Complément TP1 Administration Unix : programmation shell

1 Script de création/suppression automatique de comptes

Soit le fichier `list` contenant la liste des noms prénoms d'utilisateurs pour lesquels on doit créer des comptes informatiques sur un système Unix.

Le format de la liste est :

```
# commentaire
nom1  prénom1
nom2  prénom2
# commentaire
nom3  prénom3
...
```

1. Écrire un script-shell qui lit le fichier `list` ligne par ligne, et qui en fait l'écho à l'écran sous la forme : `<nom> <prénom>`

```
#!/bin/sh
cat list|while read w1 w2
do
  echo "<$w1> <$w2>"
done
```

2. Modifier le script pour éliminer la(les) ligne(s) de commentaires éventuellement présente(s) dans `list` (ligne commençant par un caractère '#' en début de ligne).

```
#!/bin/sh
cat list|while read w1 w2
do
  first_char=`echo $w1|cut -c1`
  if [ "$first_char" != "#" ]; then
    echo "<$w1> <$w2>"
  fi
done
```

3. Modifier le script pour remplacer les caractères accentués par leur équivalent non accentué : [éèàùç] -> [eeauc]

```
#!/bin/sh
cat list|while read w1 w2
do
  first_char=`echo $w1|cut -c1`
  if [ "$first_char" != "#" ]; then
    w1=`echo $w1|tr [éèàùç] [eeauc]`
    w2=`echo $w2|tr [éèàùç] [eeauc]`
    echo "<$w1> <$w2>"
  fi
done
```

4. Faire afficher à l'écran les informations :
   <nom> <prénom> <INITIALES>
   en tenant compte des points 1. et 2.

```
#!/bin/sh
cat list|while read w1 w2
do
  first_char=`echo $w1|cut -c1`
  if [ "$first_char" != "#" ]; then
    w1=`echo $w1|tr [éèàùç] [eeauc]`
    w2=`echo $w2|tr [éèàùç] [eeauc]`
    ini1=`echo $w1|cut -c1|tr [a-z] [A-Z]`
    ini2=`echo $w2|cut -c1|tr [a-z] [A-Z]`
    ini=$ini1$ini2
    echo "<$w1> <$w2> <$ini>"
  fi
done
```

5. Modifier le script pour passer le nom du fichier liste en argument, et contrôler l'existence du fichier avant de poursuivre le traitement du script.
   Utiliser un message de type :
   - « Usage : ... », si l'argument n'a pas été donné,
   - « Fichier <...> inexistant », si le fichier passé en argument n'existe pas.

```
#!/bin/sh
fic=$1
if [ "$fic" == "" ]; then
  echo "Usage : compte.sh file"
  exit
fi
if [ ! -f "$fic" ]; then
  echo "Fichier <$fic> inexistant!"
  exit
fi
cat $fic|while read w1 w2
do
  first_char=`echo $w1|cut -c1`
  if [ "$first_char" != "#" ]; then
    w1=`echo $w1|tr [éèàùç] [eeauc]`
    w2=`echo $w2|tr [éèàùç] [eeauc]`
    ini1=`echo $w1|cut -c1|tr [a-z] [A-Z]`
    ini2=`echo $w2|cut -c1|tr [a-z] [A-Z]`
    ini=$ini1$ini2
    echo "<$w1> <$w2> <$ini>"
  fi
done
```

6. Pour éviter les doublons éventuels au niveau des logins, on prend les 2 premiers caractères du nom et les 2 premiers caractères du prénom, tout en minuscule, pour former le nom de login de l'utilisateur. Faire afficher ce login à la place de <INITIALES> du point 4.

```
#!/bin/sh
fic=$1
if [ "$fic" == "" ]; then
  echo "Usage : compte.sh file"
  exit
fi
if [ ! -f "$fic" ]; then
  echo "Fichier <$fic> inexistant!"
  exit
fi
cat $fic|while read w1 w2
do
  first_char=`echo $w1|cut -c1`
  if [ "$first_char" != "#" ]; then
    w1=`echo $w1|tr [éèàùç] [eeauc]`
    w2=`echo $w2|tr [éèàùç] [eeauc]`
    ini1=`echo $w1|cut -c1,2|tr [A-Z] [a-z]`
    ini2=`echo $w2|cut -c1,2|tr [A-Z] [a-z]`
    ini=$ini1$ini2
    echo "<$w1> <$w2> <$ini>"
  fi
done
```

7. Utiliser le login constitué au point 6. pour créer des comptes en utilisant les informations suivantes pour la commande `useradd` :

login : comme défini au point 6.
home : /home/NOM_xy, avec xy : les 2 premières initiales du prénom en minuscule
UID : 1000 pour le premier compte, puis incrémenter de 1 à chaque compte
groupe : users (le créer si besoin)
commentaire : « prénom NOM-TP ADMIN LINUX »
shell : /bin/bash
passwd : créer un compte « guest » (useradd guest) ;
Lui affecter un mot de passe ;
utiliser le mot de passe crypté de guest comme mot de passe pour l'option -p de la commande useradd.
Créer si besoin les répertoires des utilisateurs, changer de propriétaire et de groupe si besoin.

