LABO7 – Sauvegarder ses données

1 POURQUOI SAUVEGARDER ?

Si vous vous posez la question, c'est que vous n'avez jamais perdu de données ! On réalise souvent l'importance de procéder à des sauvegardes régulières trop tard, une fois confronté à une panne de disque dur, au vol de son portable, au vandalisme, un virus, une mauvaise manipulation ou une catastrophe naturelle. Vos photos et vos documents divers sont précieux, mais en entreprise, l'importance est d'autant plus grande : mettez ces données informatique à l'abri ! S'astreindre à des sauvegardes n'est pas si contraignant, et vous vous remercierez infiniment de vous y être plié si vous êtes confronté à une perte.

Attention, le cloud, ne vous protège pas de mauvaise manipulation ou d'effacement de données si l'archivage n'est pas activé !

2 SAUVEGARDEZ LES DONNEES DE VOTRE LINUX

2.1. Script de sauvegarde

L'un des moyens les plus simples de sauvegarder un système consiste à utiliser un script shell. Par exemple, un script peut être utilisé pour configurer les répertoires à sauvegarder et transmettre ces répertoires comme arguments à l'utilitaire tar, qui crée un fichier archive. Le fichier d'archive peut ensuite être déplacé ou copié vers un autre emplacement. L'archive peut également être créée sur un système de fichiers distant tel qu'un montage NFS.

L'utilitaire ***tar*** crée un fichier d'archive de plusieurs fichiers ou répertoires. *tar* peut également filtrer les fichiers par le biais des utilitaires de compression, réduisant ainsi la taille du fichier d'archive.

2.2. Script shell simple

Le script shell suivant se sert de ***tar*** pour créer une archive sur un montage NFS. Le nom de l'archive est défini en se servant de divers utilitaires en ligne de commande.

```
#!/bin/bash
####################################
#
# Backup to NFS mount script.
#
####################################

# What to backup.
backup_files="/home /var/spool/mail /etc /root /boot /opt"

# Where to backup to.
dest="/mnt/hgfs/Sauvegardes"

# Create archive filename.
day=$(date +%A)
hostname=$(hostname -s)
archive_file="$hostname-$day.tgz"

# Print start status message.
echo "Backing up $backup_files to $dest/$archive_file"
date
echo

# Backup the files using tar.
tar czf $dest/$archive_file $backup_files


# Print end status message.
echo
echo "Backup finished"
date


# Long listing of files in $dest to check file sizes.
ls -lh $dest
```

1. *$backup_files :* variable listant les répertoires que vous souhaitez sauvegarder. La liste doit être adaptée à vos besoins.
2. *$day :* une variable contenant le jour de la semaine (lundi, mardi, mercredi, etc.) Cette fonction est utilisée pour créer un fichier d'archive pour chaque jour de la semaine, ce qui donne un historique de sauvegarde de sept jours. Il y a d'autres façons d'accomplir cela, par exemple en utilisant l'utilitaire *date*.
3. *$hostname :* variable contenant le nom d'hôte de votre système en format *court*. Vous pourrez ainsi placer les archives quotidiennes de plusieurs hôtes dans un même répertoire.
4. *$archive_file :* le nom complet de l'archive.
5. *$dest :* destination du fichier d'archive. Le répertoire doit être créé et, dans ce cas *monté* avant d'exécuter le script de sauvegarde.
6. *status messages :* messages optionnels affichés dans la console lors de l'utilisation de la commande *echo*.
7. *tar czf $dest/$archive_file $backup_files :* la commande *tar* utilisée pour créer le fichier d'archive.
    1. *c :* crée une archive.
    2. *z :* compresser l'archive avec *gzip*.
    3. *f :* sortie vers un fichier d'archive. Sinon, la sortie *tar* est envoyée vers STDOUT.
8. *ls -lh $dest :* instruction optionnelle affichant une liste du répertoire de destination détaillée *-l* et au format lisible par facilement *-h*. Ceci est utile pour vérifier rapidement la taille du fichier archive, mais ne devrait pas remplacer le test de celui-ci.

Voici un exemple simple de script shell de sauvegarde. Cependant, il y a beaucoup d'options qui peuvent être inclues dans un tel script. Voir **Références** pour avoir des liens vers des ressources fournissant des informations plus approfondies sur les scripts shell.

2.3. Exécution du script

2.3.1 *Exécution à partir d'un terminal*

La manière la plus simple d'exécuter le script de sauvegarde ci-dessus consiste à copier et coller le contenu dans un fichier. backup.sh par exemple. Le fichier doit être rendu exécutable:

```
chmod u+x backup.sh
```

Puis à partir d'une invite de commande :

```
sudo ./backup.sh
```

C'est une excellente manière de tester le script pour s'assurer que tout fonctionne comme prévu.

