Utilisation du shell Unix
11 - Scripts

Préparation

Ayez soin de noter vos `noms, prénoms et groupe` en commentaire au début de chaque script que vous créez.

1 Archivage de vos TP sur une clé USB

Lorsque vous insérez une clé USB dans une machine Linux, le contenu de la clé est automatiquement « monté » dans le répertoire `/media/`*`votreLogin`* avec comme nom le *label* de la clé ou à défaut son identifiant (*UUID*).

Faites un essai si vous avez une clé USB sous la main (celle de la SAÉ 1.3 ?).

On souhaite simplifier l'archivage d'un TP sur une clé USB en créant un script `sauveUSB` qui reçoit en paramètre le nom d'un répertoire et qui archive le répertoire avant de copier l'archive sur la clé USB, par exemple :

```
sauveUSB TPBateaux
```

va effectuer les opérations :

```
tar zcf TPBateaux.tgz TPBateaux
mv TPBateaux.tgz /media/votreLogin/nomDeLaCle
```

Pour rendre le script un peu plus robuste :

- on affichera un petit mode d'emploi si aucun nom de répertoire n'est donné en paramètre, par exemple :
  ```
  Usage : sauveUSB répertoire
  ```
- on vérifiera que le paramètre donné est bien un répertoire (on supposera qu'il est dans le répertoire courant) ;
- on s'assurera qu'une clé est bien insérée en récupérant le nom du **premier répertoire** situé dans le répertoire `/media/`*`votreLogin`*. On aura besoin du nom du répertoire pour déplacer l'archive vers la clé.

Il est recommandé de procéder par étapes :

1. Vérifier qu'il existe bien un paramètre ;
2. Vérifier que le paramètre donné est bien un répertoire ;
3. Trouver le nom du répertoire de la clé (s'il existe) et l'afficher pour tester ;
4. Finaliser le script en créant l'archive et en la déplaçant vers la clé.

2 Reconstruction d'un fichier `/etc/passwd`

Le fichier `/etc/passwd` d'un système Unix contient la base de données des utilisateurs. Il s'agit d'un fichier texte contenant en fait une table où les colonnes sont délimitées par `':'`. Voici un extrait [1] :

```
root:x:0:0:root:/root:/bin/bash
ritchied:x:1970:100:Dennis Ritchie:/home/ritchied:/bin/bash
```

1. Dennis Ritchie est l'inventeur du langage C et un des co-inventeurs d'Unix

Les machines Linux de l'IUT ne contiennent pas ces informations car elles sont regroupées dans un annuaire centralisé LDAP (*Lightweight Directory Access Protocol*).

On souhaite reconstruire les informations d'un fichier `/etc/passwd` pour tous les étudiants du BUT Informatique. On va procéder en deux étapes :

1. Création d'une ligne du fichier pour un utilisateur donné par son *login* ;
2. Parcours de tous les utilisateurs du répertoire `/users/but/info` pour reconstruire le fichier complet.

Les informations nécessaires sont : le login, le numéro de l'utilisateur, le numéro de son groupe, son nom complet, son répertoire de travail par défaut (*HOME directory*) et son shell.
Par exemple pour l'utilisateur `ritchied` dans l'extrait ci-dessus on a :

— login = `ritchied`
— numéro = `1970`
— numéro de groupe = `100`
— nom complet = `Dennis Ritchie`
— home = `/home/ritchied`
— shell = `/bin/bash`

Pour retrouver ces informations on utilisera les commandes `id` et `pinky` :
`id -u` *login* ⇒ donne le numéro de l'utilisateur
`id -g` *login* ⇒ donne le numéro de son groupe
`pinky -l` *login* ⇒ donne le nom complet, le home et le shell

Exemples :

```
$ id -g ritchied
100
$ id -u ritchied
1970
$ pinky -l ritchied
Identifiant : ritchied                    Nom réel :  Dennis Ritchie
Répertoire : /home/ritchied               Interpréteur :  /bin/bash
```

