51IF1IS1 - Introduction aux systèmes d'exploitation

Contrôle continu 1

vendredi 14 octobre 2011 - durée: 50 minutes

Aucun document n'est autorisé. Les documents électroniques sont interdits, en particulier les téléphones, ordinateurs, PDA, etc.

Le barème est donné à titre indicatif.

La situation On considère la suite de commandes suivante (le prompt est représenté par une paire de crochets, les lignes sont numérotées par commodité) :

```
1  [] ls -al /home
2  total 24
3  drwxr-xr-x 6 root  root     4096 2009-11-08 12:42 .
4  drwxr-xr-x 9 root  root     4096 2009-11-08 12:42 ..
5  drwxr-xr-x 4 Brice sciences 4096 2010-10-08 12:01 Brice
6  drwxr--r-- 2 Fabrice sciences 4096 2010-10-08 12:02 Fabrice
7  drwxr-xr-x 2 Haymadou sciences 4096 2010-10-08 12:02 Haymadou
8  drwxrwxrwx 2 Matthieu sciences 4096 2010-10-08 12:02 Matthieu
9  [] ls -al /telechargements
10 total 8328
11 drwxr-xr-x 2 root  root       4096 2009-11-08 12:42 .
12 drwxr-xr-x 9 root  root       4096 2009-11-08 12:42 ..
13 -rw-r--r-- 1 Haymadou sciences 106817 2010-10-08 12:02 plante.pdf
14 [] cd /home
15 [] ls -lR Haymadou Matthieu Brice
16 Haymadou:
17 total 24
18 -rw-r--r-- 1 Haymadou sciences 2717 2010-10-08 12:02 projet.txt
19
20 Matthieu:
21 total 508
22 -rw-r----- 1 Matthieu sciences 512766 2010-10-08 12:02 chat.jpg
23
24 Brice:
25 total 8
26 drwxrwxr-x 2 Brice sciences 4096 2010-10-08 12:01 Articles
27 drwxr-xr-x 2 Brice sciences 4096 2010-10-08 12:01 Documents
28
29 Brice/Articles:
30 total 0
31
32 Brice/Documents:
33 total 0
34 [] id
```

```
35 uid=1012(Fabrice) gid=102(sciences)
36 [] cd
37 [] pwd
38 /home/Fabrice
39 [] ls -l
40 total 125
41 -rw-r--r-- 1 Fabrice sciences 133125 2010-10-08 12:13 Hypnotize.mp3
42 -rw-r--r-- 1 Fabrice sciences      6 2010-10-08 12:13 impots.txt
```

Pour répondre aux questions, vous pourrez vous aider des extraits de quelques pages de manuel se trouvant en annexes.

Exercice 1 – *Les bases*

Dans cet exercice, on suppose qu'il n'y a pas de problème de droit et qu'aucune commande n'a été effectuée depuis les manipulations précédentes.

1. Dessiner l'arborescence de la partie du système de fichier apparaissant dans les lignes précédentes.
2. L'utilisateur `Haymadou` se connecte au système dont vous venez de décrire l'arborescence. Supposons qu'il soit dans son répertoire personnel. Il doit créer un dossier `Candidature` à l'intérieur d'un répertoire `Documents`, qui n'existe pas encore, dans son répertoire de login. Expliquer comment il peut procéder pour le faire, en minimisant le nombre de commandes.
3. Expliquer comment il peut copier le contenu du fichier `projet.txt` dans un fichier `vaisseau.txt` en utilisant le minimum de commande(s).
4. Donner une commande qui lui permettra de copier dans le répertoire `Candidature` le fichier `plante.pdf` (en utilisant le même nom). Donner une commande qui lui permettra de déplacer le fichier `vaisseau.txt` dans le répertoire `Candidature`.
5. Il doit créer une archive pour envoyer le répertoire `Candidature` à un ami. Donner la commande qu'il doit effectuer en justifiant votre choix.
6. Enfin pour nettoyer son répertoire personnel, `Haymadou` veut supprimer le répertoire `Candidature`. Comment peut-il le faire ?

Correction.