2.3.2 *Exécution avec cron*

L'exécution du script peut être automatisée avec l'utilitaire ***cron***. Le démon ***cron*** permet d'exécuter des scripts ou des commandes, à une date et une heure déterminée.

cron est configuré au travers d'entrées dans un fichier **`crontab`**. Les fichiers **`crontab`** sont séparés en champs :

```
# m h dom mon dow   command
```

1. *m:* minute de l'exécution de la commande, entre 0 et 59.
2. *h:* heure de l'exécution de la commande, entre 0 et 23.
3. *dom :* jour du mois durant lequel la commande s'exécute.
4. *mon:* mois de l'exécution de la commande, entre 1 et 12.
5. *dow :* le jour de la semaine ("day of week") où la commande s'exécute (entre 0 et 7). Dimanche peut être spécifié à l'aide de 0 ou 7, les deux valeurs sont valides.
6. *command :* la commande à exécuter.

La commande ***crontab -e*** doit être utilisée pour ajouter ou modifier les entrées d'un fichier **`crontab`**. Le contenu d'un fichier **`crontab`** peut être affiché avec la commande ***crontab -l***.

Afin d'exécuter le script *backup.sh* précédent en utilisant ***cron***, saisissez dans une invite de terminal :

```
sudo crontab -e
```

L'utilisation de ***sudo*** avec la commande ***crontab -e*** modifie le fichier **crontab** de l'utilisateur ***root***. Cela est nécessaire si vous sauvegardez des répertoires accessibles uniquement par l'utilisateur **root**.

Ajoutez l'entrée suivante au fichier **`crontab`** :

```
# m h dom mon dow   command

0 0 * * * bash /usr/local/bin/backup.sh
```

Le script ***backup.sh*** sera maintenant lancé tous les jours à minuit.

Le script ***backup.sh*** devra être copié dans le répertoire `/usr/local/bin/` pour que cette entrée s'exécute correctement. Le script peut se trouver n'importe où sur le système de fichiers, il suffira de changer le chemin d'accès du script en conséquence.

2.4. Restauration à partir d'une archive

Il est important de vérifier une archive après sa création. Une archive peut être testée en listant les fichiers qu'elle contient, mais le mieux est de *restaurer* un fichier depuis cette archive.

1. Pour voir une liste du contenu de l'archive. Saisissez à partir d'un shell :

```
tar -tzvf /mnt/backup/host-Monday.tgz
```

2. Pour restaurer un fichier à partir de l'archive dans un répertoire différent, tapez :

```
tar -xzvf /mnt/backup/host-Monday.tgz -C /tmp etc/hosts
```

L'option ***-C*** pour ***tar*** redirige les fichiers extraits vers le répertoire spécifié. L'exemple ci-dessus va extraire le fichier `/etc/hosts` vers `/tmp/etc/hosts`. ***tar*** recrée l'arborescence des dossiers qu'il contient.

Notez également que le premier « / » est enlevé du chemin du fichier à restaurer.

3. Pour restaurer tous les fichiers de l'archive saisissez ceci :

```
cd /
sudo tar -xzvf /mnt/backup/host-Monday.tgz
```

ATTENTION, cela écrasera les fichiers actuellement sur le système de fichiers.

2.5. Exercices

1. Mettez en place un partage de fichier entre votre VM et votre host.
   Le point de montage du disque partagé VMware est localisé sur ***/mnt/hgfs/nom_du_partage***

2. Mettez en place la solution de sauvegarde en modifiant le script afin que les sauvegardes soient placées sur le partage avec votre host.

3. Définissez la plage horaire le plus propice à la sauvegarde (un moment où votre VM est active et où votre système est peu utilisé) et configurez un **cronjob**.

4. Améliorez votre script de sauvegarde pour crypter les sauvegardes (avec un algorithme de cryptage symétrique). Attention à mémoriser la passe-phrase de sauvegarde.

2.6. **Références**

1. Voir Advanced Bash-Scripting Guide pour de plus amples informations à propos de l'écriture de scripts shell.
2. Le livre Teach Yourself Shell Programming in 24 Hours est disponible en ligne et est une mine d'or pour l'écriture de scripts shell.
3. La Page Wiki CronHowto contient des détails sur l'utilisation des options avancées de *cron*.
4. Voir le Manuel GNU tar pour plus d'informations concernant les options de *tar*.
5. L'article anglais de Wikipédia Backup Rotation Scheme contient des informations à propos d'autres méthodes de sauvegarde. Une ébauche en français est disponible dans l'article Sauvegarde de la version francophone de Wikipedia.
6. Le script shell utilise la commande *tar* pour la création de l'archive, mais il existe bien d'autres utilitaires en ligne de commande qui peuvent être utilisés. Par exemple :
    1. cpio : pour copier des fichiers depuis et vers des archives.
    2. dd : fait partie du paquet *coreutils*. Un utilitaire de bas niveau qui peut copier les données d'un format à un autre.
    3. rsnapshot: un utilitaire de capture instantanée du système de fichiers utilisé pour créer des copies d'un système de fichiers.
    4. rsync: un utilitaire flexible utilisé pour créer des copies incrémentielles des fichiers.