Utiliser des certificats OpenSSH

Présentation

La connexion à distance à un serveur en SSH peut se faire avec des **certificats** dont le **format est spécifique à OpenSSH**.

Rappel

- l'**authentification** sur un serveur distant en SSH consiste à renseigner votre **clé publique** dans le dossier **~/.ssh/authorized_keys** du compte existant sur le serveur.
- Votre **clé privée** reste sur votre ordinateur client et ne doit pas être communiquée. Cette clé privée peut en plus être protégée par une **passphrase**.

Configuration du service SSH du serveur

Avec Debian, la configuration d'un serveur SSH se fait dans le fichier **/etc/ssh/sshd_config** :

- La directive **PasswordAuthentication** doit être fixée à **no** pour interdire l'authentification par mot de passe. **ATTENTION** : Cela n'est à faire que si vous avez correctement configuré une authentification par clé SSH ou certificat.
- dans le dossier **/etc/ssh** se trouve plusieurs types de clés privées, avec les clés publiques associées, générées automatiquement, que peut utiliser le serveur SSH : **ssh_host_ecdsa_key** (ssh_host_ecdsa_key.pub) , **ssh_host_rsa_key** (ssh_host_rsa_key.pub) et **ssh_host_ed25519_key** (ssh_host_ed25519_key.pub).

Pour **vérifier** le paramétrage du serveur SSH, lancez la commande suivante en mode debug :

```
$ sudo /usr/sbin/sshd -d
```

La commande suivante permet d'afficher ces différents clés publiques d'un serveur SSH (RSA, ECDSA et ED25519):

```
$ ssk-keyscan adresseIPserveurssh
```

Génération du couple de clés privée / publique par le client

Utilisation de l'utilitaire **ssh-keygen** avec les options possibles suivantes :

- option **-t** : préciser le type de clés : dsa, rsa, ecdsa ou ed25519 (valeur par défaut) ;
  - option **-b** : préciser la longueur de la clé (2048 par défaut).
  - option **-C** : préciser un commentaire pour identifier la clé publique : courriel, nom, etc..

```
$ ssh-keygen -C "Charles Técher"
```

Les clés privée et publique sont enregistrées dans le dossier **~/.ssh** : **id_ed25519** et **id_ed25519.pub**.

- sécuriser la clé privée

```
$ chmod 600 ~/.ssh/id_ed25519
```

La passphrase de la clé privée peut être modifiée avec la commande suivante :

```
$ ssh-keygen -p -f .ssh/id_ed25519
```

La clé publique peut être affichée (et générée) à partir de la clé privée :

```
$ ssh-keygen -y -f .ssh/id_ed25519
```

Paramètrage du compte du serveur avec votre clé publique

Sur le serveur distant, vous souhaitez permettre une authentification avec votre **clé publique ssh**.

Pour cela, il suffit de **copier** le contenu de votre clé publique dans le fichier **~/.ssh/authorized_keys** du compte linux du serveur, compte avec lequel vous souhaitez **ouvrir une session**.

- Exemple de copie automatique de la clé publique pour un compte appelé **sio** créé sur le serveur distant :

```
ssh-copy-id sio@ipserveur
```

- Exemple de copie manuelle de la clé publique pour un compte appelé **sio** créé sur le serveur distant :

```
cat ~/.ssh/id_ed25519.pub | ssh sio@ipserveur "cat >> ~/.ssh/authorized_keys"
```

Gestion des serveurs connus

Votre client SSH va **vérifier l'identité du serveur distant** avant de se connecter. Le fichier **/etc/ssh/ssh_config** permet de configurer le client SSH pour tous les utilisateurs de votre ordinateur, et le fichier **~/.ssh/config** uniquement pour l'utilisateur qui a ouvert une session.

