SSH avec échange de clés

Tamara Crétard - 16.09.2025

Sommaire:

1. Contexte

Dans le cadre d'un cours du BTS SIO, l'objectif est de comprendre l'utilité et le fonctionnement du protocole SSH. Cet apprentissage comprend notamment la génération de paires de clés privée/publique.

2. Environnement de travail

Afin de pouvoir expérimenter l'utilité et le fonctionnement du protocole SSH, il est important de visualiser notre environnement de travail.

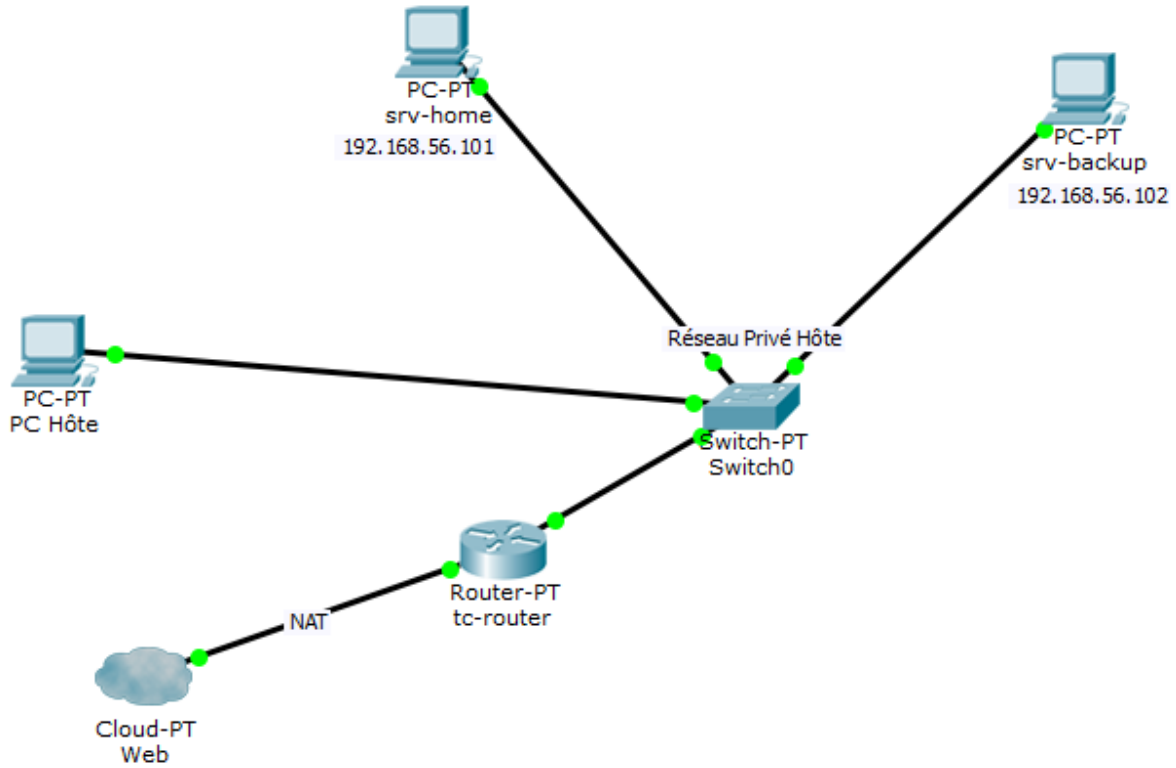
2.1. Configuration utilisée

Tout d'abord, il est intéressant de connaître la configuration utilisée par le PC hôte. Dans notre cas, la configuration utilisée est la suivante:

Processeur	11th Gen Intel(R) Core(TM) i7-11700K @ 3.60GHz 3.60 GHz
RAM	32 Go
Carte graphique	NVIDIA GeForce RTX 3080
SSD	2 To
Système d'exploitation	Microsoft Windows 11 Famille
Architecture	x64

2.2. Schéma du réseau virtuel

Afin de mieux visualiser notre installation, un schéma du réseau virtuel réalisé à l'aide de CISCO Packet Tracer est utile:



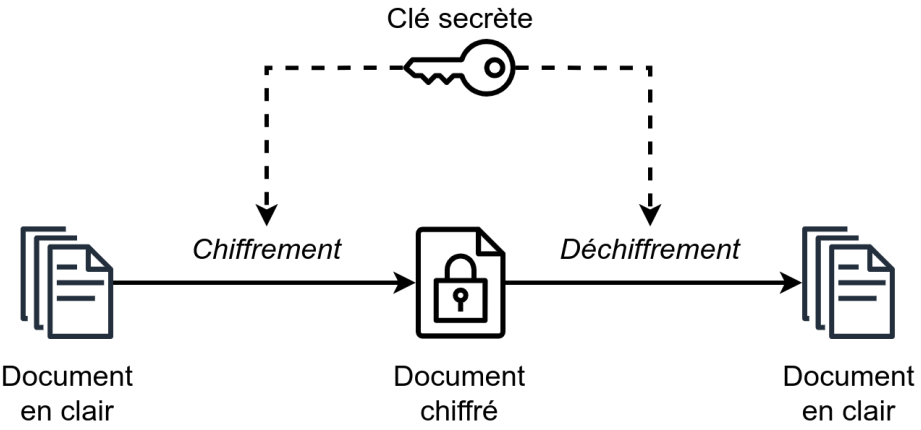
Dans notre cas, nous avons besoin d'une machine hôte, de deux machines virtuelles Linux Bullseye en Réseau Privé Hôte, ainsi que d'un routeur. La première machine virtuelle est nommée *srv-home* avec une adresse IP en 192.168.56.101 et la deuxième, *srv-backup* en 192.168.56.102.

3. SSH

SSH (Secure Shell) est un protocole sécurisé qui utilise le port 22. Il remplace Telnet et FTP. Le protocole SSH fournit une connexion de gestion sécurisée (cryptée) à un appareil distant. Cette connexion utilise le chiffrement asymétrique au départ, puis symétrique une fois la session établie.

