Api/casoft Init - Jour 3 - SSH

Contrôle à distance avec SSH

Romain de Laage et Noé Amiot

06 juillet 2022

Picasoft

Introduction

Introduction

Le principe

Besoin

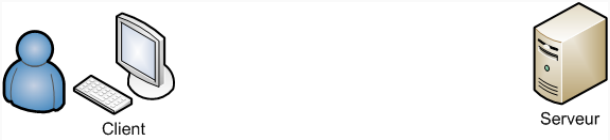
Problème

Avec des amis nous avons un serveur sur lequel on héberge nos blogs respectifs. Depuis hier il y a un problème sur celui-ci, il nous répond une erreur à chaque demande. Le serveur est à Toulouse et nous sommes tous à Compiègne, sommes nous vraiment obligés de se déplacer à Toulouse pour résoudre ce problème ?

La situation précédente est assez courante. Il est assez peu pratique de se déplacer lorsqu'on a besoin d'effectuer une action sur une machine. Heureusement il existe au moins une solution : SSH !

Le protocole SSH

- Protocole
- Sécurisé (*Secure Shell*)
- Connexion à distance
- Client-Serveur
- Utilisations dérivées (transfert de fichiers, tunnel)



Introduction

Installation de SSH

Client SSH

Le client est le logiciel qui permet de se connecter et de contrôler une machine distante.

Vous utilisez Ubuntu, le client SSH (`openssh-client`) est installé par défaut. Bonne nouvelle, vous n'avez rien de particulier à faire !

Si vous vous retrouvez bloqué avec une machine Windows (ce que nous ne vous souhaitons pas évidemment), vous devrez installer vous-même le client, vous pouvez regarder PuTTY qui est libre (*MIT*) et assez complet.

Serveur SSH

Le serveur est le logiciel qui va permettre d'autoriser les connexions à distance sur une machine, c'est le logiciel qu'il faut installer préalablement sur votre serveur ou votre ordinateur pour pouvoir le contrôler à distance.

On va utiliser `openssh-server`. Il est déjà installé par défaut sur beaucoup de distributions linux mais pas activé par défaut, s'il n'est pas activé vous pouvez exécuter `sudo systemctl start sshd`.

Attention

Le protocole SSH permet d'exécuter des commandes sur une machine distance, éventuellement de l'éteindre, mais pas de l'allumer. Votre machine doit être allumée, la connexion accessible et le serveur fonctionnel pour que vous puissiez vous y connecter.

Connexion à un hôte distant

Connexion à un hôte distant

Identification de l'hôte

Adresse IP

Attention

Le but n'est pas de faire un cours sur le réseau, nous prendrons donc des raccourcis et n'expliquerons pas tout.

Pour se connecter à une machine il faut pouvoir l'identifier sur notre réseau. Pour cela chaque machine possède un identifiant unique, l'adresse IP. Il existe deux versions d'IP, les IPv4 (comme `23.5.127.55`) et les IPv6 (comme `acbe:ff73::d0ae:ffe1:1`).

Votre machine peut être dotée d'une adresse IP publique qui permet de s'y connecter depuis partout sur Internet ou bien d'une adresse IP privée qui n'est accessible que sur un réseau local.

Pour se connecter sur notre serveur à Toulouse depuis Compiègne il nous faudra une IP publique par exemple.

Nom d'hôte

Il n'est pas facile de retenir les adresses IP, en plus celle-ci sont parfois provisoires. On utilise plutôt à la place un système de nom d'hôte, on renseigne un nom d'hôte au logiciel qui va vérifier dans un annuaire afin de récupérer l'adresse IP associée à ce nom, on utilise pour cela les DNS. Ainsi c'est l'adresse IP `91.224.148.60` qui correspond au nom d'hôte `pad.picasoft.net`.

Il existe des noms d'hôtes particuliers par exemple pour identifier votre machine qui ne passent pas par un DNS, ceux-ci sont listés dans le fichier `/etc/hosts`.