Lors de la **première connexion** du client sur le serveur distant, vous recevez un **message d'avertissement** et vous devez confirmer, à vos risques et périls, que vous acceptez de vous connecter à ce serveur. Dans ce cas, une trace du serveur (**empreinte SSHFP** - SSH FingerPrint), est enregistrée dans le fichier **~/.ssh/known_hosts**. Lors d'une prochaine connexion, si la trace d'une connexion précédente est trouvée, vous n'aurez plus le message d'avertissement.

La commande suivante permet de **prendre connaissance de l'empreinte** de la **clé publique** du serveur distant, et de la communiquer aux utilisateurs client pour **vérification** :

```
# sur le serveur distant
$ ssh-keygen -l -f /etc/ssh/ssh_host_ed25519_key.pub

# depuis votre client
$ ssh sio@adresseIP "ssh-keygen -l -f /etc/ssh/ssh_host_ed25519_key.pub"
```

Inconvénients de l'usage des clés publiques

Les clés publiques d'un serveur peuvent être changées et ne plus être celles initialement crées. C'est le cas lors d'une attaque de type **Man-In-The-Middle**.

De même, il n'est pas possible de garantir que la clé publique d'un client soit **légitime** car il est facile de générer une clé publique, avec un commentaire de son choix et de la communiquer.

L'utilisation d'un certificat ajoute des informations d'identité et ce **certificat est signé** par une autorité de certification reconnue.

La signature du certificat est un haché des données du certificat chiffrée par la **clé privée de l'autorité de certification reconnue**.

En utilisant la **clé publique** de l'autorité de certification reconnue, il est possible de **déchiffrer** la signature et de comparer ensuite les hachés.

Génération des certificats OpenSSH

OpenSSH permet de générer des **certificats spécifiques à OpenSSH**. Ce ne sont pas des certificats x509 utilisés avec SSL/TLS (https, sftp, etc.)

La création d'un certificat utilisateur ou serveur nécessite la signature par une **clé privée** qui représente alors l'**autorité de certification**.

> L'autorité de certification (CA) doit être hébergée sur un serveur très sécurisé, et, de manière idéale, offline quand il n'est pas utilisé.

Pour cette démarche, la CA sera hébergée sur le serveur distant.

Création de la clé privée sur le serveur distant avec le nom ca_key

```
$ ssh-keygen -f ca_key -C "SSH cle de la CA"
```

Cela génère 2 fichiers :

- **ca_key** : clé privée de la CA
- **ca_key.pub** : clé publique de la CA

Création d'un certificat pour le client

> La configuration d'un serveur distant pour **utiliser des certificats OpenSSH**, signés par **une ou plusieurs clés privées (CA)** connues de l'administrateur (configuration du serveur distant), **ne nécessite plus** la mise à jour sur le serveur distant d'un fichier **.ssh/authorized_keys** contenant les clés publiques des utilisateurs pour chaque compte du serveur distant pouvant être utilisé.

- sur le serveur distant, supprimez le fichier **.ssh/authorized_keys** du compte pour lequel vous souhaitez configurer l'**authentification par certificat OpenSSH**.

```
$ rm ~/.ssh/authorized_keys
```

Pour la création du certificat du client, il est nécessaire de préciser :

- **-s** : la clé privée de la CA qui signe le certificat
- **-I** : l'identifiant du certificat client. Pour l'exemple un utilisateur appelé CLIENT_PRENOM.
- **-n** : le login autorisé pour ce client sur le serveur. Il s'agit dans l'exemple du compte linux sio.
- **-V** : la validité du certificat. En général une année.
- la **clé publique** de l'utilisateur qui est nécessaire pour créer son certificat

- Transférez sur le serveur de CA la clé publique de l'utilisateur :

```
$ scp ~/.ssh/id_ed25519.pub sio@adresseIP:/home/sio
```

- Sur le serveur de CA, utilisez la commande suivante :

```
$ ssh-keygen -s ca_key -I CLIENT-PRENOM -n sio -V +365d id_ed52519.pub
```

Le certificat a été généré et le fichier obtenu porte le nom de la clé publique en ajoutant “-cert” : **id_ed25519-cert.pub**.

