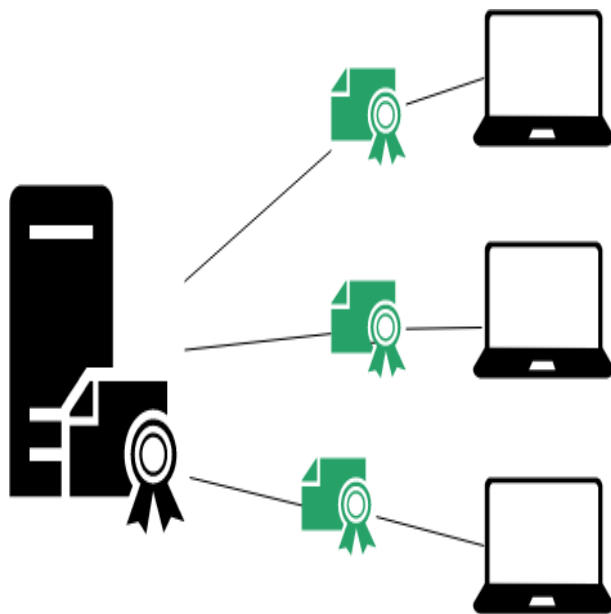


## Extension au protocole ACME pour certificat SSH



Thèse de Bachelor présentée par

**Inès MAYA**

pour l'obtention du titre Bachelor of Science HES-SO en

**Informatique et systèmes de communication avec orientation  
Sécurité Informatique**

**Septembre 2023**

Professeur-e HES responsable

**Stéphane KÜNG, Noria FOUKIA**

Légende et source de l'illustration de couverture : Autorité de certification délivrant des certificats à des utilisateurs. Réalisé par MAYA Inès.

# TABLE DES MATIÈRES

|  |             |
|--|-------------|
| <b>Remerciements</b>                             | <b>v</b>    |
| <b>Énoncé du sujet</b>                           | <b>vi</b>   |
| <b>Résumé</b>                                    | <b>vii</b>  |
| <b>Liste de acronymes</b>                        | <b>viii</b> |
| <b>Liste des illustrations</b>                   | <b>xi</b>   |
| <b>Liste des tableaux</b>                        | <b>xii</b>  |
| <b>Introduction</b>                              | <b>1</b>    |
| <b>1 Chapitre 1 : Analyse</b>                    | <b>3</b>    |
| 1.1 Situation actuelle                           | 3           |
| 1.2 Technologies utilisées                       | 3           |
| a Machines virtuelles et systèmes d'exploitation | 3           |
| b PfSense  | 3           |
| c DNS  | 4           |
| d DHCP   | 4           |
| e Serveur web                                    | 5           |
| f Step-CA  | 6           |
| <b>2 Chapitre 2 : Architecture</b>               | <b>8</b>    |
| 2.1 Réseau local                                 | 8           |
| 2.2 PfSense                                      | 8           |
| 2.3 DNS  | 9           |
| a Configuration                                  | 9           |
| 2.4 DHCP   | 9           |
| a Configuration                                  | 10          |
| 2.5 Serveur web                                  | 11          |
| a Commandes Nginx                                | 11          |
| 2.6 Provisionner                                 | 14          |
| <b>3 Chapitre 3 : PKI</b>                        | <b>15</b>   |
| 3.1 Step-CA                                      | 15          |
| a Installation                                   | 15          |
| b Désinstallation                                | 16          |
| c Configuration                                  | 16          |
| d Certificate Manager                            | 21          |
| 3.2 Certificats                                  | 21          |
| a Certificats X.509                              | 21          |
| 3.3 Renouvellement des certificats               | 22          |
| 3.4 Liste de révocation de certificats (CLR)     | 24          |

|          |  |           |
|----------|--|-----------|
| 3.5      | Certbot (Let's encrypt)                      | 24        |
| a        | Installation                                 | 25        |
| 3.6      | SSL ou TLS                                   | 27        |
| <b>4</b> | <b>Chapitre 4 : ACME Protocol</b>            | <b>28</b> |
| 4.1      | Types de défis ACME                          | 29        |
| 4.2      | Challenge HTTP (HTTP-01)                     | 29        |
| 4.3      | Challenge DNS (DNS-01)                       | 30        |
| 4.4      | Protocole ACME avec Step-CA                  | 30        |
| <b>5</b> | <b>Chapitre 5 : SSH</b>                      | <b>33</b> |
| 5.1      | Fonctionnement                               | 33        |
| 5.2      | Fonctionnement avec certificat               | 34        |
| 5.3      | Rôles  | 35        |
| a        | Serveur d'accès à distance                   | 35        |
| b        | Client SSH                                   | 36        |
| c        | Hôte   | 36        |
| 5.4      | Smallstep SSH                                | 36        |
| a        | Côté client                                  | 37        |
| b        | Côté host                                    | 39        |
| 5.5      | Authentification en SSH à l'aide d'une PKI   | 41        |
| a        | Authentification des utilisateurs par l'hôte | 43        |
| b        | Confiance des clients envers les hôtes       | 47        |
| c        | Renouvellement des certificats SSH           | 50        |
| 5.6      | Révocation des certificats émis              | 51        |
| 5.7      | Commandes SSH                                | 52        |
|          | <b>Conclusion</b>                            | <b>56</b> |
|          | <b>Références documentaires</b>              | <b>58</b> |

## REMERCIEMENTS

Je souhaite exprimer ma gratitude envers toutes les personnes qui m'ont apporté leur aide, leur soutien et qui m'ont accompagnée dans ce travail, mais aussi tout au long de mon cursus à l'HEPIA :

- Madame Noria FOUKIA, qui a cru en moi, pour son soutien, son écoute ainsi que sa bienveillance

- Monsieur Stéphane KÜNG, pour son accompagnement dans mon travail, sa disponibilité et ses conseils

- Mes camarades de classe, par leur présence, les moments de partage, leurs plaisanteries qui m'ont fait sourire dans les moments de découragement et l'aide qu'ils m'ont apportée

- Aux différents enseignants qui m'ont transmis leur savoir durant mes années d'étude

J'aimerais aussi remercier ceux qui ont été essentiels au quotidien, mes amis, ma famille, et surtout ma maman

- Estrella MAYA, n'ont seulement pour la relecture de ce travail, mais aussi pour ses encouragements, sa main toujours tendue, qui m'a aidée à relever dans les moments d'abattement et de découragement

- le pot de Nutella, qui a été d'un grand réconfort quand le moral était au plus bas

# ÉNONCÉ DU SUJET

**Descriptif :** Grâce à la fondation Let's Encrypt, la mise en place de certificats TLS pour sécuriser une application web a été grandement simplifiée, le service est proposé gratuitement, et la sécurité est augmentée, notamment grâce au renouvellement fréquent des certificats.

Toutefois, force est de constater que sur un réseau interne d'entreprise, le système proposé par la fondation, basé sur le protocole ACME, ne peut pas opérer. Les sites n'étant pas exposés, le CertBot ne peut pas leur délivrer de certificats. Il est toutefois possible aujourd'hui de coupler cet agent CertBot, utilisant le protocole ACME avec une PKI interne d'entreprise.

Le but de ce travail de Bachelor est dans un premier temps, de mettre en place une PKI privée, et de proposer des certificats TLS issue de cette PKI au travers du protocole ACME et de l'agent Certbot.

Dans un deuxième temps, d'ajouter la possibilité de distribuer des certificats d'authentification SSH aux employés par ce même agent. L'objectif est qu'un employé récupère un certificat personnel, lui permettant de s'authentifier en SSH sur les serveurs de l'entreprise, au moyen d'une authentification par PKI.

## Travail demandé :

- Délivrer des certificats web (HTTPS) au travers du protocole ACME en utilisant une PKI interne,
- Maintenir à jour la liste des certificats valides et à qui ils sont assignés,
- Explorer et développer si nécessaire, une solution permettant à des utilisateurs d'obtenir des certificats SSH personnel de courte durée, issus de la PKI interne, en utilisant l'ACME protocole,
- Permettre la révocation des certificats (CLR).

Candidat-e :

**INÈS MAYA**

Filière d'études : ISC

Professeur-e(s) responsable(s) :

**STÉPHANE KÜNG, NORIA FOUKIA**

**En collaboration avec :** ELCA Security

Travail de bachelor soumis à une convention de stage en entreprise : Non

Travail soumis à un contrat de confidentialité : Non

## RÉSUMÉ

Dans les entreprises, les employés ajoutent leurs clés publiques sur tous les services auxquels ils souhaitent se connecter. Cependant, lors d'un départ, il est nécessaire de supprimer toutes ces clés. L'utilisation de connexions SSH avec des certificats permet de les révoquer facilement lors d'un départ, évitant ainsi la présence de clés publiques qui pourraient être laissées sans surveillance.

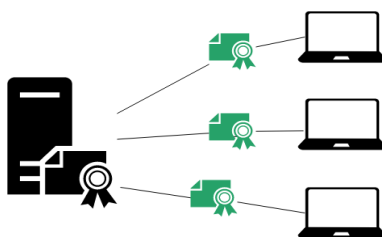
Nous avons abordé différents aspects liés à l'authentification et à la gestion des certificats dans les environnements web et SSH. Nous avons d'abord parlé de l'importance des certificats numériques, qui lient les données d'identification à une entité spécifique telle qu'une personne, une organisation ou un site web. Les certificats jouent un rôle essentiel dans la sécurité des communications et des transactions en ligne.

Nous avons également traité l'installation et l'utilisation de Certbot, un outil populaire pour l'obtention et le renouvellement automatisé de certificats SSL/TLS. Les étapes comprenaient l'installation de Certbot, la configuration du serveur web (tel que Nginx) pour l'utilisation du protocole HTTPS et la gestion des renouvellements automatiques des certificats.

En ce qui concerne l'authentification SSH, nous avons exploré l'utilisation de l'autorité de certification en ligne Step-CA pour la gestion sécurisée des certificats SSH. Nous avons expliqué comment configurer l'hôte et le client pour faire confiance à l'AC SSH, ainsi que la création et l'utilisation de certificats SSH pour les utilisateurs et les hôtes.

Nous avons également évoqué différents types de défis utilisés dans le protocole ACME, tels que le défi HTTP-01 et le défi DNS-01. Ces défis permettent de prouver le contrôle d'un identifiant lors de la délivrance de certificats.

En résumé, ce travail a couvert divers aspects liés à l'authentification et à la gestion des certificats, en mettant l'accent sur l'utilisation de Certbot, l'autorité de certification en ligne Step-CA et les défis ACME. Ces outils et concepts sont essentiels pour assurer la sécurité des communications web et SSH dans les environnements modernes.



Candidat-e :

**INÈS MAYA**

Filière d'études : ISC

Professeur-e(s) responsable(s) :

**STÉPHANE KÜNG, NORIA FOUKIA**

**En collaboration avec : ELCA Security**

Travail de bachelor soumis à une convention de stage en entreprise : Non

Travail soumis à un contrat de confidentialité : Non

## LISTE DE ACRONYMES

**AC** Autorité de certification. 8, 14, 15, 16, 17, 18, 19, 20, 21, 22, 24, 28, 29, 30, 31, 34, 35, 36, 41, 42, 43, 44, 45, 46, 47, 49, 50, 51, 52

**ACME** Automatic Certificate Management Environment. 1, 2, 14, 18, 24, 28, 29, 30, 31, 32, 56

**API** Application Programming Interface. 21

**ASCII** American Standard Code for Information Interchange. 22

**CLI** Command-Line Interface. 6, 37, 40

**CRL** Certificate Revocation List. 24, 52

**CSR** Certificate Signing Request. 28

**DHCP** Dynamic Host Configuration Protocol. 3, 4, 8, 9, 10, 56

**DNS** Domain Name System. 3, 4, 8, 9, 17, 29, 30, 32, 56

**ECDSA** Elliptic Curve Digital Signature Algorithm. 45, 46, 47, 48

**HTTP** HyperText Transfer Protocol. 11, 19, 25, 29, 32

**HTTPS** HyperText Transfer Protocol Secure. 1, 11, 17, 18, 21, 24, 26, 31, 32, 56

**IIS** Internet Information Services. 5

**IP** Internet Protocol. 4, 9, 10, 11, 17, 28, 29, 36

**ITU** International Telecommunication Union. 21

**JWK** JSON Web Key. 14

**LAN** Local Area Network. 8

**MAC** Media Access Control. 10

**NAT** Network Address Translation. 8

**OCSP** Online Certificate Status Protocol. 24, 52



**OIDC** OpenID Connect. 14

**PEM** Privacy Enhanced Mail. 22

**PID** Process ID. 54

**PKI** Public Key Infrastructure. 1, 2, 6, 15, 17, 21, 24, 51, 56

**RSA** Rivest Shamir Adleman. 45

**SHA** Secure Hash Algorithm. 46

**SSH** Secure SHell. 1, 2, 6, 25, 33, 34, 35, 36, 37, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 56, 57

**SShd** Secure SHell Daemon. 44, 46, 48

**SSL** Secure Sockets Layer. 21, 24, 27, 28, 29

**TCP** Transmission Control Protocol. 33, 34

**TLS** Transport Layer Security. 6, 21, 24, 27, 28, 29, 31, 32

**URL** Uniform Resource Locator. 29, 30, 31, 32, 36, 37, 42

**VM** Virtual Machine. 3, 8

## LISTE DES ILLUSTRATIONS

|     |  |    |
|-----|--|----|
| 2.1 | Schéma représentant l'infrastructure réseau . . . . .  | 8  |
| 2.2 | Schéma du fonctionnement du DNS . . . . .  | 9  |
| 2.3 | Capture d'écran du menu Status/DHCP Leases de l'interface web pfSense . . .                          | 10 |
| 2.4 | Capture d'écran de la commande service nginx -V . . . . .  | 12 |
| 2.5 | Capture d'écran de la commande nginx -t . . . . .  | 12 |
| 2.6 | Capture d'écran de la commande service nginx status . . . . .  | 13 |
| 2.7 | Capture d'écran de la commande systemctl status nginx . . . . .                                      | 14 |
| 2.8 | Capture d'écran des commandes service networking restart & systemctl restart<br>networking . . . . . | 14 |
| 3.1 | Capture d'écran des informations sur les versions installées des outils Step &<br>Step-CA . . . . .  | 16 |
| 3.2 | Capture d'écran de la configuration de l'AC . . . . .  | 18 |
| 3.3 | Capture d'écran de l'installation du certificat . . . . .  | 19 |
| 3.4 | Capture d'écran des informations du certificat . . . . .   | 20 |
| 3.5 | Capture d'écran d'un certificat . . . . .  | 22 |
| 3.6 | Capture d'écran de la commande de renouvellement de certificat . . . . .                             | 23 |
| 3.7 | Capture d'écran du contenu du fichier de configuration . . . . .                                     | 23 |
| 4.1 | Capture d'écran de la commande de l'ajout du provisionner . . . . .                                  | 30 |
| 5.1 | Schéma d'une connexion SSH avec une paire de clé privée/publique . . . . .                           | 34 |
| 5.2 | Schéma d'une connexion SSH avec certificat . . . . .   | 35 |
| 5.3 | Capture d'écran de la configuration du client . . . . .  | 37 |
| 5.4 | Capture d'écran du message à l'autorisation du client . . . . .                                      | 38 |
| 5.5 | Capture d'écran de la commande ssh ssh-test.app.smallstep.com . . . . .                              | 39 |
| 5.6 | Capture d'écran de la commande qui télécharge un script . . . . .                                    | 40 |
| 5.7 | Capture d'écran de la commande step ssh hosts . . . . .  | 41 |
| 5.8 | Capture d'écran de l'exécution de la commande step ca init -ssh . . . . .                            | 42 |
| 5.9 | Capture d'écran de l'exécution de la commande step ca init -ssh . . . . .                            | 43 |

|      |   |    |
|------|---|----|
| 5.10 | Capture d'écran de l'installation du certificat . . . . .   | 43 |
| 5.11 | Capture d'écran de l'exécution de la commande utilisée pour configurer SSH<br>avec des certificats et pour inspecter les certificats racine utilisés pour signer les<br>certificats . . . . . | 44 |
| 5.12 | Capture d'écran de l'émission du certificat SSH pour l'utilisateur user . . . . .   | 45 |
| 5.13 | Capture d'écran des détails du certificat SSH . . . . .   | 46 |
| 5.14 | Capture d'écran de la connexion SSH . . . . .   | 47 |
| 5.15 | Capture d'écran de la commande délivrant un certificat pour l'hôte . . . . .  | 47 |
| 5.16 | Capture d'écran des détails du certificat SSH . . . . .   | 48 |
| 5.17 | Capture d'écran de l'ajout de la configuratuon du SSHd . . . . .  | 49 |
| 5.18 | Capture d'écran de la commande affichant la clé . . . . .   | 49 |
| 5.19 | Capture d'écran de la commande affichant la clé . . . . .   | 50 |
| 5.20 | Capture d'écran de la commande faisant la vérification de renouvellement - oui  | 50 |
| 5.21 | Capture d'écran de la commande faisant la vérification de renouvellement - non  | 51 |
| 5.22 | Capture d'écran de la commande man ssh . . . . .  | 53 |
| 5.23 | Capture d'écran de la commande ssh -V . . . . .   | 53 |
| 5.24 | Capture d'écran de la commande sudo systemctl status ssh . . . . .  | 54 |

## LISTE DES TABLEAUX

|     |                                    |    |
|-----|------------------------------------|----|
| 4.1 | Types de challengea ACME . . . . . | 29 |
|-----|------------------------------------|----|

### Références des URL

- URL01 <https://smallstep.com/docs/step-ca/acme-basics/#a-typical-acme-flow>

# INTRODUCTION

## CONTEXTE

Dans les entreprises, les employés ajoutent leurs clés publiques sur tous les services auxquels ils souhaitent se connecter. Cependant, lors d'un départ, il est nécessaire de supprimer toutes ces clés. L'utilisation de connexions [SSH](#)<sup>1</sup> avec des certificats permet de les révoquer facilement lors d'un départ, évitant ainsi la présence de clés publiques qui pourraient être oubliées.