Connexion à un hôte distant

Se connecter

La commande SSH

Pour se connecter à un hôte nous allons utiliser une commande (vous commencez à avoir l'habitude maintenant), la commande en question est `ssh`.

La syntaxe la plus basique pour utiliser cette commande est `ssh <login>@<host>`, où `<login>` est à remplacer par un nom d'utilisateur et `<host>` est le nom d'hôte ou l'adresse IP de la machine à laquelle vous souhaitez vous connecter.

Exemple

Vous pouvez tester en vous connectant à votre propre ordinateur (ok, ça n'a pas beaucoup d'intérêt mais vous pourrez voir comment ça fonctionne), pour cela lancez `ssh <nom d'utilisateur>@localhost`. La console va vous demander d'entrer un mot de passe, entrez le votre.

Exercice ssh pairs

Exercice

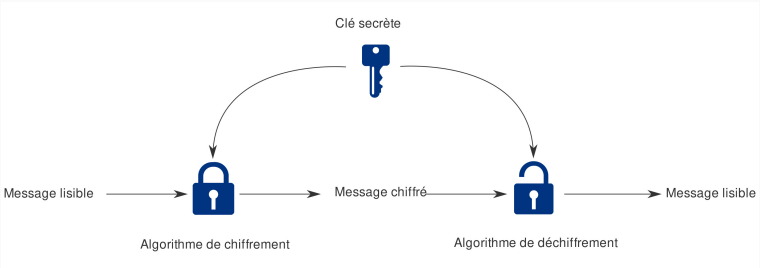
Maintenant que vous avez un serveur ssh qui tourne sur votre ordinateur, vous allez inviter vos voisins sur votre machine. Pour ce faire, hors de question qu'ils utilisent votre compte ! Créez donc un utilisateur, définissez son mot de passe et confiez à votre voisin d'amphithéatre votre adresse ip, son login et son mot de passe pour qu'il puisse se connecter.

WLAN

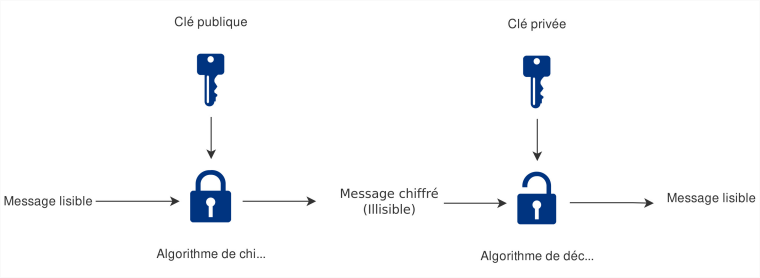
Pour que cela fonctionne, vous devez être sur le même réseau local, veillez donc à vous connecter sur le même réseau.

Sécuriser la connexion

Principe du chiffrement symétrique



Principe du chiffrement asymétrique



Connexion par mot de passe

1. Le serveur va communiquer sa clé publique au client
2. Le client va chiffrer la clé symétrique grâce à la clé publique et l'envoyer au serveur
3. Le serveur va déchiffrer la clé symétrique grâce à sa clé privée
4. Les deux machines seules connaissent maintenant la clé symétrique
5. Toutes les données échangées seront maintenant chiffrées via la clé symétrique
6. Le client peut alors envoyer son mot de passe chiffré au serveur en toute sécurité
7. Le serveur vérifie le mot de passe et autorise le cas échéant le client à poursuivre la communication (toujours chiffrée)

Tout cela est automatique

Toutes ces étapes sont transparentes, le client demande seulement à l'utilisateur de renseigner sont mot de passe.

Connexion par clé

Un autre type de connexion existe, la connexion par clé. L'utilisateur ne renseigne pas le mot de passe de l'utilisateur distant mais va fournir une clé SSH. Le serveur aura une paire de clé (une privée et une publique) tout comme le client. Les échanges sont donc chiffrés grâce au chiffrement asymétrique.