La commande suivant permet de visualiser le contenu du certificat :

```
$ ssh-keygen -L -f id_ed25519-cert.pub
id_ed25519-cert.pub:
        Type: ssh-ed25519-cert-v01@openssh.com user certificate
        Public key: ED25519-CERT SHA256:q3u+woluMmljCo+HCKHr55oztAiZ7QVDXWkw20W02UY
        Signing CA: ED25519 SHA256:5R5DL2MZrQelfe88lyf2zdv3UepovFcdm9NiHtplWUQ (using ssh-ed25519)
        Key ID: "CLIENT-PRENOM"
        Serial: 0
        Valid: from 2026-06-28T20:27:00 to 2027-06-28T20:28:40
        Principals:
                sio
        Critical Options: (none)
        Extensions:
                permit-X11-forwarding
                permit-agent-forwarding
                permit-port-forwarding
                permit-pty
                permit-user-rc
```

Remarques :

- il s'agit d'un certificat utilisateur : **Type ... user certificate**.
- ce n'est pas un certificat autosigné car l'**empreinte SHA256** de la clé publique de l'utilisateur (**Public Key**) est différente de l'empreinte de la clé qui a signé le certificat (**Signing CA**).
- le **principal** est sio, c'est à dire le **login** associé à ce certificat. Plusieurs principals peuvent être déclarés (séparé par des virgules).

La valeur du champ **Principals** est importante car cela détermine l'**identité présentée** par le client, c'est à dire le ou les comptes qu'il souhaite utiliser pour ouvrir une session.

- **Transférez** sur le client le **certificat client** nouvellement généré et placez le dans le dossier **~/.ssh/** de l'ordinateur client avec la clé privée **id_ed25519** et la clé publique **ed_ed25519.pub** :

```
$ scp sio@adresseIPCA:/home/sio/id_ed25519-cert.pub ~/.ssh/
```

Pour valider ces certificats, il faut indiquer au serveur distant, la liste des différentes clés publiques des CA qu'il accepte comme **signataires des certificats clients** présentés.

- sur le serveur distant, **copier** le contenu de la clé privée de la CA (ca_key.pub), dans le fichier **/etc/ssh/ssh_ca_keys** :

```
$ sudo cat ~/ca_key.pub >> /etc/ssh/ssh_ca_keys
```

- modifiez le fichier de configuration du service SSH **/etc/ssh/sshd_config** pour renseigner la directive **TrustedUserCAKeys** avec le nom de ce fichier :

```
...
# HostKey /etc/ssh/ssh_host_ed25519_key
TrustedUserCAKeys /etc/ssh/ssh_ca_keys
...
```

- redémarrez le service SSH

```
$ sudo systemctl restart ssh
```

- testez en mode debug la connexion en SSH au serveur distant avec votre certificat utilisateur. Seule la passsphrase de votre clé privée doit être demandée, si elle a été définie

```
$ ssh -v sio@adresseIP
...
debug1: Offering public key: /home/sio/.ssh/id_ed25519 ED25519
SHA256:q3u+woluMmljCo+HCKHr55oztAiZ7QVDXWkw20W02UY
debug1: Authentications that can continue: publickey,password
debug1: Offering public key: /home/sio/.ssh/id_ed25519 ED25519-CERT
SHA256:q3u+woluMmljCo+HCKHr55oztAiZ7QVDXWkw20W02UY
debug1: Server accepts key: /home/sio/.ssh/id_ed25519 ED25519-CERT
SHA256:q3u+woluMmljCo+HCKHr55oztAiZ7QVDXWkw20W02UY
Enter passphrase for key '/home/sio/.ssh/id_ed25519':
```

Utilisation du certificat pour s'authentifier avec un autre compte sur le serveur distant

Le certificat client **id-ed25519-cert.pub**, signé avec la clé privée de la CA **ca_key** vous a permis de vous authentifier sur le serveur distant, sans mot de passe. Si vous avez défini une passphrase pour votre clé privée, vous avez uniquement été amené à l'indiquer.

