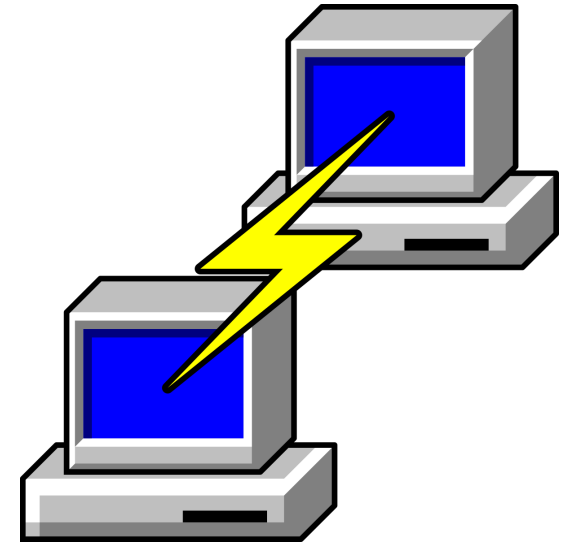
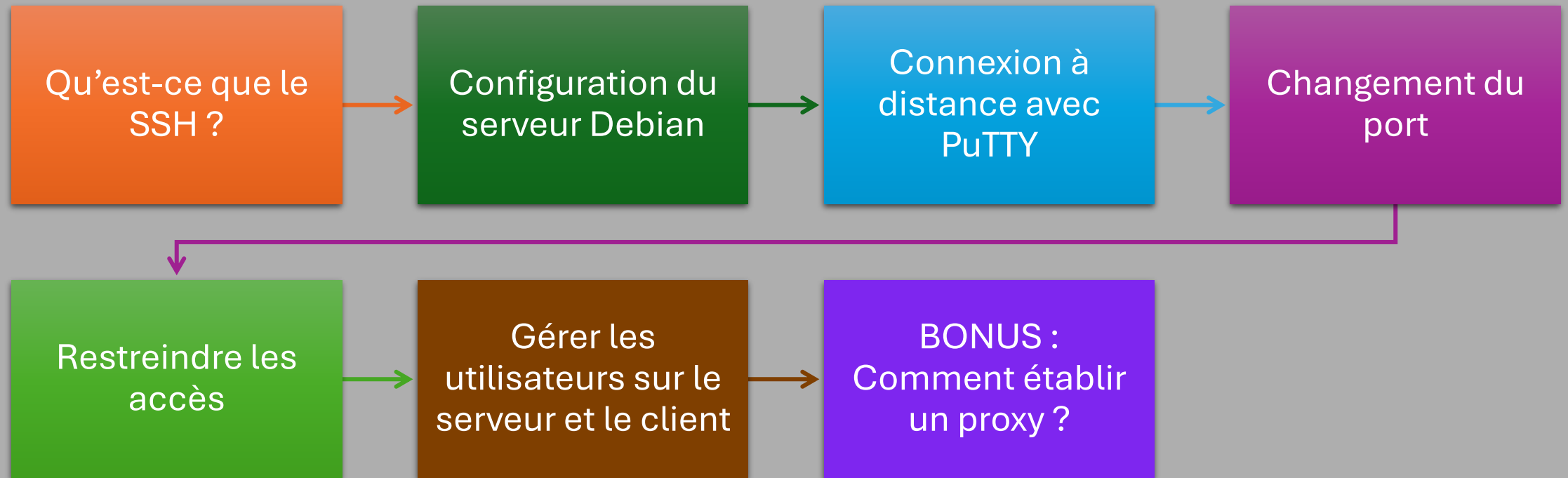




TP – SSH



Sommaire



Qu'est-ce que le SSH ?

Le **SSH**, ou Secure Shell, est un **protocole de communication sécurisé** qui permet de **chiffrer l'échange entre deux machines**.

Il est également utilisé pour **se connecter à distance** afin d'y exécuter des commandes ou de transférer des fichiers de manière sécurisée le rendant indispensable.

Par défaut, un serveur SSH écoute sur le **port TCP 22**.

Le SSH remplace le Telnet et le FTP.

De plus, **OpenSSH** est la solution la plus utilisée pour mettre en place une communication SSH.



Prérequis

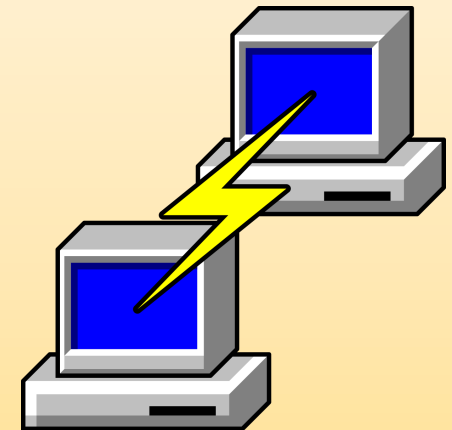
Les machines virtuelles seront en accès par pont.



Un serveur **DEBIAN**

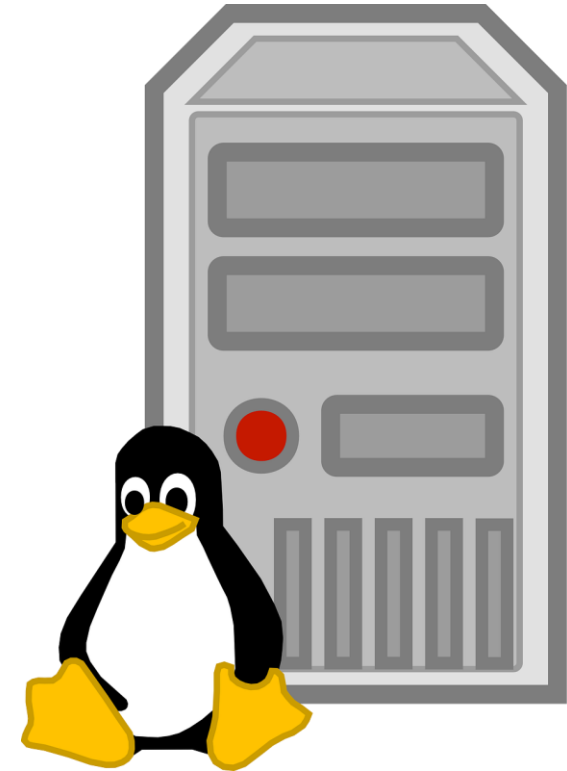


Un client **WINDOWS**



Un client **DEBIAN**
ou **PuTTY**

Configuration
du serveur
Debian



Installation du serveur sur **Debian**

```
root@Server-SSH:~# which ssh
/usr/bin/ssh

root@Server-SSH:~# apt update && apt install openssh-server -y
Atteint :1 http://deb.debian.org/debian bookworm InRelease
Réception de :2 http://deb.debian.org/debian bookworm-updates InRelease [55,4 kB]
Réception de :3 http://security.debian.org/debian-security bookworm-security InRelease [48,0 kB]
Réception de :4 http://security.debian.org/debian-security bookworm-security/main Sources [145 kB]
Réception de :5 http://security.debian.org/debian-security bookworm-security/main amd64 Packages [245 kB]
Réception de :6 http://security.debian.org/debian-security bookworm-security/main Translation-en [146 kB]
639 ko réceptionnés en 1s (588 ko/s)
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
5 paquets peuvent être mis à jour. Exécutez « apt list --upgradable » pour les voir.
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
openssh-server est déjà la version la plus récente (1:9.2p1-2+deb12u4).
openssh-server passé en « installé manuellement ».
0 mis à jour, 0 nouvellement installés, 0 à enlever et 5 non mis à jour.
root@Server-SSH:~#
```

