DATA Sec

Planification d'une sauvegarde automatique sur un serveur distant via SSH

Travail à réaliser en binôme

Introduction :

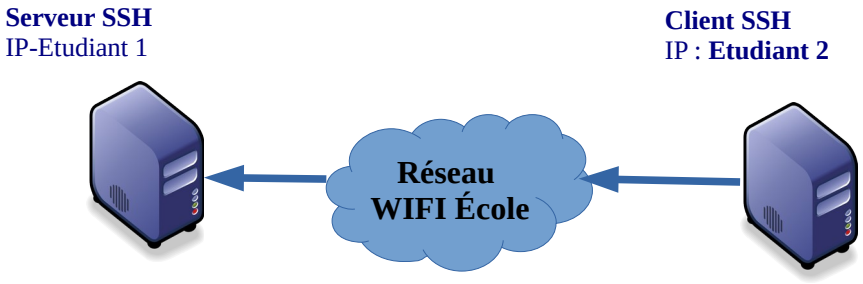
SSH (Secure Shell) est un protocole sécurisé qui permet la connexion, l'administration et le transfert de fichiers chiffrés entre deux machines.
Crontab est un outil de planification des tâches sous GNU/Linux, permettant d'exécuter automatiquement des scripts SHELL ou des commandes à intervalle régulier.

Objectifs :

Mettre en place une sauvegarde sécurisée via SSH

Automatiser la sauvegarde sur le serveur ssh avec un script shell exécuté par un client ssh et crontab

Schéma fonctionnel



Configuration du serveur SSH (Étudiant 1 : serveur)

Commandes à exécuter :
sudo su
apt-get install openssh-server
ssh-keygen -t rsa -b 1024

(Attention !! N'entrez pas de passphrase lorsque l'utilitaire vous le demande).
cat /root/.ssh/id_rsa.pub >> /root/.ssh/authorized_keys

Modifier la configuration SSH :
Éditer /etc/ssh/sshd_config

nano /etc/ssh/sshd_config
Modifier la ligne :
PermitRootLogin yes

Redémarrer SSH :
/etc/init.d/ssh restart

Explications :

ssh-keygen : génère une paire de clés de sécurité publique/privée de 1024 bits via l'algorithme RSA

Clé privée : élément cryptographique qui doit rester strictement confidentiel. Elle permet de prouver l'identité de la machine ou de l'utilisateur. Elle ne doit jamais être partagée.

Clé publique : élément associé à la clé privée, conçu pour être distribué. Elle peut être placée sur les serveurs afin de permettre l'authentification.

Les deux clés sont contenues à présent dans le répertoire /root/.ssh/

Le serveur SSH contient un fichier qui liste toutes les clés publiquesid_rsa.pub des utilisateurs possédant la

clé privée qui ont le droit de se connecter sans mot de passe. Ce fichier se nomme

authorized_keys et doit être dans /root/.ssh/authorized_keys.

Important : La clé privée Id_dsa doit être copiée sur le client SSH **(Je vous laisse le soin de faire !)**

Fonctionnement dans SSH
Lors de l'exécution du script , le serveur SSH utilise la clé publique enregistrée dans *root/.ssh/* pour vérifier que la machine qui se connecte possède bien la clé privée id_rsa correspondante. Si la vérification réussit, l'accès est accordé.

PermitRootLogin yes : autorise l'accès SSH pour root

Configuration du client SSH (Élève 2 : client) :

Commandes :
sudo su

apt-get install openssh-server

mkdir /root/sauvegarde/
cp /bin /root/sauvegarde/
cd /root/sauvegarde/
tar cvzf sauvegarde.tar.gz sauvegarde
Explications :
mkdir : création du dossier de sauvegarde
cp : copie des fichiers à sauvegarder
tar cvzf : création d'une archive compressée

Remarque : la sauvegarde du dossier */bin* sert uniquement à simuler le processus. Dans la réalité, on sauvegarde uniquement les données importantes ; */bin* n'est utilisé ici qu'à titre d'exemple.

On va créer un petit script Shell qui exécute la sauvegarde :

nano /etc/init.d/secbckp.sh

#!/bin/bash
scp -i */chemin*/id_rsa /root/sauvegarde root@ip_serveur:/home/$USER/Bureau

Rendre le script exécutable :
chmod 700 /etc/init.d/secbckp.sh

Explications :
Le script permet l'envoi automatique de la sauvegarde

scp -i : transfert sécurisé vers le serveur SSH en utilisant la clé privée *id_rsa*.

La clé *id_rsa*, générée initialement par le serveur SSH, doit au préalable être transférée sur le client SSH.
chmod 700 limite l'exécution au propriétaire (root)

Planification avec crontab

Éditer la crontab :
crontab -e

Ajouter la ligne suivante :

```
# m     h     dom    mon     dow       command
45     12      *      *       *       /etc/init.d/secbckp.sh
```

Explications :
Exécution quotidienne à 12h45
Pour tester, mettez l'heure actuelle + 1 minute pour vérifier rapidement l'éxecusion du script

Manuel crontab :

crontab utilise la syntaxe suivante : **mm hh jj MMM JJJ tâche**
Dans cette syntaxe:
mm représente les minutes (de 0 à 59)
hh représente l'heure (de 0 à 23)
jj représente le numéro du jour du mois (de 1 à 31)
MMM représente le numéro du mois (de 1 à 12) ou l'abréviation du nom du mois (jan, feb, mar, apr, ...)
JJJ représente l'abréviation du nom du jour ou le chiffre correspondant au jour de la semaine (0 représente le dimanche, 1 représente le lundi, ...)
tâche représente la commande ou le script shell à exécuter

Testes et Vérification (à exécuter du coté client et du coté serveur ssh)

Commande :
`tail -f /var/log/syslog`

Explications :
Permet de suivre en temps réel que le script est bien exécuté par cron grace au fichier journal /var/log/syslog .