Scripts Bash utiles

- A Simple Productivity Hack Using Bash Scripts Sebastian Scholl, Medium, 2019
- 5 Bash Syntax For Going Beyond Traditional Shell Scripting - Using modern Bash syntaxes for writing general-purpose programs Shalitha Suranga, 06/06/2022, Medium
- Bash vs. Python: For Modern Shell Scripting - Comparing Bash and Python options for modern automation requirements Shalitha Suranga, Medium, 21/02/2023
- Bash Scripting Tutorial Luke Reynolds, 03/04/2023
- Advanced Linux Shell Scripting OMKAR KHEDKAR, 07/03/2023

Script bash de backup

- Question via ChatGPT : **“Je souhaite disposer d'un script bash de backup pour sauvegarder une copie complète compressée d'un répertoire sous Ubuntu dans un répertoire dédié aux backups. Le script doit également effacer les backups antérieurs dès qu'ils sont plus vieux que 10 jours. Par contre, le backup de chaque premier jour du mois doit être déplacé vers un répertoire d'un autre nom.”**
- Réponse :

Voici un exemple de script bash qui répond à vos exigences pour la sauvegarde et la suppression des backups antérieurs :

backup.sh

```
#!/bin/bash

# Répertoire source à sauvegarder
source_directory="/chemin/vers/le/repertoire/source"

# Répertoire de destination pour les backups
backup_directory="/chemin/vers/le/repertoire/de/backups"

# Répertoire de destination pour les backups mensuels
monthly_directory="/chemin/vers/le/repertoire/de/backups/mensuels"

# Supprimer les backups antérieurs à 10 jours
find "$backup_directory" -type f -mtime +10 -exec rm {} \;

# Créer un répertoire pour la sauvegarde actuelle (nom basé sur la date et l'heure)
backup_folder="$(date +%Y%m%d%H%M%S)"
mkdir "$backup_directory/$backup_folder"

# Copier les fichiers du répertoire source vers la sauvegarde actuelle
cp -R "$source_directory" "$backup_directory/$backup_folder"

# Compresser la sauvegarde
tar -czf "$backup_directory/$backup_folder.tar.gz"
```

```
"$backup_directory/$backup_folder"

# Supprimer le répertoire non compressé
rm -rf "$backup_directory/$backup_folder"

# Déplacer la sauvegarde mensuelle du premier jour du mois vers le
répertoire mensuel
if [ "$(date +%d)" = "01" ]; then
    monthly_folder="$(date +%Y%m)"
    mkdir -p "$monthly_directory/$monthly_folder"
    mv "$backup_directory/$backup_folder.tar.gz"
"$monthly_directory/$monthly_folder"
fi
```

Assurez-vous de remplacer les chemins d'accès réels pour source_directory, backup_directory et monthly_directory par les chemins appropriés sur votre système. Vous pouvez également ajuster les délais de suppression ou les noms de répertoires selon vos besoins.

Enregistrez ce script dans un fichier, par exemple backup.sh, puis rendez-le exécutable en utilisant la commande suivante :

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

Vous pouvez ensuite exécuter le script en utilisant la commande suivante :

```
./backup.sh
```

Le script va créer un nouveau répertoire de sauvegarde basé sur la date et l'heure actuelles, copier les fichiers du répertoire source vers ce répertoire, compresser le répertoire de sauvegarde en un fichier tar.gz, puis supprimer le répertoire non compressé. Si la date actuelle correspond au premier jour du mois, le fichier de sauvegarde sera déplacé vers le répertoire mensuel spécifié.

N'oubliez pas d'adapter le script en fonction de vos besoins spécifiques et de tester attentivement avant de l'utiliser en production.

- Question complémentaire : **“Je souhaite placer aussi le backup effectué chaque 15 du mois. Je suppose qu'il suffit de rajouter une combinaison logique “ou” avec le test [“$(date +%d)” = “15”]. Pouvez-vous me donner la ligne complète correcte ?”**

Oui, vous pouvez ajouter une condition supplémentaire pour le 15 du mois en utilisant l'opérateur logique “||” (ou). Voici la ligne complète modifiée :

```
if [ "$(date +%d)" = "01" ] || [ "$(date +%d)" = "15" ]; then
```

Avec cette modification, la sauvegarde sera déplacée vers le répertoire mensuel à la fois le premier jour et le quinzième jour de chaque mois.

- Question complémentaire : **“Lorque le fichier de sauvegarde compressé est obtenu, et avant de considérer son déplacement éventuel vers le répertoire de backups mensuels, je souhaite copier ce fichier par rsync vers un serveur distant. Comment**

configurer de manière efficace et en respectant les règles de sécurité les clés SSH sur le serveur sauvegardé et le serveur distant, en fonction des utilisateurs (local et distant) ? Merci.”