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
    address 192.168.1.100
    netmask 255.255.255.0
```

Sur votre machine serveur **Debian**, tapez la commande « **which ssh** », si elle vous renvoie un chemin alors le SSH est installé.

Autrement, si vous n'avez pas le SSH d'installer alors exécutez la commande suivante :
apt install openssh-server -y.

On adressera une IP statique pour le serveur également.

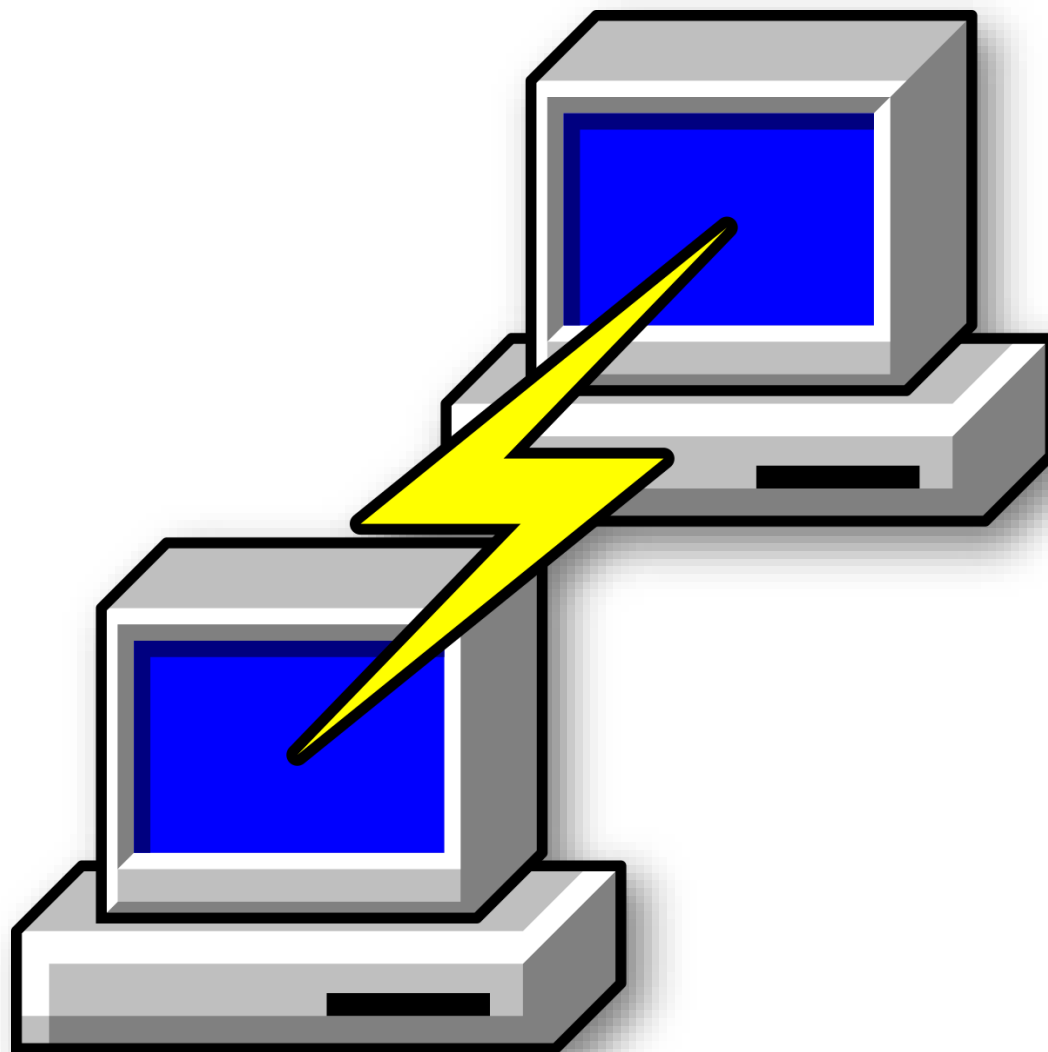
Ajout d'un utilisateur

```
root@Server-SSH:~# adduser jimmy
Ajout de l'utilisateur « jimmy » ...
Ajout du nouveau groupe « jimmy » (1001) ...
Ajout du nouvel utilisateur « jimmy » (1001) avec le groupe « jimmy » (1001) ...
Création du répertoire personnel « /home/jimmy » ...
Copie des fichiers depuis « /etc/skel » ...
Nouveau mot de passe :
Retapez le nouveau mot de passe :
passwd : mot de passe mis à jour avec succès
Modifier les informations associées à un utilisateur pour jimmy
Entrer la nouvelle valeur, ou appuyer sur ENTER pour la valeur par défaut
    NOM []: TRIQUX Jimmy
    Numéro de chambre []:
    Téléphone professionnel []:
    Téléphone personnel []:
    Autre []:
Cette information est-elle correcte ? [O/n]
Ajout du nouvel utilisateur « jimmy » aux groupes supplémentaires « users » ...
Ajout de l'utilisateur « jimmy » au groupe « users » ...
```

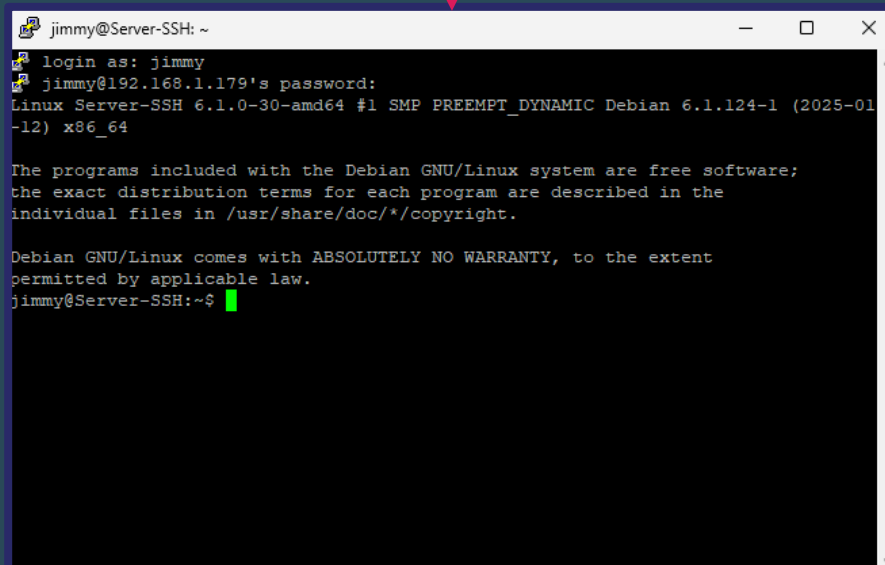
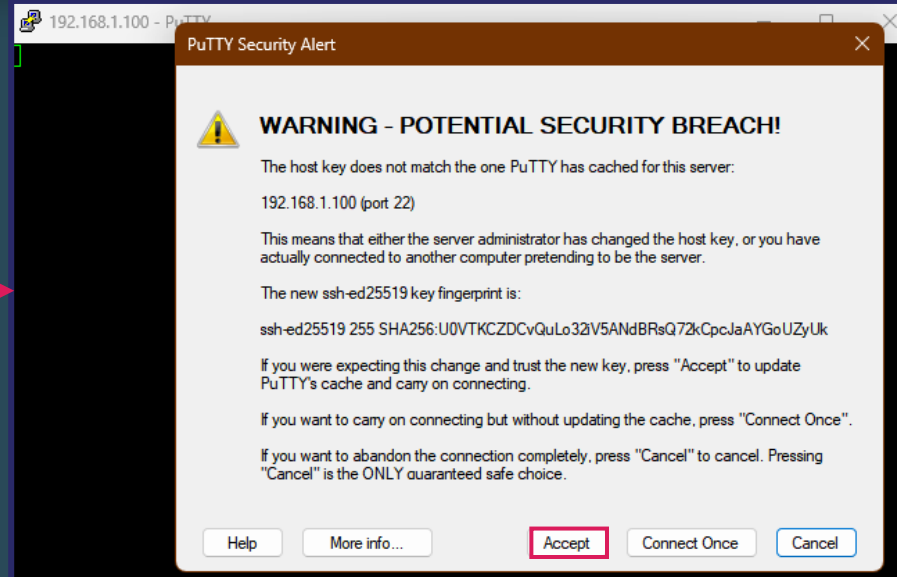
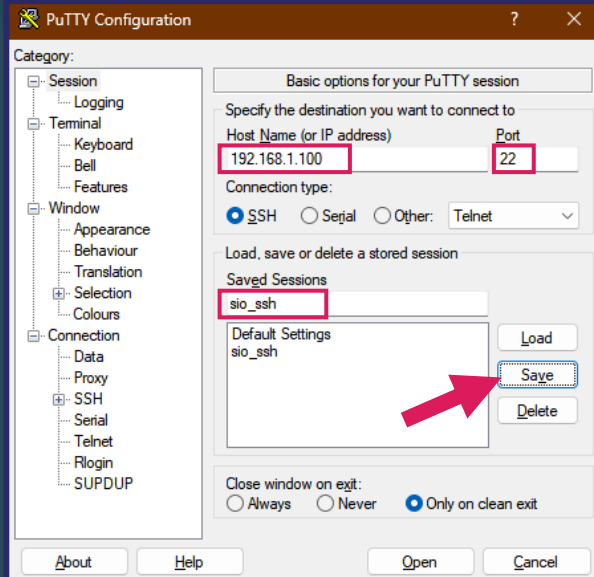
Pour se connecter à distance avec le protocole SSH, nous aurons besoin d'un compte utilisateur. Soit, vous reprenez l'utilisateur standard lors de votre installation ou alors créez en un.

Commande : adduser {nom_utilisateur}

Connexion à
distance avec
PuTTY sur
Windows



Test de la connexion SSH avec le client PuTTY sur Windows

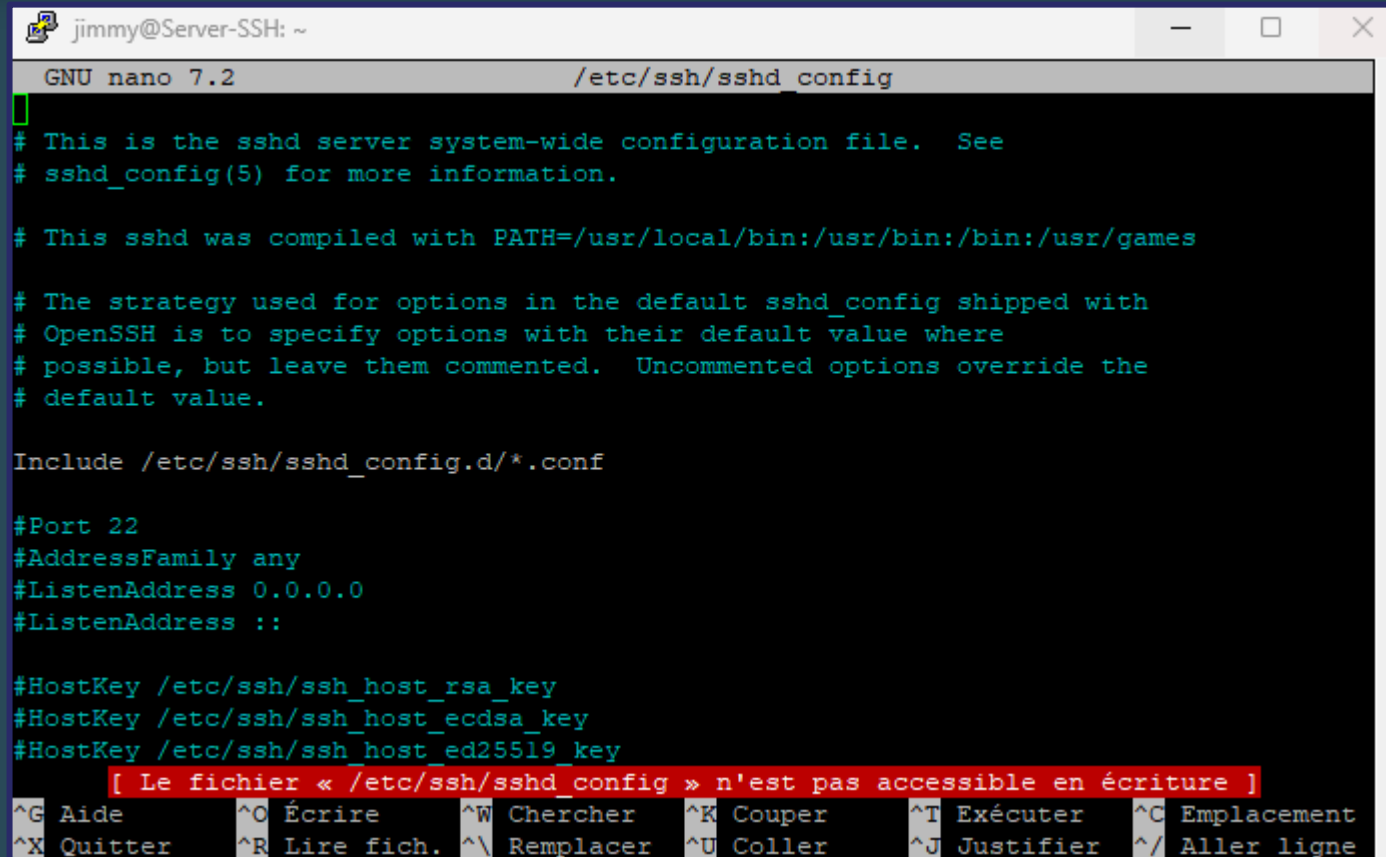


Dans **PuTTY**, rentrez l'adresse IP de votre serveur, vérifiez que vous avez bien coché **SSH** et que le **port 22** est affecté.

Avant de lancer, nous sauvegarderons les paramètres en le nommant « **sio_ssh** », cliquez sur « **Save** » et « **Open** ».
Ensuite, on vous demandera si vous souhaitez faire confiance à l'hôte, cliquez sur « **Accept** ».



Accéder au fichier « sshd_config »



