Secure Shell (SSH)

PRESENTATION

Secure Shell (SSH) est un protocole qui permet de sécuriser les communications de données entre les ordinateurs connectés au réseau.

Il permet d'assurer la confidentialité, l'intégrité, l'authentification et l'autorisation des données dans des tunnels chiffrés. Il utilise TCP habituellement sur le port 22, mais il peut en utiliser d'autres simultanément.

On utilise aujourd'hui la version SSH-2. La version SSH-1 est à proscrire.

PRESENTATION

Le protocole SSH comporte autant de fonctionnalités qui rendent obsolètes des protocoles comme Telnet ou FTP et autorise la gestion distante de ressources Linux à travers l'Internet :

- On peut l'utiliser comme console distante à la manière de Telnet, RSH ou Rlogin.
- SSH supporte les authentifications centralisées (PAM), les authentifications locales avec mot de passe ou sans échange de mot de passe (par le biais d'échange de clés).
- On peut transférer des sessions X graphiques dans un tunnel SSH.

PRESENTATION

- On peut y transférer des ports et utiliser le service comme proxy ou comme solution VPN, de manière distante ou locale.
- Les sous-protocoles SCP et SFTP offrent des services de transfert de fichiers.
- Il s’intègre à des logiciels comme ansible, systemd, x2go, …
- Et encore d’autres fonctionnalités fines en matière de sécurité ou de facilité.

PRESENTATION

En terme de cible d’attaque, le port est très sollicité par les robots qui scannent les réseaux publics en quête de configurations faibles, nulles, négligées ou exploitables. Il peut arriver qu’un port SSH exposé publiquement soit l’objet de tentatives de Déni de Service (DoS) ou de connexions Brute Force qui rendent le service inacessible.

Il est conseillé d’auditer, voire de filter les accès au service SSH avec un logiciel comme fail2ban, des sondes IPS/IDS snort ou autre. Un pot de miel tel que Cowrie peut être une arme à manier avec précaution.

INSTALLATION, CONFIGURATION ET CONNEXION OPENSSH

1- Statut du service

Avec les droits de **root**, vérifier le statut du service httpd :

sudo systemctl status sshd

2- Installation et activation du serveur openssh

Si nécessaire, on installera et activera le service sshd :

sudo apt install openssh-server

sudo systemctl enable sshd

sudo systemctl start sshd

INSTALLATION, CONFIGURATION ET CONNEXION OPENSSH

3- Fichier de configuration du serveur openssh

On peut vérifier le fichier de configuration du serveur :

sudo less /etc/ssh/sshd_config

4- Connexion locale au serveur SSH

On peut tenter une connexion locale sur le serveur :

ssh user@127.0.0.1

INSTALLATION, CONFIGURATION ET CONNEXION OPENSSH

5- Vérification de la version du client openssh

Vérifier la version du client openssh :

ssh –V OpenSSH_6.6.1p1, OpenSSL 1.0.1e-fips 11 Feb 2013

On peut afficher la bannière du service :

nc localhost 22

SSH-2.0-OpenSSH_6.6.1

Authentification par clé avec OpenSSH

L'authentification par clé fonctionne grâce à 3 composants :

- **Une clé publique** : elle sera exportée sur chaque hôte sur lequel on souhaite pouvoir se connecter.
- **Une clé privée** : elle permet de prouver son identité aux serveurs.
- **Une passphrase** : optionnelle, elle permet de sécuriser la clé privée (notons la subtilité, passphrase et pas password… donc "phrase de passe" et non pas "mot de passe").

La sécurité est vraiment accrue car la "passphrase" seule ne sert à rien sans la clé privée, et vice-versa. Cet usage peut se révéler contraignant.

Authentification par clé avec OpenSSH

1- Création de la paire de clés

La création de la paire de clés se fait avec le binaire openssh **ssh-keygen.**

Il existe 2 types de clés : **RSA** et **DSA**. Chacune pouvant être de longueur différente (les clés inférieures à 1024 bits sont à proscrire).

Pour créer une clé dans sa session avec des paramètres par défaut :

ssh-keygen

Authentification par clé avec OpenSSH

2- Clé privée / clé publique

Deux fichiers ont été créés (dans le dossier ~/.ssh/) :

- ✓ **id_rsa** (ou id_dsa dans le cas d'une clé DSA) : contient la clé privée et ne doit pas être dévoilé ou mis à disposition.
- ✓ **id_rsa.pub** (ou id_dsa.pub dans le cas d'une clé DSA) : contient la clé publique, c'est elle qui sera mise sur les serveurs dont l'accès est voulu.

Authentification par clé avec OpenSSH

2- Clé privée / clé publique

Deux fichiers ont été créés (dans le dossier ~/.ssh/) :

- ✓ **id_rsa** (ou id_dsa dans le cas d'une clé DSA) : contient la clé privée et ne doit pas être dévoilé ou mis à disposition.
- ✓ **id_rsa.pub** (ou id_dsa.pub dans le cas d'une clé DSA) : contient la clé publique, c'est elle qui sera mise sur les serveurs dont l'accès est voulu.

Authentification par clé avec OpenSSH

3- Transmission de la clé publique au serveur

Pour transmettre la clé publique au serveur, il sera nécessaire de disposer d'avance d'un mot de passe pour le compte à authentifier, à moins que d'agir comme utilisateur privilégié. Le principe est d'aller écrire la ou les clés publiques autorisées dans le fichier **~/.ssh/authorized_keys** de l'utilisateur cible.

Authentification par clé avec OpenSSH

Trois méthodes de transmission de la clé à partir d'une station distante sont proposées ici.

❖**Via une console SSH**

cat ~/.ssh/id_rsa.pub | ssh user@ip_machine "cat - >> ~/.ssh/authorized_keys"

Authentification par clé avec OpenSSH

Trois méthodes de transmission de la clé à partir d'une station distante sont proposées ici.

❖**Via le protocole de transfert SCP :**

scp ~/.ssh/id_rsa.pub user@ip_machine:/tmp

ssh user@ip_machine

cat /tmp/id_rsa.pub >> ~/.ssh/authorized_keys

rm /tmp/id_rsa.pub

Transfert de fichiers SCP et SFTP

1- Transfert de fichiers SCP

SCP est la transposition de la commande cp à travers SSH, avec des arguments, une source et une destination. On désigne la ressource distante origine ou destination par user@machine:/path. Voici quelques exemple.

Transfert de fichiers SCP et SFTP

1- Transfert de fichiers SCP

***Copie vers un serveur distant**

scp /dossier/fichier user@machine:~

***Copie venant d'un serveur distant**

scp user@machine:~/dossier/fichier .

***Copie récursive**

scp -R /dossier user@machine:~

Transfert de fichiers SCP et SFTP

2- Transfert de fichiers SFTP

SFTP s'utilise comme un client FTP.

 # sftp user@localhost

user@localhost's password:

Connected to localhost.

sftp> pwd

Remote working directory: /home/user

sftp> quit

Transfert de session graphique avec SSH

Serveur X

ssh -X user@ip_machine

ssh -MY user@ip_machine