? Guide Complet : Authentification SSH par Clés

L'authentification par clés SSH remplace avantageusement les mots de passe. Elle est plus sécurisée, insensible aux attaques par force brute, et indispensable pour l'automatisation avec des outils comme Ansible (AWX).

---

1. ? Le Concept : Clé Publique vs Clé Privée

L'authentification SSH repose sur la cryptographie asymétrique. Au lieu d'avoir un mot de passe unique, on utilise un couple de fichiers indissociables :

◫◫ La Clé Publique (Le Cadenas)	◫◫ La Clé Privée (La Clé physique)
• **Extension :** Finit toujours par `.pub` (ex: `id_ed25519.pub`). • **Rôle :** C'est le cadenas que vous installez sur toutes les portes (serveurs) où vous souhaitez entrer. • **Sécurité :** Elle n'est **pas secrète**. Vous pouvez la distribuer partout, l'envoyer par mail, la mettre sur GitHub. Sans la clé privée associée, elle ne sert à rien.	• **Extension :** Pas d'extension (ex: `id_ed25519`). • **Rôle :** C'est la clé unique que vous gardez sur vous (votre PC ou votre gestionnaire de mots de passe / AWX). Elle ouvre les cadenas. • **Sécurité : ULTRA SECRÈTE**. Elle ne doit jamais quitter votre machine ou votre coffre-fort. Si quelqu'un la vole, il a accès à tous vos serveurs.

---

2. ?? Générer sa paire de clés

On utilise aujourd'hui l'algorithme **ED25519**, qui est plus moderne, plus rapide et plus sécurisé que l'ancien RSA.

Ouvrez un terminal sur votre machine locale et tapez :

```
ssh-keygen -t ed25519 -C "cle-admin-cluster"
```

- `-t ed25519` : Spécifie l'algorithme cryptographique.

- `-C "..."` : Un commentaire pour vous rappeler à quoi sert cette clé (très utile quand on en a plusieurs).

Note : L'outil vous demandera où sauvegarder (laissez par défaut) et si vous voulez une "passphrase". Pour l'automatisation (Ansible/AWX), laissez la passphrase vide en appuyant deux fois sur Entrée.

---

3. ? Déployer la clé (Méthode classique)

Pour autoriser votre clé sur un serveur distant, il faut copier le contenu de la clé publique (le cadenas) dans le fichier `~/.ssh/authorized_keys` du serveur. La commande `ssh-copy-id` fait cela automatiquement :

```
# Remplacer 'utilisateur' et 'ip_du_serveur'
ssh-copy-id -i ~/.ssh/id_ed25519.pub utilisateur@10.151.x.x
```

Il vous demandera le mot de passe du serveur une dernière fois pour installer le cadenas. Ensuite, vous pourrez vous connecter sans mot de passe !

---

4. ? Déploiement Massif (La boucle du Sysadmin)

Si vous avez 15 serveurs ou LXC à configurer d'un coup, faire la commande précédente manuellement est fastidieux (taper "yes" pour l'empreinte, taper 15 fois le mot de passe...).

Voici comment automatiser le déploiement de la clé publique avec une boucle Bash et l'outil `sshpass`.

Prérequis :

Installez l'utilitaire `sshpass` sur votre machine d'administration :

```
sudo apt update && sudo apt install sshpass -y
```

Le Script de boucle :

Modifiez ce bloc avec vos IPs, votre utilisateur cible, et votre mot de passe, puis collez-le dans le terminal.

```
# 1. On stocke le mot de passe dans une variable d'environnement
export SSHPASS='MonMotDePasseSuperSecret'

# 2. On lance la boucle sur notre liste d'IPs
for ip in 10.151.151.230 10.151.151.244 10.151.151.100; do
    echo -e "\n=== Copie de la clé sur root@$ip ==="

    # sshpass : injecte le mot de passe automatiquement
    # StrictHostKeyChecking=no : accepte automatiquement la nouvelle empreinte (le fameux 'yes')
    sshpass -e ssh-copy-id -o StrictHostKeyChecking=no -i ~/.ssh/id_ed25519.pub root@$ip
done

# 3. On détruit la variable pour des raisons de sécurité
unset SSHPASS
```

Résultat : En 5 secondes, votre clé SSH est poussée sur l'ensemble de votre parc. Vous êtes prêt pour Ansible !

Revision #1
Created 2026-05-04 09:07:51 UTC by Nico là
Updated 2026-05-04 09:08:16 UTC by Nico là