```
jimmy@Server-SSH: ~
GNU nano 7.2 /etc/ssh/sshd_config
# This is the sshd server system-wide configuration file.  See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin:/bin:/usr/games

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

[ Le fichier « /etc/ssh/sshd_config » n'est pas accessible en écriture ]

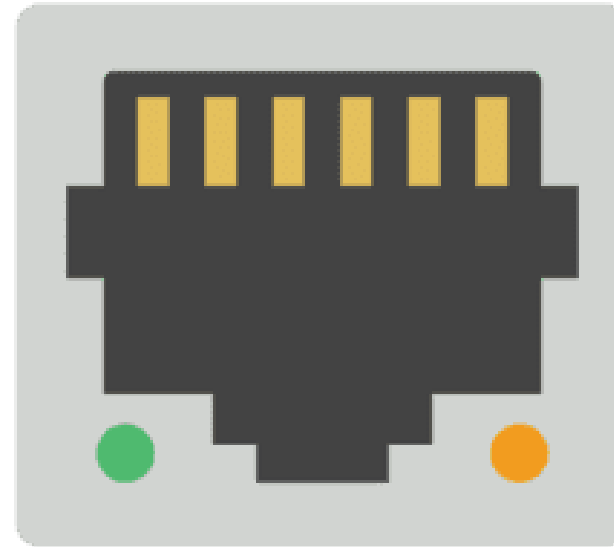
^G Aide      ^O Écrire    ^W Chercher  ^K Couper    ^T Exécuter  ^C Emplacement
^X Quitter   ^R Lire fich.^_ Remplacer  ^U Coller    ^J Justifier ^/ Aller ligne
```

Il est possible d'accéder au fichier « `/etc/ssh/sshd_config` » en étant simple utilisateur, néanmoins, l'utilisateur ne peut pas modifier le fichier.

Il lui faudrait des droits **root** pour modifier le fichier.



**Changement
du port**



Changer le port d'écoute du SSH

```
# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

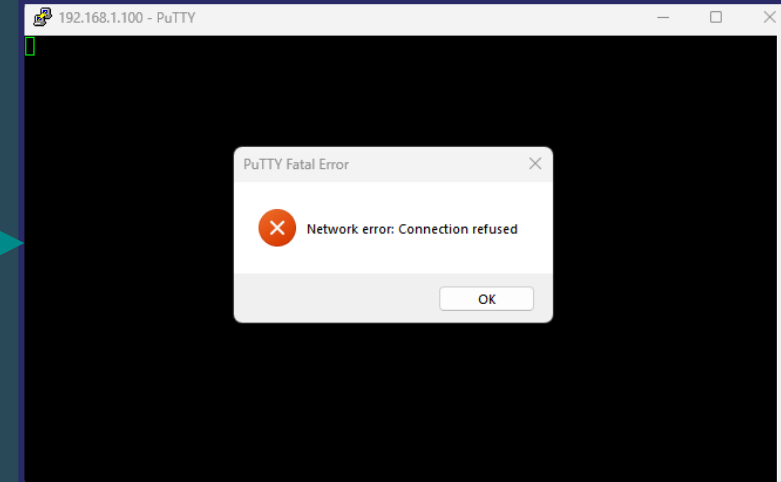
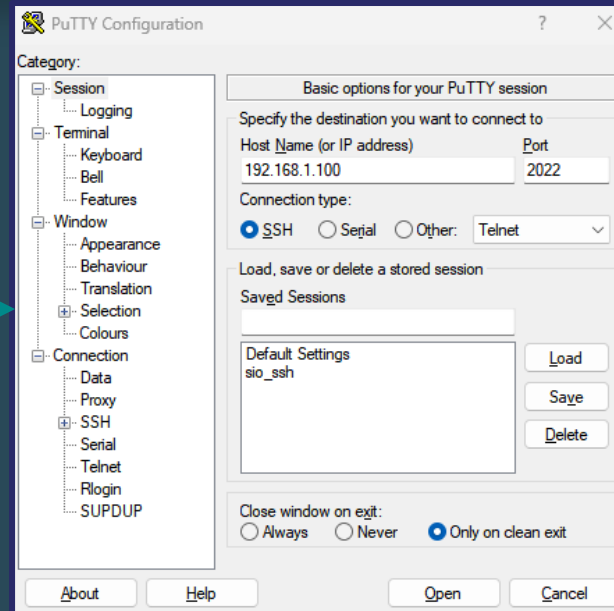
# This sshd was compiled with PATH=/usr/local/bin:/usr/bin:/bin:/usr/games

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

Port 2022_
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key
```



Comme vu précédemment, il est nécessaire d'être **root** afin de pouvoir modifier le fichier.

Retournez dans le fichier « **sshd_config** » depuis le **serveur Debian** et modifiez le port en dénotant le « **#** » et choisissez une nouvelle valeur pour le port comme **2022** par exemple.

Ensuite, en retournant sur le client **PuTTY**, re-rentrez l'adresse IP du serveur et mettez cette fois-ci « **2022** » à la place de « **22** » (ou bien du port que vous avez affecté) et cliquez sur « **Open** ».

Nous remarquons que la connexion n'est pas possible, nous recevons un message d'erreur.

Comment établir une connexion au serveur avec le changement de port ?

