Configurer une authentification SSH avec certificat

Resources

- http://www.ixany.org/fr/articles/presentation-et-utilisation-des-certificats-ssh/
- https://www.ssh.com/
- http://tech.ciges.net/blog/openssh-with-x509-certificates-how-to/
- https://linux-attitude.fr/post/certificats-x509-pour-ssh

Principes

L'utilisation de certificats permet de garantir l'identité du client et du serveur.

Le certificat présent par le client ou le serveur est signé par la clé privée d'une authorité de certification reconnue.

- **Intérêt pour le client** : la **clé publique de la CA** permet de vérifier le certificat présenté par le serveur. Il n'est pas nécessaire de renseigner le fichier **know_hosts**.
- **Intérêt pour le serveur** : la **clé publique du CA** permet de vérifier le certificat présenté par le client sans avoir à renseigner le fichier **authorized_keys**.

Les éléments nécessaires

- le certificat de la CA
- le certificat du client
- l'identité du client

Format d'un clé publique ssh

Une clé publique SSH est une chaîne de texte en une seule ligne, avec plusieurs parties distinctes : <type> <clé encodée en base64> <commentaire facultatif>

Type : Spécifie l'algorithme utilisé pour générer la clé. Par exemple :

- ssh-rsa : clé utilisant l'algorithme RSA.
- ecdsa-sha2-nistp256 : clé ECDSA avec une courbe elliptique.
- ssh-ed25519 : clé basée sur l'algorithme Ed25519.

Clé encodée en base64 : chaîne encodée en Base64 représentant la clé publique.

Commentaire facultatif : information pour identifier la clé : nom d'utilisateur et l'hôte depuis lequel la clé a été générée.

Mise en place côté serveur

- Le serveur doit disposer de la clé publique du certificat du CA
- la clé publique est indiquée dans le fichier de configuration du serveur SSH **/etc/ssh/sshd_config** avec la directive **TrustedUserCAKeys** qui précise le nom du fichier contenant la liste des clef publique des CA

```
TrustedUserCAKeys /etc/ssh/ca.pub
```

> **IMPORTANT** : le certificat du client ne sera considéré que si sa liste de nom-clés (principal) contient le nom du compte auquel il tente de se connecter.
>
> SAUF si un fichier spécifique à chaque compte indique la liste des noms-clés acceptés. Directive **AuthorizedPrincipalsFile** dans sshd_config

Mise en place côté client

Le client doit faire signer sa clé publique pour obtenir un certificat qui doit être placé dans le même répertoire que sa clé privé et publique.

```
cp client_key ~/.ssh/
cp client_cert.pub ~/.ssh/
```

Les permissions doivent être correctes :

```
chmod 600 ~/.ssh/client_key
chmod 644 ~/.ssh/client_cert.pub
```

Lors d'une connexion avec la clé privée, le certificat sera automatiquement présenté au serveur.

Le client ssh doit seulement indique quelle clé privée utiliser avec l'option -i

```
ssh -i ~/.ssh/client_key nomDNSserveur
```

Ainsi même s'il n'y a pas de clé publique client dans authorized_keys, l'authentification se fait. L'option -v permet de voir l'utilisation du certificat

```
ssh -v -i ~/.ssh/client_key nomDNSserveur
```

Vous pouvez utiliser aussi le fichier de configuraito du client ssh

```
Host <nom_du_serveur>
    HostName <adresse_du_serveur>
    User <nom_utilisateur>
    IdentityFile ~/.ssh/client_key
    CertificateFile ~/.ssh/client_cert.pub
```

Les commandes utiles

- obtenir la clé publique à partir de la clé privée d'une identité utilisateur au format openSSH

```
$ ssh-keygen -y -f utilisateur-identite.pem > id_rsa.pub
```

- Obtenir la clé publique à partir du certificat de l'utilisateur au format openSSH pour une connexion ssh
    - Extraire la clé publique du certificat de l'utilisteur

```
$ openssl x509 -in charles-cert.pem -pubkey –noout > id_rsa.pem
```

noout permet d'avoir uniquement la clé publique sans le certificat

- Convertir la clé publique du format PEM au format openssh

```
$ ssh-keygen -i -m PKCS8 -f id_rsa.pem > id_rsa.pub
```

- Obtenir la clé publique de la CA du certificat de la CA au format openSSH pour une connexion ssh
    - Extraire la clé publique du certificat du CA

```
$ openssl x509 -in pkicub-cert.pem -pubkey –noout > ca.pem
```

```
* Convertir la clé publique du format PEM au format openssh
```

```
$ ssh-keygen -f ca.pem > ca.pub
```

- Vérifiez les journaux SSH pour voir si le certificat est utilisé correctement :

```
journalctl -u ssh
```

- Sur le clien, ajoutez l'option -v pour obtenir plus de détails sur le processus de connexion :

```
ssh -v <nom_du_serveur>
```

- Facilité d'utilisation avec plusieurs serveurs en faisant confiance à la même CA en ajoutant la clé publique de la CA à ~/.ssh/known_hosts en tant qu'autorité de certification :

```
echo "@cert-authority *.example.com ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQE..." >> ~/.ssh/known_hosts
```

Cela permet au client de faire confiance à tous les serveurs avec des certificats signés par cette CA.

- Visualiser le contenu d'un certificat

```
openssl x509 -in filename.pem -text –noout
```

-noout pour ne pas afficher en base64 qui permet d'encode des données binaires ASCII

- emetteur du certificat : -issuer

```
openssl x509 -in filename.pem -issuer –noout
```

- Sujet du certificat : -subject

```
openssl x509 -in filename.pem -subject –noout
```

- Empreinte digitale : -fingeprint

```
openssl x509 -in filename.pem -fingerprint –noout
```

- Clé publique : -pubkey

```
openssl x509 -in filename.pem –pubkey -noout –out id_rsa.pub
```

From:
/ - **Les cours du BTS SIO**

Permanent link:
/doku.php/reseau/debian/clesshcertificat?rev=1736439977

Last update: **2025/01/09 17:26**