Il reste le problème de la confiance dans les clés publiques fournies, pour celle du client il faut que la clé fournie soit dans une liste de clé autorisées pour l'utilisateur distant. Pour le serveur on utilise le principe du *TOFU*.

Génération d'une paire de clés SSH

Pour générer la paire de clé on peut utiliser la commande `ssh-keygen -t rsa`, on peut renseigner un mot de passe pour chiffrer (de façon symétrique donc) la clé privée mais ce n'est pas obligatoire.

Problème

Mais comment envoyer la clé publique au serveur pour qu'il l'accepte ?

Pour envoyer la clé on peut utiliser la commande `ssh-copy-id` (on vous laisse le soin de regarder le manuel pour avoir une idée de son fonctionnement).

Exercice ssh tuxa

Les étudiants de l'UTC ont tous accès au serveur tuxa, il héberge, dans les dossiers personnels des étudiants, leurs mails, leurs documents (sessions ordinateurs de l'utc), un site internet, ...

Attention

Votre dossier existe de base mais pour pouvoir vous connecter en ssh à cette machine, vous devez activer préalablement l'accès ssh sur le site `https://comptes.utc.fr`. Puis, dans l'onglet services, activez ssh-users.

Exercice

Le ssh est activé, connectez-vous à `tuxa.sme.utc`. (Votre couple login/mot de passe est le même que pour vous connecter sur l'ent).

Exercice ssh tuxa partie 2

Question

On parlé d'un site web, c'est cool mais on y accède comment ?

Tous les éléments présents dans votre dossier `public_html` sont rendus publics à l'adresse `https://wwwetu.utc.fr/~login/whatever`.

Exercice

Créez un fichier dans votre dossier `public_html` constatez qu'il est accessible sur votre site.

Exercice ssh tuxa partie 3

Exercice

Maintenant que vous maîtrisez la connexion par mot de passe, générez une clé ssh et déclarez là sur tuxa pour pouvoir vous connecter par clé.

Aller plus loin

scp

Cette commande permet de copier des fichiers, des dossiers, ... entre un client et un serveur au travers du protocole SSH. Elle a un fonctionnement similaire à la commande cp, pour indiquer un chemin local (sur la machine du client) on le fait comme dans toutes les autres commandes. Pour indiquer un chemin distant (sur la machine du serveur) il faut faire précéder le chemin que l'on souhaite précédé de `<login>@<host>:`.

Attention

Cette commande est dépréciée, elle est assez pratique et facile à apprendre car elle ressemble à la commande cp mais il faudrait privilégier les solutions suivantes pour le transfert de fichiers.

tunnel

Le protocole SSH peut être utilisé afin de créer un tunnel entre le client et le serveur. C'est-à-dire faire passer des données entre le serveur et le client à travers un tuyau. C'est utile quand on ne peut pas faire passer les données autrement (sécurité, IPv4/IPv6, ...).

L'option de la commande `ssh` qui permet de faire cela est l'option `-L`, nous vous laissons le soins de regarder le manuel pour plus de détails.

rsync et sftp

On l'a vu, la commande `scp` est dépréciée, les commandes `rsync` et `sftp` sont donc à privilégier. Elles utilisent toutes les deux le protocole SSH et permettent chacune de transférer des fichiers, cependant elles ont des comportements différents. Nous vous laissons encore une fois le soin de consulter le manuel à leur sujet.

Penser au manuel !

Nous ne le répéterons jamais assez ! Pensez à regarder le manuel afin d'avoir plus de détails sur les commandes citées dans ce cours, il est souvent très complet et on y apprend plein de choses. Avec un peu de chance il y a même des exemples d'utilisation.

Références

- Module *Connexion à un serveur SSH*, CC BY-SA, par Picasoft (Anthony Bocquet , Lola Lejeune, Antoine Barbare) : https://school.picasoft.net/modules/ssh01/co/ssh01.html
- Module *Sécuriser un accès SSH*, CC BY-SA, par Picasoft (Anthony Bocquet , Lola Lejeune, Antoine Barbare) : https://school.picasoft.net/modules/ssh02/co/ssh02.html