1.
2. Comme le dossier `Documents` n'existe pas, on utilise l'option `-p` de la commande `mkdir` qui permet de créer un répertoire ainsi que tous les répertoires parents qui n'existent pas encore :

   ```
   mkdir -p /home/Haymadou/Documents/Candidature
   ```

3. D'après l'énoncé, le répertoire courant est `/home/Haymadou`, on peut donc utiliser les chemins relatifs à ce répertoire. La commande `cp` permet de copier et de renommer un fichier :

   ```
   cp projet.txt vaisseau.txt
   ```

4. La commande `cp` permet de copier un fichier :

   ```
   cp /telechargements/plante.pdf /home/Haymadou/Documents/Candidature
   ```

   La commande `mv` permet de déplacer un fichier :

   ```
   mv /home/Haymadou/vaisseau.txt /home/Haymadou/Documents/Candidature
   ```

5. On utilise la commande `tar` avec les options `c` pour créer l'archive `v` pour avoir des informations sur ce que va faire la commande et `f` pour manipuler l'archive dont le nom est précisé en argument :

   ```
   tar -cf /home/Haymadou/Documents/Candidature.tar /home/Haymadou/Documents/Candidature
   ```

6. Le répertoire `Candidature` n'étant pas vide, il faut utiliser l'option `-r` pour récursif de la commande `rm`

```
rm -r /home/Haymadou/Documents/Candidature
```

Si on ne se rappelle plus de cette option, on peut effectuer les commandes :

```
rm /home/Haymadou/Documents/Candidature/vaisseau.txt
rm /home/Haymadou/Documents/Candidature/plante.pdf
rmdir /home/Haymadou/Documents/Candidature
```

Exercice 2 – *Les droits*

Justifiez précisément les réponses aux questions qui suivent.

1. Est-ce que l'utilisateur `Brice` peut écouter le fichier `Hypnotize.mp3` de Fabrice ? Quels sont les utilisateurs qui peuvent supprimer le répertoire `Articles` de l'utilisateur `Brice` ?
2. Réécrire la commande suivante en utilisant la notation octale des droits :

```
chmod g=rw,u+x,o-w Hypnotize.mp3
```

3. Réécrire la commande suivante en utilisant la notation symbolique des droits :

```
chmod 553 Hypnotize.mp3
```

4. Qui est l'utilisateur qui effectue la suite de commandes présentée dans le texte de l'exercice 1 ? À quel groupe appartient-il ? Cet utilisateur peut-il afficher sur son écran l'image `chat.jpg` présente dans le répertoire maison de l'utilisateur `Matthieu` ? Peut-il la supprimer ?
5. Qui a le droit d'exécuter la commande `chmod og-r /home/Matthieu` ? Quel serait précisément son effet ?
6. Quelles commandes peut utiliser `Haymadou` pour interdire à tous les autres utilisateurs de lire le contenu de son fichier `projet.txt` ?
7. Enfin `Haymadou` voudrait créer dans son répertoire personnel un répertoire ayant les propriétés suivantes :
   – Tout le monde peut lister le contenu du répertoire.
   – Seuls l'utilisateur `Haymadou` et les membres du groupe `sciences` peuvent créer des fichiers dans le répertoire.

   Quels doivent être le propriétaire, le groupe propriétaire et les droits pour ce nouveau répertoire ?

Correction.

1. `chmod 764 Hypnotize.mp3`

2. `chmod ug=rx,o=wx Hypnotize.mp3`

3. L'utilisateur `Brice` n'a pas accès au fichier `Hypnotize.mp3` de l'utilisateur `Fabrice` car celle-ci n'a pas fourni le droit `+x` aux autres utilisateurs sur son répertoire maison (et passer par ce répertoire est le seul moyen d'accéder au fichier). Le droit en lecture `+r` sur le fichier n'est pas suffisant, il faut pouvoir accéder au fichier. Le répertoire `Articles` de l'utilisateur `Brice` se trouve dans son répertoire personnel. Son répertoire personnel n'est modifiable (`+w`) que par lui-même. L'utilisateur `Brice` est donc le seul à pouvoir supprimer son répertoire `Articles`.
4. La commande `pwd` nous informe que l'utilisateur qui l'a exécutée est l'utilisateur `Fabrice`, car elle est éxécutée après la commande **cd** sans argument, qui place l'utilisateur dans son répertoire personnel. Cette même commande nous informe également qu'il fait partie du groupe `sciences`. L'utilisateur `Fabrice` peut lire l'image `chat.jpg` de l'utilisateur `Matthieu` car :

– celle-ci lui est accessible (droit `+x` sur le répertoire maison de l'utilisateur `Matthieu` pour tous les utilisateurs).
– il possède le droit en lecture (droit `+r`) puisqu'il fait partie du groupe sciences.
Fabrice peut également supprimer cette image car l'utilisateur `Matthieu` a autorisé tous les utilisateurs à modifier son répertoire maison (droit `+w`).

