TP UNIX 1

NICOLAS Jerome

RAVAUX Alexandre

Mise en place de l'environnement

--

1) Faisable uniquement sur l'interface graphique de Linux

2) mkdir >>> Creation d'un repertoire

l'option -p permet de creer le(s) sous-repertoire(s)

cd >>> deplacement vers un repertoire donné grace au path indiqué

Quelques conseils

--

2) man [commande] pour avoir des details sur la commande

3) On utilise ici une redirection.

cmd>fichier : Si le fichier n’existe pas, il est cree par le shell

et s’il existe deja le shell detruit son contenu pour le remplacer par la sortie de la commande .

La commande echo permet d'afficher un message, en l'occurence ici "Bonjour a tous".

On ajoute la sortie (le résultat si on veut) de la commande echo dans un fichier ici appelé "fichier".

Le fichier en question n'existe pas donc linux cree le fichier avant d'y mettre le "Bonjour a tous"

Si j'avais écrit echo "Bonjour a tous" > toto, il y aurait un fichier intitulé "toto" dans mon répertoire.

touch permet de creer un fichier.

cmd>>fichier : Si le fichier n’existe pas, il est crée par le shell et s’il existe déjà la sortie de la commande est ajoutée à la fin du fichier.

Donc ici il ajoute a la fin "Nous sommes Vendredi"

4)

I - Identification de l'environnement

--

Informations concernant la session et la machine locales :

a) On utilise la commande pwd pour afficher le répertoire courant.

b) On utilise la commande id pour connaître le nom d'utilisateur et le groupe d'utilisateurs.

c) On utilise la commande hostname pour afficher le nom de la machine. On obtient: tuxws01. Il est différent de l'étiquetage. En effet, le nom étiqueté correspond au nom de la machine physique, alors que le nom affiché correspond au nom de la machine auquel on est connecté (le serveur).

d) Sur la machine, le système Unix d'installé est Linux (commande uname -o), version SMP Debian 3.2.60-1+deb7u3 (commande uname -v).

e. On n'a pas de différences.

II - Navigation et recherches dans l'arborescence

--

Navigation :

a) cd (ou cd ~) : revenir dans son répértoire personnel

cd . pas de déplacement (se déplace dans le fichier où on se trouve en réalité)

cd .. revenir en arrière

cd - répertoire on où était précedemment

b) cd

c) cd -

d) . : répertoire actuel

.. : répertoire précedent

e) le chemin absolu est la succession de répertoires à parcourir depuis la racine pour accéder au fichier spécifié ; par exemple : /users/20140113/Unix

le chemin relatif est la successionde répertoires à parcourir depuis le répertoire courant; par exemple : ../../Unix

f) 1 chemin absolu. le chemin relatif dépend du répertoire courant donc il y a une "infinité" de chemins (tous les chemins mènent à Rome)

g) Chacun a ses avantages comme ses inconvenients

Fichiers cachés :

g) Un fichier caché est un fichier non visible par défaut pour l'utilisateur s'il affiche le contenu du fichier. Le nom d'un fichier caché commence par un point.

L'utilité d'un fichier caché ? Dissimuler des éléments. Il sert la plupart du temps à des fichiers de configuration (exemple : .bashrc, .bash_history), des fichiers "cache" des logiciels (exemple : .viminfo) ou encore de simples fichiers temporaires. Ces fichiers sont cachés pour alléger l'affichage des dossiers.

h) Tout fichier commençant par un point est non visible. Pour l'afficher, il faut utiliser la commande ls -a.

Sélection par le nom :

i)

* = remplace n'importe quelle suite de zéro à n caractères sauf pour "." (le point) au début du nom de fichier.

? = remplace n'importe quel caractère.

j) ls * Cette commande listera tous les noms des fichiers du répertoire courant, à l'exception des fichiers cachés

k)

$ ls a*

afic afic2 afic1 afic3

$ ls a???

afic

$ ls b??*

bfic bfic2 bfic1 bfic3

$ ls *[12]

afic1 afic2 bfic1 bfic2 cfic1 cfic2 dfic1 dfic2 fichier2

$ ls [ac]?*[12]

afic1 afic2 cfic1 cfic2

$ ls *[!3]

afic afic1 afic2 bfic bfic1 bfic2 bfic2 cfic cfic1 cfic2 dfic dfic1 dfic2

l) ls .???

m) ls .*[12]

n) ls *[3] (??) => Tester

Recherche par criteres :

o)

```
$^ls -l /var
lrwxrwxrwx  1 root root     9 sept.  2 14:36 lock -> /run/lock
lrwxrwxrwx  1 root root     4 sept.  2 14:36 run -> /run
```

p)

```
$ ls -l
$ rm [fichier]
```

q)

```
$ ls d*
$ ls -d d*
```

III - Manipulation des fichiers et répertoires

--

Généralités :

a) Il préserve le fichier d'origine ?

b) Supprime tout de façon récursive. Il faut éviter car vous pourriez, si vous obtenez les droits du super utilisateur, supprimer les fichiers vitaux.

Exercice :

```
mkdir -p ~/home/{lea,ralf,tom}/tmp
#
mkdir ~/home/{lea,ralf,tom}/tmp/app
touch ~/home/{lea,ralf,tom}/tmp/app/{a.c,b.c,c.c}
#
mkdir -p ~/home/tom/tmp/poub{1,2}
touch ~/home/tom/tmp/poub1/z.class ~/home/tom/tmp/y.class
```

OU :

```
mkdir -p $HOME/home/{lea,ralf,tom}/tmp
#
mkdir $HOME/home/{lea,ralf,tom}/tmp/app
touch $HOME/home/{lea,ralf,tom}/tmp/app/{a.c,b.c,c.c}
#
mkdir -p $HOME/home/tom/tmp/poub{1,2}
touch $HOME/home/tom/tmp/poub1/z.class $HOME/home/tom/tmp/y.class
```

e) cp -rf home home2

f) rm -rf home

Divers :

rm -rf

IV - Affichage et filtrage du contneu des fichiers

a) cat affche tout, more n'affiche qu'un partie pour qu'on puisse lire au fur et à mesure

b) head les dix premieres lignes tail les dix dernieres lignes

c)