Ce travail représente le résultat d'un projet qui s'est étendu sur plusieurs mois. L'objectif initial est de mettre en place une [PKI](#)<sup>2</sup> et de fournir des certificats web ([HTTPS](#)<sup>3</sup>) issus de cette [PKI](#) en utilisant le protocole [ACME](#)<sup>4</sup> et l'agent Certbot. Dans un second temps, nous allons intégrer la possibilité de distribuer des certificats d'authentification [SSH](#) aux employés par le biais de cet agent. L'objectif final est de permettre à chaque employé d'obtenir un certificat personnel pour s'authentifier en [SSH](#) sur les serveurs de l'entreprise en utilisant l'authentification par certificat [SSH](#).

## MÉTHODOLOGIE

Nous avons commencé par effectuer des recherches sur les différentes technologies utiles pour l'implémentation de ce laboratoire et les compatibilités de ces dernières. Puis, nous avons monté le lab avec les différentes machines virtuelles et les configurations requises pour chacune d'elles. Nous avons configuré la [PKI](#) et délivré des certificats avec le protocole [ACME](#). Dans une seconde partie, nous avons étudié les différents moyens de délivrer des certificats [SSH](#) à l'aide d'une [PKI](#) et tester les différentes possibilités. Pour finir, nous avons mis en place un système délivrant des certificats [SSH](#) en utilisant l'outil, Step-CA [SSH](#), de Smallstep afin de permettre à des utilisateurs de se connecter ; le renouvellement a également été étudié.

## PLAN

Dans la première partie de notre travail, nous avons effectué une analyse approfondie de la situation, puis nous avons examiné les différentes technologies disponibles et sélectionné celles

- 
1. [Secure SHell \(SSH\)](#)
  2. [Public Key Infrastructure \(PKI\)](#)
  3. [HyperText Transfer Protocol Secure \(HTTPS\)](#)
  4. [Automatic Certificate Management Environment \(ACME\)](#)

qui répondaient le mieux à nos besoins.

La deuxième partie est dédiée à l'explication de l'architecture du laboratoire que nous avons utilisée, ainsi qu'à la justification des différents composants et à leur utilité.

La troisième partie est consacrée à l'étude de la **PKI** et à l'outil Step-CA, dans le but de monter une **PKI** avec cet outil. Nous avons également examiné les différents types de certificats existants, pris en charge par Step-CA.

À la suite, une autre section est consacrée à l'étude du protocole, aux différents défis qui lui sont associés, ainsi qu'au protocole **ACME** en combinaison avec l'outil step-CA.

Enfin, dans la dernière partie, nous avons exploré le protocole **SSH** ainsi que l'authentification **SSH** utilisant un certificat **SSH**.

# CHAPITRE 1 : ANALYSE

## 1.1. SITUATION ACTUELLE

Des clés publiques sont utilisées par les différents collaborateurs des entreprises pour se connecter aux différents services auxquels ils peuvent avoir accès. Le problème se pose lorsqu'ils quittent la société, les clés publiques trainent sur les différents appareils.

## 1.2. TECHNOLOGIES UTILISÉES

### a. Machines virtuelles et systèmes d'exploitation

Nous avons installé cinq machines virtuelles avec Oracle VM<sup>5</sup> VirtualBox, car ce logiciel est gratuit et ne nécessite aucune licence. De plus, nous avons l'habitude de l'utiliser.

Les quatre premières tournent sous Debian 11, étant donné que ce système d'exploitation est simple de prise en main, léger et stable. De plus, c'est un système qui compte une grande communauté, ce qui permet d'obtenir de la documentation facilement. Par ailleurs, il possède des paquets déjà installés, évitant ainsi de devoir installer les principaux. Pour finir, il s'agit d'un système d'exploitation avec lequel nous travaillons habituellement.

En ce qui concerne la dernière machine, elle tourne sous pfSense, ce système d'exploitation permettant de faire du routage et de connecter plusieurs réseaux informatiques entre eux. De plus, il permet le fonctionnement des serveurs DNS<sup>6</sup> et DHCP<sup>7</sup>.

### b. PfSense

Pour la mise en place du DNS et du DHCP<sup>8</sup>, pfSense<sup>9</sup> a été utilisé, car il présente plusieurs avantages.

Tout d'abord, pfSense permet la gestion centralisée. En utilisant pfSense comme serveur DNS et DHCP, il est possible de centraliser la gestion de ces services sur une seule plateforme. Cela facilite la configuration, le suivi et la maintenance du réseau.

PfSense offre également de nombreuses fonctionnalités avancées pour la gestion du DNS

---

5. Virtual Machine (VM)

6. Domain Name System (DNS)

7. Dynamic Host Configuration Protocol (DHCP)

8. Explication au prochain chapitre

9. Explication au prochain chapitre également

et du **DHCP**. Il est possible de configurer des résolutions **DNS** personnalisées et de définir des options **DHCP** spécifiques pour les clients.

De plus, pfSense dispose de fonctionnalités de sécurité robustes intégrées, ce qui en fait un choix solide pour les services **DNS** et **DHCP**. Il est possible de mettre en place des règles de pare-feu pour protéger le réseau, effectuer des filtrages **DNS** pour bloquer les domaines malveillants, et surveiller les activités suspectes liées aux requêtes **DNS** et **DHCP**.

En outre, pfSense offre des options de haute disponibilité pour assurer la redondance et la continuité des services **DNS** et **DHCP**. Il est possible de configurer des clusters de pfSense pour éviter les interruptions de service en cas de panne matérielle ou de défaillance d'un nœud.

Pour finir, en tant que solution open-source, pfSense est hautement personnalisable et extensible. Il est possible d'ajouter des packages supplémentaires pour étendre les fonctionnalités de **DNS** et **DHCP** selon les besoins spécifiques.

Utiliser pfSense comme serveur **DNS** et **DHCP** offre une gestion centralisée, des fonctionnalités avancées, une sécurité renforcée, une haute disponibilité et une grande flexibilité. Pour toutes ces raisons, il est le choix le plus adapté à nos besoins.

### **c. DNS**

L'utilisation d'un serveur **DNS** facilite la gestion des noms de domaine, améliore la sécurité, permet des redirections internes et favorise la scalabilité du réseau. Cela simplifie également l'accès aux services internes pour les utilisateurs et facilite la maintenance de l'infrastructure.

### **d. DHCP**

L'utilisation d'un **DHCP** simplifie la gestion des adresses **IP**<sup>10</sup>, réduit les erreurs humaines, optimise l'utilisation des adresses **IP** disponibles et offre une configuration du réseau centralisée pour les appareils qui sont connectés. Il s'agit d'un outil essentiel pour la mise en place d'un réseau efficace, flexible et évolutif.

---

10. [Internet Protocol \(IP\)](#)



## e. Serveur web

Il existe plusieurs types de serveurs web, notamment Apache, Microsoft IIS<sup>11</sup>, Lighttpd et Nginx. Apache et Nginx sont deux serveurs web qui sont compatibles avec Certbot<sup>12</sup>. Ils possèdent des différences mais également des similitudes.

Apache est un serveur logiciel gratuit et open source. Il est utilisé sur divers systèmes d'exploitation tels que Windows, Linux et MacOS. Il est considéré comme l'un des premiers logiciels de serveur web et reste une référence incontournable ; actuellement, il détient une part de marché significative, en raison de sa pré-installation sur toutes les distributions Linux majeures, comme Debian et Red Hat/CentOS.<sup>13</sup>

Du fait de sa popularité, il bénéficie d'une excellente documentation et d'un soutien intégré de la part d'autres projets logiciels. Il est choisi pour sa flexibilité, sa puissance et sa prise en charge quasi universelle.<sup>14</sup>

Nginx est un logiciel de serveur web open source renommé. Sa popularité repose sur sa rapidité et sa capacité à gérer de nombreuses connexions simultanément.<sup>15</sup>

Nginx gagne en termes de popularité en raison de sa légèreté et sa capacité à s'adapter facilement à un matériel minimal. Il est réputé pour servir rapidement des contenus statiques. Il est choisi pour son efficacité en termes de ressources et sa réactivité sous charge, ainsi que pour sa syntaxe de configuration simple. Il est plus efficace et moins exigeant en termes de ressources.

Les deux sont compatibles sous les systèmes UNIX. Cependant, les performances Nginx sous Windows ne sont pas aussi bonnes. De plus, Nginx est plus efficace, rapide et sécurisé.<sup>16</sup>

Pour ces raisons, notre choix se porte sur Nginx, car il présente de nombreux avantages indéniables. Ses performances élevées, sa gestion efficace des connexions, sa faible consommation de ressources et sa flexibilité en font un choix judicieux pour les applications nécessitant des performances optimales. De plus, il s'agit d'un serveur web moderne en pleine expansion.

---

11. Internet Information Services (IIS)

12. Explication dans le chapitre 3

13. JANKOV, 2023.

14. KRIMI, 2022.

15. KRIMI, 2022.

16. POINT, [s. d.].

## f. Step-CA

Step-CA est une autorité de certification pour la gestion automatisée et sécurisée des certificats X.509 et SSH. C'est la contrepartie serveur de Step CLI<sup>17</sup>. Elle est sécurisée par TLS<sup>18</sup> et offre plusieurs fournisseurs de certificats configurables, un modèle de certificat flexible et des bases de données enfichables pour s'adapter à une grande variété de contextes et de flux de travail.<sup>19</sup> C'est un outil développé dans l'objectif de sécuriser les infrastructures.

Il existe plusieurs raisons pour lesquelles il est envisageable d'utiliser Step-CA pour mettre en place une PKI.

Tout d'abord, la sécurité est l'argument principal. En effet, Step-CA est conçu pour offrir une sécurité robuste pour une PKI. Il utilise des algorithmes de chiffrement forts et suit les meilleures pratiques en matière de sécurité pour garantir la confidentialité et l'intégrité des certificats émis.

Une autre des raisons est la flexibilité. En effet, il offre une grande flexibilité dans la gestion des certificats. Il prend en charge différents types de certificats, tels que les certificats X.509 et les certificats SSH. Il est également possible de personnaliser les politiques de certificats pour répondre à des besoins spécifiques.

De plus, il facilite l'automatisation de nombreuses tâches liées à la gestion des certificats. Il fournit une CLI qui permet d'automatiser la création, le renouvellement et la révocation des certificats. Cela peut simplifier considérablement la gestion de la PKI, en particulier lorsqu'il faut gérer un grand nombre de certificats.

En outre, il met l'accent sur la transparence des opérations de la PKI. Il enregistre toutes les actions effectuées sur les certificats dans des journaux de log, ce qui permet de suivre les modifications et de détecter toute activité suspecte. Cela peut être utile pour résoudre les problèmes de sécurité.

Pour finir, il peut gérer efficacement une grande quantité de certificats. Il utilise une architecture distribuée et prend en charge la mise en place de plusieurs instances de Step-CA en tant que cluster pour répartir la charge et garantir la disponibilité.

Cependant, il convient de noter que Step-CA nécessite des connaissances techniques pour

---

17. Command-Line Interface (CLI)

18. Transport Layer Security (TLS)

19. SMALLSTEP, [s. d.(n)].

le déployer et le configurer correctement.

## CHAPITRE 2 : ARCHITECTURE

Il y a cinq **VM** dont quatre tournant sous Debian 11 et l'autre pfSense : une qui sert d'autorité de **AC**<sup>20</sup>, une deuxième de serveur web, une troisième qui est une machine utilisateur "Bob" et pour finir, une qui fait office de routeur, de **DNS** et de **DHCP**.

### 2.1. RÉSEAU LOCAL

Nous avons établi un **LAN**<sup>21</sup> dans lequel cinq machines sont interconnectées. Les quatre premières sont connectées au **LAN** en utilisant le mode Internal Network (intnet). Quant à la machine pfSense, elle est également connectée au **LAN** via le mode Internal Network (intnet) d'un côté, tandis que de l'autre côté, elle est connectée au réseau externe comme le montre l'image ci-dessous.

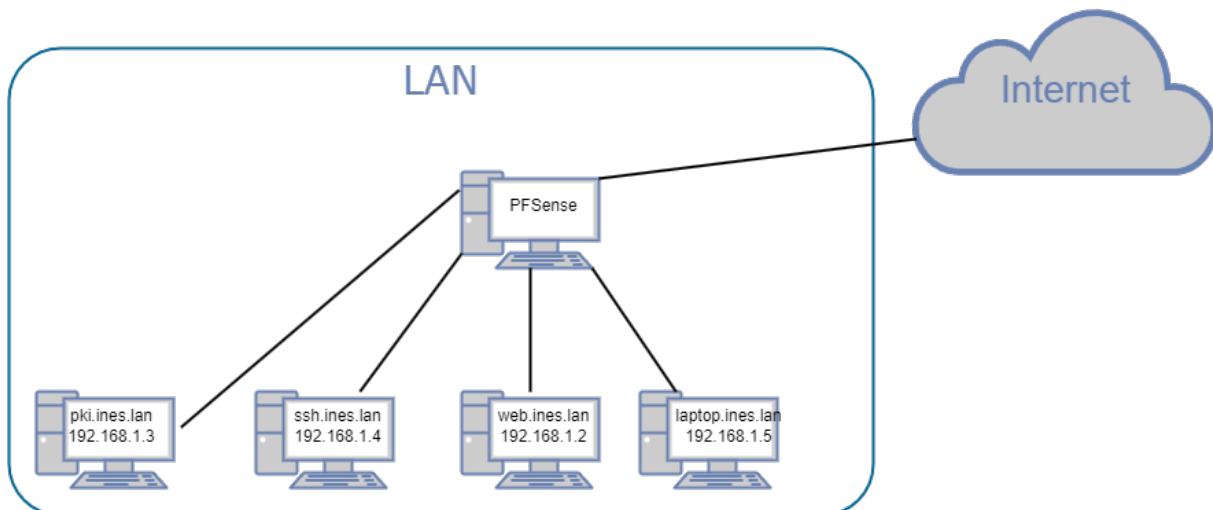


ILLUSTRATION 2.1 – Schéma infrastructure. Réalisé par Maya Inès

### 2.2. PFSense

PfSense est un système d'exploitation open source. Il possède des fonctions de routage et de **NAT**<sup>22</sup>, qui lui permettent de connecter plusieurs réseaux informatiques. Il comporte l'équivalent libre des outils et services utilisés habituellement sur des routeurs professionnels propriétaires. Il peut également servir de **DNS** et de **DHCP**.<sup>23</sup>

20. Autorité de certification (AC)

21. Local Area Network (LAN)

22. Network Address Translation (NAT)

23. WIKIPEDIA, 2022b.

## 2.3. DNS

Un serveur **DNS** est un serveur qui permet de traduire les noms de domaine en adresses **IP**. Il permet de faire correspondre un nom de domaine et une adresse **IP**.

Il est utilisé pour traduire les noms de domaine en adresses **IP**. Lorsqu'un utilisateur veut accéder à un site Web, l'ordinateur envoie une requête au serveur **DNS** pour traduire le nom de domaine en adresse **IP**. Le serveur **DNS** renvoie ensuite l'adresse **IP** à l'ordinateur, qui peut alors se connecter au site Web.

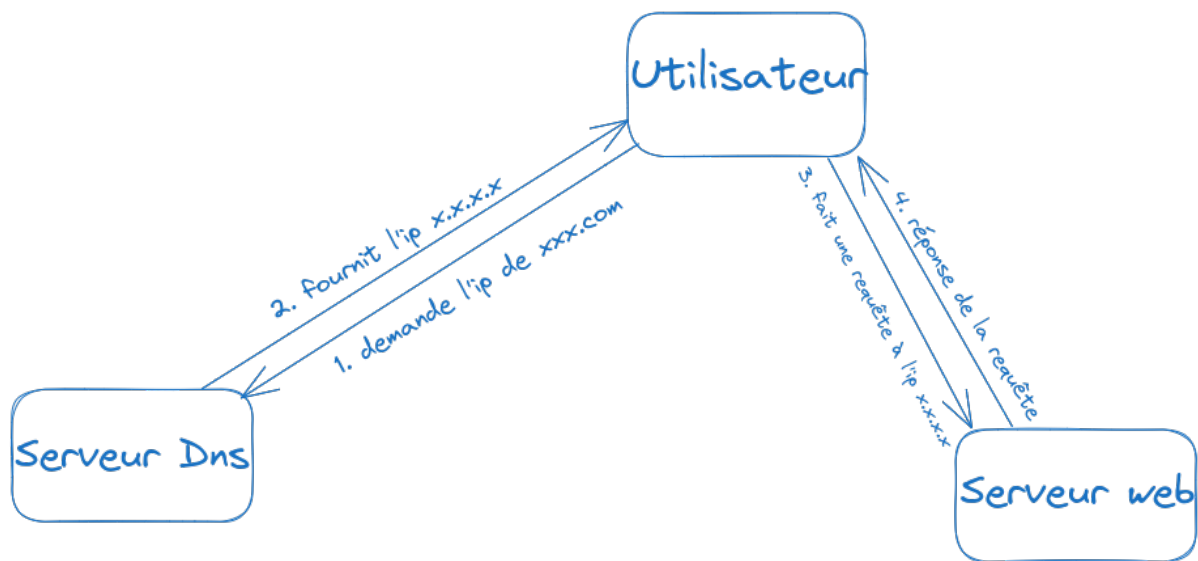


ILLUSTRATION 2.2 – Schéma du fonctionnement du DNS. Réalisé par Maya Inès

### a. Configuration

Pour configurer les serveurs **DNS**, il faut entrer les adresses **DNS** dans System > General Setup et rentrer le nom de domaine dans le champ prévu à cet effet.