Démarche pour se connecter sur le serveur distant avec un autre compte

- Sur le serveur distant, créez un nouveau compte linux appelé par exemple btssio:

```
$ sudo adduser btssio
```

- **test** en mode **debug** de la connexion en SSH au serveur distant avec votre **certificat utilisateur** et le compte **btssio** :

```
$ ssh -v btssio@adresseIP
...
debug1: Offering public key: /home/sio/.ssh/id_ed25519 ED25519
SHA256:q3u+woluMmljCo+HCKHr55oztAiZ7QVDXWkw20W02UY
debug1: Authentications that can continue: publickey,password
debug1: Offering public key: /home/sio/.ssh/id_ed25519 ED25519-CERT
SHA256:q3u+woluMmljCo+HCKHr55oztAiZ7QVDXWkw20W02UY
debug1: Authentications that can continue: publickey,password
debug1: Trying private key: /home/sio/.ssh/id_ed25519_sk
debug1: Trying private key: /home/sio/.ssh/id_xmss
debug1: Trying private key: /home/sio/.ssh/id_dsa
debug1: Next authentication method: password
```

```
btssio@10.187.35.100's password:
```

L'authentification par certificat n'a pas aboutie et vous devez vous authentifier avec le mot de passe du compte.

Rappel : lors de la création de votre certificat, seul le principal sio a été indiqué.

Il est nécessaire de **recréer votre certificat client** en définissant comme **principal**, les **deux comptes**, sio (ou celui à votre nom) et le nouveau (btssio).

```
* Création à nouveau du certificat client sur le serveur jouant le rôle de CA en définissant les deux
comptes comme principal :
```

- Sur le serveur de CA, utilisez la commande suivante :

```
$ ssh-keygen -s ca_key -I CLIENT-PRENOM -n sio,btssio -V +365d id_ed52519.pub
```

- Visualiser le contenu du certificat :

```
$ ssh-keygen -L -f id_ed25519-cert.pub
id_ed25519-cert.pub:
        Type: ssh-ed25519-cert-v01@openssh.com user certificate
        Public key: ED25519-CERT SHA256:q3u+woluMmljCo+HCKHr55oztAiZ7QVDXWkw20W02UY
        Signing CA: ED25519 SHA256:5R5DL2MZrQelfe88lyf2zdv3UepovFcdm9NiHtplWUQ (using ssh-ed25519)
        Key ID: "CLIENT-PRENOM"
        Serial: 0
        Valid: from 2026-06-28T20:27:00 to 2027-06-28T20:28:40
        Principals:
                sio
                btssio
        Critical Options: (none)
        Extensions:
                permit-X11-forwarding
                permit-agent-forwarding
                permit-port-forwarding
                permit-pty
                permit-user-rc
```

Remarques : les deux comptes sio et btssio sont définis comme principal

- depuis votre client, récupérer le **certificat client** nouvellement généré et placez le dans le dossier **~/.ssh/** de l'ordinateur client avec la clé privée **id_ed25519** et la clé publique **ed_ed25519.pub** :

```
$ scp sio@adresseIPCA:/home/sio/id_ed25519-cert.pub ~/.ssh/
```

- **test** en mode **debug** de la connexion en SSH au serveur distant avec votre **nouveau** certificat utilisateur et le nouveau compte **btssio** :

```
$ ssh -v btssio@adresseIP
...
debug1: Offering public key: /home/sio/.ssh/id_ed25519 ED25519
SHA256:q3u+woluMmljCo+HCKHr55oztAiZ7QVDXWkw20W02UY
debug1: Authentications that can continue: publickey,password
debug1: Offering public key: /home/sio/.ssh/id_ed25519 ED25519-CERT
SHA256:q3u+woluMmljCo+HCKHr55oztAiZ7QVDXWkw20W02UY
debug1: Server accepts key: /home/sio/.ssh/id_ed25519 ED25519-CERT
SHA256:q3u+woluMmljCo+HCKHr55oztAiZ7QVDXWkw20W02UY
Enter passphrase for key '/home/sio/.ssh/id_ed25519':
```