Pour configurer efficacement et en respectant les règles de sécurité les clés SSH entre le serveur local (où se trouve le script de sauvegarde) et le serveur distant, vous pouvez suivre ces étapes :

1. Génération des clés SSH sur le serveur local :
   - Connectez-vous au serveur local en tant qu'utilisateur souhaité (par exemple, utilisateur_local).
   - Ouvrez un terminal et exécutez la commande suivante pour générer une paire de clés SSH :

     ```
     ssh-keygen -t rsa -b 4096
     ```

   - Vous pouvez appuyer sur Entrée pour accepter l'emplacement et le nom de fichier par défaut pour la clé.
   - Vous serez invité à entrer une phrase de passe optionnelle pour protéger votre clé privée. Il est recommandé d'en utiliser une pour renforcer la sécurité.
2. Ajout de la clé publique sur le serveur distant :
   - Copiez la clé publique générée sur le serveur local vers le serveur distant en utilisant la commande suivante :

     ```
     ssh-copy-id -i ~/.ssh/id_rsa.pub
     utilisateur_distant@adresse_du_serveur
     ```

   - Remplacez utilisateur_distant par le nom de l'utilisateur sur le serveur distant et adresse_du_serveur par l'adresse IP ou le nom d'hôte du serveur distant.
   - Vous serez invité à entrer le mot de passe de l'utilisateur distant. Entrez-le pour autoriser la copie de la clé publique.
3. Vérification de la connexion SSH sans mot de passe :
   - Essayez de vous connecter au serveur distant à partir du serveur local en utilisant la commande suivante :

     ```
     ssh utilisateur_distant@adresse_du_serveur
     ```

   - Si tout est configuré correctement, vous devriez vous connecter au serveur distant sans être invité à entrer un mot de passe.
4. Utilisation de rsync avec la clé SSH :
   - Pour utiliser rsync avec la clé SSH pour copier le fichier de sauvegarde vers le serveur distant, vous pouvez utiliser la commande suivante dans votre script de sauvegarde :

     ```
     rsync -e "ssh -i /chemin/vers/clé_privée" fichier_sauvegarde
     utilisateur_distant@adresse_du_serveur:/chemin/destination
     ```

   - Remplacez /chemin/vers/clé_privée par le chemin complet vers votre clé privée générée précédemment, fichier_sauvegarde par le nom du fichier de sauvegarde compressé et /chemin/destination par le répertoire de destination sur le serveur distant.

Veillez à protéger vos clés privées en les gardant sécurisées et en leur accordant des permissions appropriées (par exemple, en utilisant chmod 600). De plus, assurez-vous que les utilisateurs locaux et distants disposent des droits appropriés pour accéder aux fichiers et répertoires nécessaires.

Ces étapes vous permettront de configurer une connexion SSH sécurisée entre le serveur local et le serveur distant, en utilisant des clés SSH pour l'authentification sans mot de passe lors de l'utilisation de rsync.

- Question complémentaire : **“Quels sont les rôles des options -e de rsync, et -i de ssh ?”**

Les options -e de rsync et -i de ssh ont les rôles suivants :

1. Option -e de rsync :
   - L'option -e de rsync permet de spécifier la commande à utiliser pour établir la connexion SSH entre le client rsync (le serveur local) et le serveur distant. Elle permet de définir l'interpréteur de shell ou le programme d'accès distant à utiliser. Par défaut, rsync utilise ssh comme protocole de transfert sécurisé, mais vous pouvez utiliser l'option -e pour spécifier d'autres programmes si nécessaire.
   - Par exemple, pour utiliser ssh comme protocole de transfert sécurisé, vous pouvez utiliser

     ```
     -e "ssh"
     ```

     dans votre commande rsync. Si vous avez besoin de spécifier un fichier de clé privée SSH, vous pouvez utiliser

     ```
     -e "ssh -i /chemin/vers/clé_privée"
     ```

2. Option -i de ssh :
   - L'option -i de ssh permet de spécifier le chemin vers le fichier de clé privée à utiliser pour l'authentification SSH. Lorsque vous vous connectez à un serveur distant, ssh utilise par défaut la clé privée située dans ~/.ssh/id_rsa. L'option -i permet de spécifier une autre clé privée si vous en avez généré une ou si vous souhaitez utiliser une clé différente.
   - Par exemple, vous pouvez utiliser

     ```
     -i /chemin/vers/clé_privée
     ```

     pour spécifier le chemin complet vers votre clé privée lors de l'utilisation de ssh.

En combinant ces deux options, vous pouvez utiliser rsync avec une connexion SSH sécurisée en spécifiant la commande ssh à utiliser avec l'option -e et en spécifiant la clé privée à utiliser avec l'option -i.

From:
https://dvillers.umons.ac.be/wiki/ - **Didier Villers, UMONS - wiki**

Permanent link:
https://dvillers.umons.ac.be/wiki/floss:bash_scripts?rev=1685521147

Last update: **2023/05/31 10:19**