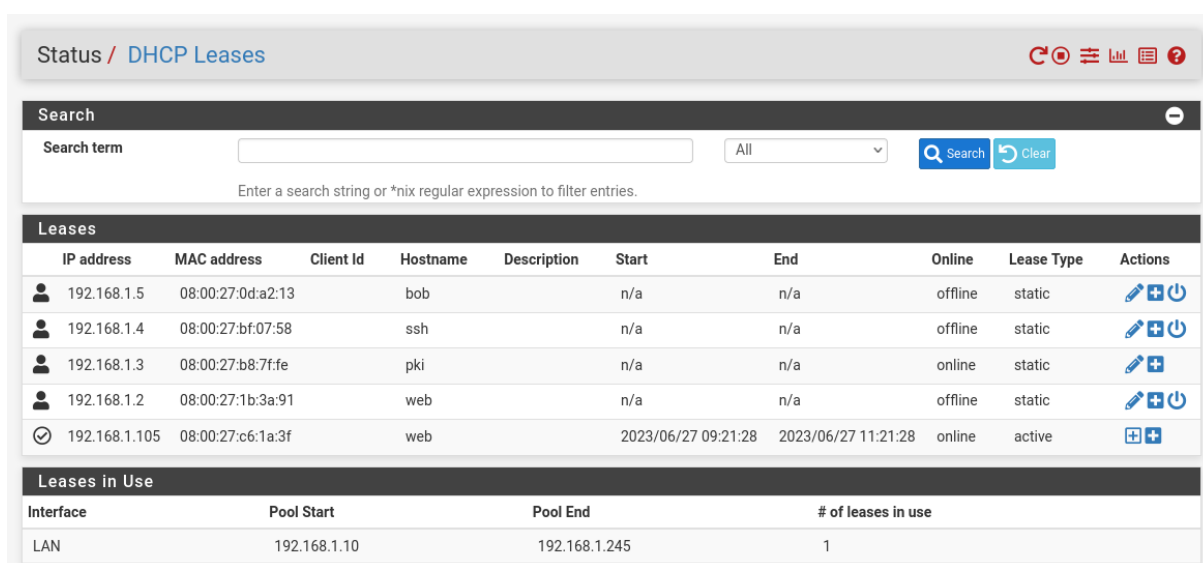
## 2.4. DHCP

Le **DHCP** est un protocole réseau qui simplifie la gestion des adresses **IP** en attribuant automatiquement des adresses **IP** et d'autres paramètres de configuration aux dispositifs connectés à un réseau afin de faciliter la mise en réseau et la configuration des dispositifs.

## a. Configuration

Il est possible configurer des adresses IP statiques pour chaque machine à l'aide de pfSense. La machine pfSense ainsi que les autres nécessitant des adresses IP fixes doivent impérativement être allumées, car ces manipulations nécessitent leur disponibilité.

Tout d'abord, il se faut se connecter à l'interface web de pfSense et se rendre dans "Status", "DHCP Leases". Les machines connectées sont répertoriées avec leur adresse IP, adresse MAC<sup>24</sup> et leur nom. Dans la colonne "Actions" se trouve deux boutons "+" à côté de chaque machine. Il suffit de cliquer sur le premier, en survolant avec la souris, "add static mapping" apparait.



The screenshot shows the pfSense web interface for "Status / DHCP Leases". It includes a search bar at the top, a table of DHCP leases, and a table of leases in use.

| Leases |               |                   |           |          |             |                     |                     |         |            |         |
|--------|---------------|-------------------|-----------|----------|-------------|---------------------|---------------------|---------|------------|---------|
|        | IP address    | MAC address       | Client Id | Hostname | Description | Start               | End                 | Online  | Lease Type | Actions |
| 👤      | 192.168.1.5   | 08:00:27:0d:a2:13 |           | bob      |             | n/a                 | n/a                 | offline | static     | ✎ + 🔌   |
| 👤      | 192.168.1.4   | 08:00:27:bf:07:58 |           | ssh      |             | n/a                 | n/a                 | offline | static     | ✎ + 🔌   |
| 👤      | 192.168.1.3   | 08:00:27:b8:7f:fe |           | pki      |             | n/a                 | n/a                 | online  | static     | ✎ +     |
| 👤      | 192.168.1.2   | 08:00:27:1b:3a:91 |           | web      |             | n/a                 | n/a                 | offline | static     | ✎ + 🔌   |
| 👤      | 192.168.1.105 | 08:00:27:c6:1a:3f |           | web      |             | 2023/06/27 09:21:28 | 2023/06/27 11:21:28 | online  | active     | + +     |

| Leases in Use |              |               |                    |
|---------------|--------------|---------------|--------------------|
| Interface     | Pool Start   | Pool End      | # of leases in use |
| LAN           | 192.168.1.10 | 192.168.1.245 | 1                  |

ILLUSTRATION 2.3 – Capture d'écran du menu Status/DHCP Leases de l'interface web pfSense. Réalisé par Maya Inès

Une fois dans le menu "Edit Static Mapping", saisir une adresse IP dans le champ prévu à cet effet. L'adresse ne doit pas faire partie de la plage d'adresses IP attribuées par le pool DHCP, dans notre cas, une adresse entre 192.168.1.10 et 192.168.1.245. Ensuite, dans le champ prévu pour le nom de domaine, écrire le nom de domaine du serveur. Dans notre cas, il s'agit d'"ines.lan".

Pour finir, tout en bas de la page, il suffit de cliquer sur le bouton "save" et ensuite sur celui "Apply change" afin de rendre effectives les modifications apportées.

Après toutes ces étapes, il est désormais possible de configurer des adresses IP statiques

24. Media Access Control (MAC)

pour les machines du réseau en utilisant pfSense. Il est aussi important de prendre note des adresses IP statiques attribuées à chaque machine pour une référence ultérieure.

Dans notre cas, les adresses attribuées sont les suivantes :

- machine web : 192.168.1.2
- machine pki : 192.168.1.3
- machine ssh : 192.168.1.4
- machine utilisateur : 192.168.1.5

Ces données sont aussi disponibles sur le schéma de l'architecture qui se trouve dans la partie réseau local.

## 2.5. SERVEUR WEB

Un serveur web est un logiciel permettant de stocker des fichiers et de les rendre accessibles sur internet. Il est capable de recevoir des requêtes HTTP<sup>25</sup>/HTTPS et de renvoyer des réponses HTTP/HTTPS à ces requêtes. Les serveurs web sont utilisés pour héberger des sites web, des applications web et d'autres contenus web.

### a. Commandes Nginx

Les commandes mentionnées à continuation sont parmi les plus populaires et utiles pour utiliser Nginx. Cependant, il existe d'autres commandes moins courantes que nous ne citerons pas.

Il est important de noter que l'utilisation de ces commandes nécessite des privilèges de superutilisateur (root). Pour obtenir ces privilèges, la commande "sudo" est utilisée. Elle permet d'exécuter une commande en tant qu'utilisateur root ou administrateur. Il suffit de la placer au début de la commande afin de bénéficier des droits nécessaires.

L'installation du serveur Nginx sur un système d'exploitation basé sur Debian est très simple puisqu'il est disponible dans les dépôts officiels. La commande "*sudo apt install nginx -y*" installe le serveur web Nginx sur le système sans nécessiter l'intervention utilisateur pour confirmer l'installation. Une fois la commande exécutée avec succès, Nginx doit être installé et est prêt à être utilisé comme serveur web sur le système.

Lorsque l'installation est effectuée, il est possible de vérifier la version installée.

```
1 service nginx -V
```

---

25. HyperText Transfer Protocol (HTTP)

```
user@web:~$ sudo service nginx -V
service ver. 1.60
```

ILLUSTRATION 2.4 – Capture d'écran de la commande service nginx -V. Réalisé par Maya Inès

La commande "*nginx -t*" permet de vérifier si la configuration de Nginx est correcte ou s'il y a des erreurs de syntaxe. Si la configuration est correcte, aucun message d'erreur ne sera affiché. En revanche, si des erreurs sont détectées, Nginx affichera des messages spécifiques indiquant où se trouvent les erreurs et quelles sont les modifications nécessaires pour corriger la configuration. Cette commande est utile lors de la modification de la configuration de Nginx pour s'assurer que les modifications apportées sont valides et n'entraînent pas d'erreurs lors du redémarrage du serveur.

```
user@web:~$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

ILLUSTRATION 2.5 – Capture d'écran de la commande nginx -t. Réalisé par Maya Inès

La commande "*systemctl enable nginx*" permet de faire démarrer automatiquement le serveur Web Nginx lorsque la machine Linux démarre ou redémarre. La commande correspondante est "*service nginx enable*".

En exécutant la commande "*service nginx stop*" avec les privilèges appropriés, le service Nginx sera arrêté, ce qui signifie que le serveur web ne sera plus accessible jusqu'au redémarrage. Il convient de noter que l'arrêt du service Nginx peut affecter les sites web ou les applications qui dépendent de celui-ci. Sa commande correspondante est *systemctl stop nginx*.

Quitter Nginx est très similaire à l'arrêter, mais il le fait avec élégance, ce qui signifie qu'il finira de servir les connexions ouvertes avant de s'arrêter. Pour quitter Nginx, il faut employer l'une des commandes suivantes :

- 1 service nginx quit
- 2 systemctl quit nginx

Le redémarrage de Nginx se résume à un arrêt suivi d'un démarrage. Les commandes suivantes doivent être utilisées pour redémarrer Nginx :

- 1 service nginx restart
- 2 systemctl restart nginx

Cette commande revient à faire :



```
1 sudo systemctl start nginx
2 sudo systemctl stop nginx
```

Le rechargement est un peu différent du redémarrage, il est plus gracieux. Le rechargement est défini comme "démarrer le nouveau processus de travail avec une nouvelle configuration, arrêter gracieusement les anciens processus de travail". Les commandes pour recharger Nginx sont les suivantes :

```
1 service nginx reload
2 systemctl reload nginx
```

La vérification de l'état actuel du serveur web Nginx se fait à l'aide de l'une des commandes suivantes :

```
1 service nginx status
2 systemctl status nginx
```

```
user@web:~$ sudo service nginx status
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset:
   Active: active (running) since Fri 2023-09-08 10:34:17 CEST; 4h 35min ago
     Docs: man:nginx(8)
   Process: 570 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_proces
   Process: 581 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (co
   Main PID: 583 (nginx)
    Tasks: 3 (limit: 2307)
   Memory: 12.0M
      CPU: 94ms
   CGroup: /system.slice/nginx.service
           └─583 nginx: master process /usr/sbin/nginx -g daemon on; master_p
             └─584 nginx: worker process
               └─585 nginx: worker process

Sep 08 10:34:16 web.ines.lan systemd[1]: Starting A high performance web server
Sep 08 10:34:16 web.ines.lan systemd[1]: nginx.service: Failed to parse PID fro
Sep 08 10:34:17 web.ines.lan systemd[1]: Started A high performance web server
```

ILLUSTRATION 2.6 – Capture d'écran de la commande service nginx status. Réalisé par Maya Inès

```

user@web:~$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset:
   Active: active (running) since Fri 2023-09-08 10:34:17 CEST; 4h 35min ago
     Docs: man:nginx(8)
   Process: 570 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_proces
   Process: 581 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (co
   Main PID: 583 (nginx)
     Tasks: 3 (limit: 2307)
    Memory: 12.0M
       CPU: 94ms
    CGroup: /system.slice/nginx.service
           └─583 nginx: master process /usr/sbin/nginx -g daemon on; master_p
             └─584 nginx: worker process
               └─585 nginx: worker process

Sep 08 10:34:16 web.ines.lan systemd[1]: Starting A high performance web server
Sep 08 10:34:16 web.ines.lan systemd[1]: nginx.service: Failed to parse PID fro
Sep 08 10:34:17 web.ines.lan systemd[1]: Started A high performance web server

```

ILLUSTRATION 2.7 – Capture d’écran de la commande `systemctl status nginx`. Réalisé par Maya Inès

En exécutant les commandes ci-dessous avec les privilèges appropriés, le service de réseau sera redémarré. Cela signifie que toutes les interfaces réseau seront désactivées puis réactivées, et les configurations réseau seront réinitialisées. Le redémarrage du service de réseau est généralement utilisé lorsque des modifications de configuration réseau ont été effectuées et nécessitent une prise en compte immédiate.

```

1 service networking restart
2 systemctl restart networking

```

```

user@web:~$ sudo service networking restart
user@web:~$ sudo systemctl restart networking

```

ILLUSTRATION 2.8 – Capture d’écran des commandes `service networking restart` & `systemctl restart networking`. Réalisé par Maya Inès

## 2.6. PROVISIONNER

Les provisionners sont des méthodes d’utilisation de l’AC afin d’obtenir des certificats pour des personnes ou des machines. Ils offrent différents modes d’autorisation pour l’AC.<sup>26</sup> Cela permet l’automatisation des tâches et garanti des ressources bien configurées.

Il existe plusieurs types de provisionners, notamment [JWK](#)<sup>27</sup>, [OIDC](#)<sup>28</sup>, [ACME](#), [SSHPOP](#). Le choix du provisionner dépend des besoins spécifiques.

26. SMALLSTEP, [s. d.(f)].

27. JSON Web Key (JWK)

28. OpenID Connect (OIDC)

## CHAPITRE 3 : PKI

Une **PKI** a un grand nombre d'utilisations différentes. Cependant, son objectif premier est le chiffrement et/ou la signature des données. Elle permet aux entités, utilisateurs et serveurs, de sécuriser des échanges d'information en utilisant des certificats digitaux.

Dans une telle infrastructure, l'utilisateur doit faire une demande auprès de l'**AC** afin d'obtenir un certificat numérique. Le client génère un couple de clés (une privée et une publique) et envoie la clé publique à l'**AC**. Ensuite, elle signe la clé publique et joint la signature sur le certificat, qu'elle renvoie au client.

### 3.1. STEP-CA

#### a. Installation

Pour installer les outils Step et Step-CA sur un système Linux (Debian), nous avons suivi les instructions fournies par la documentation de Smallstep. Ci-dessous se trouvent les commandes nécessaires afin d'effectuer cette installation.

Tout d'abord, télécharger le package d'installation de Step-cli.

```
1 wget https://dl.step.sm/gh-release/cli/docs-ca-install/v0.23.2/step-cli_0.23.2_amd64.deb
```

Puis installer Step-cli en utilisant dpkg.

```
1 sudo dpkg -i step-cli_0.23.2_amd64.deb
```

Mais, il faut aussi télécharger le package d'installation de Step-CA.

```
1 wget https://dl.step.sm/gh-release/certificates/docs-ca-install/v0.23.2/step-ca_0.23.2_amd64.deb
```

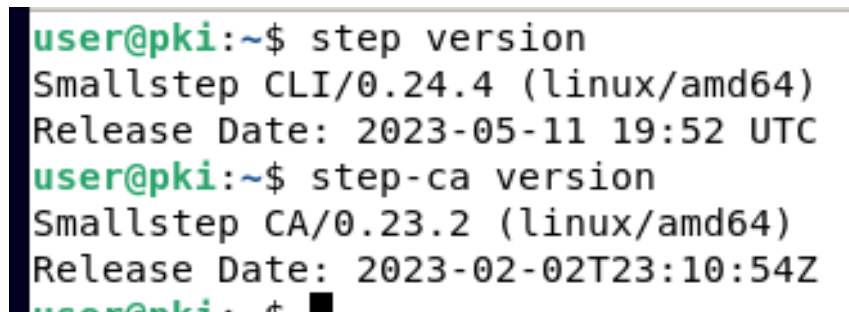
Pour terminer, il est nécessaire d'installer Step-CA en utilisant dpkg.

```
1 sudo dpkg -i step-ca_0.23.2_amd64.deb
```

Une fois l'installation terminée, il est possible de vérifier que les outils ont été installés avec succès en exécutant les commandes ci-dessous. La première sert à vérifier la version de Step et la seconde celle de Step-CA.

```
1 step version
2 step-ca version
```

En tapant ces commandes, les informations sur les versions installées des outils Step et Step-CA seront affichées, confirmant ainsi que l’installation s’est déroulée correctement.



```

user@pki:~$ step version
Smallstep CLI/0.24.4 (linux/amd64)
Release Date: 2023-05-11 19:52 UTC
user@pki:~$ step-ca version
Smallstep CA/0.23.2 (linux/amd64)
Release Date: 2023-02-02T23:10:54Z

```

ILLUSTRATION 3.1 – Capture d’écran des informations sur les versions installées des outils Step & Step-CA. Réalisé par Maya Inès

## b. Désinstallation

Pour désinstaller les outils Step et Step-CA, il faut utiliser la commande ci-dessous. Elle désinstalle le package Step-CLI du système.

```
1 sudo dpkg -r step-cli
```

De plus, pour effectuer une désinstallation complète, il est également indispensable de supprimer la configuration du répertoire HOME/.step. Il est possible de le faire en exécutant la commande ci-dessous.

