommer `texte.txt` puis `/bin/sh`, puis `/home/tux/TP1/sh`. Que se passe t-il ? Pourquoi ?**

Q9 Essayez d'effacer `/bin/sh` puis `/home/tux/TP1`. Que se passe t-il ? Pourquoi ?

Note importante : certains explorateurs de fichiers mettent les fichiers effacés dans une poubelle ou on peut les récupérer en cas de fausse manœuvre. Ce n'est pas le cas de pcmanfm, donc soyez attentifs à ce que vous faites.

Q10 Regardez comment changer les permissions sur `/home/tux/TP1`. Quel incidence cela a-t il sur la possibilité de copier un fichier dans ce répertoire ? Vers ce répertoire ? Sur la possibilité d'effacer un fichier de ce répertoire ? De renommer un fichier ?

2.2 Terminal et shell

Unix a été inventé à un moment où l'utilisateur avait la possibilité d'interagir avec l'ordinateur via un *terminal*, c'est-à-dire la combinaison d'un clavier et d'un écran pouvant écrire (en général) 25 lignes de 80 caractères (en une seule couleur, généralement vert ou orange sur fond noir). Cette façon d'interagir avec la machine peut paraître archaïque de nos jours mais elle est pourtant d'une puissance diabolique.

Vous trouverez un émulateur de terminal dans le menu « Applications », sous-menu « Accessoires ». Le démarrer. Celui-ci démarre un programme, appelé *shell* ou *interprète de commandes*. Ce *shell* vous donne quelques informations, et affiche un symbole `$`, appelé invite (ou *prompt* en anglais) signe qu'il attend vos ordres.

Pour lui donner un ordre, il suffit de taper le nom de la commande désirée, puis de valider par la touche Entrée.

Q11 Que font les commandes `cal`, `date`, `exit`, `leafpad`, `midori`, `pcmanfm` ?

Une commande très pratique est `man` : elle permet d'obtenir le manuel de quasiment toutes les commandes. On l'utilise sous la forme `man` *page* où *page* est la page de manuel désirée.

Q12 Tapez `man cal`. Que se passe t-il ?

Vous pouvez faire défiler le texte ligne par ligne avec Entrée ou page par page avec la barre d'espace et quitter `man` avec la touche q.

Q13 Que fait la commande `ls` ?

La commande `cd` permet de changer de répertoire courant. `pwd` permet d'afficher le répertoire courant, en particulier, `cd` *d* où *d* est le nom absolu d'un répertoire change le répertoire du *shell* en *d*. Essayez avec `cd /usr/bin` par exemple.

Q14 Que fait `cd` sans argument ? (i.e. `cd` non suivi du nom d'un répertoire)

Q15 Que fait `cd ~` ? `cd ~/TP1` ? De quoi `~` est-il l'abréviation ?

Q16 Après vous être assurés que vous êtes dans le répertoire `/home/tux`, faire `cd TP1`. Qu'affiche alors `pwd` ?

`ls -a` affiche, comme `ls` le contenu d'un répertoire, mais cette commande affiche également les fichiers dont le nom commence par un point, cachés par défaut.

Q17 `ls -a` dans `/home/tux/TP1` doit montrer deux répertoires cachés. Quels sont leurs noms ?

En fait, dans chaque répertoire du système, il existe deux répertoires cachés avec ces deux mêmes noms.

Q18 Que donne un `cd` sur chacun de ces répertoires ?

Q19 Placez-vous dans le répertoire `/usr/bin`. Que fait alors `cd ../sbin/../../etc` ?

On dit que `../sbin/../../etc` est un nom de répertoire *relatif* au répertoire courant. La plupart des commandes qui acceptent des noms de fichiers ou de répertoires acceptent aussi bien les noms relatifs que les noms absolus.

2.3 Processus

Unix est un système multitâche, c'est-à-dire qu'il peut faire tourner en parallèle plusieurs programmes. Chacun de ces programmes en cours d'exécution est appelé un *pro-*

cessus.

Pour visualiser les processus du système : `ps aux`.

Comme vous pouvez le constater, le résultat est difficile à lire. Les shells sous Unix possèdent un mécanisme très puissant, appelé redirection, qui permet de rediriger le résultat d'une commande vers un fichier. Essayez `ps aux > resultat.txt` et ouvrez le fichier produit avec leafpad pour regarder la liste des processus. La première ligne explique la signification des différentes colonnes (PID signifie : *process identifier*).

On peut aussi rediriger la sortie d'une commande vers l'entrée d'une autre. Par exemple la commande `more` affiche son entrée page par page.

Q20 Que fait `ps aux | more` ?

Q21 Que fait `ps aux | grep pcmanfm` ?

La commande `kill` permet de tuer un processus à partir de son numéro, pour autant qu'on en ait la permission. Ce peut être utile dans le cas d'un processus qui continue à tourner (donc à consommer des ressources du système) alors qu'il aurait dû s'arrêter.

Lancez l'explorateur de fichiers (pcmanfm), trouvez son pid puis tuez ce processus par la ligne de commande.

Pour les processus ayant une interface graphique et qui seraient bloqués, la commande `xkill` est utile.

Q22 Que fait xkill ?

2.4 Ajouter ses commandes au système

On peut ajouter ses propres commandes au système pour l'enrichir. Par exemple, imaginons que nous voulions une commande affichant le calendrier du mois puis la date et l'heure. Créez, à l'aide de leafpad, un fichier `/home/tux/TP1/caldate` dans lequel vous mettrez les deux lignes :

```
cal
date
```

Vous pouvez exécuter les commandes de ce fichier en tapant `/bin/sh /home/tux/TP1/caldate` ou plus simplement `sh /home/tux/TP1/caldate`. La commande `sh` est en effet un *shell* interprétant chaque ligne du fichier qu'on lui donne en argument comme une commande. On appelle *scripts* les commandes créées avec une suite de commandes *shell*.

On peut encore raccourcir cette commande, en ajoutant à la première ligne du fichier la ligne suivante :

```
#! /bin/sh
```

Cette ligne signifie que le fichier doit être exécuté par `/bin/sh`.

Q23 Essayez d'exécuter alors `/home/tux/TP1/caldate`. Que se passe t-il ? Comment corriger le problème ?

De même que le *shell* comprend que la commande `sh` fait référence au programme `/bin/sh`, on peut faire en sorte d'abréger `/home/tux/TP1/caldate` en `caldate`. Sans entrer dans les détails, disons que la façon dont Slitaz est configurée fait qu'il suffit de créer un répertoire `/home/tux/.local/bin` et d'y placer le script.

2.5 Un premier script python

Dans un terminal lancer `python`. On peut demander à python d'afficher bonjour en tapant `print('bonjour')`.

De même qu'on peut écrire un script *shell*, on peut écrire un script python : avec un éditeur de texte, créer un fichier `/home/tux/TP1/salut` contenant

```
#! /usr/bin/python
print('bonjour')
```

et faire en sorte que la commande `salut` fonctionne.

2.6 Un environnement de développement python : IDLE

S'il est possible d'écrire des programmes python à partir de n'importe quel éditeur de texte, il existe cependant des environnements de développement facilitant le travail, en mettant par exemple en évidence les problèmes de syntaxe et en facilitant le test des programmes. IDLE est l'environnement de développement livré par défaut avec python.

Lancer IDLE. À première vue, IDLE ressemble à `python` lancé depuis un terminal avec lequel on peut interagir. Cependant, IDLE permet aussi de créer des scripts : choisir « New Window » dans le menu « File ». Dans la fenêtre qui s'ouvre, taper le code du script à écrire, par exemple

```
#! /usr/bin/python
print('Guten Tag')
```

Vous pouvez constater que python met en couleur les différents éléments du programme.

Sauvez le fichier (par exemple dans `/home/tux/TP1/Hallo`). Vous pouvez alors l'exécuter directement depuis IDLE : menu Run/Run Module.

Notez enfin qu'à partir du moment où un fichier commence par la ligne `#! /usr/bin/python`, pcmanfm le reconnaît comme programme python et il est possible de lui demander de l'ouvrir avec IDLE par défaut (clic-droit, « Ouvrir avec », « Ouvrir avec un autre programme », choisir idle et cocher « Définir l'application par défaut »)