Remarques :

- le certificat a été **reconnu et utilisé**. Vous êtes invité à saisir la passphrase de votre clé privée.
- vous pouvez maintenant vous connecter avec le compte **sio** ou le compte **btssio** sans que l'administrateur doivent renseigner le fichier **~/.ssh/authorized_keys** des comptes sio et btssio.

Création du certificat pour le serveur distant

Pour la création du certificat du serveur il est nécessaire de préciser :

- **-s** : la clé privée de la CA qui signe le certificat
- **-I** : l'identifiant du certificat client. Pour l'exemple un utilisateur appelé charles.
- **-h** : précise qu'il s'agit d'un certificat serveur.

- **-n** : les noms / adresse IP du serveur.
- **-V** : la validité du certificat. En général une année.
- la **clé publique** du serveur situé dans le dossier **/etc/ssh** qui est nécessaire pour créer son certificat.

- Sur le serveur de CA, utilisez la commande suivante, en utilisant **sudo** afin que le certificat puisse être sauvegardé dans le dossier **/etc/ssh** :

```
$ sudo ssh-keygen -s ca_key -I SERVEUR-LOCAL -h -n localhost,127.0.0.1,adresseIP -V +365d
/etc/ssh/ssh_host_ed25519_key.pub
```

La commande suivante permet de visualiser le contenu du certificat serveur :

```
$ ssh-keygen -L -f /etc/ssh/ssh_host_ed25519_key-cert.pub
/etc/ssh/ssh_host_ed25519_key-cert.pub:
        Type: ssh-ed25519-cert-v01@openssh.com host certificate
        Public key: ED25519-CERT SHA256:++xMosJ1oo91GoxCV77GSb7tSBzKqKVvY2kDXvXzBr0
        Signing CA: ED25519 SHA256:5R5DL2MZrQelfe88lyf2zdv3UepovFcdm9NiHtplWUQ (using ssh-ed25519)
        Key ID: "SERVEUR-LOCAL"
        Serial: 0
        Valid: from 2026-06-28T20:58:00 to 2027-06-28T20:59:19
        Principals:
                localhost
                127.0.0.1
                adresseIP
        Critical Options: (none)
        Extensions: (none)
```

Remarques : il s'agit d'un certificat utilisateur : **Type ...host certificate**.

- Configuration du serveur SSH pour utiliser ce certificat en modifiant le fichier **/etc/ssh/sshd_config** :

```
...
HostKey /etc/ssh/ssh_host_ed25519_key
HostCertificate /etc/ssh/ssh_host_ed25519_key-cert.pub
TrustedUserCAKeys /etc/ssh/ssh_ca_keys
...
```

- lançez manuellement le serveur en mode debug, pour visualiser qu'il charge bien les 2 fichiers (la clé privée et le certificat) :

```
$  sudo /usr/sbin/sshd -d
sudo /usr/sbin/sshd -d
```

- redémarrez le service SSH

```
$ sudo systemctl restart ssh
```

Configuration du client pour pour vérifier le certificat du serveur

Rappel

Jusqu'à présent, au niveau du client, le fichier **~/.ssh/known_hosts** était renseigné avec la **clé publique** de chaque serveur distant auquel vous avez fait confiance pour vous connecter.

Désormais, avec l'utilisation d'un **certificat signé par une CA**, cela **n'est plus nécessaire**.

Il suffit que le client dispose de la clé publique de la CA pour vérifier les certificats présenté par les serveurs distants.