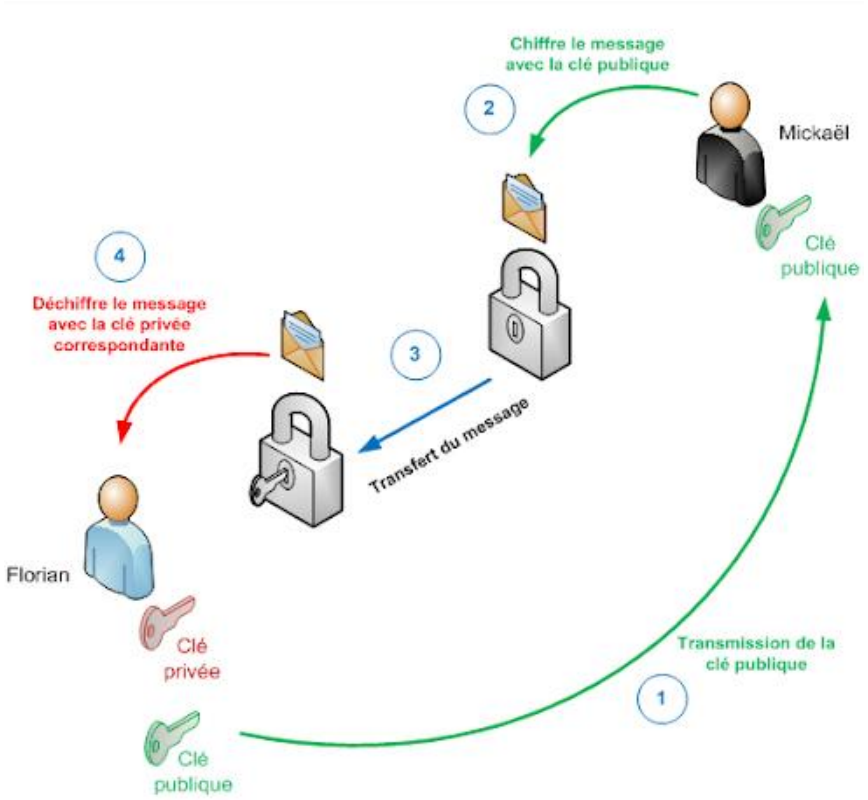
3.1. Chiffrement symétrique

Le chiffrement symétrique utilise la même clé pour chiffrer et déchiffrer. Parmi les algorithmes de chiffrement symétrique, on retrouve AES, DES et 3DES. Ce type de chiffrement est plus efficace pour une grande quantité de données. L'inconvénient est que la confidentialité est moindre car la même clé est utilisée.



3.2. Chiffrement asymétrique

Le chiffrement asymétrique utilise une paire de clés publique/privée. En effet, la clé publique du destinataire est utilisée pour chiffrer et ce dernier utilise sa clé privée pour déchiffrer. Parmi les algorithmes de chiffrement asymétrique, on retrouve RSA et Diffie-Hellman. Ce type de chiffrement est plus lent et moins adapté pour une grande quantité de données. Cependant, il garantit un haut niveau de confidentialité car une paire de clés est utilisée.



4. Expérimentations

Maintenant que nous avons exposé le protocole SSH, son utilité, ainsi que son fonctionnement, nous allons mettre en pratique cela.

4.1. Prérequis

Avant toute expérimentation, il est nécessaire d'installer deux machines: l'une appelée *srv-home* avec une adresse IP 192.168.56.101 et l'autre, *srv-backup* en 192.168.56.102. Les deux machines doivent être en Réseau Privé Hôte.

Le paquet SSH doit être installé sur *srv-home* à l'aide de la commande **apt install ssh,** après avoir réalisé un **apt update**:

```
root@srv-home:~# apt update
```

```
root@srv-home:~# apt install ssh
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
ssh est déjà la version la plus récente (1:8.4p1-5+deb11u3).
0 mis à jour, 0 nouvellement installés, 0 à enlever et 8 non mis à jour.
```

Nous pouvons vérifier le bon fonctionnement de SSH avec la commande **systemctl status ssh**:

```
root@srv-home:~# systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2025-10-08 20:13:00 CEST; 5min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 358 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 394 (sshd)
      Tasks: 1 (limit: 1115)
     Memory: 5.1M
        CPU: 19ms
     CGroup: /system.slice/ssh.service
             └─394 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

oct. 08 20:13:00 srv-home systemd[1]: Starting OpenBSD Secure Shell server...
oct. 08 20:13:00 srv-home sshd[394]: Server listening on 0.0.0.0 port 22.
oct. 08 20:13:00 srv-home sshd[394]: Server listening on :: port 22.
oct. 08 20:13:00 srv-home systemd[1]: Started OpenBSD Secure Shell server.
```

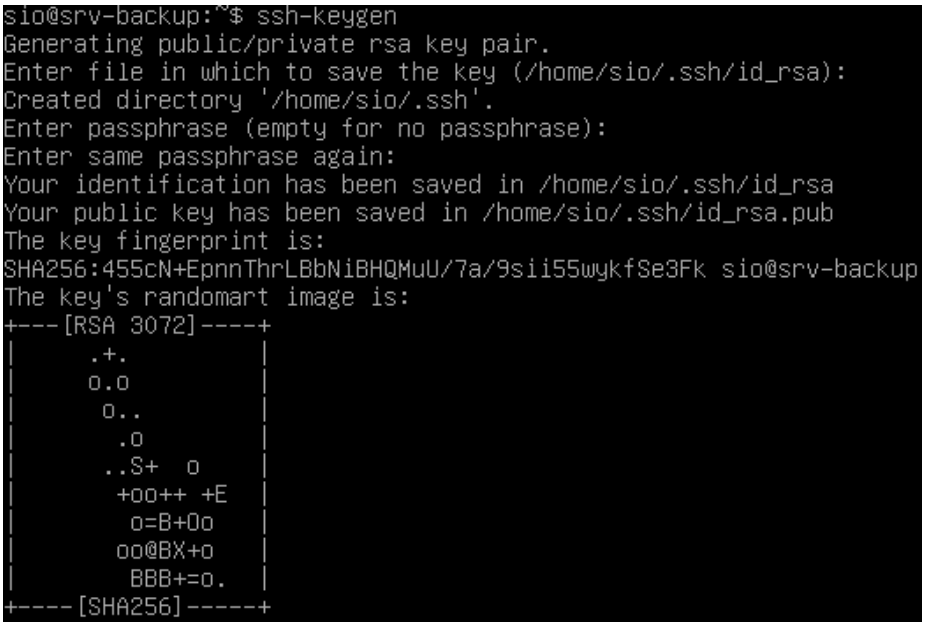
Également, il est important de changer le nom des machines avec la commande **nano /etc/hostname** pour éviter toute confusion lors des manipulations:

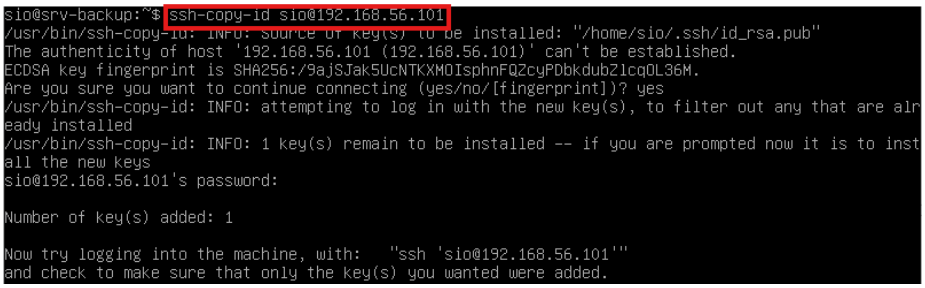
```
root@bullseye:~# nano /etc/hostname root@srv-home:~#
root@bullseye:~# nano /etc/hostnameroot@srv-backup:~#
```

4.2. Connexion SSH de *srv-backup* vers *srv-home*

Dans cette expérimentation, l'objectif est d'accéder à *srv-home* depuis *srv-backup*, sans avoir à utiliser de mot de passe à chaque connexion avec SSH.

Pour y arriver, plusieurs étapes doivent être suivies:

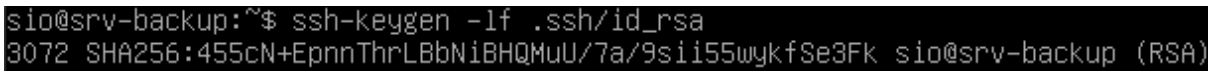
Etape	Description
1	Sur *srv-backup*, générer une clé publique/privée avec la commande **ssh-keygen.**
	Représentation
	
Etape	Description
2	Il n'est pas nécessaire d'entrer de passphrase, simplement presser la touche "Enter".
	Représentation
	
Etape	Description
3	Copier la clé publique de *srv-backup*, sur *srv-home* avec la commande **ssh-copy-id sio@192.168.56.101.** Cela permet de faire en sorte que lors des futures connexions, nous puissions nous connecter à l'utilisateur *sio* de *srv-home* (192.168.56.101) sans utiliser de mot de passe.

	Représentation
	

Après avoir suivi ces étapes, il nous est possible de se connecter depuis *srv-backup* au compte *sio* de *srv-home* sans utiliser de mot de passe grâce à la commande **ssh sio@192.168.56.101**:

Cela est possible car *srv-backup* possède la clé privée qui correspond à sa clé publique enregistrée dans *srv-home*.

Si l'on souhaite connaître la clé publique associée à la paire de clé, il est possible de taper la commande **ssh-keygen -lf .ssh/id_rsa**:



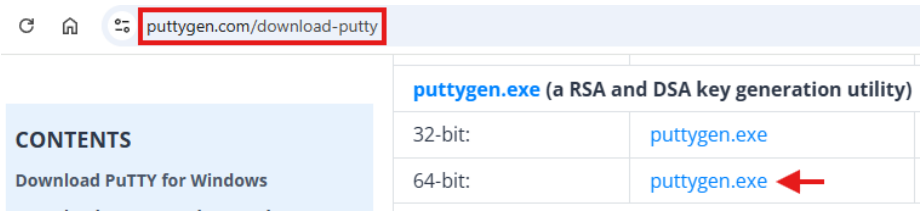
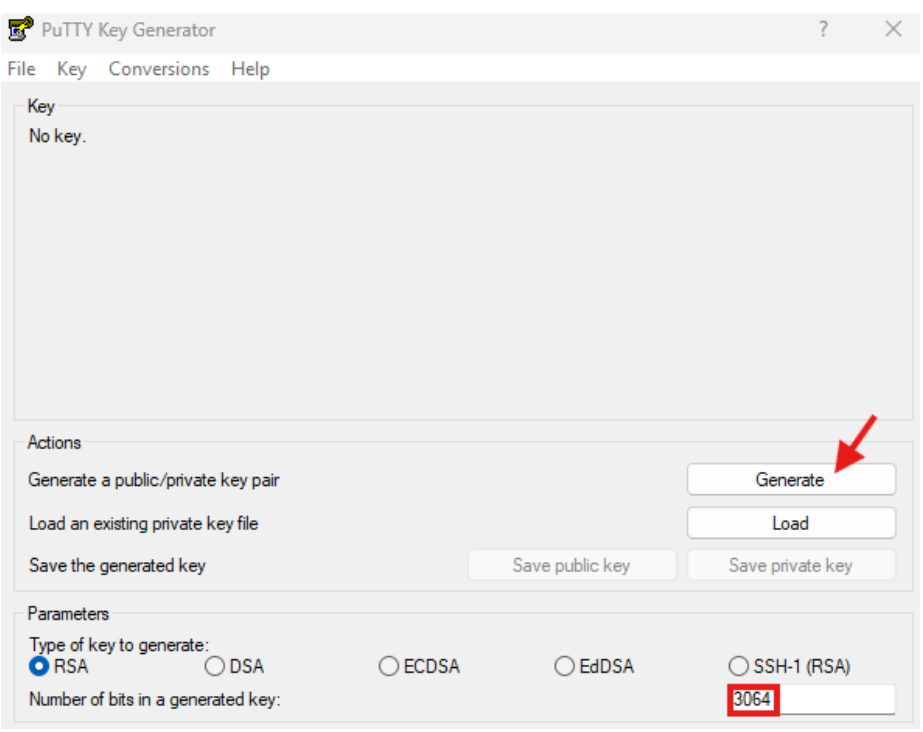
4.3. Connexion SSH du PC hôte vers *srv-home*