```
root@Server-SSH:~#  
root@Server-SSH:~# systemctl status ssh  
• ssh.service - OpenBSD Secure Shell server  
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)  
   Active: active (running) since Mon 2025-02-17 20:13:52 CET; 1min 52s ago  
     Docs: man:sshd(8)  
           man:sshd_config(5)  
  Process: 650 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)  
 Main PID: 651 (sshd)  
    Tasks: 1 (limit: 2306)  
   Memory: 1.4M  
      CPU: 20ms  
   CGroup: /system.slice/ssh.service  
           └─651 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"  
  
févr. 17 20:13:52 Server-SSH systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...  
févr. 17 20:13:52 Server-SSH sshd[651]: Server listening on 0.0.0.0 port 22.   
févr. 17 20:13:52 Server-SSH sshd[651]: Server listening on :: port 22.  
févr. 17 20:13:52 Server-SSH systemd[1]: Started ssh.service - OpenBSD Secure Shell server.  
root@Server-SSH:~#  
root@Server-SSH:~# service ssh restart  
root@Server-SSH:~#  
root@Server-SSH:~# systemctl status ssh  
• ssh.service - OpenBSD Secure Shell server  
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)  
   Active: active (running) since Mon 2025-02-17 20:15:48 CET; 2s ago  
     Docs: man:sshd(8)  
           man:sshd_config(5)  
  Process: 670 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)  
 Main PID: 671 (sshd)  
    Tasks: 1 (limit: 2306)  
   Memory: 1.4M  
      CPU: 20ms  
   CGroup: /system.slice/ssh.service  
           └─671 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"  
  
févr. 17 20:15:48 Server-SSH systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...  
févr. 17 20:15:48 Server-SSH sshd[671]: Server listening on 0.0.0.0 port 2022.   
févr. 17 20:15:48 Server-SSH sshd[671]: Server listening on :: port 2022.  
févr. 17 20:15:48 Server-SSH systemd[1]: Started ssh.service - OpenBSD Secure Shell server.  
root@Server-SSH:~#
```

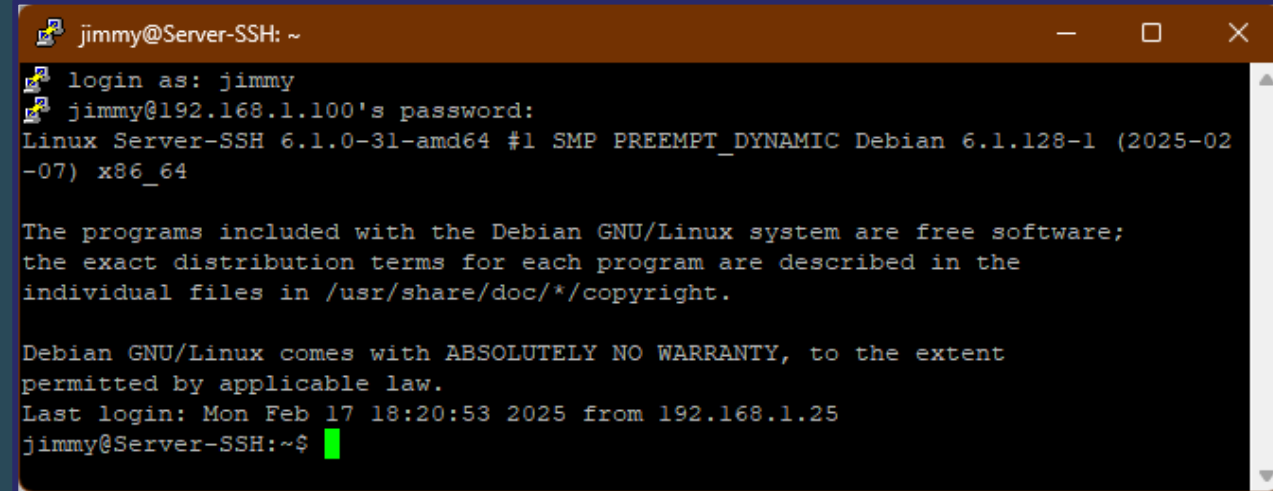
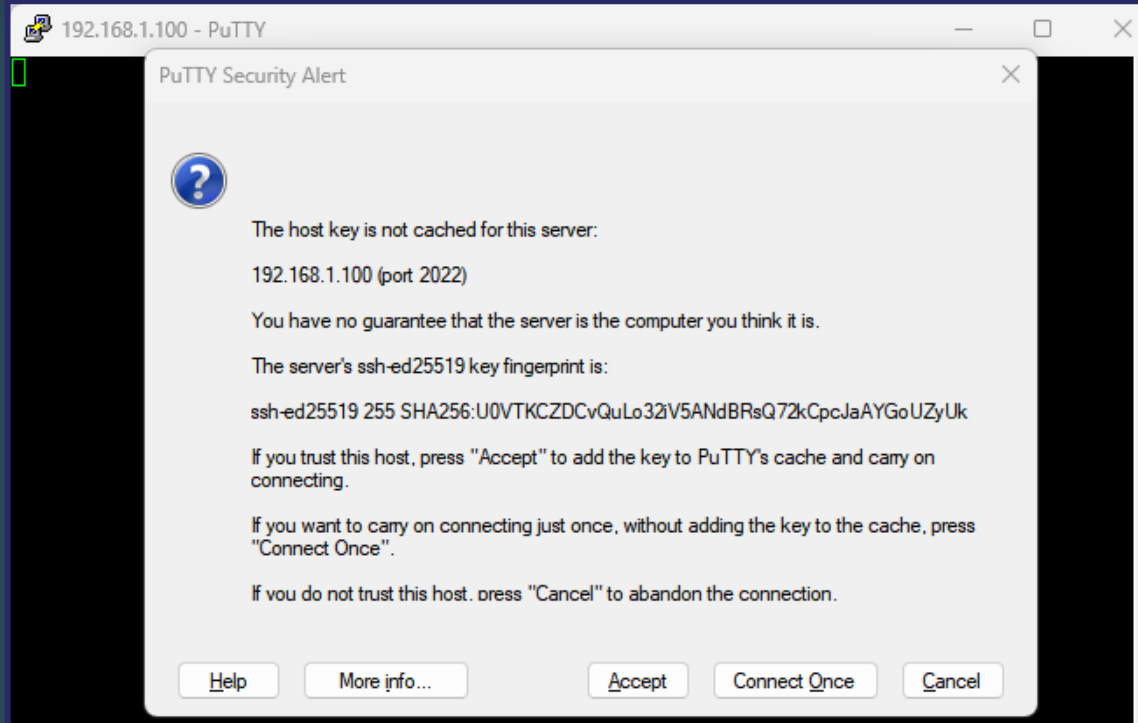
Afin de pouvoir établir une connexion au serveur, il est important de redémarrer le service SSH.

En effet, simplement enregistrer et quitter le fichier de configuration n'est pas suffisant comme nous pouvons le voir sur l'image avec la commande « **systemctl status ssh** » nous pouvons voir que le port d'écoute est toujours le **port 22**.

Pour remédier à cela, nous exécuterons la commande « **service ssh restart** ».

En réexécutant la commande de statut du service SSH, nous apercevons désormais que le port d'écoute est bien le **2022** comme indiqué dans le fichier **sshd_config**.

Re-tentative de connexion



Lors de la re-tentative de connexion, on vous redemandera d'accepter une clé puisque nous avons changé de port.

En acceptant, nous pouvons de nouveau accéder à notre serveur en **SSH** avec un port différent de celui imposé par défaut.

Quel est l'intérêt d'un changement de port ?



Changer le port SSH, qui est par défaut **22**, est une pratique courante pour renforcer sa sécurité.

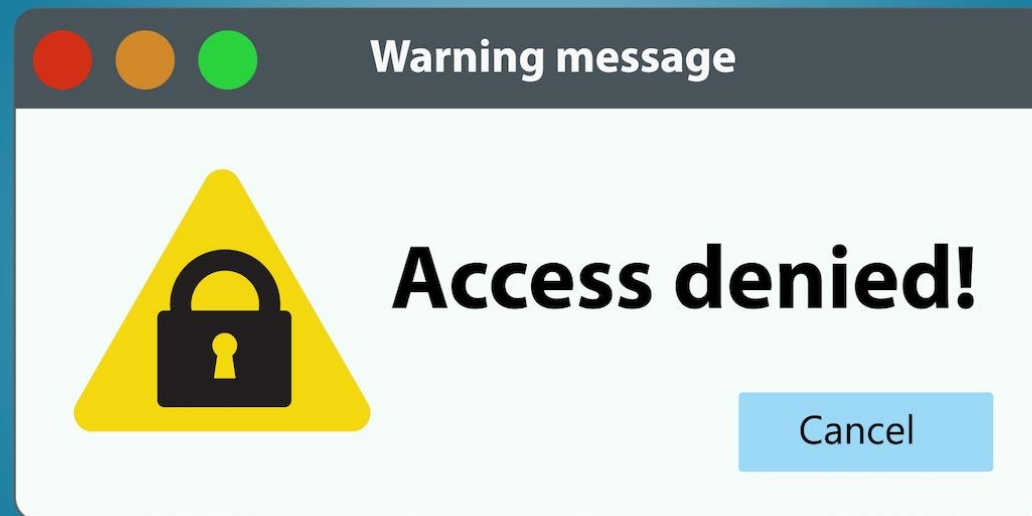
Dans le cadre d'une tentative de **bruteforce** par un cyberattaquant, il est préférable d'avoir un port personnalisé afin d'éviter les attaques basiques qui viseront le port par défaut du SSH.

Le changement de port est également efficace lorsqu'il est couplé à d'autres bonnes pratiques comme :

- Authentification par clé SSH uniquement (**PasswordAuthentication no**) ;
- Utiliser **fail2ban** pour bloquer les IP après des échecs répétés.

Néanmoins, un scan complet de l'adresse IP du serveur avec un outil comme **nmap** révélera le port personnalisé. Il est alors recommandé d'utiliser un pare-feu afin de restreindre l'accès au port SSH uniquement à des **IP autorisées**.

**Restreindre
les accès**



Retour dans le fichier « sshd_config »