On créé d'abord un compte guest et on lui affecte un mot de passe:

```
useradd guest
passwd guest
Changing password for user guest.
```

```
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

```
#!/bin/sh
fic=$1
if [ "$fic" == "" ]; then
  echo "Usage : compte.sh file"
  exit
fi
if [ ! -f "$fic" ]; then
  echo "Fichier <$fic> inexistant!"
  exit
fi
DEL=0
if [ "$2" = -del ]; then
  DEL=1
fi

let uid=1000
cat $fic|while read w1 w2
do
  first_char=`echo $w1|cut -c1`
  if [ "$first_char" != "#" ]; then
    w1=`echo $w1|tr [éèàùç] [eeauc]`
    w2=`echo $w2|tr [éèàùç] [eeauc]`
    ini1=`echo $w1|cut -c1,2|tr [A-Z] [a-z]`
    ini2=`echo $w2|cut -c1,2|tr [A-Z] [a-z]`
    ini=$ini1$ini2
    echo "<$w1> <$w2> <$ini>"
    echo -n "creation du compte $ini : "
    pass=`grep guest /etc/shadow |cut -d: -f 2`
     useradd -c "$w3 $w1-TP ADMIN UNIX" -d /home/${w1}_$ini2 -g users -u $uid -s /
bin/bash -p "$pass" $ini
    mkdir /home/${w1}_$ini2
    chown -R $ini /home/${w1}_$ini2
    chgrp -R users /home/${w1}_$ini2
  fi
  let uid=$uid+1
fi
done
```

8. Tester manuellement la connexion sur les comptes ainsi créés.
On change de console (ALT+F2) par exemple, et on se connecte avec le login à tester.

9. Modifier le script pour prévoir une option `-del` : cette option détruit les comptes des utilisateurs (`userdel`) et affiche à chaque fois un message :

```
  suppression du compte <....> : [OK]
```

[ERROR] + message

[OK] : si la suppression a réussi
[ERROR] : dans le cas contraire, suivi du message d'erreur de `userdel`

fichier compte.sh

```
#!/bin/sh
fic=$1
if [ "$fic" == "" ]; then
  echo "Usage : compte.sh file"
  exit
fi
if [ ! -f "$fic" ]; then
  echo "Fichier <$fic> inexistant!"
  exit
fi
DEL=0
if [ "$2" = -del ]; then
  DEL=1
fi
let uid=1000
cat $fic|while read w1 w2
do
  first_char=`echo $w1|cut -c1`
  if [ "$first_char" != "#" ]; then
    w1=`echo $w1|tr [éèàùç] [eeauc]`
    w2=`echo $w2|tr [éèàùç] [eeauc]`
    ini1=`echo $w1|cut -c1,2|tr [A-Z] [a-z]`
    ini2=`echo $w2|cut -c1,2|tr [A-Z] [a-z]`
    ini=$ini1$ini2
    echo "<$w1> <$w2> <$ini>"
    if [ "$DEL" = 1 ];then
      echo -n "suppression du compte $ini : "
      mess=`userdel $ini 2>&1`
      if [ $? = 0 ];then
        echo [OK]
      else
        echo [ERROR] $mess
      fi
    else
      echo -n "creation du compte $ini : "
      pass=`grep guest /etc/shadow |cut -d: -f 2`
     useradd -c "$w3 $w1-TP ADMIN UNIX" -d /home/${w1}_$ini2 -g users -u $uid -s /
bin/bash -p "$pass" $ini
      mkdir /home/${w1}_$ini2
      chown -R $ini /home/${w1}_$ini2
      chgrp -R users /home/${w1}_$ini2
    fi
    let uid=$uid+1
  fi
done
```

2 Recherche/signatureMD5/vérification des scripts set_UID/set_GUID appartenant à root

1. Écrire un script qui recherche tous les fichiers du PC à partir de la racine, correspondant aux critères :
- propriétaire : root
- « set_UID » bit ou « set_GID » bit monté.

Pour réduire le temps d'exécution du script, on limitera la recherche à la liste des répertoires mentionnés par la variable PATH.