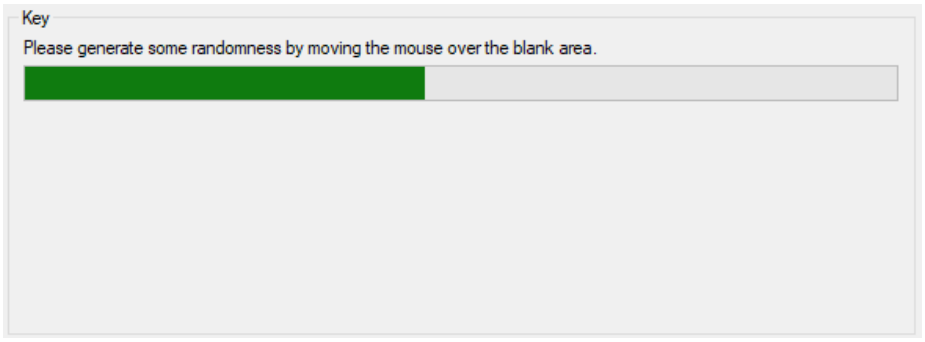
Maintenant, nous allons nous connecter du PC hôte avec un système d'exploitation Windows au serveur Debian *srv-home*, sans avoir à utiliser de mot de passe à chaque connexion avec SSH.

Pour cela, il est nécessaire de suivre plusieurs étapes, certaines depuis le PC hôte et d'autres depuis *srv-home*.

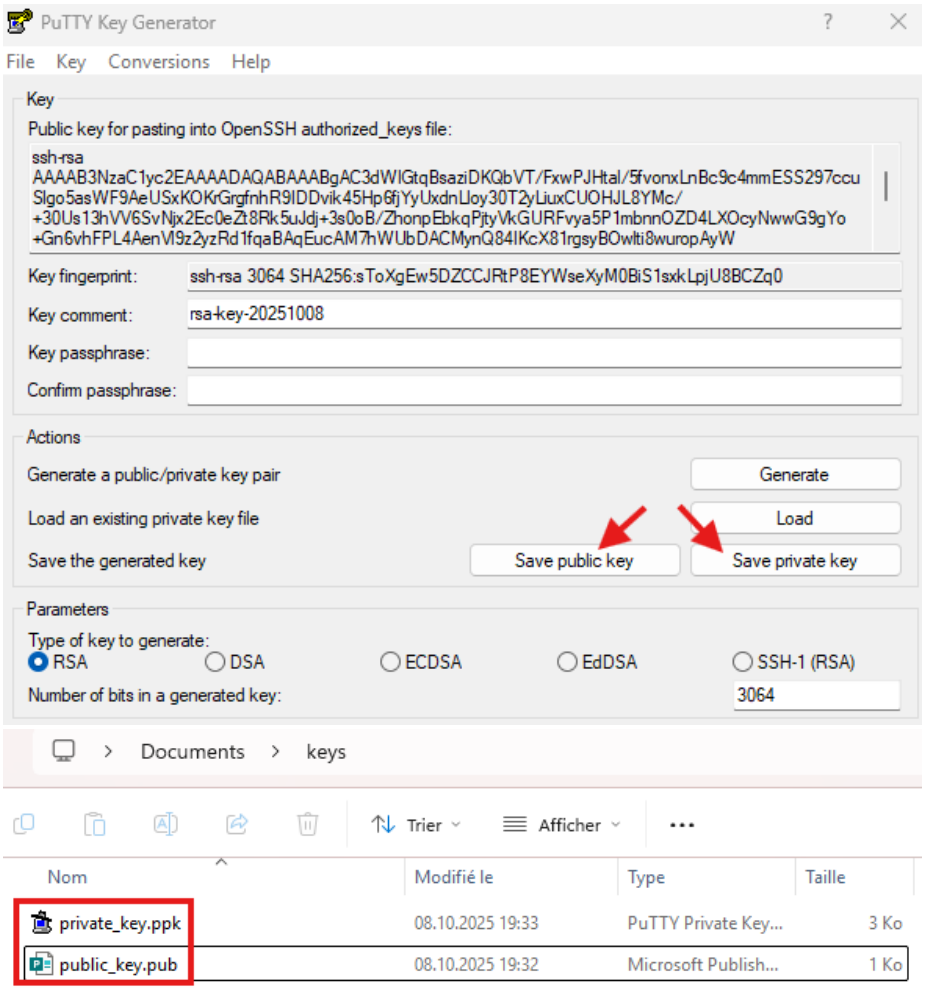
4.3.1. Génération de clés sur le PC hôte

Les premières manipulations sont à réaliser sur le PC hôte Windows, il faut en effet générer une paire de clés en suivant les instructions suivantes:

Etape	Description
1	Installer PuTTy Key Generator via https://puttygen.com/download-putty.
	Représentation
	
Etape	Description
2	Après avoir lancé PuTTy Key Generator, choisir 3064 pour le nombre de bits et simplement cliquer sur “Generate”. Il faut ensuite bouger la souris pour générer la paire de clés.
	Représentation
	

<table>
<tr><td></td><td> 
</td></tr>
<tr><td>Etape</td><td>Description</td></tr>
<tr><td>3</td><td>Cliquer sur “Save public key” pour sauvegarder la clé publique, puis sur “Save private key” pour sauvegarder la clé privée. L’extension de fichier pour la clé privée est .ppk, tandis que pour la clé publique, l’extension est .pub.