```
# Authentication:  
  
#LoginGraceTime 2m  
#PermitRootLogin prohibit-password  
#StrictModes yes  
#MaxAuthTries 6  
#MaxSessions 10
```



La ligne « **PermitRootLogin** » permet d'autoriser ou non la connexion au serveur SSH avec le login **root**.

Les différentes valeurs possibles de **PermitRootLogin** sont :

- **yes** : Accès autorisé avec mot de passe ou clé SSH ;
- **no** : Accès **root** interdit ;
- **prohibit-password** (ou **without-password**) : Accès **root** uniquement par clé SSH ;
- **forced-commands-only** : Accès **root** uniquement pour exécuter des commandes spécifiques (scripts).

PermitEmptyPasswords

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

La ligne « **PermitEmptyPasswords** » contrôle si les utilisateurs peuvent se connecter avec un **mot de passe vide**. On renseigne notre choix avec la valeur **no** et la valeur **yes**.

Par défaut, cette option est définie sur **no**, ce qui est une bonne pratique de sécurité et également un prérequis dans un environnement professionnel afin d'éviter toute erreur interne possible.

De plus, c'est obligatoire pour se prévenir des attaques extérieures de personnes malveillantes car autrement il pourrait se connecter instantanément au serveur.

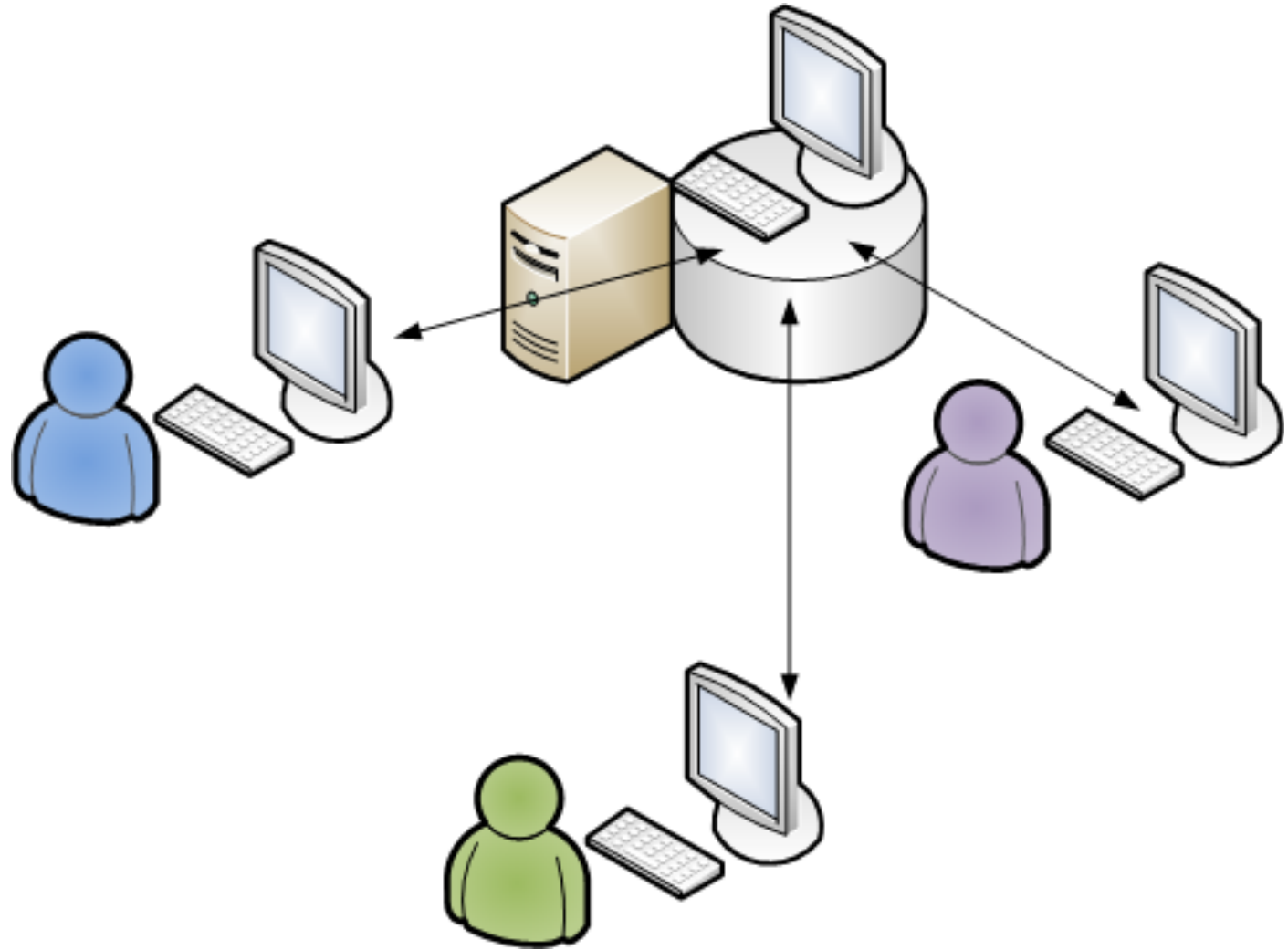
La différence entre ces deux commandes

	PermitRootLogin without-password	PermitEmptyPasswords no
Objectif	Contrôle les connexions avec mot de passe vide.	Contrôle l'accès root via SSH.
Portée	S'applique à tous les utilisateurs .	S'applique à tous les utilisateurs root .
Impact sur la sécurité	Bloque les connexions non sécurisées (mot de passe vide).	Restreint l'accès root (mot de passe ou clé SSH).
Exemple	Un utilisateur avec un mot de passe vide ne peut pas se connecter.	L'utilisateur root ne peut pas se connecter avec un mot de passe, uniquement avec une clé SSH.

En somme, **PermitEmptyPasswords no** sécurise tous les utilisateurs contre les mots de passe vides, tandis que **PermitRootLogin without-password** protège spécifiquement le compte **root**.

Ensemble, elles renforcent la sécurité SSH.

**Gérer les
utilisateurs
sur le
serveur et le
client**



Création des groupes et utilisateurs

```
root@Server-SSH:~# groupadd etudiant
root@Server-SSH:~#
root@Server-SSH:~# groupadd ssh
root@Server-SSH:~#
root@Server-SSH:~# useradd -m -g etudiant -G ssh user1
root@Server-SSH:~#
root@Server-SSH:~# useradd -m -g ssh user2
root@Server-SSH:~#
root@Server-SSH:~# useradd -m -g etudiant user3
```

Sur le serveur **Debian**, ajoutons dans un premier temps, les groupes « **etudiant** » et « **ssh** ».

Ensuite, avec la commande « **useradd** », nous pourrions créer des utilisateurs avec leur répertoire personnel (**-m**), leur groupe principal (**-g**) et secondaire (**-G**) en une seule commande. Répétez pour les 2 autres avec les groupes correspondants.