```
1 rm -rf ~/.step
```

Cela supprimera le répertoire .step et tous ses contenus, y compris la configuration associée.

Il est important de noter que ces commandes doivent être exécutées avec les privilèges de superutilisateur (root) en utilisant la commande sudo. Par ailleurs, il faut être attentif aux actions qui sont effectuées lors de la désinstallation des outils Step et Step-CA, car elles peuvent avoir des conséquences sur le fonctionnement du système.

## c. Configuration

Pour cette partie, nous avons suivi le tutoriel Getting Started de Smallstep<sup>29</sup>.

Pour configurer l’AC, il suffit d’exécuter la commande suivante :

```
1 step ca init
```

29. Disponible à l’adresse : <https://smallstep.com/docs/step-ca/getting-started/#run-your-certificate-authority>

Il est crucial de noter qu'il est nécessaire d'exécuter cette commande avec des privilèges administrateurs. Il faut s'assurer d'avoir les droits nécessaires (privilèges administrateurs ou utilisateur root) avant de lancer la commande pour garantir un fonctionnement correct et sécurisé de l'AC.

Lors de sa configuration, certaines informations sont demandées, telles que le nom de la PKI, l'adresse IP ou le nom du DNS à ajouter à l'AC, l'adresse d'écoute de l'AC, le nom du premier provisionner de l'AC ainsi qu'un mot de passe.

L'adresse d'écoute de la nouvelle AC correspond à l'adresse IP et au port sur lequel le serveur de l'AC écoute les connexions entrantes. Il est possible de spécifier une adresse et un port différents de ceux utilisés dans l'exemple.

Il est important de mémoriser le mot de passe fourni, car celui-ci est demandé pour effectuer diverses opérations, notamment la demande de certificat.

Lors de notre configuration, nous avons fourni les informations suivantes :

- Nom de la PKI : *pki-ines*
- Nom DNS ou adresses IP à ajouter à l'AC : *192.168.1.3*
- Adresse d'écoute de l'AC : *443*
- Nom du premier provisionner de l'AC : *user@pki-ines.com*

Pour spécifier l'adresse d'écoute de la nouvelle AC, il faut fournir l'adresse IP du serveur ou de la machine sur laquelle l'AC est exécutée. Cette adresse indique l'interface réseau par laquelle l'AC sera accessible. Dans notre cas, l'adresse IP entrée est 192.168.1.3.

Il est crucial de s'assurer de fournir la bonne adresse IP du serveur sur lequel l'AC sera exécutée afin que les demandes de certificat et autres opérations puissent être correctement gérées par l'AC.

En ce qui concerne l'adresse d'écoute, le port par défaut est le port 443, qui est généralement utilisé pour les connexions HTTPS.

Cependant, si le souhait est que l'AC écoute sur toutes les interfaces réseau disponibles, spécifiez l'adresse IP "0.0.0.0". Cela signifie que l'AC sera accessible via toutes les adresses IP du serveur.

```

user@pki:~$ sudo step ca init
[sudo] password for user:
✓ Deployment Type: Standalone
What would you like to name your new PKI?
✓ (e.g. Smallstep): pki-ines
What DNS names or IP addresses will clients use to reach your CA?
✓ (e.g. ca.example.com[,10.1.2.3,etc.]): pki.ines.lan
What IP and port will your new CA bind to? (:443 will bind to 0.0.0.0:443)
✓ (e.g. :443 or 127.0.0.1:443): :443
What would you like to name the CA's first provisioner?
✓ (e.g. you@smallstep.com): user@pki.ines.lan
Choose a password for your CA keys and first provisioner.
✓ [leave empty and we'll generate one]:

Generating root certificate... done!
Generating intermediate certificate... done!

✓ Root certificate: /root/.step/certs/root_ca.crt
✓ Root private key: /root/.step/secrets/root_ca_key
✓ Root fingerprint: b850e841aeb47bf6c2d87b27cdad493d16efaf78fa6e85310125bf1ff367b9bc
✓ Intermediate certificate: /root/.step/certs/intermediate_ca.crt
✓ Intermediate private key: /root/.step/secrets/intermediate_ca_key
✓ Database folder: /root/.step/db
✓ Default configuration: /root/.step/config/defaults.json
✓ Certificate Authority configuration: /root/.step/config/ca.json

Your PKI is ready to go. To generate certificates for individual services see 'step help ca'.

FEEDBACK 🍌 🍌
The step utility is not instrumented for usage statistics. It does not phone
home. But your feedback is extremely valuable. Any information you can provide
regarding how you're using `step` helps. Please send us a sentence or two,
good or bad at feedback@smallstep.com or join GitHub Discussions
https://github.com/smallstep/certificates/discussions and our Discord
https://u.step.sm/discord.

```

### ILLUSTRATION 3.2 – Capture d’écran de la configuration de l’AC. Réalisé par Maya Inès

Il est important de noter l’empreinte de la racine. Cette information est nécessaire pour les étapes ultérieures afin d’établir la confiance avec l’AC à partir d’autres environnements ou hôtes.

Pour voir la configuration qui a été mise en place, il est possible de le faire avec cette commande-ci :

```
1 cat .step/config/defaults.json
```

Pour commencer, il faut démarrer l’AC avec la commande suivante :

```
1 step-ca
```

Il est nécessaire de s’assurer l’exécution de la commande ci-dessus en ayant des privilèges administrateurs, afin de pouvoir écouter le port d’écoute HTTPS par défaut.

Pour accéder à l’AC à distance, il faut installer et utiliser la commande Step sur les clients. Il est également possible d’utiliser n’importe quel client ACME pour obtenir des certificats auprès de l’AC.

Le certificat racine de l’AC étant un certificat auto-signé, il n’est pas automatiquement ap-

prouvé par les clients. Tout nouveau client Step doit établir une relation de confiance avec l'AC. Il est possible de le faire en fournissant l'empreinte digitale de l'AC à *step ca bootstrap*. L'empreinte digitale de l'AC est une signature cryptographique qui identifie le certificat racine de l'AC.<sup>30</sup>

La commande ci-dessous permet d'afficher l'empreinte digitale de l'AC.

```
step certificate fingerprint <.step/certs/root_ca.crt>
```

Pour configurer step de manière à ce qu'il accède à l'AC à partir d'une nouvelle machine, il faut exécuter :

```
step ca bootstrap --ca-url <url> --fingerprint <fingerprint>
```

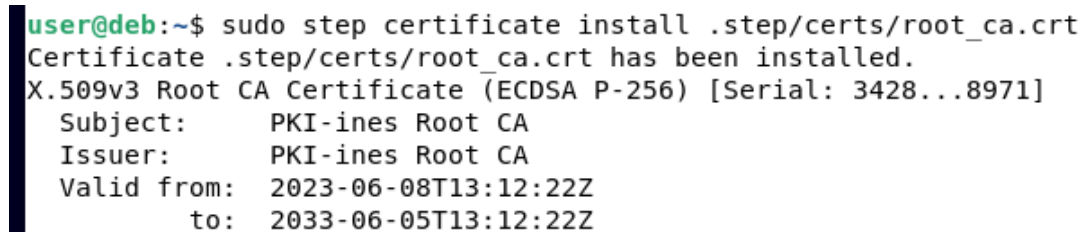
Cette commande télécharge le certificat racine de l'AC et écrit les détails de la connexion dans `.step/config/defaults.json`. À partir de là, step fait confiance à l'AC.

Pour établir une confiance globale dans un certificat spécifique, de sorte qu'il soit reconnu et accepté par les programmes et les services du système qui vérifient les certificats, il existe la commande "step certificate install". Elle permet d'installer un certificat dans le magasin de confiance d'un système.

Lors de l'exécution de la commande, il faut spécifier le chemin vers le fichier du certificat que nous souhaitons installer, comme cela :

```
step certificate install <step path>
```

Dans notre cas, le step path est `.step/certs/root_ca.crt`



```
user@deb:~$ sudo step certificate install .step/certs/root_ca.crt
Certificate .step/certs/root_ca.crt has been installed.
X.509v3 Root CA Certificate (ECDSA P-256) [Serial: 3428...8971]
  Subject:      PKI-ines Root CA
  Issuer:       PKI-ines Root CA
  Valid from:   2023-06-08T13:12:22Z
               to:   2033-06-05T13:12:22Z
```

ILLUSTRATION 3.3 – Capture d'écran de l'installation du certificat. Réalisé par Maya Inès

Après avoir exécuté cette commande, le certificat sera ajouté au magasin de confiance du système, ce qui signifie que les programmes tels que les navigateurs Web, les clients HTTP ou d'autres outils qui vérifient les certificats pour établir des connexions sécurisées reconnaîtront le certificat comme étant valide.

---

30. SMALLSTEP, [s. d.(g)].

Cela peut être particulièrement utile lorsque nous travaillons avec des certificats auto-signés ou des autorités de certification internes qui ne sont pas déjà incluses dans le magasin de confiance par défaut du système. L'installation du certificat permet de contourner les avertissements de sécurité ou les erreurs liées à la vérification du certificat lors de l'accès à des services sécurisés.

Il convient de noter que, selon les restrictions du système, l'exécution de cette commande peut nécessiter des privilèges d'administrateur ou de superutilisateur. Si les privilèges d'administrateur ne sont pas possédés, le message ci-dessous s'affiche.

```
failed to install .step/certs/root_ca.crt: install is not supported on this system
```

Après avoir installé le certificat, en exécutant la commande "step certificate inspect <step path>", l'outil step analysera le certificat spécifié et affichera des informations détaillées sur celui-ci. Cela peut inclure des détails tels que le sujet du certificat, l'émetteur (l'AC qui a délivré le certificat), la date de validité, les extensions du certificat, etc.

```
user@debian:~$ step certificate inspect .step/certs/root_ca.crt
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number: 84449001759808324559597630403536676560 (0x3f88493f9271f645a782e4b40ba2e6d0)
    Signature Algorithm: ECDSA-SHA256
    Issuer: O=pki-ines,CN=pki-ines Root CA
    Validity
      Not Before: Jun 7 08:32:49 2023 UTC
      Not After : Jun 4 08:32:49 2033 UTC
    Subject: O=pki-ines,CN=pki-ines Root CA
    Subject Public Key Info:
      Public Key Algorithm: ECDSA
      Public-Key: (256 bit)
      X:
        8a:a5:96:68:98:0b:65:78:3e:39:ed:35:6b:cf:b3:
        6b:5b:ca:c1:9f:8e:06:c9:80:48:8b:f1:39:3a:37:
        84:c9
      Y:
        f1:e2:5a:82:43:fb:3a:ca:c1:c6:5e:69:10:e4:b3:
        42:7e:61:7c:ea:86:9c:05:ad:e3:75:c6:09:e3:80:
        0e:07
      Curve: P-256
    X509v3 extensions:
      X509v3 Key Usage: critical
        Certificate Sign, CRL Sign
      X509v3 Basic Constraints: critical
        CA:TRUE, pathlen:1
      X509v3 Subject Key Identifier:
        45:91:ED:16:D5:D1:30:A0:BA:9E:58:1B:14:5D:29:E2:A9:F5:F1:12
    Signature Algorithm: ECDSA-SHA256
      30:46:02:21:00:b2:14:20:9d:7c:7a:54:f3:a1:58:4e:e3:d0:
      8f:aa:96:cc:6e:d2:f8:f2:47:0a:3c:44:6b:6e:be:5f:a2:e5:
      50:02:21:00:c2:68:7e:c8:cf:da:9e:9d:6d:7b:bc:69:9c:82:
      f1:0b:d7:49:4d:b6:11:19:33:33:06:f4:08:58:e9:ea:91:36
```

ILLUSTRATION 3.4 – Capture d'écran des informations du certificat. Réalisé par Maya Inès



## d. Certificate Manager

Il est aussi possible de gérer des certificats avec Smallstep Certificate Manager, un produit commercial construit sur Step-CA qui offre une [AC](#) hébergée hautement disponible, des notifications d'expiration et des alertes, un tableau de bord de gestion, une révocation active, une [API](#) <sup>31</sup>, et d'autres fonctionnalités. Smallstep Certificate Manager permet d'émettre facilement des certificats [TLS/SSL](#) <sup>32</sup> privés pour tous les objets. <sup>33</sup>

Cette partie ne sera pas développée dans ce travail.

## 3.2. CERTIFICATS

Un certificat est un fichier électronique qui établit un lien entre des données d'identification et une entité spécifique, telle qu'une personne, une organisation ou un site web. Ce certificat est délivré par une [AC](#) de confiance. Il joue un rôle important dans la sécurité des communications et des transactions en ligne.

Il contient diverses informations, par exemple le nom de l'entité, sa clé publique, la période de validité du certificat et l'identité de l'[AC](#) qui l'a émis. Il est utilisé pour vérifier l'authenticité et l'intégrité des données échangées lors de communications sécurisées, telles que les connexions [HTTPS](#), les signatures électroniques et les échanges de clés.

Les certificats numériques sont un gage de confiance et garantissent la sécurité des transactions et des communications en ligne, tout en assurant l'identification des entités et la protection des données échangées.

Il existe différents types de certificats numériques. Chacun a ses utilisations spécifiques et est émis par une [AC](#) de confiance pour garantir son intégrité et sa validité.

### a. Certificats X.509

Un certificat X.509 est un type courant de certificat numérique qui suit le standard X.509 défini par l'[ITU](#) <sup>34</sup>. Il est largement utilisé dans les [PKI](#) afin de sécuriser les communications et les transactions en ligne.

Il sert dans diverses applications, notamment pour sécuriser les connexions [HTTPS](#), les

---

31. [Application Programming Interface \(API\)](#)

32. [Secure Sockets Layer \(SSL\)](#)

33. [SMALLSTEP](#), [s. d.(k)].

34. [International Telecommunication Union \(ITU\)](#)

signatures électroniques, l'authentification des utilisateurs et la protection des données lors des communications sur Internet. Il joue un rôle essentiel dans l'établissement de la confiance et la garantie de l'authenticité des entités et des données échangées.

Il contient plusieurs informations telles que l'identité du détenteur, la clé publique, la signature numérique, la période de validité et des informations sur l'AC.

Il se trouve généralement au format PEM<sup>35</sup>, format le plus utilisé. Il se présente sous forme d'un fichier texte contenant des parties encodées en ASCII<sup>36</sup> Base64. Chaque élément du certificat est délimité par un en-tête et un pied de page écrits en texte brut. Les formules simples telles que « begin certificate » et « end certificate » indiquent le début et la fin du certificat.<sup>37 38</sup>

Il existe plusieurs types d'extensions pour ce type de format tels que .cert, .crt, .pem et .key.

```
-----BEGIN CERTIFICATE-----
MIIBoDCCAUAwIBAgIRAM0ivKbq0xZ9Pbctj0VxbZswCgYIKoZIzj0EAwIwLjER
MA8GA1UEChMlcGtpLWluZXMxGTAXBgNVBAMTEHBraS1pbmVzIFJvb3QgQ0EwHhcN
MjMwNjEzMTUyMzA5WhcNMzMwNjEwMTUyMzA5WjAuMREwDwYDVQQKEWhwa2ktaW5l
czEZMBcGA1UEAxMQcGtpLWluZXMgUm9vdCBDQTBCZMBMGBYqGSM49AgEGCCqGSM49
AwEHA0IABLsatxvUNth6mLezo4KNECgCaiILigWDfyNy6+LMZhZUqjyQg4oXuXP6
xU6yqAJ0SpSqBvKxFe0qvbGY9zg1PBajRTBDMA4GA1UdDwEB/wQEAwIBBjASBgNV
HRMBAf8ECDAGAQH/AgEBMB0GA1UdDgQWBBSdJI1Vpj9rIAYrWfYmDYIIDSSrSTAK
BggqhkJ0PQQDAgNIADBFAiA4yYdNeIX3DfXacQ7L/zwM1TRf/VVpbybDnRZM//X
uAIhAK2h4M8oEHSlvqVAqmIHMokQVL+sN5rGWxlZ8bkFmHz4
-----END CERTIFICATE-----
```

ILLUSTRATION 3.5 – Capture d'écran d'un certificat. Réalisé par Maya Inès

### 3.3. RENOUELEMENT DES CERTIFICATS

La commande ci-dessous permet de renouveler tous les certificats installés à l'aide de Certbot en utilisant Step-CA.

```
1 sudo REQUESTS_CA_BUNDLE=$(step path)/certs/root_ca.crt certbot renew
```

La tâche de renouvellement par défaut de Certbot est conçue pour s'aligner sur la durée de vie de 90 jours des certificats de Let's Encrypt. Elle est exécutée toutes les 12 heures et les certificats sont renouvelés si leur expiration est dans les 30 jours.

Par conséquent, tenant compte de ce qui précède, la ligne de commande ci-dessus exécute la commande "certbot renew" en utilisant le chemin du certificat racine fourni par Step-CA.

35. Privacy Enhanced Mail (PEM)

36. American Standard Code for Information Interchange (ASCII)

37. ADELIND, 2020.

38. GENIORAMA, 2021.