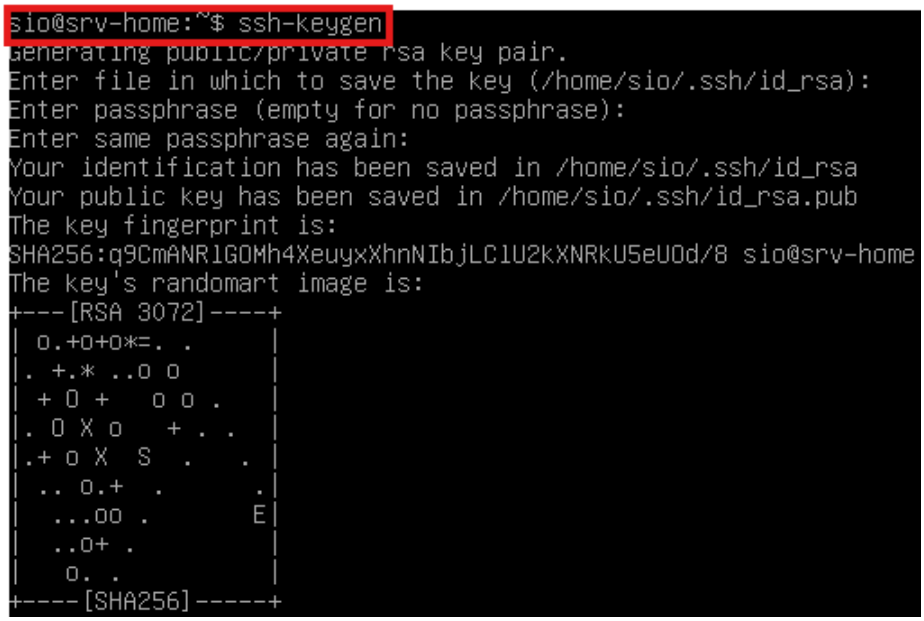
Représentation

 
</td></tr>
</table>

4.3.2. Génération de clés sur *srv-home*

Après avoir généré une paire de clés sur le PC hôte, il est nécessaire de générer une paire de clés sur la machine *srv-home*, car nous allons devoir copier la clé publique du PC hôte dans le répertoire *authorized_keys* de *srv-home*.

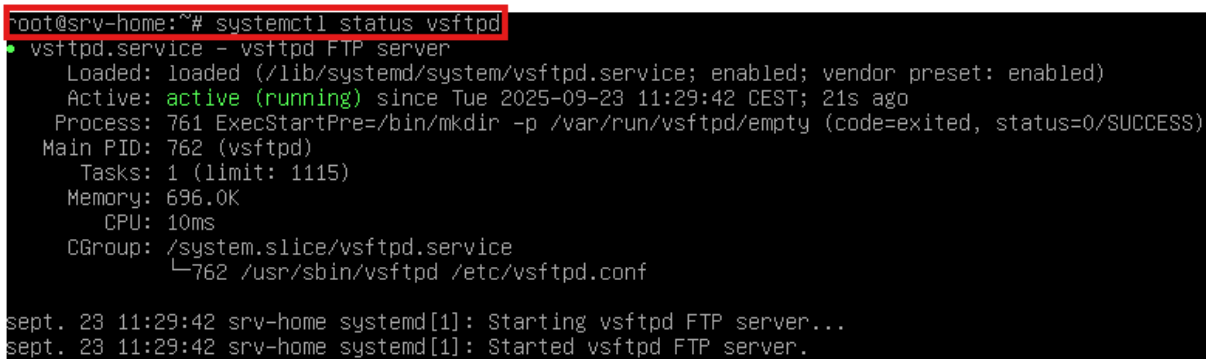
Pour y arriver, voici les étapes à suivre:

Etape	Description
1	Sur *srv-home*, se connecter à l'utilisateur *sio* et générer une paire de clés avec **ssh-keygen**. Il n'est pas nécessaire d'entrer de passphrase, simplement presser la touche "Enter".
	Représentation
	

Etape	Description
2	Mettre à jour les paquets avec la commande **apt update**.
	Représentation
	

Etape	Description
3	Installer FTP sur srv-home avec la commande **apt install vsftpd**.
	Représentation
	`root@srv-home:~# apt install vsftpd`

Nous pouvons ensuite vérifier que le service est bien activé avec la commande **systemctl status vsftpd**:



Afin de pouvoir effectuer les manipulations suivantes sur la machine Windows, il est nécessaire de modifier certaines autorisations.

Pour le répertoire .ssh, seul le propriétaire doit pouvoir lire, écrire et exécuter. Il faut donc entrer la commande **chmod 700 /home/sio/.ssh**:

```
root@srv-home:~# chmod 700 /home/sio/.ssh
```

Pour le fichier authorized_keys, seul le propriétaire doit pouvoir lire et écrire. Il faut donc entrer la commande **chmod 600 /home/sio/.ssh/authorized_keys**:

```
root@srv-home:~# chmod 600 /home/sio/.ssh/authorized_keys
```

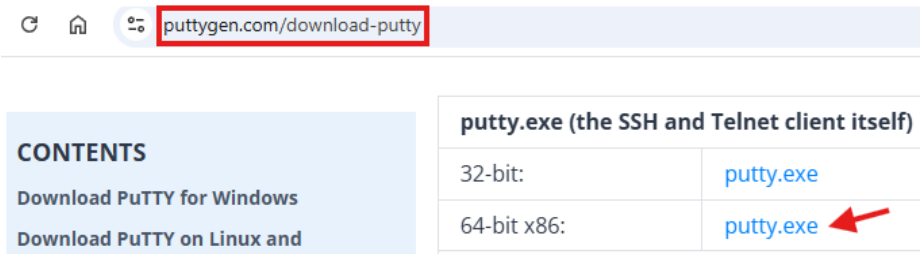
Enfin, il faut s'assurer que sio est bien l'utilisateur propriétaire avec la commande **chown -R sio:sio /home/sio/.ssh**:

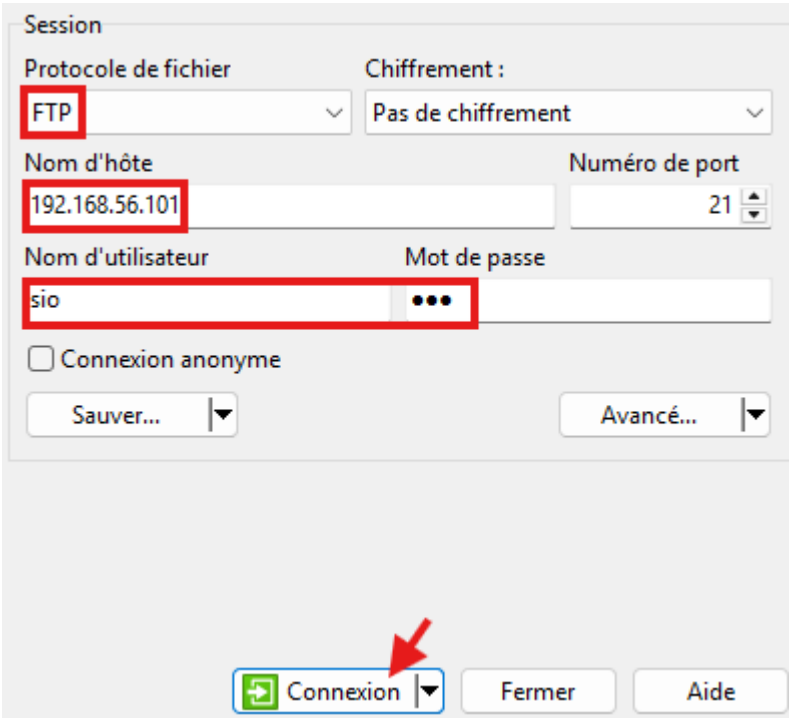
```
root@srv-home:~# chown -R sio:sio /home/sio/.ssh
```