```
root@Server-SSH:~# chpasswd
user1:*Sio2020
user2:*Sio2020
user3:*Sio2020
```

Après avoir créé les trois utilisateurs, nous devons leur attribuer des mots de passe.

Exécutez la commande « **chpasswd** », renseignez l'identifiant et rentrez un mot de passe séparé par « **:** », réitérez pour les 2 autres utilisateurs puis faites « **Ctrl+D** » afin de quitter.

Création du répertoire « .ssh » pour chaque utilisateur

192.168.1.100 - PuTTY

user1

```
login as: user1
user1@192.168.1.100's password:
Linux Server-SSH 6.1.0-31-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.128-1 (2025-02-07) 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: Tue Feb 18 00:05:30 2025 from 192.168.1.24
$ mkdir /home/user1/.ssh
```

192.168.1.100 - PuTTY

user2

```
login as: user2
user2@192.168.1.100's password:
Linux Server-SSH 6.1.0-31-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.128-1 (2025-02-07) 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.
$ mkdir /home/user2/.ssh
```

192.168.1.100 - PuTTY

user3

```
login as: user3
user3@192.168.1.100's password:
Linux Server-SSH 6.1.0-31-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.128-1 (2025-02-07) 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.
$ mkdir /home/user3/.ssh
```

Serveur **Debian**

```
root@Server-SSH:~# mkdir -p /root/.ssh
root@Server-SSH:~#
root@Server-SSH:~# chmod 0770 /root/.ssh
root@Server-SSH:~#
root@Server-SSH:~# chmod 0770 /home/user1/.ssh
root@Server-SSH:~#
root@Server-SSH:~# chmod 0770 /home/user2/.ssh
root@Server-SSH:~#
root@Server-SSH:~# chmod 0770 /home/user3/.ssh
root@Server-SSH:~#
```

Sur les clients de chacun de vos utilisateurs, créez un répertoire « **/home/{user}/.ssh** »

Voir

TP SAMBA

concernant les permissions chmod.

Ici, nous donnons les permissions de contrôle total pour l'utilisateur et son groupe sur son répertoire « **.ssh** » mais les autres utilisateurs n'ont aucun accès.

Générer une clé DSA

```
192.168.1.100 - PuTTY
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Feb 18 00:07:45 2025 from 192.168.1.25
$ ssh-keygen -t dsa -f ~/.ssh/id_dsa
Generating public/private dsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user1/.ssh/id_dsa
Your public key has been saved in /home/user1/.ssh/id_dsa.pub
The key fingerprint is:
SHA256:bsXSzMXyhrjODGuYvv25c8yuuqLHHcgM1cA9gZqAY2o user1@Server-SSH
The key's randomart image is:
+---[DSA 1024]-----+
|oo..          |
|*oo   . .     |
|=oo.   . . o   |
|=E .   . @     |
|o+ . . . S O   |
|  +o.* . o     |
| .o.o.+oo      |
| .o. .o+       |
|.ooo+==*       |
+---[SHA256]-----+
$
```

```
192.168.1.100 - PuTTY
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Feb 18 00:11:01 2025 from 192.168.1.25
$ ssh-keygen -t dsa -f ~/.ssh/id_dsa
Generating public/private dsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user2/.ssh/id_dsa
Your public key has been saved in /home/user2/.ssh/id_dsa.pub
The key fingerprint is:
SHA256:fcMwBhPtxGtuZel2npglflWTB56uyOullj09Q3m2g0 user2@Server-SSH
The key's randomart image is:
+---[DSA 1024]-----+
| . . +o |
| * . o...|
| . + .o.+|
| . =ooo=B|
| S . *++X|
| ..oE+*|
| .+B+.|
| .=*.|
| .+o+ .|
+---[SHA256]-----+
$
```

```
192.168.1.100 - PuTTY
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Feb 18 00:11:40 2025 from 192.168.1.25
$ ssh-keygen -t dsa -f ~/.ssh/id_dsa
Generating public/private dsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user3/.ssh/id_dsa
Your public key has been saved in /home/user3/.ssh/id_dsa.pub
The key fingerprint is:
SHA256:dOGhcxfCOc0SknZv//RxZGx972EsEOBzkOaeLE/Hpbs user3@Server-SSH
The key's randomart image is:
+---[DSA 1024]-----+
| .B= . |
| +*B=o. |
| .*++=o ..|
| . =ooo. B|
| S o.+..+o|
| . = + ..+=|
| + . . o+=|
| . . . +|
| . E. |
+---[SHA256]-----+
$
```

Sur chaque client PuTTY pour les 3 utilisateurs, nous générons une clé DSA avec la commande « **ssh-keygen -t dsa -f ~/.ssh/id_dsa** » qui permet de générer à la fois une clé privée et une clé publique.

Si, vous recevez un message du type « **Permission denied** » ou « **Connection refused** », retournez dans votre fichier « **sshd_config** » et dénoter la ligne **PermitRootLogin** yes.

Il est également possible de le faire sur un **client Linux**.

Envoyer la clé publique au serveur

```
192.168.1.100 - PuTTY

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: Thu Feb 27 20:24:17 2025 from 192.168.1.25
root@192.168.1.100:~# ssh-copy-id -i ~/.ssh/id_dsa.pub root@192.168.1.100
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/user2/.ssh/
id_dsa.pub"
The authenticity of host '192.168.1.100 (192.168.1.100)' can't be established.
ED25519 key fingerprint is SHA256:U0VTKCZDCvQuLo32iV5ANdBRsQ72kCpcJaAYGoUZYUk.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompt
ed now it is to install the new keys
root@192.168.1.100's password:

Number of key(s) added: 1

Now try logging into the machine, with:  "ssh 'root@192.168.1.100'"
and check to make sure that only the key(s) you wanted were added.
```

```
$ cat ~/.ssh/authorized_keys
ssh-dss AAAAB3NzaC1kc3MAAACBAJrepqs9nnaHMxpyk7Y6cFggNR2ML18MMmlxuC/QRaA/pv+CvU9n
ZTb23Lr57EKoYJlYC5Mc2C9w4vXxFNczMhR6pUG/7zcxvdcDVTd2A/7Y0m6M2mhoG2Lr/oLo6eWU+EX1
cMd+bmYS/hZ46Wr8jJMrmQC1/rkH5WrWjOkNsIxhAAAAFQDj4iveI/t/sSuf8SsH+mQYSleAowAAAIAY
22+TfdYX9Yo7VGqiZw52SpZ1LZjk+x2T21q0bg7ZCsCvssMC8Y3bo6cW4u8g2xAPiLXFrBolXCSxuFY3
SHuVqwss5sLlK6STo6oAKljAEusHgRjcKtV9vCnLGtZYiXTk4ojQtqaEaVQ831AjG/788BH2H5B3FV6b
Qtk7ulVMjwAAAIAG3f3x8DTmWKjaPb4VQcMrvus6ttutYBBZkGq5z+N6J3ZzpuGtKN5y3Bosd67iL37z
aMkrXFlkJRYsGGRcfHK3dL6yDfj3mEv/uYJKzOW+SXD2CPRzNCP8bu55aJoyxLiXlYJR0G7IR3jOSqe
5GghEIA/KkVS8kHDZh6CuukAbA== user1@Server-SSH
$
```

Une fois que les clés sont générées, on envoie la clé publique de chaque utilisateur au serveur avec la commande « **ssh-copy-id -i ~/.ssh/id_dsa.pub user@votre_serveur** ».

La clé publique est envoyée ensuite dans le répertoire « **~/.ssh/authorized_keys** ».

À quoi font références les deux fichiers générés ?

```
root@Server-SSH:~# ssh user1@192.168.1.100 -p 22
user1@192.168.1.100's password:
Linux Server-SSH 6.1.0-30-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.124-1 (2025-01-12) 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.

