SSH (Secure Shell)

Qu'est-ce que SSH ?

- SSH, ou Secure Shell, est une technologie qui permet de contrôler et gérer une machine à distance via un terminal de manière sécurisée.

Pourquoi "Secure Shell" ?

- Shell 💻 : Un shell est une interface qui permet d'exécuter des commandes sur un système - (comme un terminal).

- Secure 🔒 : Contrairement aux anciennes méthodes (comme Telnet), SSH chiffre les communications, empêchant ainsi les écoutes et les attaques.

Pourquoi SSH est utile ?

- Gérer un serveur à distance sans être physiquement devant la machine.

- Exécuter des commandes sur un autre ordinateur via le réseau.

- Transférer des fichiers de façon sécurisée (ex : SFTP).

- Automatiser des tâches (déploiement, gestion de serveurs, etc.).

Telnet : Un protocole ancien ??

- Le protocole Telnet permettait de se connecter à distance à un serveur, mais envoyait toutes les données en texte clair.

- Cela exposait les informations sensibles, comme les identifiants, à des risques d'interception.

SSH : La solution sécurisée !

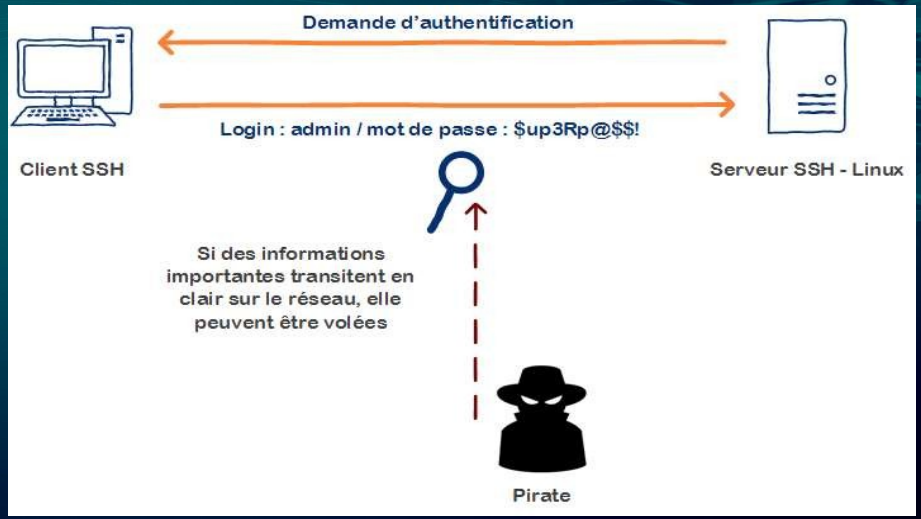
- Pour résoudre ce problème, SSH a été créé pour chiffrer toutes les communications entre l'utilisateur et le serveur.

- Ce protocole sécurisé a remplacé Telnet, garantissant ainsi une connexion à distance privée et protégée.

Les risques de Telnet !

- Le manque de chiffrement dans Telnet permettait à des intrus de capturer facilement les données transmises

- Notamment les mots de passe et les commandes exécutées, ce qui rendait ce protocole vulnérable.



Clé SSH : Qu'est-ce que c'est ?

- Une clé SSH est une méthode d'authentification utilisée pour se connecter de manière sécurisée à un serveur distant.

- Contrairement à un mot de passe classique, une clé SSH repose sur un système de cryptographie asymétrique : il y a deux clés, une clé publique et une clé privée.

Pourquoi utiliser des clés SSH ?