2.1 Reconstruction de la ligne pour un utilisateur donné

Écrire un script `passUser` qui reçoit en paramètre le login d'un utilisateur et qui fonctionne comme sur les exemples suivants :

```
$ passUser
    Usage : passUser login
$ passUser ritchied
ritchied:x:1970:100:Dennis Ritchie:/home/ritchied:/bin/bash
$ passUser schtroumpf
    Utilisateur inconnu 'schtroumpf'
```

Remarques :

— en cas d'erreur le script retournera une valeur différente de 0 (par exemple avec `exit 1`) ;
— en cas d'erreur les commandes affichent les messages sur la sortie des erreurs, on peut donc éliminer les messages parasites en redirigeant la sortie des erreurs d'une commande vers `/dev/null` (périphérique virtuel qui ne fait rien des caractères qu'on lui envoie) ;
— la variable `$?` du shell contient le résultat de la dernière commande exécutée (0 en cas de succès, une valeur non nulle en cas d'échec) ;

— pour extraire les bonnes informations du résultat de `pinky` il faudra combiner des `grep` et des `cut` ;

— pour éliminer les espaces superflus dans une valeur, on peut la passer à la commande `xargs` :

```
$ echo "   Une poule    sur un   mur" | xargs
Une poule sur un mur
```

— il est recommandé de calculer chaque valeur nécessaire et de la stocker dans une variable avant d'afficher la ligne demandée en affichant toutes les variables.

2.2 Reconstruction pour tous les utilisateurs

Écrire un script `passInfo` qui appelle le script précédent pour chaque utilisateur du BUT Info. Le résultat du script sera un fichier baptisé `PasswdInfo` stocké dans votre répertoire de travail par défaut (*HOME*).

Indications :

— sur les machines Linux des salles de TP, les répertoires utilisateurs ne sont montés qu'à la demande, donc si vous affichez le contenu de `/users/but/info` vous ne voyez que votre login et `Public`.

Pour avoir la liste complète, vous pouvez utiliser le serveur `gigondas` sur lequel tous les répertoires utilisateurs sont montés. Deux solutions s'offrent à vous :

1. Faire tourner votre script sur `gigondas`.
2. Obtenir la liste en allant lire le contenu de `/users/but/info` sur `gigondas` grâce à une requête SSH :

   ```
   ssh gigondas ls /users/but/info
   ```

   Mais cela suppose que vous avez bien installé une clé SSH dans votre compte pour pouvoir accéder à `gigondas` sans donner le mot de passe (sinon il faudra retaper votre mot de passe à chaque lancement de `passInfo` ce qui n'est pas très confortable).

— on n'enregistrera le résultat de `passUser` dans le fichier `PasswdInfo` que si la commande a réussi.

3 S'il vous reste du temps...

Améliorez le script `sauveUSB` pour donner la possibilité d'archiver un répertoire qui n'est pas dans le répertoire courant, par exemple :

```
sauveUSB ../mesTPs/Algo/TPBateaux
```

Il faut découper le paramètre en deux parties (après avoir vérifié que c'était bien un répertoire) : `../mesTPs/Algo` et `TPBateaux` pour pouvoir faire les commandes :

```
cd ../mesTPs/Algo
tar zcf TPBateaux.tgz TPBateaux
mv TPBateaux.tgz /media/votreLogin/nomDeLaClé
```

Vous aurez besoin des commandes `dirname` et `basename` (voir le manuel de ces commandes).

4 Compte-rendu

Vous devez rendre à l'enseignant un listing des scripts que vous avez écrits et un extrait du fichier `PasswdInfo`. Vous pouvez obtenir un listing complet avec :

```
cd $HOME/bin
head $HOME/PasswdInfo > PasswdInfo.extrait
imprime sauveUSB passUser passInfo PasswdInfo.extrait
```

Si vous ne pouvez pas imprimer, enregistrez une copie du fichier PDF produit par la commande `imprime` puis envoyez-la par courriel à l'enseignant en ayant soin de nommer le fichier avec vos noms.