Solution de sauvegarde Rsync/Cron

Projet m2l.org

DUMAS Lucie

Table des matières

Que sont Rsync et Cron ?

Rsync

Rsync (signifiant « Remote Sync ») est une utilité de ligne de commande très utilisée dans les systèmes d'exploitation Unix et Linux. Elle est utilisée pour la synchronisation et la copie de fichiers et de répertoires entre deux emplacements, qu'ils soient sur la même machine ou sur des machines distantes via SSH.

Rsync est particulièrement apprécié pour les tâches de sauvegarde, de transfert de données, de mise à jour de répertoires et de synchronisation entre serveurs.

Cron

Cron est un outil utilisé dans les systèmes d'exploitation Unix et Linux pour la planification de tâches. Il permet aux utilisateurs et aux administrateurs de définir des tâches à exécuter automatiquement selon un horaire prédéfini. Ces tâches peuvent inclure des opérations de maintenance, des sauvegardes, des mises à jour logicielles, des scripts personnalisés ou toute autre action pouvant être automatisée.

Bien que Cron soit un outil puissant pour automatiser les tâches système, il nécessite cependant une compréhension appropriée de sa syntaxe et de son fonctionnement pour l'utiliser efficacement.

Création et configuration du conteneur Backup

Création du conteneur backup

Dans cette mission, nous mettrons en place un système de sauvegarde grâce à l'outil Rsync. Nous verrons ensuite comment automatiser la tâche de sauvegarde à l'aide de l'outil Cron.

Nous créons dans un premier temps le conteneur backup qui servira à héberger nos sauvegardes. Nous y sauvegarderons les fichiers de configuration des DNS, du serveur web, de la base de données et du serveur. Ce nouveau conteneur aura pour IP : 10.31.96.99. Avant de le démarrer, nous insérons la ligne suivante dans le fichier /var/lib/lxc/backup/config pour que ce dernier démarre automatiquement au lancement du serveur :

```
lxc.start.auto = 1
```

Configuration des clés pour SSH

Pour que notre conteneur de sauvegarde puisse se connecter à nos différentes machines, nous mettons en place l'authentification SSH par clés. L'authentification SSH par clés est plus sécurisée qu'une authentification SSH par mot de passe car elle s'appuie sur trois éléments :

- la clé privée
- la clé publique
- la passphrase

Dans un premier temps, nous allons générer une clé SSH publique, qui sera transmise à notre serveur et permettra au conteneur backup de s'authentifier auprès de ce dernier. Nous génèrerons également une clé privée dans le dossier /.ssh du compte root. Pour générer les clés, nous devons entrer la commande suivante avec le compte root :

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

Cette commande contient plusieurs paramètres :

- -t : cet argument permet de choisir le type de chiffrement (dans notre cas RSA)
- -b : cet argument permet de définir la taille de la clé (dans notre cas 4096 bits). Une taille de clé inférieure à 2048 n'est pas souhaitable étant donné qu'elle peut facilement être contournée.
- -f : cet argument le chemin du fichier ainsi que son nom. Par défaut, les noms de fichiers de la clé privée et de la clé publique seront respectivement « id_rsa » et « id_rsa.pub ». Les fichiers seront enregistrés dans le document «~/.ssh »

Dans un second temps, nous allons partager la clé publique de notre conteneur backup avec toutes les machines dont il doit avoir l'accès, c'est-à-dire notre serveur, nos DNS et notre serveur Web. Pour cela, nous devons copier cette clé dans le fichier /home/sio/.ssh/authorized_keys de toutes nos machines :

```
sio@srv-g6:    $ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDHbgW4SkGpTG7DRSz2NB5/78UGoNgR7/VX+61XUangdr6RnBHJE8KyqKZmSxNxE+MIMUFqvfD65ec3+qp9
uYbs1YqBCKCvwWuAwfDYeH1S+qazgsiMh4F7TsGnxal6y2V+fYKLc4Yxdp7CdDC3FMz/bmeB/D0I1FI++nhqFAcnHgRUZ1FJ1ica1kAcCdZqTzz1ioMShY61
+cMzKECQdTGJHe1YZsE+2qYqXLFxqcSvvf+v2ax+V9vhn+T/UmcngWI3u5NDAYOZA4a4RC1cxAGkAG6+rMGQMoti6vDIzrWITdQSjV1hE9ImPmxJ2FWzcAI5
nE4afb0+8Vbxt0aAk9cJ+nCBuvTrkdspWFNwET63BLhaGscmJQgOBP1DyjOVFCoNqiW1NHbcfYyHIIKS1OOXfwqmQfk6JIua29fv0TpHX3JxZU9PQ2xnakHC
76kariNjWff5N7Yw/hP48P1sgMW7CLu08J+PKEVmJacGhy3JAj15fciBuGXrKwEUWAagKRdeFHDNfH5RHfYP5i7j5+TunJ0eWWOMzM16azK0tN+L1Ev7Eyn9
x3JLtFF1cbuc8MiRG8+YMOQU9OSKK6pL4mf8jbBQcdkwmWD6iH9mwbFHVARvPzKziRGYjdi8fZIueeZpgK1PYagcQL8ovEIbz1/2tJ/MKi7jybq7/CECmoIX
zQ== pedago\ldumas@Eloi04
```