5. L'utilisateur `Matthieu` est propriétaire du répertoire `/home/Matthieu`, il est le seul a pouvoir changer ses permissions (Note : le super-utilisateur peut aussi le faire, mais cette précision n'est pas demandée). Enlever le droit en lecture (`+r`) sur un répertoire empêche les utilisateurs concernés de lister le contenu du répertoire (avec `ls` par exemple). Cela ne les empêche pas d'accéder à son contenu dans le cas où il connaissent les noms des fichiers/répertoires qu'il contient.

6. Au choix, l'utilisateur `Haymadou` peut empêcher directement la lecture du fichier, ou indirectement il peut interdire l'accès au répertoire :
```
–chmod og-r ~/projet.txt
–chmod 600 ~/projet.txt
–chmod og-x ~
–chmod 744 ~
```

7. Le propriétaire est `Haymadou`, le groupe propriétaire est `sciences` et les droits correspondent à rwxrwxr-x.

Exercice 3 (34%) – *Les liens*

Dans la suite de l'exercice, supposez que vous êtes l'utilisateur `Haymadou` et que les commandes suivantes viennent d'être effectuées.

```
1 [] cd ~
2 [] echo "Le ciel est bleu" > projet.txt
```

1. Donnez les commandes qui permettent de :
– Créer, dans votre répertoire maison, un sous-répertoire nommé `liens` et un lien symbolique nommé `lsliens` du répertoire `liens`.
– Créer, dans le répertoire `liens`, un lien symbolique nommé `lsprojet` et un lien physique nommé `lpprojet` du fichier `projet.txt`.

2. Sur votre copie, en face de chaque numéro de ligne vide, indiquer ce que le shell afficherait sur l'écran en réponse aux commandes suivantes. Justifier vos réponses.

```
1  [] cd
2  [] echo "La terre est ronde" > liens/lpprojet
3  [] cat lsliens/lpprojet
4
5  [] cat projet.txt
6
7  [] cat liens/lsprojet
8
9  [] echo "Les voitures sont dangereuses" > liens/lsprojet
10 [] cat lsliens/lsprojet
11
12 [] cat liens/lpprojet
13
14 [] cat projet.txt
15
16 [] ls -l liens
17 -rwxrwxr--   2 Haymadou  sciences  42 Oct 14 23:47 projet
18 -rwxrwxr--   2 Haymadou  sciences  42 Oct 14 23:47 lpprojet
19 lrwxrwxrwx   1 Haymadou  sciences   9 Oct 14 23:59 lsprojet -> ../projet.txt
20 [] chmod g-rx liens/lpprojet
```

```
21 [] ls -l liens
22
23 [] ls -i lsprojet
24 1577715 lsprojet
25 [] ls -i lpprojet
26 1577700 lpprojet
27 [] ls -i projet
28
29 [] exit
```

3. Donnez les commandes qui permettent de créer, dans le répertoire `Matthieu`, un lien symbolique nommé `lschat` et un lien physique nommé `lpchat` du fichier `chat.jpg`. Sur votre copie indiquer en face du numéro de la ligne vide ce que le shell afficherait sur l'écran en réponse aux commandes suivantes. Justifier votre réponse.

```
1  [] cd ~Matthieu
2  [] ls -il
3  total 680
4  4571714 -rw-r--r--  2 Matthieu  sciences  169522 Aug 18 14:39 chat.jpg
5  4571714 -rw-r--r--  2 Matthieu  sciences  169522 Aug 18 14:39 lpchat
6  4571717 lrwxrwxrwx  1 Matthieu  sciences       8 Oct 14 21:46 lschat -> chat.jp
7  [] rm lpchat
8  [] ls -il
9
10 [] exit
```

Correction.