```

user@serveur:~$ sudo REQUESTS_CA_BUNDLE=$(step path)/certs/root_ca.crt certbot renew
Saving debug log to /var/log/letsencrypt/letsencrypt.log

- - - - -
Processing /etc/letsencrypt/renewal/serveur.conf
- - - - -
Cannot extract OCSP URI from /etc/letsencrypt/archive/serveur/cert1.pem
Cert is due for renewal, auto-renewing...
Plugins selected: Authenticator standalone, Installer None
Renewing an existing certificate for serveur
Performing the following challenges:
http-01 challenge for serveur
Waiting for verification...
Cleaning up challenges

- - - - -
new certificate deployed without reload, fullchain is
/etc/letsencrypt/live/serveur/fullchain.pem
- - - - -

Congratulations, all renewals succeeded:
  /etc/letsencrypt/live/serveur/fullchain.pem (success)
- - - - -

```

ILLUSTRATION 3.6 – Capture d’écran de la commande de renouvellement de certificat. Réalisé par Maya Inès

Il est également possible de configurer Certbot pour ne renouveler les certificats que lorsqu’ils sont à quelques heures de leur expiration. Il suffit de décommander cette ligne `renew_before_expiry = 30 days` qui se trouve dans le fichier de configuration spécifique au domaine, `/etc/letsencrypt/renewal/serveur.conf`

```

user@serveur:~$ sudo cat /etc/letsencrypt/renewal/serveur.conf
renew_before_expiry = 60 days
version = 1.12.0
archive_dir = /etc/letsencrypt/archive/serveur
cert = /etc/letsencrypt/live/serveur/cert.pem
privkey = /etc/letsencrypt/live/serveur/privkey.pem
chain = /etc/letsencrypt/live/serveur/chain.pem
fullchain = /etc/letsencrypt/live/serveur/fullchain.pem

# Options used in the renewal process
[renewalparams]
account = 889a3cb4fc3683ec7fd7fb3d8943414b
server = https://192.168.1.103:4444/acme/acme/directory
authenticator = standalone

```

ILLUSTRATION 3.7 – Capture d’écran du contenu du fichier de configuration. Réalisé par Maya Inès

De plus, il existe la possibilité de modifier le nombre de jours avant expiration à partir duquel le renouvellement doit être effectué. Il suffit de remplacer le nombre "30" par "60" jours ou le nombre de jours souhaité.

Ces options permettent de personnaliser le processus de renouvellement des certificats selon des besoins spécifiques.

### 3.4. LISTE DE RÉVOCATION DE CERTIFICATS (CLR)

Une **CRL**<sup>39</sup> est la liste contenant tous les certificats qui ne sont plus valides et auxquels il ne faut plus faire confiance. Elle est créée par l'**AC**. Elle est régulièrement mise à jour et répertorie les certificats numériques qui ont été révoqués avant leur date d'expiration.

Bien que les certificats numériques aient une période de validité, ladite liste est nécessaire étant donné que parfois le certificat de l'entité peut être révoqué de façon anticipée suite à la perte de la clé privée, la résiliation prématurée de la période de validité ou la fin de l'accès à un système ou à des ressources.

Elle est publiée par l'**AC** qui a émis le certificat et peut être consultée par les utilisateurs ou les systèmes lorsqu'ils souhaitent vérifier la validité d'un certificat. Lorsque l'un d'eux se trouve dans la liste de révocation, il est considéré comme non valide et les communications qui l'utilisent sont refusées.

Les **CRL** peuvent être distribuées de différentes manières, notamment via des protocoles tels que **OCSP**<sup>40</sup>, qui permet de vérifier l'état d'un certificat en temps réel, ou via des fichiers périodiquement mis à jour ; offrant la possibilité d'être téléchargés et consultés par des utilisateurs.

Il s'agit d'un mécanisme crucial dans une **PKI** pour garantir que les certificats numériques révoqués ne soient pas utilisés pour des activités malveillantes ou non autorisées.

### 3.5. CERTBOT (LET'S ENCRYPT)

Certbot est un logiciel libre et gratuit qui permet d'utiliser automatiquement des certificats Let's Encrypt sur des sites web administrés manuellement afin d'activer le protocole **HTTPS**.<sup>41</sup>

Il existe plusieurs façons d'obtenir un certificat **SSL** et/ou **TLS** pour un site web dont Let's Encrypt.

Let's Encrypt est une **AC** gratuite et automatisée qui fournit des certificats **SSL/TLS**. Afin d'obtenir un certificat pour le domaine, il faut prouver avoir le contrôle du domaine en utilisant un logiciel qui utilise le protocole **ACME**.<sup>42</sup>

---

39. Certificate Revocation List (CRL)

40. Online Certificate Status Protocol (OCSP)

41. CERTBOT, [s. d.(a)].

42. ENCRYPT, [s. d.].

## a. Installation

Pour l'installation de Certbot, nous avons suivi la marche à suivre disponible sur le site officiel<sup>43</sup>. Il faut renseigner le type de serveur utilisé ainsi que le système d'exploitation sur lequel il tourne. En ce qui nous concerne, il s'agit de Nginx et Debian 11. Ensuite il faut respecter la marche à suivre figurant sur la documentation officielle.

Il faut commencer par taper les commandes suivantes :

```
1 sudo apt-get update
2 sudo apt-get install software-properties-common
3 sudo add-apt-repository ppa:certbot/certbot
4 sudo apt-get update
5 sudo apt-get install certbot
```

La première permet de mettre à jour les paquets sur le système. La seconde installe le paquet "software-properties-common" nécessaire pour gérer les référentiels logiciels. La troisième ajoute le référentiel PPA Certbot pour obtenir la dernière version de Certbot. À la suite, il faut remettre à jour à nouveau les paquets sur le système pour inclure le nouveau référentiel. Pour finir, la dernière commande installe certbot.

Il faut se connecter via [SSH](#) sur le serveur qui héberge le site web [HTTP](#) en utilisant les privilèges sudo. Puis installer snapd en suivant les instructions pour activer le support classic snap. Pour finir, exécuter les lignes de commande dans le terminal de la machine afin de s'assurer de disposer de la dernière version de snapd.

```
1 sudo snap install core ; sudo snap refresh core
```

Dans le cas où des paquets Certbot ont été installés à l'aide d'un gestionnaire de paquets du système d'exploitation comme apt, dnf ou yum, il faut les supprimer avant d'installer le snap Certbot afin de s'assurer que lorsque la commande certbot est exécutée, c'est le snap qui est utilisé et non l'installation effectuée à partir du gestionnaire de paquets du système d'exploitation. La commande exacte pour ce faire dépend du système d'exploitation, mais pour les distributions basées sous Debian, la commande est :

```
1 sudo apt-get remove certbot
```

Pour installer Certbot sur la machine en utilisant Snap, il faut exécuter la commande suivante.

---

43. Disponible à l'adresse : <https://certbot.eff.org/instructions>

```
1 sudo snap install --classic certbot
```

Pour s'assurer que la commande certbot peut être effective, il est nécessaire d'exécuter l'instruction suivante en ligne de commande sur la machine.

```
1 sudo ln -s /snap/bin/certbot /usr/bin/certbot
```

Après avoir installé Certbot, il existe deux options pour l'exécuter. Les deux se valent.

La première option est d'obtenir et installer des certificats avec configuration automatique pour Nginx. Pour obtenir un certificat et permettre à Certbot de modifier automatiquement la configuration de Nginx pour activer l'accès [HTTPS](#) en une seule étape, il suffit d'exécuter :

```
1 sudo certbot --nginx
```

Sinon la seconde option est d'obtenir un certificat avec configuration manuelle de Nginx. Dans le cas où l'utilisateur préfère modifier manuellement la configuration de Nginx, il faut exécuter cette commande-ci.

```
1 sudo certbot certonly --nginx
```

En cas d'erreur comme celle-ci :

```
1 Could not choose appropriate plugin : The requested nginx plugin does not appear to be
  installed
2 The requested nginx plugin does not appear to be installed
```

Il faut installer le plugin avec cette commande :

```
1 sudo apt-get install python3-certbot-nginx
```

Les paquets Certbot du système sont livrés avec une tâche cron ou un timer systemd qui renouvelle automatiquement les certificats avant leur date d'expiration. Cela permet à l'utilisateur de ne pas avoir besoin de relancer Certbot, à moins qu'il ne change la configuration.

Il est possible de tester le renouvellement automatique des certificats. La commande ci-dessous est utilisée pour effectuer une simulation (dry run) du renouvellement des certificats gérés par Certbot. Elle permet de vérifier si le renouvellement automatique des certificats fonctionne sans l'effectuer réellement.

```
1 sudo certbot renew --dry-run
```

En exécutant cette commande, Certbot simule le processus de renouvellement des certificats en vérifiant leur statut et en effectuant les étapes nécessaires sans réellement renouveler les

certificats. Cela permet de vérifier si tout est correctement configuré et si le renouvellement se déroulera sans problème lorsqu'il sera exécuté automatiquement selon le calendrier prédéfini.

L'utilisation de l'option "`--dry-run`" est recommandée afin d'effectuer des tests périodiques pour s'assurer que le processus de renouvellement des certificats fonctionne correctement. Cela permet de détecter et de résoudre les éventuelles erreurs ou problèmes avant que les certificats ne se rapprochent de leur expiration réelle.

La commande `certbot` possède de nombreuses sous-commandes, que ce soit pour l'installation, l'obtention et la rénovation mais également pour le management de certificats et de comptes. Il est possible de toutes les retrouver avec la commande : `certbot help`.

### 3.6. SSL ou TLS

[SSL](#) et [TLS](#) sont deux protocoles de sécurité utilisés dans le but de sécuriser les données lors de communication sur Internet. Bien qu'ils remplissent essentiellement le même objectif, [TLS](#) est la version plus récente et améliorée de [SSL](#).

[SSL](#) crée une connexion sécurisée entre un client (tel qu'un navigateur web) et un serveur web. Il utilise des protocoles de chiffrement afin de garantir la confidentialité et l'intégrité des données échangées. La communication commence par une poignée de main (handshake) où le client et le serveur négocient les paramètres de sécurité, y compris les algorithmes de chiffrement à utiliser. [SSL](#) a connu plusieurs vulnérabilités au fil du temps, ce qui a conduit à son abandon progressif au profit de [TLS](#). [SSL 3.1](#) a été remplacé par [TLS 1.0](#).

[TLS](#) est une évolution de [SSL](#) pour remédier à ses vulnérabilités. Il fonctionne de manière similaire à [SSL](#). Il établit également une connexion sécurisée en utilisant un processus de poignée de main pour négocier les paramètres de sécurité entre le client et le serveur. Il utilise des algorithmes de chiffrement modernes et résistants aux attaques. Il est considéré comme plus sécurisé que [SSL](#) en raison de ses améliorations continues en matière de sécurité.

La version utilisée est [TLS 1.2](#) ou [1.3](#).

## CHAPITRE 4 : ACME PROTOCOL

Le protocole **ACME** est un protocole de communication dont le but est d'automatiser les relations entre une machine et l'**AC** qui est chargée de délivrer les certificats.

Il permet à l'**AC** de communiquer directement avec le serveur afin d'obtenir et installer des certificats **SSL/TLS**.<sup>44</sup>

Step-CA prend en charge le protocole **ACME**. Il est possible d'obtenir des certificats X.509 de sa propre **AC** à l'aide de clients et de bibliothèques **ACME** populaires, ou via le client **ACME** intégré à la commande step.

**ACME** est un protocole moderne et standardisé pour la validation et l'émission automatique de certificats X.509 d'une **AC** à des clients.

Les machines peuvent obtenir des certificats d'une **AC** sans aucune interaction humaine avec **ACME**.

Il est possible d'utiliser **ACME** en production pour émettre des certificats X.509 pour les charges de travail internes, les proxies, les files d'attente, les bases de données, etc. afin d'utiliser **TLS** mutuel pour l'authentification et le cryptage.

Un client **ACME** peut demander un certificat X.509 à une **AC**. Il crée un compte auprès d'un serveur **ACME** et soumet une commande de certificat. Il faut considérer un compte **ACME** comme un emplacement où sont stockées les demandes de certificat en cours pour ce client particulier.

L'**AC** répond par une série de défis. Pour les relever, le client doit prouver qu'il contrôle chaque nom de sujet (nom de domaine, adresse **IP** ou ID de périphérique) qu'il demande dans le certificat. Elle vérifie ensuite les réponses du client.

Une fois que le client a relevé avec succès les défis de l'**ACME**, il soumet une **CSR**<sup>45</sup> à l'**AC**. La **CSR** est un message crypté spécialement formaté envoyé par un demandeur de certificat numérique **SSL** à une **AC**. La **CSR** valide les informations dont l'**AC** a besoin pour délivrer un certificat. Elle vérifie que le client contrôle la clé privée associée à la demande de certificat. Puis, elle délivre un certificat au client.<sup>46</sup>

---

44. SMALLSTEP, [s. d.(a)].

45. **Certificate Signing Request (CSR)**

46. SMALLSTEP, [s. d.(a)].



## 4.1. TYPES DE DÉFIS ACME

Il n'existe pas de méthode standard unique pour prouver le contrôle d'un identifiant dans le cadre de l'ACME. La spécification de base de l'ACME reconnaît cette diversité et prévoit des extensions pour répondre à ce besoin.

Les clients ACME utilisent généralement les protocoles HTTP, DNS ou TLS pour relever les défis de validation. Ces protocoles sont couramment utilisés pour prouver le contrôle d'un identifiant lors du processus d'obtention de certificats.

| Type d'identifiant | dns-01 | tls-alpn-01 | device-attest-01 |
|--------------------|--------|-------------|------------------|
| Adresse IP         | O      | X           | O                |
| Hostname           | O      | O           | O                |
| ID du périphérique | X      | X           | X                |

TABLEAU 4.1 – Types de challenge ACME. Source : tiré de <https://smallstep.com>, ref. URL01.

## 4.2. CHALLENGE HTTP (HTTP-01)

Ce défi est un type de défi utilisé par les AC pour vérifier le contrôle sur un domaine lors de la demande d'un certificat SSL/TLS.

Afin de réussir le défi HTTP-01, il faut créer un fichier contenant un jeton aléatoire et l'empreinte de la clé de compte sur le serveur web.<sup>47</sup> Ce fichier doit être accessible à partir d'une URL<sup>48</sup> spécifique qui est fournie par l'AC.<sup>49</sup>

Le principal avantage du défi HTTP-01 est sa facilité d'automatisation pour les plateformes de serveurs web telles qu'Apache et Nginx.<sup>50</sup>

L'AC ACME demande au client d'héberger un nombre aléatoire à une URL aléatoire sous `/.well-known/acme-challenge` sur le port 80. L'AC vérifie le contrôle du client en émettant une requête HTTP GET vers cette URL.

Il s'agit d'un bon type de défi à usage général. En hébergeant la réponse au défi via HTTP sur le port 80, le client prouve qu'il contrôle un port protégé sur le domaine demandé. Le type de défi HTTP-01 est le plus facile à mettre en place, car n'importe quel serveur web permet d'héberger la réponse au défi sous la forme d'un fichier statique.

47. Team, 2020.

48. Uniform Resource Locator (URL)

49. WEB, [s. d.].

50. Team, 2020.

### 4.3. CHALLENGE DNS (DNS-01)

L'ACME AC demande au client de fournir un enregistrement DNS TXT aléatoire pour le domaine en question. Elle vérifie le défi en interrogeant le DNS pour cet enregistrement TXT.

Le type de défi DNS-01 est utile si le serveur ACME ne peut pas atteindre directement le domaine demandé. Le serveur doit seulement être en mesure d'effectuer une recherche DNS afin de confirmer le défi. Cependant, comme le client ACME doit modifier les enregistrements DNS, la configuration d'un client DNS-01 est généralement plus complexe.

### 4.4. PROTOCOLE ACME AVEC STEP-CA

Dans cette partie, nous allons utiliser Step-CA installé dans le chapitre précédent. Nous avons suivi le tutoriel "Run your own private CA & ACME server using step-ca"<sup>51</sup>

Afin d'activer ACME, il suffit d'ajouter un provisionner ACME à la configuration Step-CA en exécutant la commande suivante :

```
1 step ca provisioner add acme --type ACME
```

```
user@step-ca:~$ step ca provisioner add acme --type ACME
No admin credentials found. You must login to execute admin commands.
✓ Please enter admin name/subject (e.g., name@example.com): user@pki.com
✓ Provisioner: smallstep-sso (OIDC) [client: a2821165-4b7c-4ac5-ac6c-6202623a2dec]
Your default web browser has been opened to visit:
https://auth.smallstep.com/oidc/ines/auth?client_id=a2821165-4b7c-4ac5-ac6c-6202623a2dec&code_challenge=8vZmrrbxAgAZSHvYa-eLP3hePrDDk8j63Q-I-sRGVDA&c
6c5aa381f76cbc4735beb6503771e6f226&redirect_uri=http%3A%2F%2F127.0.0.1%3A10000&response_type=code&scope=openid+email&state=pztuZrBrLuLy09jaxap409ALWY
adminHandler.authorizeToken; unable to load admin with subject(s) [aa6b494e-6754-461e-b8f2-2555579c39ed ines.maya@etu.hesge.ch] and provisioner 'step
```

ILLUSTRATION 4.1 – Capture d'écran de la commande de l'ajout du provisionner. Réalisé par Maya Inès

Ensuite, il faut redémarrer Step-CA afin qu'il prenne en compte la nouvelle configuration.

Pour configurer un client ACME afin qu'il se connecte à Step-CA, il suffit de diriger le client vers l'URL du répertoire ACME approprié et d'indiquer au client qu'il doit faire confiance au certificat racine de l'AC.

Une fois les certificats émis, il faut également s'assurer qu'ils sont renouvelés avant leur expiration.

La majorité des clients ACME se connecte par défaut à l'AC de Let's Encrypt. Pour se connecter à Step-CA, il faut pointer le client vers la bonne URL du répertoire ACME.

Une seule instance de Step-CA peut avoir plusieurs provisionners ACME, chacun avec sa

51. Disponible à l'adresse : <https://smallstep.com/blog/private-acme-server/>

propre URL de répertoire ACME qui ressemble à : `https://ca-host/acme/nom du provisionner/-répertoire`

Plus haut, un provisionner ACME nommé "acme" a été ajouté. L'URL de son répertoire ACME est : `https://pki.ines.lan/acme/acme/directory`

La communication entre un client et un serveur ACME utilise toujours HTTPS. Par défaut, les clients valident le certificat HTTPS du serveur en utilisant les certificats racine publics de la liste de confiance par défaut du système. Cela ne pose pas de problème lors d'une connexion à Let's Encrypt : il s'agit d'une AC publique et son certificat racine se trouve déjà dans la liste de confiance par défaut du système. Le certificat racine interne ne l'est pas, donc les connexions HTTPS des clients ACME à Step-CA échouent.

Il y a deux façons de résoudre ce problème. Soit en configurant explicitement le client ACME pour qu'il fasse confiance au certificat racine de Step-CA, ou alors en ajoutant le certificat racine de Step-CA à la liste de confiance par défaut du système (par exemple, grâce à la commande `step certificate install`).

En cas d'utilisation d'AC pour TLS en production, configurer explicitement le client ACME pour qu'il ne fasse confiance qu'au certificat racine est une meilleure option.

En simulation de Let's Encrypt en préproduction, l'installation du certificat racine est une simulation plus fidèle de la production. Une fois le certificat racine installé, aucune autre configuration du client n'est nécessaire.

Pour obtenir un certificat de Step-CA à l'aide de Certbot, il faut diriger Certbot vers l'URL du répertoire ACME à l'aide de l'option `--server`, mais également indiquer à Certbot de faire confiance au certificat racine en utilisant la variable d'environnement `REQUESTS_CA_BUNDLE`.

La commande complète est la suivante :