Nous allons ensuite activer la fonctionnalité de connexion par clé SSH. Pour cela, nous devons décommenter la ligne « PubkeyAuthentication yes » du fichier /etc/ssh/sshd_config de notre serveur :

```
PubkeyAuthentication yes
```

Nous pouvons maintenant nous déconnecter du serveur et retenter de nous connecter en SSH. Nous observons maintenant que nous ne devons plus entrer d'identifiant ni de mot de passe. Nous utiliserons dorénavant la passphrase pour nous connecter :

```
PS C:\Users\ldumas\.ssh> ssh sio@10.31.96.1
Enter passphrase for key 'C:\Users\ldumas/.ssh/id_rsa':
Linux srv-g6 5.10.0-20-amd64 #1 SMP Debian 5.10.158-2 (2022-12-13) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Jan 20 14:12:04 2023 from 10.187.20.53
sio@srv-g6:~$
```

Pour toutes les machines que nous souhaitons sauvegarder, nous modifions leur fichier /etc/ssh/ssh_config :

```
# Autoriser la connexion avec root
PermitRootLogin yes
# Interdire la connexion par mot de passe
PasswordAuthentication no
# Autoriser la connexion par clé publique
PubkeyAuthentication yes
```

Nous allons ajouter dans le fichier /etc/bind/db.m2l.org de notre DNS Master les lignes nécessaires afin que nous puissions attribuer un nom au serveur de sauvegarde (ex : faire une connexion SSH : ssh sio@backup.m2l.org.). Cette étape n'est pas indispensable mais elle permet de pouvoir nous connecter avec un nom de domaine à la place d'une adresse IP.

```
@ IN SOA ns1.m2l.org. root.m2l.org. (
        2020122601;
        43200;
        3600;
        3600000;
        172800 );

@ IN A 10.31.96.80;
@ IN NS ns1.m2l.org.;
@ IN NS ns2.m2l.org.;

ns1 IN A 10.31.96.80;
ns2 IN A 10.31.96.54;

backup IN A 10.31.96.99;
www IN A 10.31.96.80;

console IN CNAME www;
```

Mise en place de Rsync

Dans un premier temps, nous devons télécharger Rsync sur chaque machine dont nous voulons en sauvegarder le contenu :

```
apt update && apt upgrade
apt install rsync
```

Pour utiliser Rsync, nous devons une succession de commandes comme celle-ci :

```
# Serveur web
rsync -azv -e ssh root@10.31.96.80:/var/www/html /home/backup/web_conf/ >> $FILENAME
```

Voici la signification de chaque argument :

- a : signifie « archive mode ». C'est un raccourcis permettant d'activer un ensemble d'options qui sont généralement utiles lors de la copie de fichiers.
- v : signifie « verbose ». Cela permet d'afficher toutes les informations à l'écran dès l'exécution de la commande.
- z : signifie « compression ». Cela permet de faire une compression des fichiers.

Notre commande peut donc être décomposée comme suit :

- rsync : utilisation de la commande rsync
- azv : activation des arguments a, z et v
- e : spécification d'une commande externe pour établir une connexion SSH
- ssh root@10.31.96.80: : connexion SSH vers la machine à sauvegarder
- /var/www/html : choix du dossier à sauvegarder
- /home/backup/web_conf/ : choix du dossier accueillant les fichiers sauvegardés sur le serveur de sauvegarde
- >> $FILENAME : nom du fichier du journal d'événements qui accueillera les logs liées à la sauvegarde