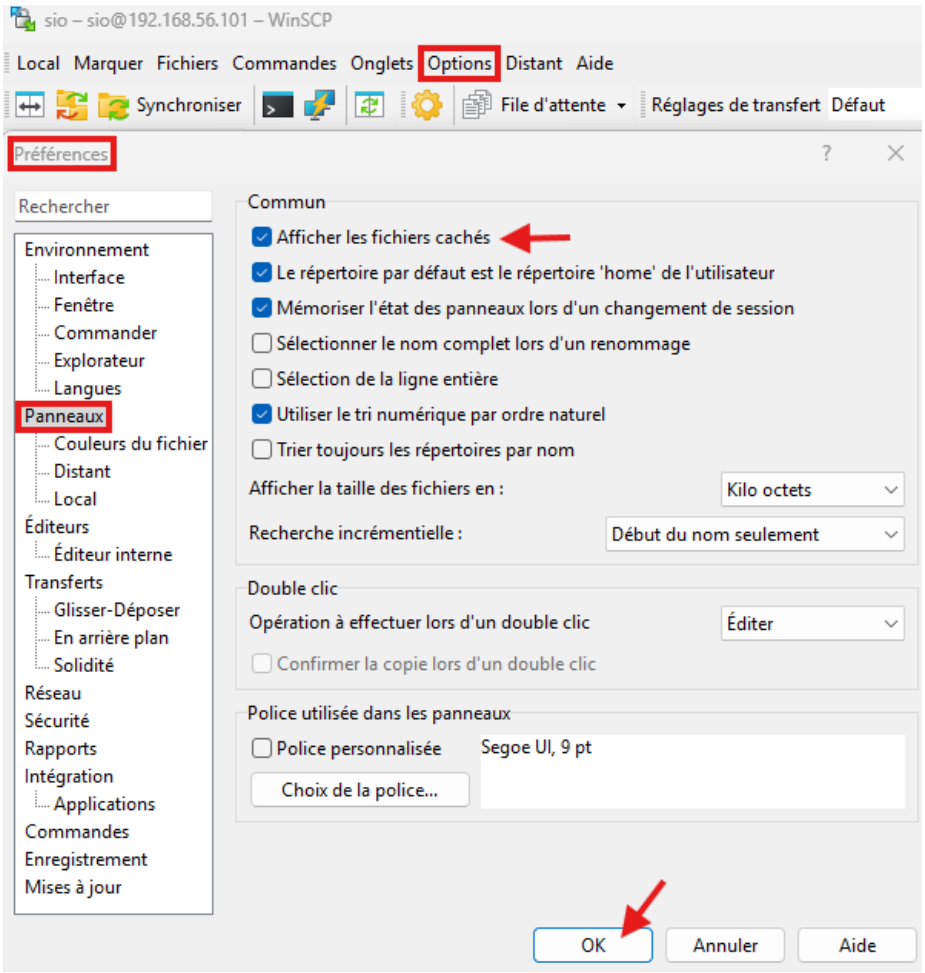
4.3.3. Copie de la clé publique du PC hôte vers *srv-home*

Maintenant qu'un répertoire *Authorized_keys* existe sur *srv-home*, nous allons pouvoir y copier la clé publique de la machine Windows afin que *srv-home* la connaisse et que la connexion SSH depuis le poste Windows soit possible sans entrer de mot de passe.

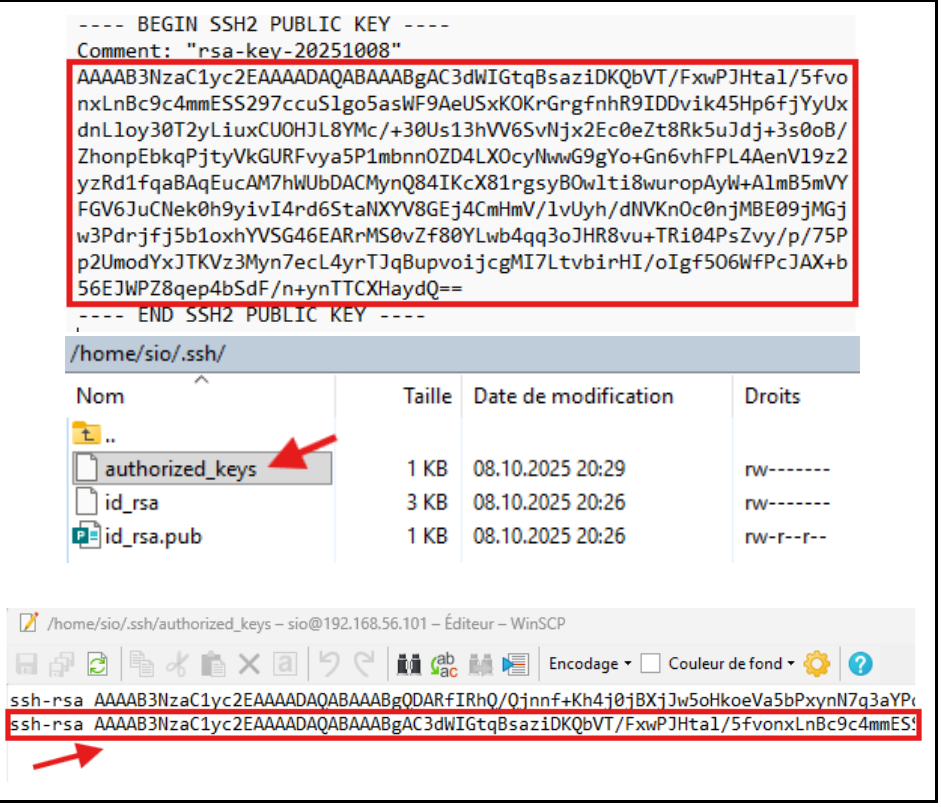
Pour cela, il est nécessaire de suivre plusieurs étapes:

Etape	Description
1	Installer PuTTy via https://puttygen.com/download-putty.
	Représentation
	
Etape	Description
2	Installer WinSCP via https://winscp.net/eng/download.php.
	Représentation
	

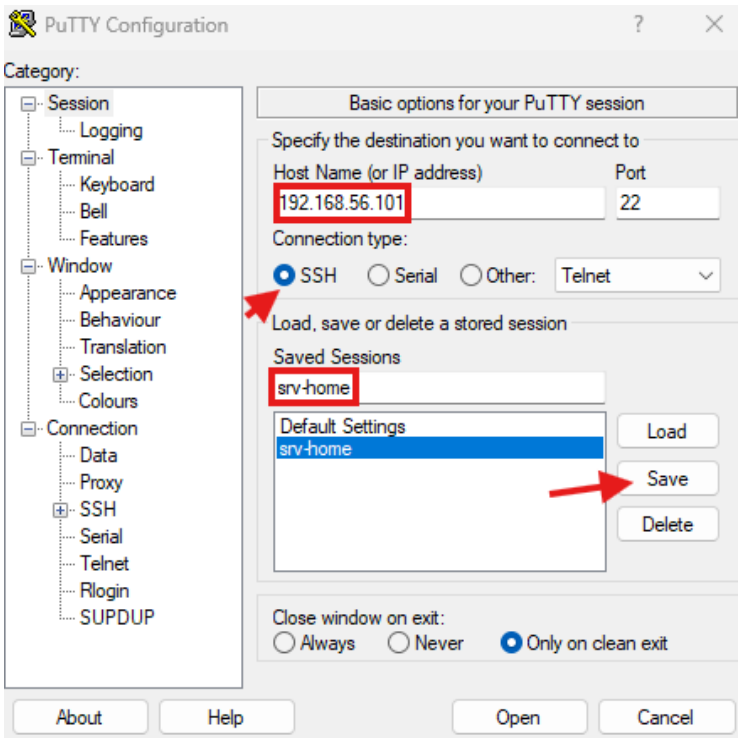
Etape	Description
3	Ouvrir WinSCP et se connecter en FTP de Windows à Debian. Pour cela, il faut choisir FTP, entrer l'adresse IP de *srv-home* et le nom de l'utilisateur ainsi que son mot de passe. La connexion FTP est temporaire, elle n'est pas sécurisée et sert simplement à copier la clé publique de Windows vers *srv-home*.
	Représentation
	

Etape	Description
4	Ensuite, aller dans l'onglet "Outils", puis "Préférences", et dans la catégorie "Panneaux" cocher "Afficher les fichiers cachés".
	Représentation
	

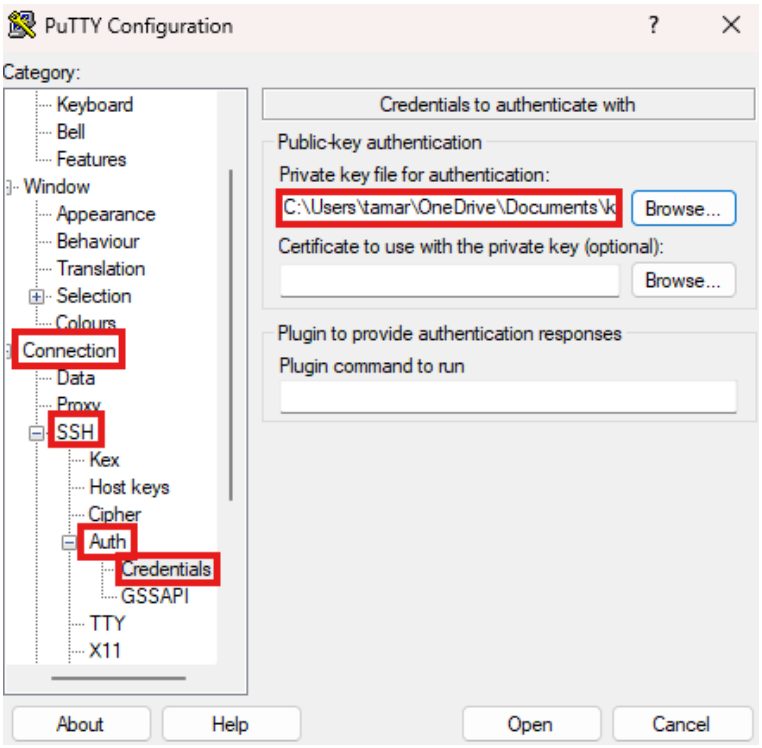
Etape	Description
5	Il faut ensuite copier la clé publique de Windows sauvegardée précédemment, dans le fichier *authorized_keys* de *srv-home*. Il est nécessaire de réarranger la clé publique pour qu'elle soit sur la même ligne.