```
1 sudo REQUESTS_CA_BUNDLE=(step path)/certs/root_ca.crt Certbot certonly -n --webroot -d
    web.ines.lan --server https://pki.ines.lan/acme/acme/directory --agree-tos --email "
    ines.maya@etu.hesge.ch"
```

où `(step path)/certs/root_ca.crt` représente le chemin au certificat root.

Dans notre cas, `web.ines.lan`, après l'option `-d`, est le nom de la machine (hostname) serveur sur lequel le programme tourne.

L'URL, `pki.ines.lan`, a été donnée lors de la config Step-CA dans le chapitre précédent.

Dans l'URL, le 1er ACME est le nom du provisionner donné un peu plus tôt, lors de l'activation de l'ACME.

L'instruction sudo est nécessaire dans le mode autonome de Certbot afin qu'il puisse écouter sur le port 80 pour relever le défi HTTP-01.

Certbot prend en charge le défi DNS-01 en utilisant le plugin approprié, ce qui permet de valider les certificats via des enregistrements DNS au lieu de la méthode HTTP. Ce défi est compatible avec la plupart des fournisseurs de services DNS.

En plus du défi DNS-01, Certbot dispose également d'intégrations plus avancées avec les serveurs web populaires tels que Nginx et Apache. Ces intégrations permettent de configurer automatiquement le serveur web pour utiliser HTTPS une fois que le certificat est obtenu et installé avec succès. Cela facilite grandement la mise en place de connexions sécurisées sur le site web.

Toutes ces fonctionnalités de Certbot, y compris le support du défi DNS-01 et les intégrations avec les serveurs web, sont compatibles avec l'utilisation de Step-CA. Il est possible d'utiliser Certbot avec Step-CA pour obtenir, renouveler et configurer des certificats TLS de manière automatisée et sécurisée.

## CHAPITRE 5 : SSH

SSH est un protocole qui permet d'établir une communication chiffrée et sécurisée sur un réseau informatique entre une machine locale (client) et une machine distante (serveur). Cela peut servir à exécuter des commandes ou transférer des fichiers par exemple.

Il existe plusieurs raisons pour lesquelles SSH est utilisé.

La première est la sécurité. SSH offre une communication sécurisée grâce à des méthodes de chiffrement robustes. Les données échangées entre le client SSH et le serveur sont protégées contre les attaques d'interception et d'écoute non autorisées. Cela garantit que les informations sensibles soient protégées pendant la transmission.

De plus, SSH possède une authentification forte, basée sur des clés publiques et privées. Cela signifie que pour se connecter à un serveur SSH, l'utilisateur doit fournir une paire de clés, composée d'une clé publique et d'une clé privée. Cette méthode d'authentification est plus sécurisée que les simples combinaisons de nom d'utilisateur et de mot de passe, car les clés privées sont généralement stockées de manière sécurisée sur l'ordinateur local de l'utilisateur.

En outre, SSH permet d'accéder et de contrôler à distance des systèmes informatiques. Cela signifie qu'il est possible de se connecter à un serveur distant depuis n'importe quel endroit et exécuter des commandes, transférer des fichiers ou administrer le système à distance. Cela facilite la gestion des serveurs et des réseaux.

Pour finir, SSH est natif. En effet, il est largement pris en charge sur différentes plateformes, dont Linux, MacOS et Windows. Il est possible d'utiliser des clients SSH disponibles sur ces systèmes d'exploitation pour se connecter à des serveurs distants, ce qui facilite la gestion des systèmes hétérogènes.

### 5.1. FONCTIONNEMENT

Il utilise un modèle client-serveur afin de permettre l'authentification de deux systèmes distants et le chiffrement des données qui transitent entre eux. Il utilise par défaut le port TCP<sup>52</sup> 22, mais celui-ci peut être changé.

Une connexion SSH utilise un processus à plusieurs étapes pour établir une communication sécurisée entre un client SSH et un serveur SSH.

---

52. Transmission Control Protocol (TCP)

La première étape est d'établir la connexion. Le client **SSH** envoie une demande de connexion au serveur **SSH** sur le port **TCP 22**. Le serveur **SSH** écoute les demandes de connexion entrantes.

Une fois la demande de connexion reçue, le serveur **SSH** répond avec son identification et les paramètres de chiffrement pris en charge. Le client **SSH** vérifie l'authenticité du serveur en vérifiant sa clé publique.

L'étape suivante est l'échange de clés. Le client **SSH**, qui a généré auparavant une paire de clés cryptographiques : une publique et une privée, envoie la clé publique au serveur **SSH**. Ce dernier l'utilise pour chiffrer une clé de session générée aléatoirement.

Le serveur **SSH** envoie la clé de session chiffrée au client **SSH**, qui la déchiffre à l'aide de sa clé privée. À partir de ce moment, le client et le serveur partagent une clé de session commune pour le chiffrement des données échangées.

Une fois que la clé de session est partagée, une session **SSH** est établie entre le client et le serveur. Toutes les données échangées entre eux sont chiffrées à l'aide de ladite clé.

Une fois la session établie, le client **SSH** envoie les informations d'identification de l'utilisateur (nom d'utilisateur et éventuellement le mot de passe ou la clé publique) au serveur **SSH** pour l'authentification. Le serveur vérifie les informations d'identification et permet ou refuse l'accès en conséquence. Une fois l'authentification réussie, l'utilisateur peut utiliser le terminal **SSH** pour exécuter des commandes sur le serveur distant.

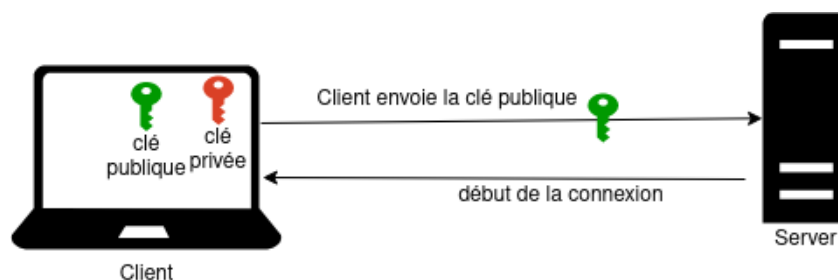


ILLUSTRATION 5.1 – Schéma d'une connexion avec une paire de clé privée/publique. Réalisé par Maya Inès

## 5.2. FONCTIONNEMENT AVEC CERTIFICAT

Le fonctionnement d'une connexion avec certificat diffère de celui avec une paire de clés.

Tout d'abord, l'utilisateur génère une paire de clés **SSH** (une clé privée et une clé publique) sur son client **SSH**. Ensuite, la clé publique est signée par une **AC** pour créer un certificat. Le

certificat contient la clé publique de l'utilisateur et est signé par l'**AC**, ce qui garantit l'authenticité de la clé publique.

La deuxième étape est la configuration de l'**AC**. Elle est configurée pour accepter et gérer les certificats des utilisateurs. Elle possède une liste des certificats de confiance et peut vérifier si un certificat est valide et s'il est associé à un utilisateur autorisé.

L'étape suivante est la connexion **SSH** avec certificats. Lorsque l'utilisateur tente de se connecter à l'hôte distant, le client **SSH** envoie son certificat signé à l'hôte. Ce dernier vérifie la validité du certificat en utilisant la clé publique de l'**AC** pour vérifier la signature. Si le certificat est valide, l'hôte accepte la connexion et authentifie l'utilisateur associé au certificat.

Une fois l'authentification réussie, une session **SSH** sécurisée est établie entre le client et l'hôte. Toutes les communications entre eux sont cryptées afin d'assurer la confidentialité des données échangées pendant la session.

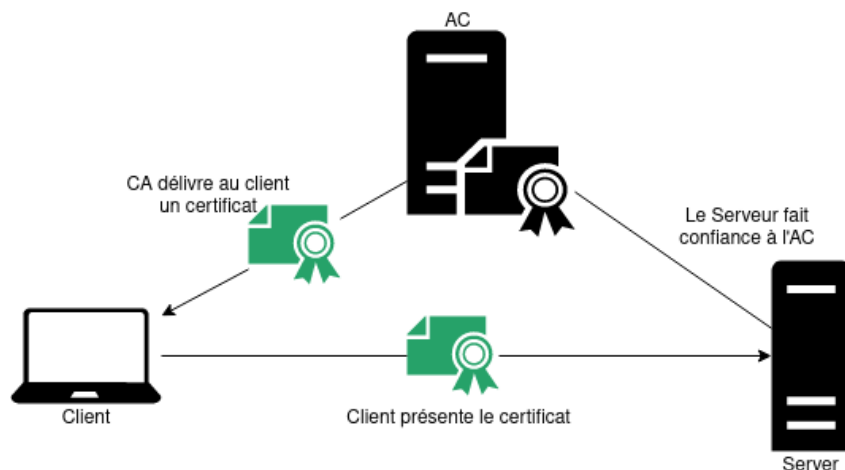


ILLUSTRATION 5.2 – Schéma d'une connexion SSH avec certificat. Réalisé par Maya Inès

L'utilisation de certificats **SSH** offre plusieurs avantages, tels que la simplification de la gestion des clés, la possibilité de révoquer rapidement des certificats en cas de besoin, et une meilleure sécurité en évitant la transmission des clés privées sur le réseau.

### 5.3. RÔLES

#### a. Serveur d'accès à distance

Le serveur d'accès à distance est responsable de l'authentification et de l'autorisation des connexions **SSH**. Il gère les clés et les certificats des utilisateurs, ainsi que les politiques de sécu-

rité. Le serveur **AC** est configuré pour autoriser ou refuser les connexions en fonction de règles définies. Il peut également effectuer des tâches supplémentaires telles que l'enregistrement des activités des utilisateurs, le suivi des connexions, etc.

## **b. Client SSH**

Le client **SSH** est l'outil utilisé par les utilisateurs pour se connecter à distance à un hôte via **SSH**. Il peut être une application en ligne de commande, comme OpenSSH ou une interface graphique. Le client **SSH** permet à l'utilisateur d'initier une connexion sécurisée avec un hôte distant en fournissant les informations d'authentification nécessaires, telles que l'adresse **IP** de l'hôte, le nom d'utilisateur et les clés privées.

## **c. Hôte**

L'hôte est le serveur distant auquel le client **SSH** se connecte. Il peut s'agir d'un serveur Linux, MacOS ou tout autre système d'exploitation prenant en charge le protocole **SSH**. Il reçoit les connexions **SSH** et vérifie les informations d'authentification fournies par le client. Si les informations sont valides, il autorise la connexion et ouvre une session sécurisée entre le client et lui-même.

## **5.4. SMALLSTEP SSH**

Cette partie montre comment se connecter en **SSH** à l'aide de certificats grâce à l'outil Smallstep **SSH**, un produit commercial qui offre un flux de travail **SSH** complet de bout en bout, combinant les avantages des fournisseurs d'identité modernes, des certificats **SSH** et des technologies standard éprouvées comme OpenSSH et OAuth.<sup>53</sup>

Pour cette partie, nous avons suivi le guide "Getting Started with Smallstep SSH" fourni par Smallstep<sup>54</sup>.

Avant d'utiliser Smallstep **SSH**, il est nécessaire de créer un compte. Pour ce faire, il faut fournir un nom d'équipe, une adresse e-mail, un nom d'utilisateur et un mot de passe.

Une fois que ces informations ont été fournies, il est possible d'accéder à la page qui s'ouvre et suivre les instructions fournies.

L'**URL** de la page à laquelle il faut accéder pour continuer est : [https ://small-](https://small-)

---

53. SMALLSTEP, [s. d.(h)].

54. Disponible à l'adresse : [https ://smallstep.com/docs/ssh/#step-1-set-up-your-client](https://smallstep.com/docs/ssh/#step-1-set-up-your-client)



step.com/app/team/cm/authorities/add/acme

Team dans l'URL ci-dessus est le nom de l'équipe fournie précédemment lors de la création du compte. Dans notre cas, le nom de l'équipe est mycom. Ce qui donne l'URL suivant : <https://smallstep.com/app/mycom/cm/authorities/add/acme>.

### a. Côté client

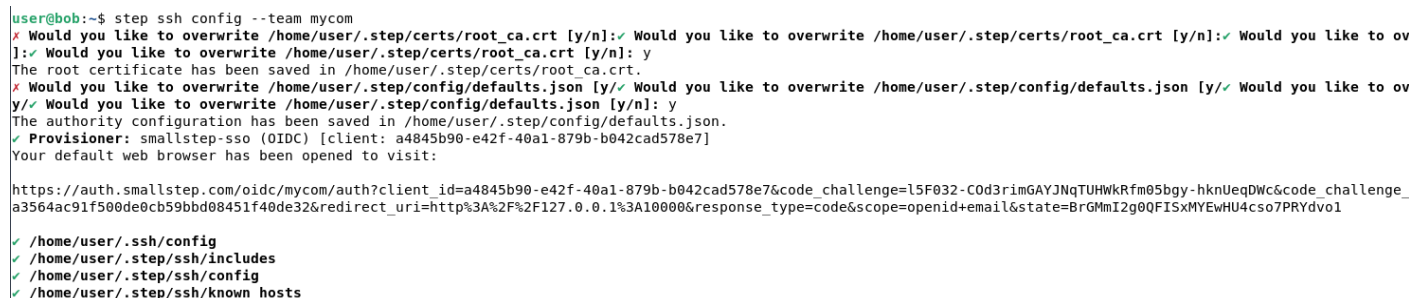
Pour la machine client, qui va se connecter, il est nécessaire de taper la commande ci-dessous pour une machine Linux, comme dans notre cas. Cette commande permet de configurer le client OpenSSH pour se connecter aux hôtes gérés par Smallstep en utilisant l'authentification unique.

```
curl -L -o step https://dl.step.sm/s3/cli/ui-cli-install/step_latest_linux_amd64 && sudo
install -m 0755 -t /usr/bin step
```

Cette commande télécharge et installe l'outil de ligne de commande Step, qui est utilisé pour configurer le client SSH. Ensuite, il faut le configurer en exécutant la commande suivante :

```
step ssh config --team mycom
```

Cette commande configure les paramètres SSH pour le client afin de se connecter aux hôtes gérés par Smallstep. Dans notre cas, "mycom" est le nom de notre équipe mais il peut être changé en fonction du nom de l'équipe choisie précédemment.