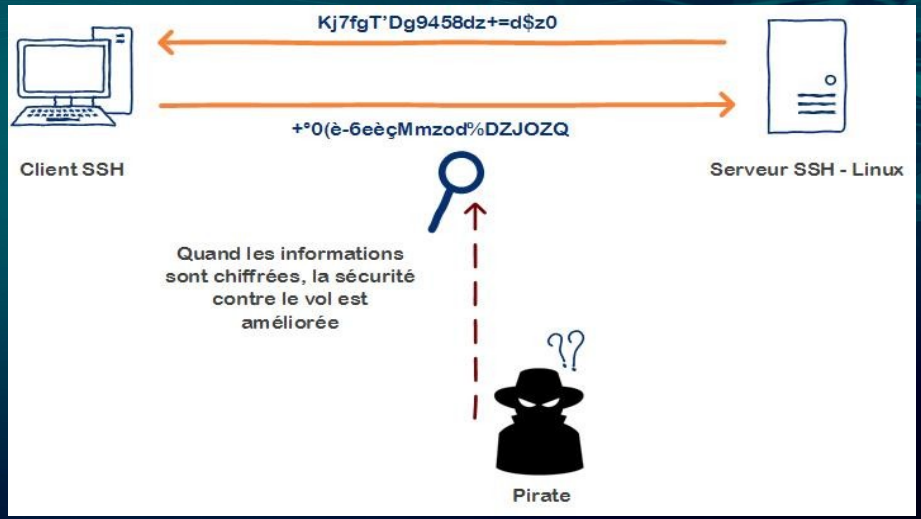
- Sécuriser l'accès : Les clés SSH sont plus sûres que les mots de passe, qui peuvent être devinés ou interceptés.

- Automatisation : Elles permettent des connexions automatiques sans avoir à saisir un mot de passe à chaque fois.

Clé publique et clé privée !

- Clé publique : Elle est installée sur le serveur distant. C'est elle qui permet de vérifier que la connexion vient de la bonne personne.

- Clé privée : Elle est conservée sur l'ordinateur local. C'est cette clé qui permet de s'authentifier en prouvant que l'utilisateur est bien celui qu'il prétend être.

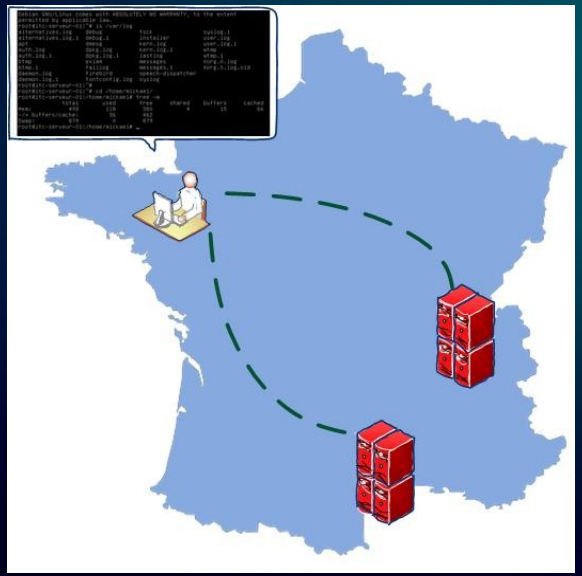


Comment ça fonctione ?

- Connexion sécurisée entre l'ordinateur et le serveur (Tunnel SSH)

- L’ordinateur local établit une connexion avec un serveur distant via SSH.

- SSH crée un tunnel sécurisé entre les deux machines, garantissant que les données échangées ne puissent pas être interceptées.



```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@itc-serveur-01:~# ls /var/log
alternatives.log    debug          fsck             syslog.1
alternatives.log.1  debug.1        installer        user.log
apt                 dmesg          kern.log         user.log.1
auth.log            dpkg.log       kern.log.1       wtmp
auth.log.1          dpkg.log.1     lastlog          wtmp.1
btmp                exim4          messages         Xorg.0.log
btmp.1              faillog        messages.1       Xorg.0.log.old
daemon.log          firebird       speech-dispatcher
daemon.log.1        fontconfig.log  syslog
root@itc-serveur-01:~#
root@itc-serveur-01:~# cd /home/mickael/
root@itc-serveur-01:/home/mickael# free -m
             total       used       free     shared    buffers     cached
Mem:           498        118        380          4         15         66
-/+ buffers/cache:         36        462
Swap:          879          0        879
root@itc-serveur-01:/home/mickael# _
```

Qu'est-ce qu'un tunnel SSH ?

