Maëllie BRIENT--BERT

Serveur SSH sécurisé avec clés :

Objectif :

- Mettre en place un serveur SSH fonctionnel sur une machine Linux
- Supprimer la connexion par mot de passe pour garder seulement l'authentification par clés
- Apprendre à :
- Générer et utiliser des clés SSH
- Configurer le service sshd
- Tester et vérifier la sécurité de la connexion

Prérequis :

- Une VM (Machine Virtuelle) Linux sur Debian.
- Un client (PC ou autre VM)
- Accès administrateur (sudo) sur la VM Linux
- Paquet OpenSSH Server installé sur la VM Linux

Rappel :

Voici un rappel pour :

- Créer l'accès administrateur
- Installer OpenSSH Server

Accès administrateur :

Pour pouvoir utiliser sudo, il faut rajouter l'utilisateur dans le fichier sudoers.

Pour cela on doit se connecter en tant que root (admin) sur le terminal :

```
user@ssh:~$ su -
Mot de passe :
root@ssh:~#
```

On va ensuite modifier le fichier sudoers.

```
root@ssh:~# visudo
```

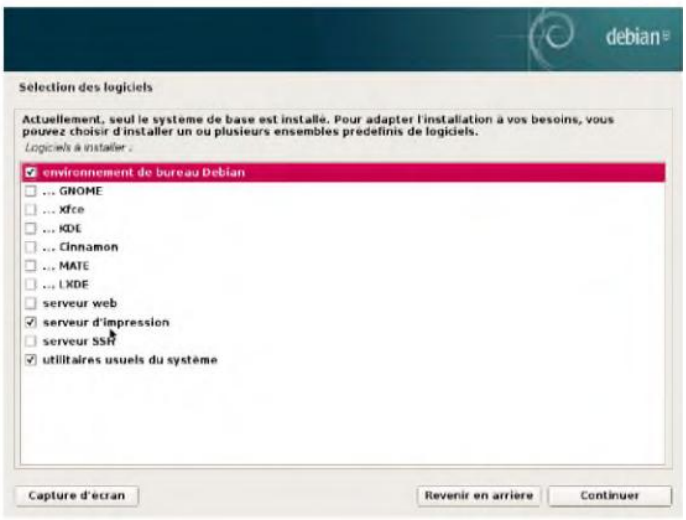
On ajoute l'utilisateur dans le fichier.

```
# User privilege specification
root    ALL=(ALL:ALL) ALL
user    ALL=(ALL:ALL) ALL
```

Ensuite on peut quitter la connexion root.

Installation OpenSSH Server :

Pour l'installer on va directement cocher la case OpenSSH lors de l'installation de Debian.



Si vous n'avez pas installer OpenSSH depuis le menu d'installation de Debian, alors veuillez suivre la méthode qui se trouve dans la rebrique Problèmes rencontrés et solutions : 1- Installation OpenSSH Server.

Etapes :

Une fois que l'utilisateur peut utiliser sudo et que OpenSSH server est installé, on va pouvoir suivre les étapes de ce projet.

1- Vérification du service SSH :

Vu que ce projet demande l'utilisation de OpenSSH nous allons vérifier que ce dernier est bien activer :

```
user@ssh:~$ sudo systemctl status ssh
[sudo] Mot de passe de user :
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)
     Active: active (running) since Tue 2025-08-26 12:33:07 CEST; 9h ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 721 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 732 (sshd)
      Tasks: 1 (limit: 2236)
     Memory: 3.4M
        CPU: 107ms
     CGroup: /system.slice/ssh.service
             └─732 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

août 26 12:33:07 ssh systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
août 26 12:33:07 ssh sshd[732]: Server listening on 0.0.0.0 port 22.
août 26 12:33:07 ssh sshd[732]: Server listening on :: port 22.
août 26 12:33:07 ssh systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
août 26 17:32:50 ssh sshd[3467]: Connection reset by 192.168.150.1 port 49677 [preauth]
```

Si votre service apparait comme étant inactif, veuillez suivre la solution qui se trouve dans la partie Problèmes et solutions : 2- Démarrage du service SSH.

2- Génération d'une paire de clé SSH sur le client :

On créer la clé SSH public et privé qui vont être stocké sur le client et dans deux fichiers :

- Un fichier pour la clé publique
- Un fichier pour la clé privée

```
PS C:\Users\mbrie> ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\mbrie/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\mbrie/.ssh/id_rsa
Your public key has been saved in C:\Users\mbrie/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:o0ChCcBMurN9ffi/eL59Gz6/2BLoX6cwm0KP3sxz++Q mbrie@Ma♦llie
```

3- Copie de la clé publique sur le serveur SSH :

On va pouvoir copier la clé publique SSH sur notre VM qui sert de serveur. On reste sur le client pour faire la copie.