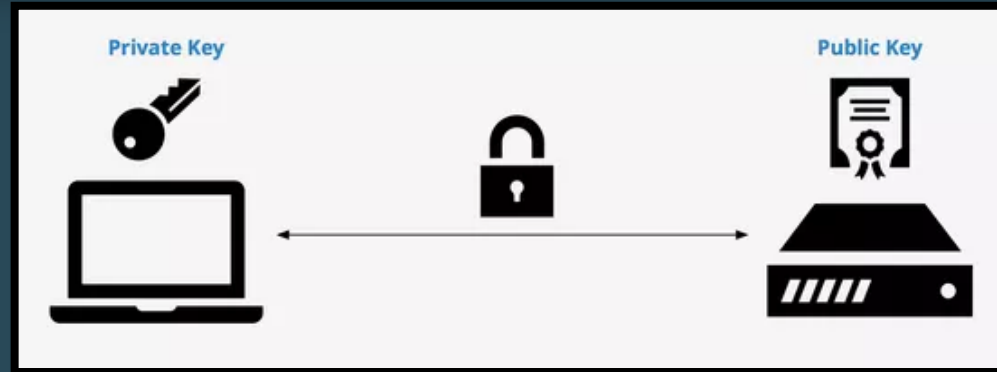
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Mar  4 14:16:32 2025 from 192.168.1.100
$ cd ~/.ssh
$ ls -la
total 28
drwxrwx--- 2 user1 ssh 4096  4 mars  14:14 .
drwxr-xr-x 3 user1 ssh 4096  4 mars  14:12 ..
-rw----- 1 user1 ssh  606  4 mars  14:14 authorized_keys
-rw----- 1 user1 ssh 1438  4 mars  14:14 id_dsa
-rw-r--r-- 1 user1 ssh  606  4 mars  14:14 id_dsa.pub
-rw----- 1 user1 ssh  978  4 mars  14:14 known_hosts
-rw----- 1 user1 ssh  142  4 mars  14:14 known_hosts.old
$ _
```

En se connectant avec **user1**, nous pouvons remarquer deux fichiers « **id_dsa** » qui est la clé privée et « **id_dsa.pub** » qui est la clé publique.

La différence réside dans la longueur de la chaîne de caractères qui est plus courte pour la clé publique suivit d'un commentaire comme le nom d'utilisateur et l'hôte du serveur.

Cela est tout à fait normal puisque la **clé privée est utilisée pour déchiffrer les données** et donc doit rester secrète (ce pourquoi elle est plus longue et complexe). Tandis que la **clé publique est utilisée pour chiffrer les données** et peut être partagée librement (ce pourquoi elle est courte et contient des informations supplémentaires comme l'utilisateur et la machine hôte).

Pourquoi ?



Cette différence s'explique pour des raisons de sécurité. En effet, la clé privée doit être protégée, car c'est celle-ci qui permet d'accéder au système à distance.

Dans le fonctionnement du **SSH**, le serveur SSH utilise la clé publique pour authentifier l'utilisateur, alors que le client utilise la clé privée pour prouver son identité.

Donc, si la clé privée est compromise alors un cyberattaquant pourra accéder au système (d'où l'intérêt d'affecter des permissions strictes et un mot de passe fort) tandis que la clé publique ne pose aucun risque de sécurité si elle est divulguée.

Test de connexion

```
root@Server-SSH:~# ssh user2@192.168.1.100 -p 22
user2@192.168.1.100's password:
Linux Server-SSH 6.1.0-30-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.124-1 (2025-01-12) 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: Tue Mar  4 14:25:44 2025 from 192.168.1.100
$ cd ~/.ssh
$ ls -la
total 28
drwxrwx--- 2 user2 user2 4096  4 mars  14:24 .
drwxr-xr-x 3 user2 user2 4096  4 mars  14:13 ..
-rw----- 1 user2 user2  606  4 mars  14:24 authorized_keys
-rw----- 1 user2 user2 1434  4 mars  14:24 id_dsa
-rw-r--r-- 1 user2 user2  606  4 mars  14:24 id_dsa.pub
-rw----- 1 user2 user2  978  4 mars  14:24 known_hosts
-rw----- 1 user2 user2  142  4 mars  14:24 known_hosts.old
$ exit
Connection to 192.168.1.100 closed.
root@Server-SSH:~#
root@Server-SSH:~# ssh user3@192.168.1.100 -p 22
user3@192.168.1.100's password:
Linux Server-SSH 6.1.0-30-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.124-1 (2025-01-12) 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: Tue Mar  4 14:13:33 2025 from 192.168.1.2
$ cd ~/.ssh
$ ls -la
total 28
drwxrwx--- 2 user3 user3 4096  4 mars  14:25 .
drwxr-xr-x 3 user3 user3 4096  4 mars  14:13 ..
-rw----- 1 user3 user3  606  4 mars  14:25 authorized_keys
-rw----- 1 user3 user3 1434  4 mars  14:25 id_dsa
-rw-r--r-- 1 user3 user3  606  4 mars  14:25 id_dsa.pub
-rw----- 1 user3 user3  978  4 mars  14:25 known_hosts
-rw----- 1 user3 user3  142  4 mars  14:25 known_hosts.old
$ _
```

Depuis la machine **Serveur**, nous testerons chaque utilisateur avec la commande « **user@lpserveur -p numéroport** » puis je liste chacun de leur répertoire « **/.ssh** » avec « **ls -la** ».

Je me connecte sur user2 et user3 et je liste leur répertoire « **.ssh** », nous remarquons effectivement la présence des fichiers **id_dsa** et **id_dsa.pub**.

Autoriser que certains groupes à avoir accès au SSH

```
# Authentication:

#LoginGraceTime 2m
#PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
AllowGroups ssh root

#PubkeyAuthentication yes

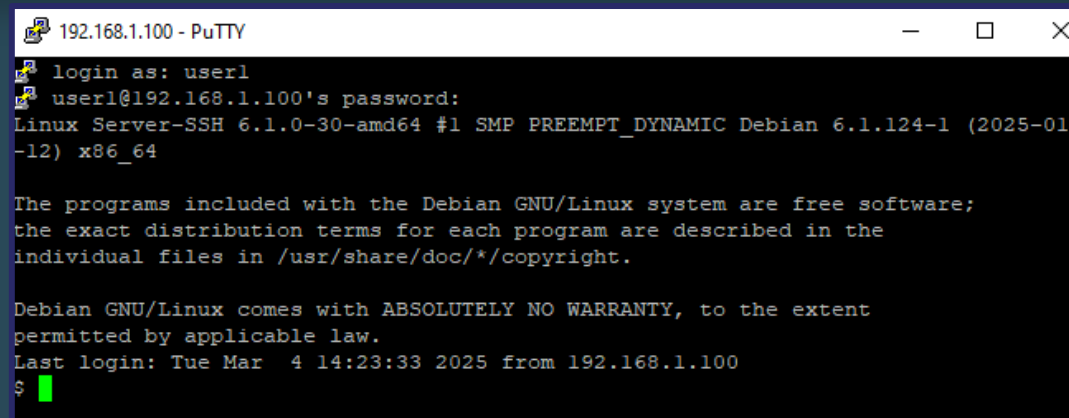
# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile .ssh/authorized_keys .ssh/authorized_keys2

#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 yes
#PermitEmptyPasswords no
```

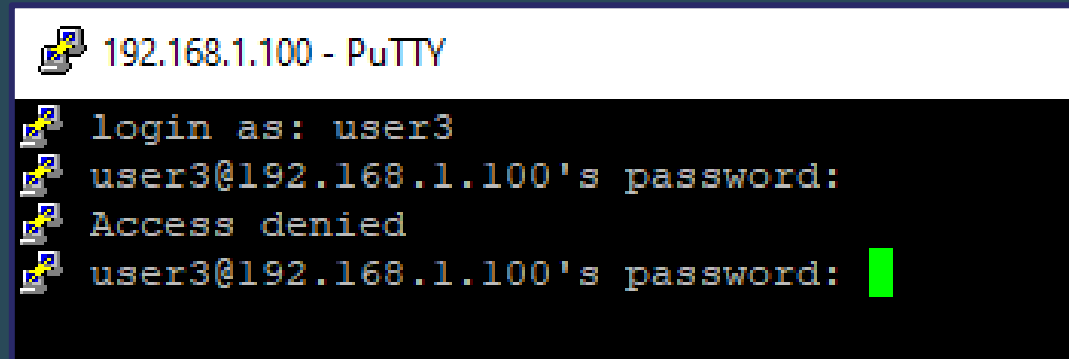


192.168.1.100 - PuTTY