1. Voici les commandes.

```
1 [] mkdir liens
2 [] ln -s liens lsliens
3 [] cd liens
4 [] ln ../projet.txt lpprojet
5 [] ln -s ../projet.txt lsprojet
```

2. Voici les réponses.

`l 4: La terre est ronde`

Comme `lsprojet` est un lien symbolique vers `../projet.txt`, la commande `cat` affiche le contenu du fichier à cette adresse.

`l 6: La terre est ronde`

Comme `lsliens` est un lien symbolique vers le répertoire `liens`, la commande `cat` affiche le contenu du fichier `lsprojet` contenu dans le répertoire `liens`.

`l 8: La terre est ronde`

En effet, `projet.txt` et `lpprojet` sont deux noms pour le même fichier.

```
l11: Les voitures sont dangereuses
l13: Les voitures sont dangereuses
l15: Les voitures sont dangereuses
```

En effet, la commande `echo` va écrire dans le fichier contenu dans l'adresse désignée dans le lien `lsprojet`, c'est-à-dire dans le fichier `projet.txt`.

```
l22:
 total 24
 -rwx--xr-- 2 Haymadou sciences 42 Oct 14 23:47 projet
 -rwx--xr-- 2 Haymadou sciences 42 Oct 14 23:47 lpprojet
 lrwxrwxrwx 1 Haymadou sciences 9 Oct 14 23:59 lsprojet -> ../projet.txt
```

En effet, comme `projet` et `lpprojet` sont deux liens symboliques vers le même inoeud, lorsque l'on change les droits de `lpprojet`, on change les caractéristiques de l'inoeud et donc les droits de `projet`.

```
l28:
 1577700 projet
```

3. `lpchat` et `chat.jpg` sont deux liens physique (deux noms différents) vers le même fichier de numéro d'inoeud 45717714. En effaçant le fichier `lpchat`, on supprime un lien physique vers cet inœud qui tombe à 1 :

```
1  [] cd ~Matthieu
2  [] ls -il
3  total 680
4  4571714 -rw-r--r--  2 Matthieu  sciences  169522 Aug 18 14:39 chat.jpg
5  4571714 -rw-r--r--  2 Matthieu  sciences  169522 Aug 18 14:39 lpchat
6  4571717 lrwxrwxrwx  1 Matthieu  sciences       8 Oct 14 21:46 lschat -> chat.jpg
7  [] rm lpchat
8  [] ls -il
9  total 680
10 4571714 -rw-r--r--   1 Matthieu  sciences  169522 Aug 18 14:39 chat.jpg
11 4571717 lrwxrwxrwx   1 Matthieu  sciences       8 Oct 14 21:46 lschat -> chat.jpg
12 [] exit
```