Nous pouvons maintenant créer un script permettant de sauvegarder la configuration de notre serveur, notre serveur web, la configuration de nos DNS et notre base de données :

```
# Logs
FILENAME="/home/backup/logs/log_$(date '+%d-%m-%Y').log"

echo "----------------------------" >> $FILENAME
echo "Début de la sauvegarde : $(date '+%d-%m-%Y-%T')" >> $FILENAME
echo "----------------------------" >> $FILENAME

# Commandes

# Serveur
rsync -azv -e ssh root@10.31.96.1:/etc/rc.local /home/backup/server_conf/ >> $FILENAME
rsync -azv -e ssh root@10.31.96.1:/etc/system/system/rc-local.service
/home/backup/server_conf/ >> $FILENAME

# Serveur web
rsync -azv -e ssh root@10.31.96.80:/var/www/html /home/backup/web_conf/ >> $FILENAME

# DNS1
rsync -azv -e ssh root@10.31.96.53:/etc/bind/named.conf.local /home/backup/dns1_conf/ >>
$FILENAME
rsync -azv -e ssh root@10.31.96.53:/etc/bind/named.conf.options /home/backup/dns1_conf/
>> $FILENAME
rsync -azv -e ssh root@10.31.96.53:/etc/bind/db.m2l.org/home /backup/dns1_conf/ >>
$FILENAME

# DNS2
rsync -azv -e ssh root@10.31.96.54:/etc/bind/named.conf.local /home/backup/dns2_conf/ >>
$FILENAME
rsync -azv -e ssh root@10.31.96.54:/etc/bind/named.conf.local /home/backup/dns2_conf/ >>
$FILENAME

# Base de données
mysqldump -h 10.31.96.80 -u backup -pdrowssap –all-databases >>
/home/backup/db/backup_db_$(date "+%d-%m-%Y").sql

echo "----------------------------" >> $FILENAME
echo "FIN" >> $FILENAME
echo "----------------------------" >> $FILENAME
```

Pour pouvoir faire fonctionner la migration de la base de données, nous devons installer dans notre conteneur backup le client de MariaDB :

```
apt update && apt upgrade
apt install mariadb-client
```

Nous devons ensuite revenir sur notre conteneur web pour créer un nouvel utilisateur qui sera utilisé par le conteneur backup, et qui aura la possibilité d'accéder à l'extérieur :

```
# Connexion à la base de données en tant que root
mysql -u root -p
```

```
-- Création de l'utilisateur
create user'backup'@'%' identified by 'drowssap';

-- Attribution des privilèges sur toutes les bases de données
grant all privileges on *.* to 'backup'@'%' with grant option;

-- Mise à jour des privilèges de la base de données
flush privileges ;
```

Ensuite, nous devons changer le fichier de configuration de MariaDB (/etc/mysql/mariadb.conf.d/50-server.cnf) afin que ce service puisse écouter sur le réseau. Pour ce faire, nous devons changer la ligne bind = 127.0.0.1 en bind = 0.0.0.0 comme ceci :

```
#
# * Basic Settings
#

user                    = mysql
pid-file                = /run/mysqld/mysqld.pid
basedir                 = /usr
datadir                 = /var/lib/mysql
tmpdir                  = /tmp
lc-messages-dir         = /usr/share/mysql
lc-messages             = en_US
skip-external-locking

# Broken reverse DNS slows down connections considerably and name resolve is
# safe to skip if there are no "host by domain name" access grants
#skip-name-resolve

# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address            = 0.0.0.0
```

Mise en place de Cron

Dans un premier temps, nous devons installer l'outil Cron sur notre serveur de sauvegarde afin que ce dernier puisse lancer le script précédemment réalisé avec l'outil Rsync :

```
apt update && apt upgrade
apt install cron
```

Pour planifier les différentes tâches, l'outil Cron s'appuie sur ses fichiers de configuration appelés « crontab ». Chaque utilisateur peut avoir son propre fichier crontab, où il spécifie des tâches qu'il souhaite exécuter et la fréquence à laquelle elles doivent être exécutées. Les tâches peuvent être planifiées à des minutes, des heures, des jours, des mois ou des jours de la semaine spécifiques, offrant ainsi une grande flexibilité dans la planification des travaux.

Voici un exemple de ligne de commande crontab :

```
* * * * *    user    /dossier_mon_script/mon_script.sh
```

Notre commande peut être décomposée comme suit :

- Le premier astérisque correspond aux minutes. Il signifie que le script sera exécuté toutes les minutes.
- Le deuxième astérisque correspond aux heures. Il signifie que le script sera exécuté toutes les heures.
- Le troisième astérisque correspond aux jours. Il signifie que le script sera exécuté tous les jours.
- Le quatrième astérisque correspond aux mois. Il signifie que le script sera exécuté tous les mois.
- Le cinquième astérisque correspond aux jours de la semaine (lundi, mardi…). Il signifie que le script sera exécuté tous les jours de la semaine.
- User signifie que le compte devant exécuter le script est le compte user.
- Le chemin affiché correspond au chemin absolu du fichier de script à exécuter.

Nous devons faire la sauvegarde de nos machines tous les jours à 23h. Le script devra également être exécuté par l'utilisateur root. Nous entrons donc la ligne suivante dans la crontab :

```
0  23  *  *  *    root  /home/backup/save.sh
```