```
user@bob:~$ step ssh config --team mycom
x Would you like to overwrite /home/user/.step/certs/root_ca.crt [y/n]: y Would you like to overwrite /home/user/.step/certs/root_ca.crt [y/n]: y Would you like to ov
]: y Would you like to overwrite /home/user/.step/certs/root_ca.crt [y/n]: y
The root certificate has been saved in /home/user/.step/certs/root_ca.crt.
x Would you like to overwrite /home/user/.step/config/defaults.json [y/n]: y Would you like to overwrite /home/user/.step/config/defaults.json [y/n]: y Would you like to ov
y: y Would you like to overwrite /home/user/.step/config/defaults.json [y/n]: y
The authority configuration has been saved in /home/user/.step/config/defaults.json.
✓ Provisioner: smallstep-sso (OIDC) [client: a4845b90-e42f-40a1-879b-b042cad578e7]
Your default web browser has been opened to visit:
https://auth.smallstep.com/oidc/mycom/auth?client_id=a4845b90-e42f-40a1-879b-b042cad578e7&code_challenge=L5F032-C0d3rimGAYJNqTUHWkRfm05bgy-hknUeqDWc&code_challenge_
a3564ac91f500de0cb59bbd08451f40de32&redirect_uri=http%3A%2F%2F127.0.0.1%3A10000&response_type=code&scope=openid+email+state=BrGMmI2g0QFISxMYEwHU4cso7PRYdvo1
✓ /home/user/.ssh/config
✓ /home/user/.step/ssh/includes
✓ /home/user/.step/ssh/config
✓ /home/user/.step/known_hosts
```

ILLUSTRATION 5.3 – Capture d'écran de la configuration du client. Réalisé par Maya Inès

Il faut s'assurer que l'agent SSH est en cours d'exécution avant de configurer OpenSSH sur la machine à l'aide de l'outil Step CLI. L'agent SSH gère les clés d'authentification et facilite les connexions sécurisées.

Une fois que vous avez exécuté ces commandes, le client SSH est configuré pour se connecter aux hôtes gérés par Smallstep en utilisant l'authentification unique. Il est maintenant possible d'utiliser l'outil Step CLI pour gérer les connexions SSH de manière sécurisée

Par la suite, une page s'ouvre avec le message ci-dessous apparaît.

*"Votre client Smallstep a demandé à vous identifier en utilisant votre compte Smallstep et votre adresse électronique. L'acceptation de cette demande permettra à votre autorité de certification d'émettre un certificat vous identifiant en tant qu'administrateur de l'autorité de certification. Ce certificat est utilisé pour les commandes d'administration de l'AC (c'est-à-dire le provisionneur, le modèle et la gestion de l'administrateur)"*

Une fois que celle-ci apparaît, il faut se connecter au compte et autoriser en cliquant sur le bouton prévu à cet effet. Le client devient autorisé et le message ci-dessous s'affiche.

*"Votre client est autorisé! Retournez à votre terminal pour continuer le flux. N'hésitez pas à fermer cette page."*

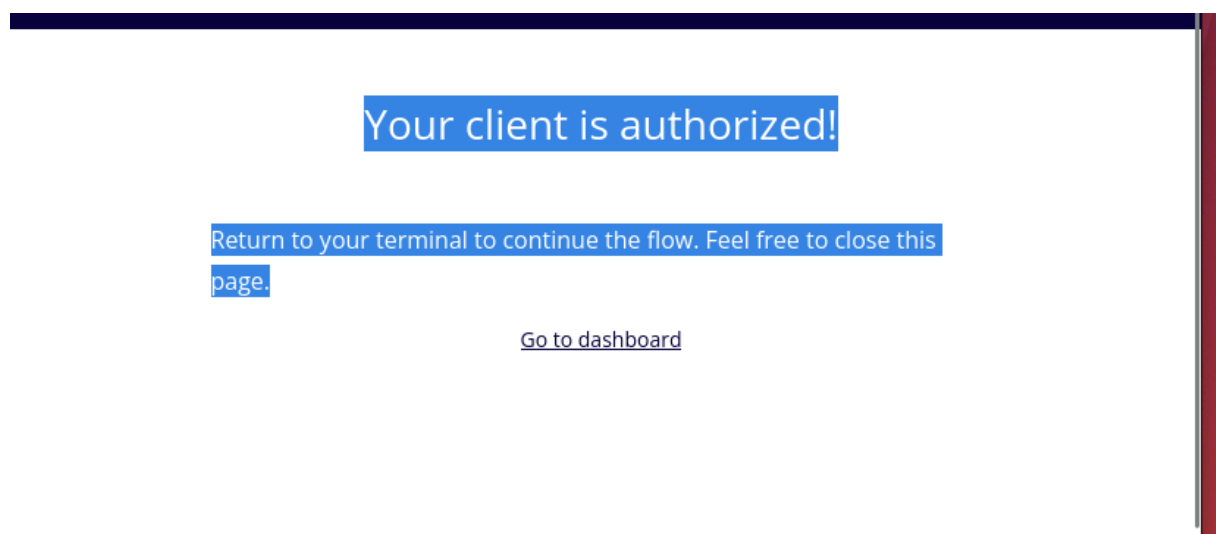


ILLUSTRATION 5.4 – Capture d'écran du message à l'autorisation du client. Réalisé par Maya Inès

Ensuite il faut exécuter la commande suivante :

```
1 ssh ssh-test.app.smallstep.com
```

En exécutant la commande, `step ssh hosts`, il est possible de demander la liste des hôtes gérés par Smallstep sur lesquels le client possède des droits d'accès.

```
user@bob:~$ ssh ssh-test.app.smallstep.com
👉 pah-pa-rah! You have successfully connected to the smallstep test server 🏆

We have authenticated you as:
  fingerprint SHA256:4kk0gxGtkap9SAGEQTB27P2ETf9b8c5nkp0CNUF6zNI
  username 'bob'
  uid:gid 128000:128000
  team 'project'

Your next step is to register a host. This will allow you to test ssh access
in your own environment.

Connection to ssh-test.app.smallstep.com closed.
```

ILLUSTRATION 5.5 – Capture d’écran de la commande `ssh ssh-test.app.smallstep.com`. Réalisé par Maya Inès

En dernier lieu, pour se connecter, il faut exécuter la commande ci-dessous Elle permet la connexion sans mot de passe.

```
1 ssh nom-hote
```

Dans notre cas, le nom de l’hôte est `ssh.ines.lan`.

## b. Côté host

Sur la machine hôte, sur laquelle la connexion sera effectuée, il faut exécuter les commandes suivantes en tant que root :

```
1 curl -sSL0 https://files.smallstep.com/ssh-host.sh
2 bash ssh-host.sh --is-bastion
```

Ces commandes téléchargent un script appelé "`ssh-host.sh`" depuis le site Smallstep et l’exécutent avec l’option "`--is-bastion`" pour configurer la machine comme un bastion (hôte intermédiaire) pour les connexions SSH.

```

root@ssh:/home/user# curl -sSLO https://files.smallstep.com/ssh-host.sh bash ssh
-host.sh --is-bastion
curl: option --is-bastion: is unknown
curl: try 'curl --help' or 'curl --manual' for more information
root@ssh:/home/user# curl -sSLO https://files.smallstep.com/ssh-host.sh
bash ssh-host.sh --is-bastion

A team is required!
Team: project

A SSH enrollment token is required!
Enrollment token (hidden):
One or more tags are required!
Example: foo=bar bax=fax mop.top=cat-bat
Tag(s): foo=user bax=b

Required variables:
    Team:          project
    Token:         project...snip...
    Hostname:      ssh.ines.lan
    Tag(s):        foo=user bax=b

Optional variables:
    Is-Bastion:    true

Downloading and installing step CLI and step-ssh utilities...
dpkg: warning: 'ldconfig' not found in PATH or not executable
dpkg: warning: 'start-stop-daemon' not found in PATH or not executable
dpkg: error: 2 expected programs not found in PATH or not executable
Note: root's PATH should usually contain /usr/local/sbin, /usr/sbin and /sbin
Removing temporary download directory /tmp/tmp.oXVlW0kjsx6!

```

ILLUSTRATION 5.6 – Capture d’écran de la commande qui télécharge un script. Réalisé par Maya Inès

Ensuite, il faut se rendre sur la page suivante : <https://smallstep.com/app/team/ssh/hosts>

Dans notre cas, team sera remplacé par mycom.

Sur cette page, il faut fournir le nom de l’équipe et le jeton qui se trouvent dans la page qui s’ouvre après avoir cliqué sur le lien ci-dessus. Ces informations permettent de lier la machine hôte au service Smallstep SSH.

En suivant ces étapes, la configuration pour la machine hôte, pour qu’elle puisse utiliser Smallstep SSH et établir des connexions sécurisées avec d’autres hôtes autorisés, est faite.

Une fois que l’hôte SSH est configuré et qu’au moins un client est enregistré, il est possible d’utiliser le compte Smallstep en SSH.

Tout d’abord, il faut utiliser le CLI de Step afin de récupérer la liste des hôtes auxquels l’utilisateur peut accéder en utilisant la commande suivante :

```
1 step ssh hosts
```

```
user@bob:~$ step ssh hosts
HOSTNAME      ID                                TAGS
ssh.ines.lan  ef447970-b4df-4b66-b196-e4d58e094916  bax=bob,foo=user
```

ILLUSTRATION 5.7 – Capture d’écran de la commande step ssh hosts. Réalisé par Maya Inès

Puis se connecter en **SSH** à l’hôte normalement :

```
ssh <hostname>
```

Il est possible maintenant d’utiliser le compte Smallstep pour l’accès **SSH** aux hôtes. L’accès est basé sur des certificats clients de 16 heures. Lorsque le certificat expire, il est demandé de se reconnecter avec smallstep. Si nous souhaitons ajouter d’autres utilisateurs et activer l’authentification unique, il faut se connecter à notre fournisseur d’identité. Ceci permet de découvrir toute l’étendue de Smallstep **SSH**.

## 5.5. AUTHENTIFICATION EN SSH À L’AIDE D’UNE PKI

Cette partie aborde les bases de l’émission et du renouvellement des certificats **SSH** pour les utilisateurs et les hôtes. Elle se base sur le tutoriel "Basic Certificate Authority Operations Working With **SSH** Certificates" <sup>55</sup>

Les certificats **SSH** sont différents des certificats X.509. Ils possèdent leur propre format.

Une **AC SSH** est représentée par une clé publique de confiance au lieu d’un certificat d’**AC** racine. Une **AC SSH** est similaire aux autres clés publiques **SSH**, mais elle est préfixée par une annotation spéciale cert-authority lorsqu’elle est utilisée dans un fichier known\_hosts. <sup>56</sup>

Pour commencer, nous avons besoin d’une **AC** Step-CA avec le support **SSH** activé. Il est possible de la créer en exécutant la commande ci-dessous.

```
step ca init --ssh
```

Tout d’abord, l’hôte authentifie l’utilisateur. Lorsqu’il est configuré pour faire confiance à la clé de l’**AC** utilisateur, un hôte délègue l’identité de l’utilisateur à l’**AC SSH**. Lorsqu’un utilisateur présente son certificat à un hôte au cours de l’échange **SSH**, l’hôte lui fait confiance s’il est signé par la clé de l’**AC** de l’utilisateur, et il autorise tous les noms d’utilisateur répertoriés à se connecter.

Lorsqu’un hôte est configuré pour faire confiance à la clé de l’**AC** utilisateur, il peut authentifier les utilisateurs. Lorsqu’un utilisateur présente son certificat à un hôte lors d’un échange

55. Disponible à l’adresse : <https://smallstep.com/docs/step-ca/basic-certificate-authority-operations/index.html#working-with-ssh-certificates>

56. SMALLSTEP, [s. d.(b)].

SSH, l'hôte lui fait confiance s'il est signé par la clé de l'AC de l'utilisateur. L'hôte autorise alors tous les noms d'utilisateur répertoriés à se connecter.

En utilisant cette approche, l'hôte délègue l'authentification de l'utilisateur à l'AC SSH, en vérifiant la validité de son certificat signé par l'AC.

Ainsi, lorsque l'utilisateur présente son certificat SSH lors d'une connexion, l'hôte peut vérifier sa validité en utilisant la clé publique de l'AC. Si le certificat est valide et que le nom d'utilisateur est autorisé par l'hôte, l'utilisateur est autorisé à se connecter.

Ce mécanisme de confiance basé sur les certificats permet une authentification sécurisée des utilisateurs lors des connexions SSH.

Tout d'abord, il faut lancer la commande `step ca init` comme précédemment mais avec l'option `-sh`. La commande "`step ca init`" exécuté avec l'option `-ssh` permet de créer deux paires de clés AC SSH : une pour l'AC hôte et une pour l'AC utilisateur. La clé de l'AC utilisateur signe les certificats utilisateur SSH et la clé de l'AC hôte signe les certificats hôte SSH.

```
user@pki:~$ step ca init --ssh
⚠ It looks like step is already configured to connect to an authority.
You can use 'contexts' to easily switch between teams and authorities.
Learn more at https://smallstep.com/docs/step-cli/the-step-command#contexts.

✓ Deployment Type: Standalone
What would you like to name your new PKI?
✓ (e.g. Smallstep): pki-ines
What DNS names or IP addresses will clients use to reach your CA?
✓ (e.g. ca.example.com[,10.1.2.3,etc.]): 192.168.1.3
What IP and port will your new CA bind to? (:443 will bind to 0.0.0.0:443)
✓ (e.g. :443 or 127.0.0.1:443): :4444
What would you like to name the CA's first provisioner?
✓ (e.g. you@smallstep.com): ines@pki-ssh.com
Choose a password for your CA keys and first provisioner.
✓ [leave empty and we'll generate one]:

Generating root certificate... done!
Generating intermediate certificate... done!
Generating user and host SSH certificate signing keys... done!
```

ILLUSTRATION 5.8 – Capture d'écran de l'exécution de la commande `step ca init -ssh`. Réalisé par Maya Inès

Ensuite, comme dans le chapitre trois, il faut distribuer la Root CA aux différentes machines. Pour cela, la commande `step ca bootstrap -ca-url [url] -fingerprint [url]` est également utilisée. La différence réside dans l'URL de l'AC et le fingerprint. Ensuite, il faut installer le certificat root distribué mais comme pour le reste, la commande reste la même que celle vue précédemment (chapitre trois). Dans notre cas, ci-dessous se trouvent les commandes qui ont été effectuées.

```
1 step ca bootstrap --ca-url https://192.168.1.3:4444 --fingerprint 067
    b713e5703c7b9ec7361a8e726248067222f92813136f31c627d3ba99121b9
2 sudo step certificate install $(step path)/certs/root_ca.crt
```

Ces commandes ont été effectuées tant sur la machine hôte SSH que sur la machine client, celle de l'utilisateur bob.

```
bob@bob:~$ step ca bootstrap --ca-url https://192.168.1.3:4444 --fingerprint 067b713e5703c7b9ec7
361a8e726248067222f92813136f31c627d3ba99121b9
✓ Would you like to overwrite /home/bob/.step/certs/root_ca.crt [y/n]: y
The root certificate has been saved in /home/bob/.step/certs/root_ca.crt.
✓ Would you like to overwrite /home/bob/.step/config/defaults.json [y/n]: y
The authority configuration has been saved in /home/bob/.step/config/defaults.json.
```

ILLUSTRATION 5.9 – Capture d'écran de l'exécution de la commande `step ca init -ssh`. Réalisé par Maya Inès

```
bob@bob:~$ sudo step certificate install $(step path)/certs/root_ca.crt
[sudo] password for bob:
Certificate /home/bob/.step/certs/root_ca.crt has been installed.
X.509v3 Root CA Certificate (ECDSA P-256) [Serial: 1858...7502]
Subject:      pki-ines Root CA
Issuer:       pki-ines Root CA
Valid from:   2023-08-28T09:43:50Z
to:          2033-08-25T09:43:50Z
```

ILLUSTRATION 5.10 – Capture d'écran de l'installation du certificat. Réalisé par Maya Inès

### a. Authentification des utilisateurs par l'hôte

Pour commencer, il faut déléguer l'authentification des utilisateurs à l'AC SSH. Il faut faire en sorte que l'hôte fasse confiance à l'AC SSH. Sur la machine hôte, une fois la configuration de l'AC terminée, il faut exécuter la commande suivante :

```
1 step ssh config --roots > /home/user/.step/certs/ssh_user_key.pub
```

La première partie de la commande imprime les clés publiques utilisées pour vérifier les certificats utilisateur ou hôte.<sup>57</sup>.

La seconde partie de la commande redirige la sortie de la commande précédente vers un fichier spécifique, "`$(step path)/certs/ssh_user_key.pub`". "`$(step path)`" est une variable qui contient le chemin du répertoire où Step stocke la configuration et l'état.

57. Disponible à l'adresse suivante : <https://smallstep.com/docs/step-cli/reference/ssh/config/#name>

```

user@ssh:~$ step ssh config --roots > .step/certs/ssh_user_key.pub
user@ssh:~$ ls -l .step/certs/
total 24
-rw----- 1 user user 672 Jul 26 15:01 intermediate_ca.crt
-rw----- 1 user user 623 Aug 28 11:50 root_ca.crt
-rw-r--r-- 1 user user 627 Aug 16 15:44 ssh_host_ca_key-cert.pub
-rw----- 1 user user 161 Jul 26 15:01 ssh_host_ca_key.pub
-rw----- 1 user user 161 Jul 29 15:28 ssh_user_ca_key.pub
-rw-r--r-- 1 user user 161 Aug 28 11:55 ssh_user_key.pub

```

ILLUSTRATION 5.11 – Capture d’écran de l’exécution de la commande utilisée pour configurer SSH avec des certificats et pour inspecter les certificats racine utilisés pour signer les certificats. Réalisé par Maya Inès

Le fichier contenant les clés publiques de l’AC SSH doit être spécifié dans la configuration du démon SSH (SSHd<sup>58</sup>). Sur l’hôte, il faut exécuter la commande ci-dessous afin d’ajouter ces clés à la configuration SSHd :

```
1 cat <<EOF >> /etc/ssh/sshd_config
```

La commande ci-dessus permet d’ajouter du contenu à la fin du fichier /etc/ssh/sshd\_config en utilisant la syntaxe du document «EOF».

Il faut compléter la commande avec ce contenu-ci :

```

1 # This is the CA's public key for authenticating user certificates:
2 TrustedUserCAKeys /home/user/.step/certs/ssh_user_key.pub
3 EOF

```

Avec ces lignes, nous indiquons au démon où se trouve la clé publique de l’AC pour l’authentification des certificats des utilisateurs.

Ensuite, il faut redémarrer le SSHd afin que les modifications prennent effet. Sur une machine Debian, la commande ci-dessous doit être exécutée. Une fois cela fait, l’hôte fera confiance à tout certificat utilisateur émis par l’AC SSH. Cela signifie que les utilisateurs authentifiés par l’AC SSH sont autorisés à se connecter à l’hôte.