- Un tunnel SSH est une connexion chiffrée qui permet de transmettre des données de manière sécurisée à travers un réseau non sécurisé (comme Internet).

- Ce tunnel empêche les pirates d’intercepter ou de modifier les informations échangées. Il agit comme un canal privé pour sécuriser les données en transit.

Le protocole SSH : Communiquer avec une machine distante

Qu’est-ce que fait exactement le protocole SSH ?

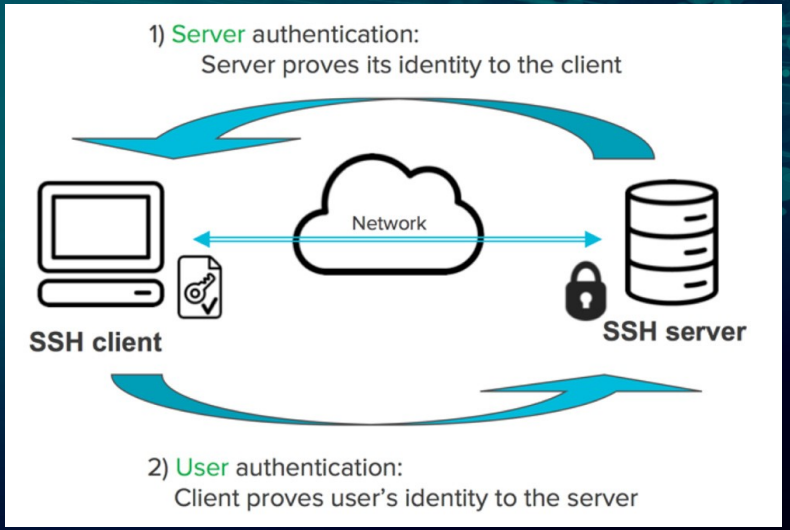
- SSH établit une connexion sécurisée entre deux machines, généralement entre ton ordinateur et un serveur distant.
- Il permet d’envoyer des commandes, de recevoir des réponses, de transférer des fichiers, tout ça comme si tu étais connecté localement sur la machine distante.

Comment ça marche ?

- Lors de la connexion, les deux machines se reconnaissent mutuellement grâce à un système de clé cryptographique ou via un mot de passe.
- Une fois l’identité vérifiée, SSH crée un canal chiffré entre ton ordi et le serveur.
- Toutes les actions (commandes, réponses, transferts) passent alors par ce tunnel sécurisé.

Ce que tu peux faire avec SSH :

- Lancer des programmes à distance.
- Gérer des fichiers (les déplacer, créer, supprimer...).
- Configurer ou surveiller un serveur.
- Déployer un site web, exécuter des scripts automatiques, etc.



L’agent SSH : le gardien silencieux

C’est quoi un agent SSH ?

- L’agent SSH est un petit programme qui tourne en tâche de fond.
- Son rôle ? Garder ta clé privée en mémoire pour que tu n’aies pas besoin de ressaisir ton mot de passe à chaque connexion.

Pourquoi c’est utile ?

- Il évite de retaper la passphrase de la clé privée à chaque commande.
- C’est indispensable quand on travaille avec Git ou quand on se connecte souvent à plusieurs serveurs.

En résumé :

- L’agent SSH te permet de gagner du temps, renforce la sécurité (car ta clé reste protégée), et améliore ton workflow au quotidien.

Et avec Git, alors ?

Quand on relie une paire de clés SSH à son compte GitHub ou GitLab :

- La clé publique est ajoutée à ton compte distant.
- Quand tu fais un push, pull ou clone, Git va vérifier si la clé privée locale correspond à la clé publique enregistrée sur ton compte.
- Si c'est bon → Connexion instantanée et sécurisée, sans avoir à entrer ton identifiant/mot de passe.

Pourquoi c'est génial ?

- Fini les connexions non sécurisées via HTTPS.
- Tu évites de mettre ton mot de passe Git en clair dans les scripts ou outils.
- Tu peux travailler en toute confiance, surtout en équipe.