```
#!/bin/sh
# former la liste des répertoires séparés par des espaces
liste_rep=`echo $PATH|tr : ' '`
# rechetcher les fichiers de root ayant au moins le set-uid-bit
liste=`find $liste_rep -user root -perm -4000`
```

2. Pour chacun des fichiers trouvés, faire calculer le checksum MD5, et enregistrer dans un fichier (idéalement sur une disquette ...) les couples « Checksum, nom_absolu_du_fichier ».

fichier set_sugid_liste.sh

```
#!/bin/sh
WRITE_DIR=/mnt/floppy
# le repertoire /mnt/floppy existe-t-il? si non, le creer
test -d  $WRITE_DIR || mkdir $WRITE_DIR
# la disquette est-elle deja montee ? si non, la monter
mount |grep -q fd0
if [ $? = 1 ]; then
    mount /dev/fd0  $WRITE_DIR 2> /dev/null
    if [ $? != 0 ]; then
      echo "Pas de disquette trouvée."
      echo "Insérer une disquette dans le lecteur et relancer le script SVP"
      exit 1
    fi
fi

# former la liste des répertoires séparés par des espaces
liste_rep=`echo $PATH|tr : ' '`
# rechetcher les fichiers de root ayant au moins le set-uid-bit
liste=`find $liste_rep -user root -perm -4000`

> $WRITE_DIR/root_set_uid_gid_file_liste.txt
for file in $liste
do
  md5sum $file >>  $WRITE_DIR/root_set_uid_gid_file_liste.txt
done
#démonter la disquette
umount /dev/fd0
exit 0
```

3. Écrire un script qui vérifie :
- qu'il n'y a pas de nouveau fichier set_UID ou set_GID par rapport à la liste enregistrée
- que le checksum calculé est identique au checksum calculé, pour chaque fichier.

fichier check_sugid.sh

```
#!/bin/sh
WRITE_DIR=/mnt/floppy
# le repertoire /mnt/floppy existe-t-il? si non, le creer
test -d  $WRITE_DIR || mkdir $WRITE_DIR
# la disquette est-elle deja montee ? si non, la monter
mount |grep -q fd0
if [ $? = 1 ]; then
    mount /dev/fd0  $WRITE_DIR 2> /dev/null
    if [ $? != 0 ]; then
      echo "Pas de disquette trouvée."
      echo "Insérer une disquette dans le lecteur et relancer le script SVP"
      exit 1
    fi
fi

# former la liste des répertoires séparés par des espaces
liste_rep=`echo $PATH|tr : ' '`
# rechetcher les fichiers de root ayant au moins le set-uid-bit
liste=`find $liste_rep -user root -perm -4000`

for file in $liste
do
  #récupérer l'ancien MD5 correpondant au fichier $file
   md5sum_old=`grep "$file\$" $WRITE_DIR/root_set_uid_gid_file_liste.txt|cut -d' '
-f1`
  if [ "$md5sum_old" = "" ]; then
    # pas d'ancien MD5 : c'est un nouveau fichier
    printf "%64s %s\n" "Nouveau fichier détecté $file :" "`ls -l $file` !!! ALERTE
!!!"
  else
    # calculer le nouveau MD5 et comparer
    md5sum_new=`md5sum $file|cut -d' ' -f1`
    if [ "$md5sum_old" = "$md5sum_new" ]; then
      printf "%64s %s\n" "$file :" "checksum MD5 inchangé"
    else
      printf "%64s %s\n" "$file :" "checksum MD5 changé   !!! ALERTE !!!"
    fi
  fi
done

#démonter la disquette
umount /dev/fd0
exit 0
```

4 Programmer (cron) le déclenchement de ce script tous les matins à 8h.

Liste des actions de crontab :

```
crontab -l
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (/tmp/crontab.19549 installed on Fri Nov  4 16:56:55 2005)
# (Cron version -- $Id: crontab.c,v 2.13 1994/01/17 03:20:37 vixie Exp $)
0,5,10,15,20,25,30,35,40,45,50,55 * * * * chmod 777 /home/graveur/
0,10,20,30,40,50 * * * * sh /root/check_cups.sh
```

copie des actions crontab dans le fichier cron_root.txt:

```
crontab -l > /root/cron_root.txt
```

Puis on édite cron_root.txt en enlevant les commentaires superflus (début de fichier) pour ajouter la ligne :

```
0 8 * * * sh /root/check_sugid.sh > /root/log 2>&1
```

On soumet le nouveau fichier :

```
crontab /root/cron_root.txt
```

On vérifie le nouveau crontab :

```
crontab -l
```

Pour que cela marche, il faut laisser la disquette dans le lecteur.