Préparation du cilent

- Dans l'ordinateur client, supprimez le contenu du fichier **~/.ssh/known_hosts** de votre compte client (ici sio) :

```
echo > ~/.ssh/known_hosts
```

- depuis le client, récupérez le fichier de la clé publique de la CA **ca_key.pub** afin de disposer de son contenu :

```
$ scp sio@adresseIPCA:/home/sio/ca_key.pub ~/.ssh/
```

Configuration du client

La configuration du compte du client (côté du client) se fait en ajoutant la directive **@cert-authority** dans le fichier **~/.ssh/known_hosts**.

Pour reconnaître tous les certificats signés par une CA précise, il faut ajouter une ligne avec le format suivant :

```
markers (optional) hostnames public key, comment
```

Remarques :

- Les différents champs sont **séparés par des espaces**.
- Pour les noms de machines (**hostnames**), vous pouvez utiliser une **liste de valeurs** séparée par des **virgules**.
- Les caractères joker (***, ? et !**) sont autorisés.
- Il est possible d'**interdire** une ou plusieurs valeurs avec le **caractère !**.
- le **#** permet de commenter une ligne.

- Autorisez tous les certificats signés par la clé privée de la CA en écrivant **UNE SEULE ligne** dans le fichier **~/.ssh/known_hosts** en indiquant la clé publique **ca_key.pub**, pour toutes les adresse IP qui commencent par **10.**.

```
@cert-authority 10.*,*.educ-valadon-limoges.fr ssh-ed25519 AAAAC3NzaC1lZDI1N... cle publique de la CA
```

Les tests

- **testez** en mode **debug** l'accepation du certificat du serveur distant

```
$ ssh -v sio@adresseIP
...
debug1: Server host certificate: ssh-ed25519-cert-v01@openssh.com
SHA256:++xMosJ1oo91GoxCV77GSb7tSBzKqKVvY2kDXvXzBr0, serial 0 ID "SERVEUR-LOCAL" CA ssh-ed25519
SHA256:5R5DL2MZrQelfe88lyf2zdv3UepovFcdm9NiHtplWUQ valid from 2026-06-28T23:19:00 to 2027-06-28T23:20:54
...
debug1: Host '10.xxx.yyy.zzz' is known and matches the ED25519-CERT host certificate.
debug1: Found CA key in /home/sio/.ssh/known_hosts:1
...
Enter passphrase for key '/home/sio/.ssh/id_ed25519':
```

Remarques :

- le certificat serveur est **présenté au client**;
- le **nom d'hôte** utilisé avec l'adresse IP **correspond** bien avec ce qui est renseigné dans le certificat (**principal**) ;
- la **clé publique de la CA** est bien **trouvé** chez le client pour **vérifier la signature** du certificat ;
- seule la **passphrase** de votre **clé privée** est demandée.

Vous n'avez pas de message d'avertissement et aucune ligne n'est ajoutée au fichier **~/.ssh/known_hosts**.

```
* **Modifiez le contenu du fichier **~/.ssh/known_hosts** pour interdire le adresses IP qui commencent
par 10. (ajout du caractère !).
```

```
@cert-authority !10.*,*.educ-valadon-limoges.fr ssh-ed25519 AAAAC3NzaC1lZDI1N... cle publique de la CA
```

```
* **testez** en mode **debug** la non acceptation du certificat du serveur distant :
```

```
$ ssh -v sio@adresseIP
...
debug1: Server host certificate: ssh-ed25519-cert-v01@openssh.com
SHA256:++xMosJ1oo91GoxCV77GSb7tSBzKqKVvY2kDXvXzBr0, serial 0 ID "SERVEUR-LOCAL" CA ssh-ed25519
SHA256:5R5DL2MZrQelfe88lyf2zdv3UepovFcdm9NiHtplWUQ valid from 2026-06-28T23:19:00 to 2027-06-28T23:20:54
...
debug1: Host '10.xxx.yyy.zzz' is known and matches the ED25519-CERT host certificate.
debug1: Found CA key in /home/sio/.ssh/known_hosts:1
...
Enter passphrase for key '/home/sio/.ssh/id_ed25519':
```