Mais nous avons besoin de l'adresse IP de notre serveur :

```
user@ssh:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:29:22:b2 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.150.151/24 brd 192.168.150.255 scope global dynamic noprefixroute ens33
       valid_lft 1538sec preferred_lft 1538sec
    inet6 fe80::20c:29ff:fe29:22b2/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

Pour la copie de la clé publique, si vous rencontrer ce message d'erreur :

```
PS C:\Users\mbrie> ssh-copy-id user@192.168.150.151
ssh-copy-id : Le terme «ssh-copy-id» n'est pas reconnu comme nom d'applet de commande, fonction, fichier de script ou programme exécutable. Vérifiez
l'orthographe du nom, ou si un chemin d'accès existe, vérifiez que le chemin d'accès est correct et réessayez.
Au caractère Ligne:1 : 1
+ ssh-copy-id user@192.168.150.151
+ ~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (ssh-copy-id:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException
```

C'est normal, en effet cette commande fonctionne uniquement sous Linux ou Mac. Si vous avez un Linux ou un Mac vous pouvez utiliser la commande ssh-copy-id name_session@ip_serveur. Sinon veuillez utiliser la méthode qui se trouve dans la rebrique Problèmes rencontrés et solutions : 3- Problème de copy de clé.

On retourne ensuite sur notre VM, sur laquelle on va créer un dossier caché. on va sécuriser notre dossier caché ainsi que le fichier qui va contenir notre clé.

```
user@ssh:~$ mkdir -p ~/.ssh
```

Ensuite nous ajoutons notre clé publique dans un fichier qui sera dans ce dossier.

```
user@ssh:~$ nano ~/.ssh/authorized_keys
```

On sécurise le dossier caché.

```
user@ssh:~$ chmod 700 ~/.ssh
```

On sécurise ensuite notre fichier qui contient notre clé publique.

```
user@ssh:~$ chmod 600 ~/.ssh/authorized_keys
```

4- Configuration du serveur SSH :

Nous allons maintenant configurer notre serveur SSH pour pouvoir utiliser une authentification seulement par clé et plus par mot de passe.

Pour cela on va commencer par modifier ce fichier de configuration :

```
user@ssh:~$ sudo nano /etc/ssh/sshd_config
```

On va modifier certaines lignes :

```
#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

PubkeyAuthentication yes

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile     .ssh/authorized_keys .ssh/authorized_keys2

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no
#PermitEmptyPasswords no
```

Une fois ces modifications faites ils nous restent plus qu'à redémarrer le service SSH.

```
user@ssh:~$ sudo systemctl restart ssh
```

Résultat :

Maintenant que nous avons terminé notre serveur SSH avec notre clé publique.

Pour savoir si cela fonctionne nous allons retourner sur notre client pour tester la connexion entre ce dernier et notre VM.

1- Connexion avec le mot de passe :

On va commencer par tester que le mot de passe a bien été désactivé.

```
PS C:\Users\mbrie> ssh user@192.168.150.151 -o PreferredAuthentications=1234
user@192.168.150.151: Permission denied (publickey).
```

La connexion par mot de passe a bien été désactivé, la permission de se connecter au serveur en passant par un mot de passe a bien été refusé.

2- Connexion par clé publique :

On va terminer se test en se connectant à notre serveur par notre clé.

```
PS C:\Users\mbrie> ssh user@192.168.150.151
Linux ssh 6.1.0-38-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.147-1 (2025-08-02) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Aug 29 17:39:59 2025 from 192.168.150.1
user@ssh:~$
```

Problèmes rencontrés et solutions :

1- Installation OpenSSH server :

Voici comment installer OpenSSH si cela n'a pas été fait come dans le rappel :

- Mettez à jour votre Debian avec la commande : sudo apt update
- Installez OpenSSH avec la commande : sudo apt install openssh-server -y

2- Démarrage du service SSH :

Voici comment démarrer le service SSH : sudo systemctl enable –now ssh

Vérifier ensuite que le service a bien démarré.

3- Problème de copie de clé :

Si vous êtes sur Windows ou Git Bash, vous allez devoir copier manuellement votre clé publique et uniquement votre clé publique, on va devoir copier notre clé publique manuellement.

```
PS C:\Users\mbrie\.ssh> type $env:USERPROFILE\.ssh\id_rsa.pub
```

Conclusion et apprentissage :

1- Conclusion :

Pour rappel l'objectif de ce projet était de sécuriser l'accès SSH au serveur en utilisant un système d'authentification par clé publique et privé à la place de mot de passe.

Une fois qu'une paire de clé a été générée, la clé publique copié sur le serveur et les bons droits configurés pour pouvoir réussir à se connecter sans mot de passe depuis le client. Cela permet une connexion plus pratique et sécurisé.

Une amélioration possible pourrait être de désactiver complétement l'authentification par mot de passe dans la configuration du serveur SSH.

2- Apprentissage :

Voici ce que l'on peut retenir de ce projet :

- Gestion des clés SSH : génération, rôle de la clé privé et publique.
- Importances des permissions sur les dossiers et fichiers (chmod 700, chmod 600).
- Différence entre authentification par mot de passe et par clé.
- Utilisation d'outils sous Windows (OpenSSH, PowerShell) et Linux.
- Rigueur sur le nom des fichiers (authorized_keys).
- Notion de sécurité par réduction de surface d'attaque (éviter le brute force sur les mots de passe).