	Représentation
	

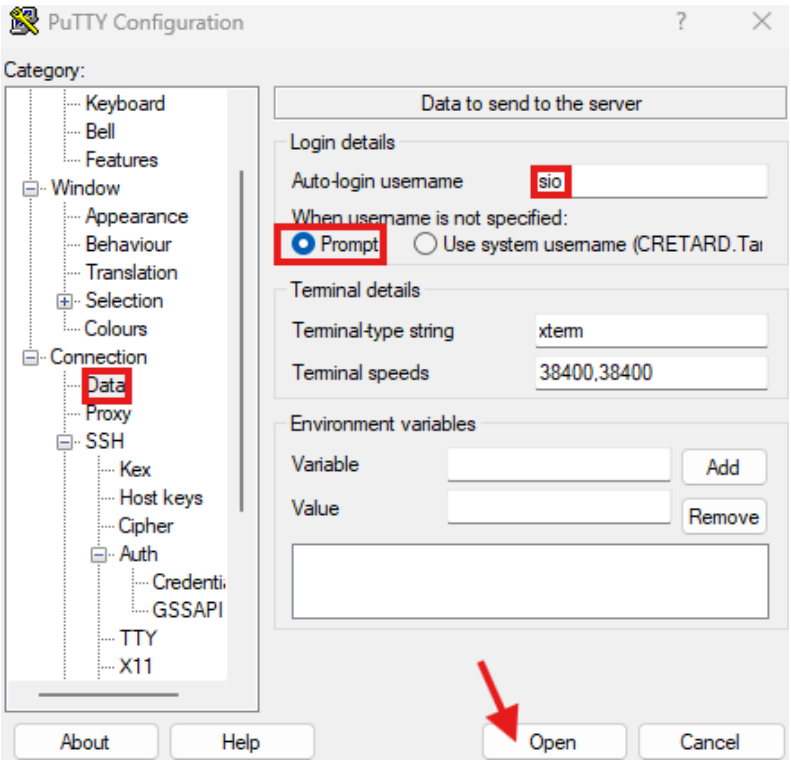
Afin de pouvoir se connecter sans utiliser de mot de passe avec Putty, dans l'onglet principal "Session", nous entrons l'adresse IP de *srv-home* et sélectionnons la connexion SSH. Nous sauvegardons également la session pour pouvoir se connecter plus rapidement les prochaines fois:



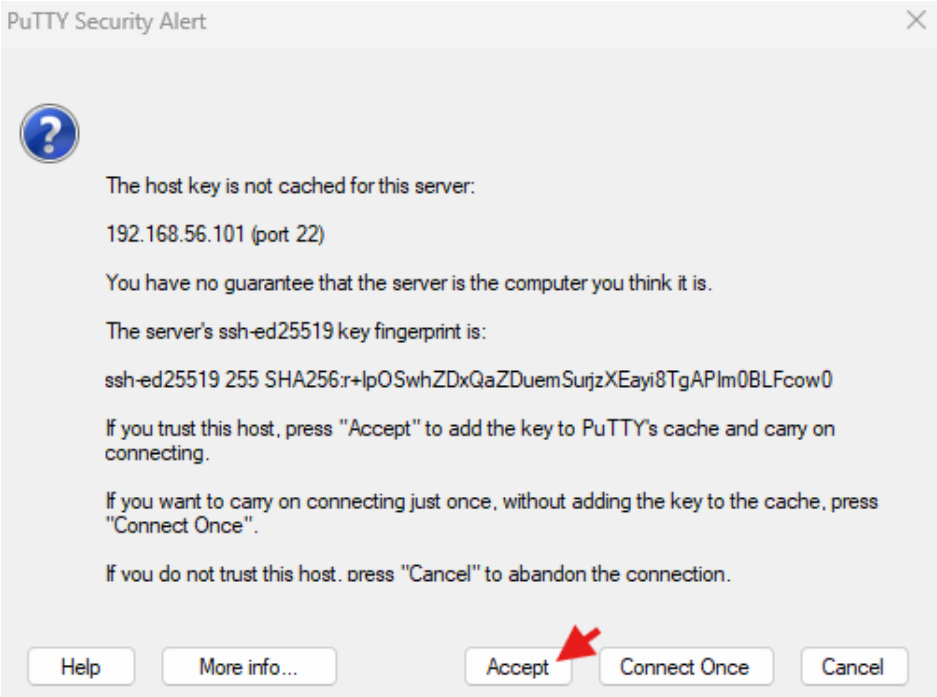
Ensuite, nous sélectionnons la clé privée du PC hôte dans “Credentials” qui se trouve dans l’onglet “Connection”, “SSH”, puis “Auth”:



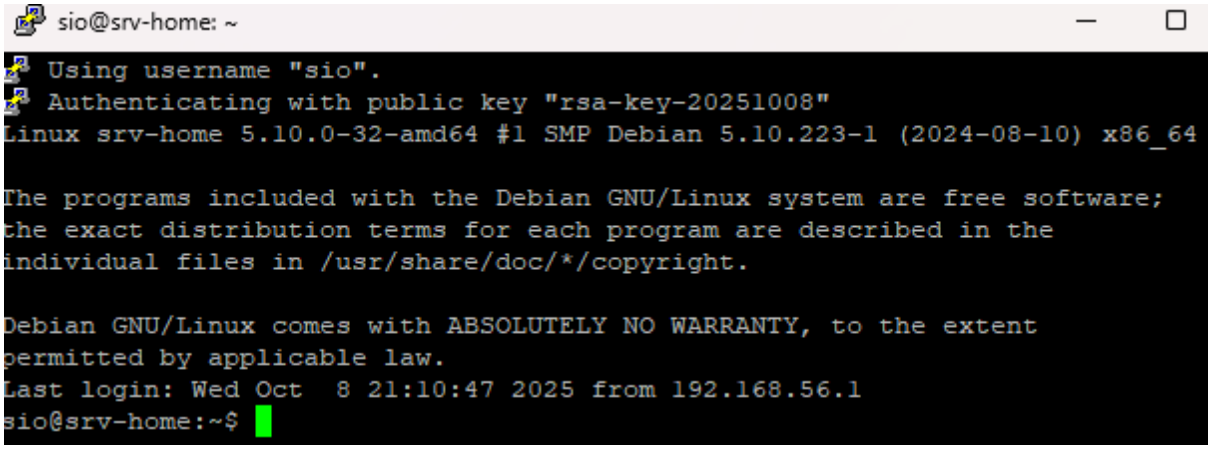
Enfin, dans l’onglet “Data” qui se trouve dans “Connection”, nous choisissons l’utilisateur auquel nous souhaitons nous connecter, dans notre cas, *sio,* et cliquons sur “Open”:



Après cela, un message s'affiche nous demandant si nous faisons confiance à *srv-home*, nous cliquons sur "Accept":



La connexion SSH se fait alors sans avoir besoin d'entrer de mot de passe:



4.4. Remarques

Il faut noter que les étapes 1 à 5 de la partie 4.3.3 peuvent être réalisées simplement en se connectant une première fois en SSH avec le mot de passe de l'utilisateur *sio*, via Powershell. Il est nécessaire de copier la clé publique de la machine Windows dans le fichier *authorized_keys*, grâce à la commande **nano /home/sio/.ssh/authorized_keys**: