Solution de Sauvegarde rsync/cron

Cours sur la Sauvegarde Automatique avec Rsync et SSH en Mode Push

I. Introduction

La sauvegarde automatique des fichiers via Rsync et SSH en mode Push assure la sécurité et la disponibilité des données.
Ce processus automatisé synchronise les fichiers depuis plusieurs machines vers un serveur de sauvegarde.

II. Compréhension de Rsync et SSH

Rsync (Remote Sync) : Rsync est un outil de synchronisation de fichiers à distance et locaux, réputé pour sa capacité à transférer uniquement
les parties modifiées des fichiers, réduisant ainsi la bande passante utilisée et accélérant les transferts.

SSH (Secure Shell) : SSH est un protocole de communication sécurisé utilisé pour l'accès distant aux machines. Il fournit un canal crypté pour
les communications, ce qui le rend idéal pour les transferts de données sensibles.

III. Mode Push avec Rsync et SSH

Dans le mode Push, c'est la machine source qui initie la connexion et envoie les fichiers au serveur de sauvegarde. Cela offre une flexibilité accrue et permet un contrôle
fin sur les sauvegardes.

IV. Fonctionnement de la Sauvegarde en Mode Push

Initialisation de la sauvegarde : La machine source lance une connexion SSH vers le serveur de sauvegarde pour envoyer les fichiers à sauvegarder.

Identification des fichiers modifiés : Rsync compare les fichiers sur la machine source avec ceux sur le serveur de sauvegarde pour déterminer les modifications.

Transfert des données modifiées : Seules les parties modifiées des fichiers sont transférées via SSH vers le serveur de sauvegarde, minimisant ainsi la quantité de données échangées.

Synchronisation des fichiers : Les fichiers sur le serveur de sauvegarde sont mis à jour pour refléter les dernières versions des fichiers sur la machine source.

V. Avantages de la Sauvegarde en Mode Push

Contrôle granulaire : La machine source contrôle l'initiation des sauvegardes, offrant ainsi un contrôle fin sur le processus.

Facilité de configuration : Les machines source peuvent être configurées pour envoyer des sauvegardes sans nécessiter de configuration spéciale sur le serveur de sauvegarde.

Transferts sécurisés : SSH assure un transfert sécurisé des données, garantissant que les informations sensibles ne sont pas compromises pendant le transfert.

VI. Intégration avec Cron

Pour automatiser le processus de sauvegarde en mode Push, nous pouvons utiliser Cron, un outil de planification de tâches sous Unix.

1 - Créez un script Rsync qui initie la sauvegarde depuis la machine source vers le serveur de sauvegarde via SSH.

2 - Utilisez Cron pour planifier l'exécution périodique du script de sauvegarde sur la machine source.

VII. Conclusion

La sauvegarde automatique en mode Push avec Rsync et SSH offre un moyen efficace de sécuriser et de sauvegarder les fichiers de configuration. En suivant ce modèle,
les organisations peuvent garantir la disponibilité et l'intégrité de leurs données tout en réduisant la nécessité d'interventions humaines.

TP

création d'un nouveau containers

création d'un containers et configuration de l'adresse IP du nouveau containers qui s'appellera backup

```
nano /etc/network/interfaces
```

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
        address 10.31.80.98/20
        gateway 10.31.95.254
        dns-nameservers 8.8.8.8
```

```
systemctl restart networking
```

FQDN

il suffit d'ajouter un enregistrement dans le fichier de zone du DNS

```
@ IN NS backup2.m2l.org.
backup2 IN A 10.31.80.99
```

installation de Rsync

pour ce TP nous auront besoins de Rsync pour pouvoir créer une solution de sauvegarde

```
apt install rsync
```

Récupération des clé publique

Méthode utilisé : Pull

Nous allons générer plusieurs clés SSH sur plusieurs serveurs, tels que **WEB, DNS1 et DNS2**, puis nous les ajouterons dans le fichier authorized_keys.

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

Exemple de clé

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAACAQDK50h6TQo2lJVvVClPOCBzMCA3RL9MTjPacDN6SuMBNR0Q
g+/lieH83g/noBSnTP3xNc66RWd05b+JMzvTfD7/oES6AAD1Q/MMqoM6IeU5sbwNX6K/JsqVDI5y
jkq+gKZu5jqNBAycOY3cf2Z/ZKNhIiOowTj3KjnJNk6bLthq5vHoqLVOtvzEKUB+py5p9Od9j99L
LbfPnBTTS/8l/tGB2F2FFgYYC6qJwL4f5XAapVaxQzlbwdEclQb75kzNqyKREg3mJPNKCigUpPrK
MqlGNCNbu5OXDhAhCb8UaQ4FChGcBGa2HtxSwS3ncmhQ6WBlF3GGTfH7jQSwOtMCZUAVtMw7bzVn
Gw68cYPq/wNgxGZYSj/ldTx6TG0Q6JY9cH5gIumgdbEP/rI79v+Fs1K7yRX/lSAOmDLxNuI9ORqa
8jCxPUbVbhS/XCgwrB8ljMf6NhjPR45YM7/crwal3o2pzfCvdfdrZ6VMGSt3/dVl8LaClecrvGeX
p6Ubn2ZrLkC8h1PheeffYVCWTc1EBI2uQUDPZvakyAf789yijylkiN7w4JXpbAbNY2fEAB7+Swfo
/gIFGO2sCAPDwtr91KzvzJ7JzP6oWfnbbpRqEsfTlyxRH4C85nTCUAXI3mreWmnja56aFlktUPuf
uKOvcFMGcei6bAI56D5oZj+wJSeeCw== root@web
```

Après avoir copié les clés, il est nécessaire de les coller dans le fichier authorized_keys dans le serveur backup.

```
# pour se connecter au containers backup

lxc-attach backup

# pour éditer le fichiers authorized_keys
```

```
nano /root/.ssh/authorized_keys

# /!\ le répertoire .ssh commence par un point il ne sera pas lister par la commande ls, ajouter cette options
ls -a
```

puis création des répertoires dans /home du serveur backup pour recevoir les données

```
cd /home
mkdir backup
cd backup
mkdir web
mkdir dns
```

schéma :

```
        /home
          |
          └── backup
                |
                ├── web
                └── dns
```

puis utiliser ses propres DNS sur la totalité des serveurs

```
nano /etc/resolv.conf

#contenu
10.31.80.53
10.31.80.63
```

Les scripts Bash

Les scripts Bash qui permettent d'envoyer les données vers le serveur de sauvegarde seront placés sur les serveurs où se trouvent les données à récupérer.

Au total 3 script

1 - script DNS1
1 - script DNS2
1 - script WEB

Script Bash WEB

```
#! /bin/bash


DATE=$(date +"%Y-%m-%d")
HEURE=$(date +"%H:%M:%S")
LOG_FILE="./logs/backup_log_$DATE.log"

echo "-- Début de la sauvegarde : $DATE à $HEURE" >> "$LOG_FILE"
echo "-----------------------------------------------------" >> "$LOG_FILE"

rsync -avz --delete /etc/apache2 root@backup2.m2l.org:/home/backup/web >>
"$LOG_FILE"
rsync -avz --delete /var/www/html root@backup2.m2l.org:/home/backup/web/ >>
"$LOG_FILE"
rsync -avz --delete /var/log/apache2 root@backup2.m2l.org:/home/backup/web
>> "$LOG_FILE"

mysqldump -u dba -pdrowssap --all-databases > "./sql/all-databases-
$DATE.sql"
rsync -avz "./sql/all-databases-$DATE.sql"
root@backup2.m2l.org:/home/backup/web >> "$LOG_FILE"

echo "-- FIN -------------------------------------------" >> "$LOG_FILE"
```

Script Bash DNS1

```
#!/bin/bash

DATE=$(date +"%Y-%m-%d")
LOG_FILE="./backup_log_$DATE.log"

echo "-- Début de la sauvegarde : $DATE" >> "$LOG_FILE"
echo "-----------------------------------------------------" >> "$LOG_FILE"

rsync -avz --delete /etc/bind root@backup2.m2l.org:/home/backup/dns/dns1 >>
"$LOG_FILE"
rsync -avz --delete /var/log/syslog
root@backup2.m2l.org:/home/backup/dns/dns1 >> "$LOG_FILE"

echo "-- FIN -------------------------------------------" >> "$LOG_FILE"
```

Script Bash DNS2

```
#!/bin/bash

DATE=$(date +"%Y-%m-%d")
LOG_FILE="./backup_log_dns2_$DATE.log"

echo "-- Début de la sauvegarde : $DATE" >> "$LOG_FILE"
echo "------------------------------------------------" >> "$LOG_FILE"

rsync -avz --delete /etc/bind root@backup2.m2l.org:/home/backup/dns/dns2 >>
"$LOG_FILE"
rsync -avz --delete /var/log/syslog
root@backup2.m2l.org:/home/backup/dns/dns2 >> "$LOG_FILE"

echo "-- FIN ------------------------------------------" >> "$LOG_FILE"
```

Ensuite il faut donner les droits et les permissions pour chaque fichiers Bash

```
# serveur WEB
chmod +x backupweb.sh

# serveur DNS1
chmod +x backupdns1.sh

# serveur DNS2
chmod +x backupdns2.sh
```

CRON

Maintenant, nous allons automatiser l'exécution des fichiers Bash tous les jours à 2 heures du matin grâce à l'outil cron.

Voici comment procéder pour planifier l'exécution des scripts Bash avec cron :

1 - Installation de cron

```
apt-get update
apt-get install cron
```

2 - Ouvrez la table cron en éditant le fichier crontab avec la commande suivante :

```
crontab -e
```

3 - Ajoutez une ligne pour chaque tâche que vous souhaitez planifier. Par exemple, pour exécuter un script à 2 heures du matin tous les jours, ajoutez :

```
# sur le serveur WEB
0 2 * * * /root/backupweb.sh

# sur le serveur DNS1
0 2 * * * /root/backupdns1.sh

# sur le serveur DNS2
0 2 * * * /root/backupdns2.sh
```

Enregistrez et quittez le fichier. Les modifications seront automatiquement prises en compte par cron.
Avec cette configuration, votre script sera exécuté automatiquement tous les jours à 2 heures du matin.