```
login as: user1
user1@192.168.1.100's password:
Linux Server-SSH 6.1.0-30-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.124-1 (2025-01-12) 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: Tue Mar  4 14:23:33 2025 from 192.168.1.100
$
```



192.168.1.100 - PuTTY

```
login as: user3
user3@192.168.1.100's password:
Access denied
user3@192.168.1.100's password:
```

La commande « **AllowGroups {groupe1} {groupe2}** » permet d'autoriser uniquement les groupes assignés à pouvoir se connecter en **SSH**.

PasswordAuthentication yes permet aux utilisateurs de se connecter via un mot de passe et

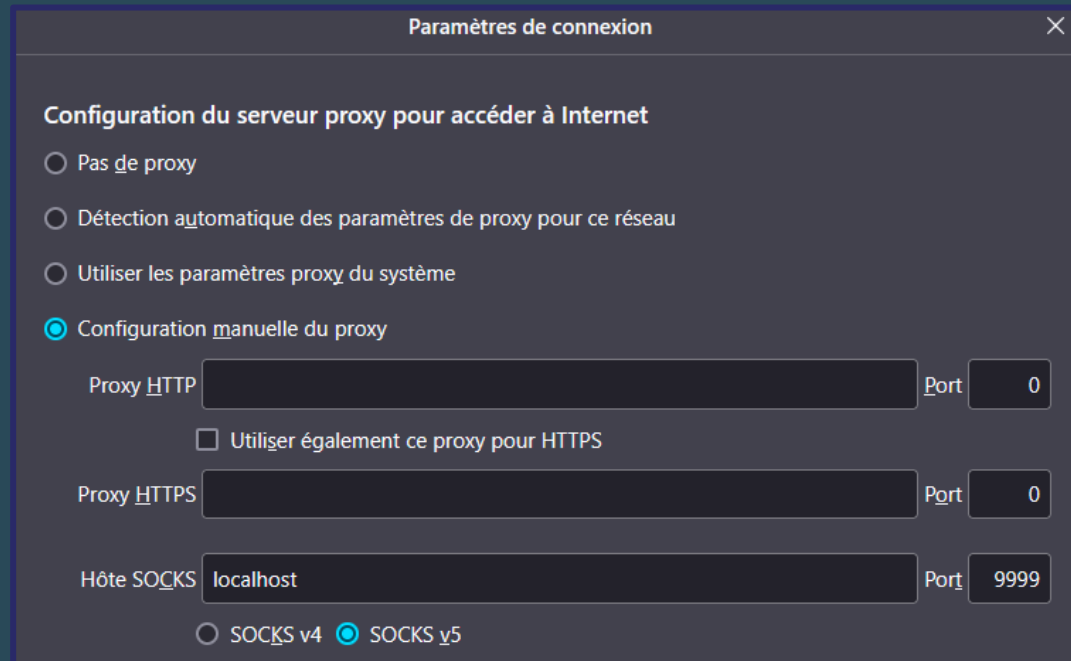
PasswordAuthentication no désactive l'authentification par mot de passe (seulement l'authentification par clé publique fonctionnera).

En se connectant avec **user3** qui n'est que dans le groupe **etudiant**, nous recevons le message « **Access denied** » car cet utilisateur n'est ni dans le groupe **ssh** ni **root** alors il ne peut plus se connecter en **SSH**.

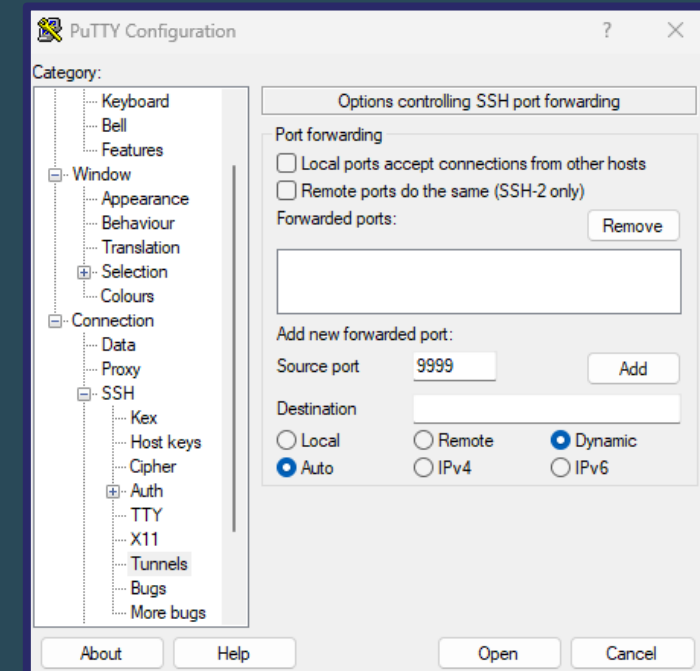
BONUS :
Comment
établir un
proxy ?



Configuration du client web et du client PuTTY



Firefox : Paramètre, Général, Paramètres réseau, Sélectionner « **Configuration manuelle du proxy** » puis rentrez « localhost » dans **Hôte SOCKS** et port **9999**.



Dans PuTTY, allez dans « **SSH** », « **Tunnels** » et rentrez dans source port **9999** et cochez la case **Dynamic**.

Capture de trame avec Wireshark

37800	590.165575	2001:861:20d3:ece0::...	2a02:ec80:600:ed1a::...	TCP	86	35820 → 443	[ACK]	Seq=11458	Ack=607185	Win=538368	Len=0	TSval=3520724589	TSecr=3888131889	
37801	590.165586	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=610041	Ack=11458	Win=42496	Len=1428	TSval=3888131890	TSecr=3520724572	[TCP PDU reassembled in 37811]
37802	590.165586	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=611469	Ack=11458	Win=42496	Len=1428	TSval=3888131890	TSecr=3520724572	[TCP PDU reassembled in 37811]
37803	590.165614	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=612897	Ack=11458	Win=42496	Len=1428	TSval=3888131890	TSecr=3520724572	[TCP PDU reassembled in 37811]
37804	590.165614	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=614325	Ack=11458	Win=42496	Len=1428	TSval=3888131890	TSecr=3520724572	[TCP PDU reassembled in 37811]
37805	590.165614	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=615753	Ack=11458	Win=42496	Len=1428	TSval=3888131890	TSecr=3520724572	[TCP PDU reassembled in 37811]
37806	590.165642	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=617181	Ack=11458	Win=42496	Len=1428	TSval=3888131890	TSecr=3520724572	[TCP PDU reassembled in 37811]
37807	590.165642	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=618609	Ack=11458	Win=42496	Len=1428	TSval=3888131890	TSecr=3520724572	[TCP PDU reassembled in 37811]
37808	590.165674	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[PSH, ACK]	Seq=620037	Ack=11458	Win=42496	Len=1428	TSval=3888131890	TSecr=3520724572	[TCP PDU reassembled in 37811]
37809	590.166191	2001:861:20d3:ece0::...	2a02:ec80:600:ed1a::...	TCP	86	35820 → 443	[ACK]	Seq=11458	Ack=621465	Win=538368	Len=0	TSval=3520724589	TSecr=3888131890	
37810	590.166399	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=621465	Ack=11458	Win=42496	Len=1428	TSval=3888131891	TSecr=3520724572	[TCP PDU reassembled in 37811]
37811	590.166399	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TLSv1.3	1090	Application Data								
37812	590.166557	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=623897	Ack=11458	Win=42496	Len=1428	TSval=3888131891	TSecr=3520724572	[TCP PDU reassembled in 37820]
37813	590.166557	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=625325	Ack=11458	Win=42496	Len=1428	TSval=3888131891	TSecr=3520724572	[TCP PDU reassembled in 37820]
37814	590.166584	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=626753	Ack=11458	Win=42496	Len=1428	TSval=3888131891	TSecr=3520724572	[TCP PDU reassembled in 37820]
37815	590.166584	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=628181	Ack=11458	Win=42496	Len=1428	TSval=3888131891	TSecr=3520724572	[TCP PDU reassembled in 37820]
37816	590.166584	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=629609	Ack=11458	Win=42496	Len=1428	TSval=3888131891	TSecr=3520724572	[TCP PDU reassembled in 37820]
37817	590.166623	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=631037	Ack=11458	Win=42496	Len=1428	TSval=3888131891	TSecr=3520724572	[TCP PDU reassembled in 37820]
37818	590.166623	2a02:ec80:600:ed1a::...	2001:861:20d3:ece0::...	TCP	1514	443 → 35820	[ACK]	Seq=632465	Ack=11458	Win=42496	Len=1428	TSval=3888131891	TSecr=3520724572	[TCP PDU reassembled in 37820]

Voici un exemple de trame en rentrant l'adresse de Google.