Annexes : extraits des pages du manuel

```
CP(1)                          User Commands                          CP(1)

NAME
       cp - copy files and directories

SYNOPSIS
       cp [OPTION]... [-T] SOURCE DEST
       cp [OPTION]... SOURCE... DIRECTORY
       cp [OPTION]... -t DIRECTORY SOURCE...

DESCRIPTION
       Copy SOURCE to DEST, or multiple SOURCE(s) to DIRECTORY.

       Mandatory arguments to long options are mandatory for short options
       too.

       -a, --archive
              same as -dR --preserve=all

       --backup[=CONTROL]
              make a backup of each existing destination file

       -b     like --backup but does not aprojetept an argument

       --copy-contents
              copy contents of special files when recursive

       -d     same as --no-dereference --preserve=links

       -f, --force
              if an existing destination file cannot be opened, remove it and
              try again (redundant if the -n option is used)

       -i, --interactive
              prompt before overwrite (overrides a previous -n option)

       -H     follow command-line symbolic links in SOURCE

       -l, --link
              link files instead of copying

       -L, --dereference
              always follow symbolic links in SOURCE

       -R, -r, --recursive
              copy directories recursively

       -s, --symbolic-link
              make symbolic links instead of copying

       -v, --verbose
              explain what is being done

       --help display this help and exit
```

```
       --version
              output version information and exit

AUTHOR
       Written by Torbjorn Granlund, David MacKenzie, and Jim Meyering.

REPORTING BUGS
       Report cp bugs to bug-coreutils@gnu.org
       GNU coreutils home page: <http://www.gnu.org/software/coreutils/>
       General help using GNU software: <http://www.gnu.org/gethelp/>
       Report cp translation bugs to <http://translationproject.org/team/>

COPYRIGHT
       Copyright (C) 2010 Free Software Foundation, Inc. License GPLv3+: GNU
       GPL version 3 or later <http://gnu.org/licenses/gpl.html>.
       This is free software: you are free to change and redistribute it.
       There is NO WARRANTY, to the extent permitted by law.

GNU coreutils 8.5             April 2010                          CP(1)
```

```
MKDIR(1)                      User Commands                     MKDIR(1)

NAME
       mkdir - make directories

SYNOPSIS
       mkdir [OPTION]... DIRECTORY...

DESCRIPTION
       Create the DIRECTORY(ies), if they do not already exist.

       Mandatory arguments to long options are mandatory for short options
       too.

       -m, --mode=MODE
              set file mode (as in chmod), not a=rwx - umask

       -p, --parents
              no error if existing, make parent directories as needed

       -v, --verbose
              print a message for each created directory

       -Z, --context=CTX
              set the SELinux security context of each created directory to
              CTX

       --help display this help and exit

       --version
              output version information and exit

AUTHOR
       Written by David MacKenzie.

REPORTING BUGS
```

```
      Report mkdir bugs to bug-coreutils@gnu.org
      GNU coreutils home page: <http://www.gnu.org/software/coreutils/>
      General help using GNU software: <http://www.gnu.org/gethelp/>
      Report mkdir translation bugs to <http://translationproject.org/team/>

COPYRIGHT
      Copyright (C) 2010 Free Software Foundation, Inc. License GPLv3+: GNU
      GPL version 3 or later <http://gnu.org/licenses/gpl.html>.
      This is free software: you are free to change and redistribute it.
      There is NO WARRANTY, to the extent permitted by law.

GNU coreutils 8.5             April 2010                      MKDIR(1)
```

```
TAR(1)                 BSD General Commands Manual              TAR(1)

NAME
     tar - The GNU version of the tar archiving utility

SYNOPSIS
     tar [-] A --catenate --concatenate | c --create | d --diff --compare |
        --delete | r --append | t --list | --test-label | u --update | x
        --extract --get [options] [pathname ...]

DESCRIPTION
     Tar stores and extracts files from a tape or disk archive.

     The first argument to tar should be a function; either one of the letters
     Acdrtux, or one of the long function names. A function letter need not
     be prefixed with ‘‘-’’, and may be combined with other single-letter
     options. A long function name must be prefixed with --. Some options
     take a parameter; with the single-letter form these must be given as sep-
     arate arguments. With the long form, they may be given by appending
     =value to the option.

FUNCTION LETTERS
     Main operation mode:

     -A, --catenate, --concatenate
          append tar files to an archive

     -c, --create
          create a new archive

     -d, --diff, --compare
          find differences between archive and file system

     --delete
          delete from the archive (not on mag tapes!)

     -r, --append
          append files to the end of an archive

     -t, --list
          list the contents of an archive

     -u, --update
          only append files newer than copy in archive
```

```
    -x, --extract, --get
         extract files from an archive

OTHER OPTIONS

    -f, --file ARCHIVE
         use archive file or device ARCHIVE

    -v, --verbose
         verbosely list files processed

    -z, --gzip, --gunzip --ungzip
         filter the archive through gzip

    -Z, --compress, --uncompress
         filter the archive through compress

ENVIRONMENT
    The behavior of tar is controlled by the following environment variables,
    among others:

    SIMPLE_BACKUP_SUFFIX
           Backup prefix to use when extracting, if --suffix is not speci-
           fied. The backup suffix defaults to '~' if neither is specified.

    TAR_OPTIONS
           Options to prepend to those specified on the command line, sepa-
           rated by whitespace. Embedded backslashes may be used to escape
           whitespace or backslashes within an option.

    TAPE   Device or file to use for the archive if --file is not specified.
           If this environment variable is unset, use stdin or stdout
           instead.

EXAMPLES
    Create archive.tar from files foo and bar.
         tar -cf archive.tar foo bar
    List all files in archive.tar verbosely.
         tar -tvf archive.tar
    Extract all files from archive.tar.
         tar -xf archive.tar

SEE ALSO
    tar(5), symlink(7), rmt(8)

HISTORY
    The tar command appeared in Version 7 AT&T UNIX.

                              Sep 22, 2010
```