```
1 systemctl restart ssh.service
```

La seconde étape consiste en créer un certificat SSH pour l’utilisateur. Pour commencer, il faut créer un certificat SSH pour l’utilisateur "user" de la machine ssh.ines.lan, en utilisant la commande ci-dessous. Elle doit être exécutée sur la machine client.

```
1 step ssh certificate user@ssh.ines.lan id_ecdsa
```

---

58. Secure SHell Daemon (SSHd)



Cette commande génère un certificat **SSH** pour l'utilisateur "user" sur le serveur "ssh.ines.lan" en utilisant l'algorithme **ECDSA** <sup>59</sup>.

**ECDSA** est un algorithme de signature numérique. Il est considéré comme un bon algorithme de signatures pour diverses raisons. Tout d'abord, il est sûr. En effet, il est basé sur les mathématiques des courbes elliptiques, ce qui le rend résistant aux attaques de force brute et aux attaques basées sur les propriétés des nombres premiers, comme le problème du logarithme discret. Il est également efficace, car comparé à certains autres algorithmes de signature numérique basés sur **RSA** <sup>60</sup>, il offre des clés plus courtes, ce qui permet des opérations plus rapides en termes de traitement des données. De plus, la taille des signatures est réduite et facilite ainsi le traitement. Pour finir, son utilisation est très répandue dans les systèmes de cryptographie modernes et a fait ses preuves dans des applications telles que la sécurisation des communications et les protocoles d'authentification.

L'**AC** émet une clé privée, une clé publique et un certificat **SSH** pour l'utilisateur "user". Step ajoute automatiquement le certificat et la clé privée à l'agent **SSH** local. Cependant, seuls la clé privée et le certificat sont nécessaire pour l'authentification par certificat **SSH**.

```
bob@bob:~$ step ssh certificate user@ssh.ines.lan id_ecdsa
✓ Provisioner: ines@pki-ssh.com (JWK) [kid: baK2H1ngRu1dQ1qRfW9d4K1ayHS4d2gLaBLa5CWVicI]
Please enter the password to decrypt the provisioner key: █
✓ CA: https://192.168.1.3:4444
Please enter the password to encrypt the private key:
✓ Private Key: id_ecdsa
✓ Public Key: id_ecdsa.pub
✓ Certificate: id_ecdsa-cert.pub
✓ SSH Agent: yes
```

ILLUSTRATION 5.12 – Capture d'écran de l'émission du certificat SSH pour l'utilisateur user. Réalisé par Maya Inès

Il est possible d'inspecter le certificat **SSH** généré avec la commande ci-dessous. Elle permet d'afficher à la sortie des informations sur le certificat telles que son type, sa clé publique, son **AC** signé, l'identifiant de la clé, le numéro de série, la période de validité, les principaux (noms d'utilisateurs), les options critiques et des extensions.

```
cat id_ecdsa-cert.pub | tail -1 | step ssh inspect
```

L'identifiant de la clé est un identifiant unique pour le certificat. Les principaux sont les utilisateurs autorisés à utiliser le certificat. Les options critiques sont des options qui doivent être

59. Elliptic Curve Digital Signature Algorithm (ECDSA)

60. Rivest Shamir Adleman (RSA)

comprises par le serveur **SSH** afin que le certificat soit accepté. Les extensions sont des options supplémentaires qui peuvent être utilisées pour spécifier des contraintes supplémentaires sur l'utilisation du certificat.

```
bob@bob:~$ cat id_ecdsa-cert.pub | tail -1 | step ssh inspect
-:
  Type: ecdsa-sha2-nistp256-cert-v01@openssh.com user certificate
  Public key: ECDSA-CERT SHA256:Ew4USkHqxa7/Rx6LNRl0t5e8Wl1C2lXs3RjYkrD95ps
  Signing CA: ECDSA SHA256:kgiJ9M98EoDSGKijv+MKZfnKzHVyJdhIFNnjdScC6IY (using ecdsa-sha2-n
  istp256)
  Key ID: "user@ssh.ines.lan"
  Serial: 6752740654396751812
  Valid: from 2023-08-28T12:02:06 to 2023-08-29T04:03:06
  Principals:
    user
    user@ssh.ines.lan
  Critical Options: (none)
  Extensions:
    permit-X11-forwarding
    permit-agent-forwarding
    permit-port-forwarding
    permit-pty
    permit-user-rc
  Signature:
    00:00:00:20:7c:44:b6:1d:bc:a9:54:70:9a:70:44:af:
    a5:6e:72:da:72:f6:98:34:a1:e7:d0:28:0f:ed:d8:7b:
    0a:b2:9a:db:00:00:00:20:4d:7b:cb:e2:30:de:80:bf:
    9b:be:83:bc:af:6c:3c:b1:15:ac:15:90:4d:e5:85:67:
    d0:32:f1:b9:17:fc:c9:14
```

ILLUSTRATION 5.13 – Capture d'écran des détails du certificat SSH. Réalisé par Maya Inès

Sur la figure ci-dessus, nous constatons qu'il s'agit d'un certificat utilisateur `ecdsa-sha2-nistp256-cert-v01@openssh.com`.

Un certificat **SSH** utilisateur est un fichier qui contient une clé publique et des informations telles que le nom, la date d'expiration et les autorisations qui sont signées par une **AC** de confiance.

La principale différence entre un certificat **SSH** utilisateur et un certificat d'hôte est qu'un certificat utilisateur authentifie les utilisateurs auprès des serveurs, tandis qu'un certificat d'hôte authentifie les hôtes de serveur auprès des utilisateurs.

Par défaut, lors de l'utilisation de certificats **SSH**, le démon **SSHd** autorise la connexion en tant que l'un des principaux (utilisateurs) répertoriés dans le certificat. Si la liste des principaux est vide, le **SSHd** autorise l'utilisateur à s'authentifier en tant que n'importe quel utilisateur sur l'hôte.

Nous pouvons voir également sur la figure ci-dessus que l'algorithme de signature numérique **ECDSA** est utilisé avec **SHA**<sup>61</sup>256.

Maintenant un certificat a été généré pour l'utilisateur "bob". Il est possible de se connecter à l'hôte en utilisant ce certificat pour l'authentification.

---

61. Secure Hash Algorithm (SHA)

```

bob@bob:~$ ssh user@ssh.ines.lan
The authenticity of host 'ssh.ines.lan (192.168.1.4)' can't be established.
ECDSA key fingerprint is SHA256:utnverbZK9nRMYh7GZ0pSvcPdWhyQiWC/8UAW3LCNko.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ssh.ines.lan,192.168.1.4' (ECDSA) to the list of known hosts.
Linux ssh.ines.lan 5.10.0-25-amd64 #1 SMP Debian 5.10.191-1 (2023-08-16) 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: Thu Aug 24 13:02:48 2023 from 192.168.1.5

```

ILLUSTRATION 5.14 – Capture d’écran de la connexion SSH. Réalisé par Maya Inès

## b. Confiance des clients envers les hôtes

Le but de cette partie est de configurer la confiance des clients **SSH** envers les hôtes. Ces derniers peuvent également avoir des certificats **SSH**, au lieu de la paire de clés publiques typique de l’hôte. Les certificats d’hôte permettent aux clients qui font confiance à l’**AC** de l’hôte d’éviter de demander une confirmation de confiance lors de la première utilisation.

La première étape consiste à délivrer un certificat pour l’hôte. Pour cela, l’**AC** doit signer un certificat d’hôte lié à la clé **SSH ECDSA** de l’hôte. Sur la machine hôte, il faut exécuter les commandes suivantes :

```

1 cd /etc/ssh
2 sudo -E step ssh certificate --host --sign pki.ines.lan ssh_host_ecdsa_key.pub

user@ssh:/etc/ssh$ sudo -E step ssh certificate --host --sign pki.ines.lan ssh_h
ost_ecdsa_key.pub
[sudo] password for user:
Sorry, try again.
[sudo] password for user:
Sorry, try again.
[sudo] password for user:
✓ Provisioner: ines@pki-ssh.com (JWK) [kid: baK2H1ngRu1dQ1qRfW9d4K1ayHS4d2gLABLa
5CWVicI]
Please enter the password to decrypt the provisioner key:
✓ CA: https://192.168.1.3:4444
✓ Would you like to overwrite ssh_host_ecdsa_key-cert.pub [y/n]: y
✓ Certificate: ssh_host_ecdsa_key-cert.pub

```

ILLUSTRATION 5.15 – Capture d’écran de la commande délivrant un certificat pour l’hôte. Réalisé par Maya Inès

La première commande permet de se déplacer dans le répertoire `/etc/ssh`, afin d’accéder au répertoire contenant les fichiers de configuration du service **SSH**.

La seconde commande exécute le programme Step-CA avec des privilèges administratifs (sudo). Le paramètre -E est utilisé afin de préserver les variables d'environnement lors de l'exécution avec sudo. La commande `step ssh certificate` est utilisée afin de générer ou signer des certificats SSH. Les options `-host` et `-sign` indiquent que le certificat généré est pour un hôte et qu'il faut le signer. `pki.ines.lan` représente le nom de l'hôte pour lequel le certificat est généré et `ssh_host_ecdsa_key.pub` est le fichier contenant la clé publique ECDSA de l'hôte. Cette commande générera et signera un certificat SSH pour l'hôte spécifié.

En utilisant la commande ci-dessous, il est possible de voir ce qui a été créé.

```
1 cat ssh_host_ecdsa_key-cert.pub | step ssh inspect
```

```
user@ssh:/etc/ssh$ cat ssh_host_ecdsa_key-cert.pub | step ssh inspect
-:
    Type: ecdsa-sha2-nistp256-cert-v01@openssh.com host certificate
    Public key: ECDSA-CERT SHA256:utnverbZK9nRMYh7GZ0pSvcpdWhyQiWC/8UAW3LCNk
o
    Signing CA: ECDSA SHA256:JrAidAbLLJNWeg8k6BZ9FNyUF260SUYSruYt4qAxp3Q (using ecdsa-sha2-nistp256)
    Key ID: "pki.ines.lan"
    Serial: 7307147936261645388
    Valid: from 2023-08-28T14:22:02 to 2023-09-27T14:23:02
    Principals:
        pki.ines.lan
    Critical Options: (none)
    Extensions: (none)
    Signature:
        00:00:00:21:00:f9:7d:64:43:ed:39:bb:24:8a:d5:07:
        9d:d4:6e:29:03:1d:34:bf:eb:57:91:27:26:f4:01:3d:
        fb:9e:92:07:a2:00:00:00:20:40:36:05:fd:e0:5c:23:
        26:6b:27:5f:42:c6:12:72:56:08:22:18:c2:06:a9:e3:
        36:1a:d2:c6:d2:7d:ea:8e:b7
```

ILLUSTRATION 5.16 – Capture d'écran des détails du certificat SSH. Réalisé par Maya Inès

La deuxième partie est d'installer la clé et le certificat de l'hôte. Pour que l'hôte puisse utiliser cette clé, il faut déplacer les fichiers dans `/etc/ssh` et ajouter ce qui suit à la configuration du SSHd. Sur la machine hôte, il faut taper :

```
1 cat <<EOF | sudo tee -a /etc/ssh/sshd_config
2 # Il s'agit de la cle privée et du certificat de notre hôte :
3 HostKey /etc/ssh/ssh_host_ecdsa_key
4 HostCertificate /etc/ssh/ssh_host_ecdsa_key-cert.pub
5 EOF
```

En exécutant cette commande, le contenu entre «EOF et EOF sera ajouté à la fin du fichier `/etc/ssh/sshd_config` en utilisant `sudo` pour obtenir les permissions nécessaires. Cela permet de

mettre à jour les configurations **SSH** en ajoutant ces lignes spécifiques au fichier.

```
user@ssh:/etc/ssh$ cat <<EOF | sudo tee -a /etc/ssh/sshd_config
> # This is our host private key and certificate:
> HostKey /etc/ssh/ssh_host_ecdsa_key
> HostCertificate /etc/ssh/ssh_host_ecdsa_key-cert.pub
> EOF
# This is our host private key and certificate:
HostKey /etc/ssh/ssh_host_ecdsa_key
HostCertificate /etc/ssh/ssh_host_ecdsa_key-cert.pub
```

ILLUSTRATION 5.17 – Capture d’écran de l’ajout de la configuration du SSHd. Réalisé par Maya Inès

Ensuite, il est possible d’automatiser la rotation des clés de l’hôte. Par défaut, les certificats d’hôte ont une durée de validité d’un mois. Cette période peut être modifiée en fonction des besoins et de la politique de sécurité souhaitée. Le certificat de l’hôte expirant dans un mois, il faut également mettre en place un renouvellement automatique du certificat. Il faut ajouter sur l’hôte un script cron de renouvellement hebdomadaire qui exécute l’étape **SSH** renew.

```
1 cat <<EOF | sudo tee /etc/cron.weekly/rotate-ssh-certificate
2 #!/bin/sh
3 export STEPPATH=/root/.step
4 cd /etc/ssh && step ssh renew ssh_host_ecdsa_key-cert.pub ssh_host_ecdsa_key --force 2> /
   dev/null
5 exit 0
6 EOF
7 sudo chmod 755 /etc/cron.weekly/rotate-ssh-certificate
```

En exécutant ces commandes, un script, dans le fichier `/etc/cron.weekly/rotate-ssh-certificate`, qui renouvelle automatiquement le certificat **SSH** chaque semaine, est créé. Ensuite les permissions appropriées sont attribuées au fichier afin qu’il puisse être exécuté.

La quatrième et dernière partie est la configuration des clients **SSH** afin qu’ils fassent confiance à l’autorité de certification de l’hôte. Pour cela, il faut ajouter la clé de l’**AC** de l’hôte au fichier `.ssh/known_hosts` de **SSH**. Pour afficher la clé de l’hôte, il suffit d’exécuter :

```
1 step ssh config --host --roots

user@ssh:~$ step ssh config --host --roots
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBNhlXmtv
FL5WVV3UZZFINDZL4zGS4UyjuflYErBNDxCdkZgwh8NPBT1QUj f+8dn+4o8u8HLP36kWKDdr2zewT0=
```

ILLUSTRATION 5.18 – Capture d’écran de la commande affichant la clé. Réalisé par Maya Inès

Il faut l'ajouter au fichier `.ssh/known_hosts` du client **SSH**, en faisant précéder `@cert-authority *` pour le marquer en tant qu'autorité de certification, comme dans l'exemple ci-dessous.

```
@cert-authority * ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBNHLXmtvFL5WVV3UZZFINDZL4zGS4UyjuflYErBNDxCdkZg
```

```
bob@bob:~$ cat .ssh/known_hosts
|1|xfoWZXgdddgS9PCBKt+2435eb+U=|MqETx0Z0rd9woeZDt9GLz3hGX5U= ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNo
oYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBF5v/5nNQGn7Lxdc68RtB9wStCbBfwm0/p0YLTv9IYnqSe/KoDo7VUCDcbt
24H2Vt0WVm+8459kkosA8dNNp0jc=
|1|4w9kbvuVZbXvFuo5Q8X/C5syW50=|pClG1am0M6uxSQn9voB3uA/Sx64= ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNo
oYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBF5v/5nNQGn7Lxdc68RtB9wStCbBfwm0/p0YLTv9IYnqSe/KoDo7VUCDcbt
24H2Vt0WVm+8459kkosA8dNNp0jc=
@cert-authority * ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBNHLXm
tvFL5WVV3UZZFINDZL4zGS4UyjuflYErBNDxCdkZgwh8NPBT1QUjf+8dn+4o8u8HLP36kWKDdr2zewT0=
```

ILLUSTRATION 5.19 – Capture d'écran de la commande affichant la clé. Réalisé par Maya Inès

Le client **SSH** fait désormais confiance à tout hôte disposant d'un certificat valide signé par l'**AC**.

Il reste plus qu'à se connecter afin de tester la configuration.

### c. Renouvellement des certificats SSH

Les certificats **SSH** ont une courte durée de vie. Pour les certificats hôte, la durée de vie est d'un mois et pour l'utilisateur moins de 24 heures.

Il est possible de vérifier si un certificat **SSH** doit être renouvelé avec la commande ci-dessous. Elle renvoie '0' si le certificat **SSH** doit être renouvelé en fonction de sa durée de vie restante. Il renvoie "1" si le certificat **SSH** est dans les limites de sa durée de validité et n'a pas besoin d'être renouvelé. Par défaut, un certificat **SSH** a besoin d'être renouvelé lorsqu'il a dépassé 66% (seuil par défaut) de sa durée de vie. Ce seuil peut être ajusté en utilisant l'option `'--expires-in'`.

```
step ssh needs-renewal id_ecdsa-cert.pub
```

```
bob@bob:~$ step ssh needs-renewal id_ecdsa-cert.pub -v
certificate _needs renewal
```

ILLUSTRATION 5.20 – Capture d'écran de la commande faisant la vérification de renouvellement - oui. Réalisé par Maya Inès

L'option `-v` utilisée dans la figure ci-dessus, permet d'afficher l'information, si le certificat doit être renouvelé, de manière lisible pour l'être humain.

```
bob@bob:~$ step ssh needs-renewal id_ecdsa-cert.pub -v
certificate does not need renewal
```

ILLUSTRATION 5.21 – Capture d'écran de la commande faisant la vérification de renouvellement - non. Réalisé par Maya Inès

Il existe d'autres options disponibles pour cette commande. Il est possible de regarder si le certificat va expirer dans un délai donné ou s'il a dépassé un certain pourcentage de sa durée de vie (75% dans l'exemple ci-dessous).

```
1 # Verifier si le certificat va expirer dans un delai donne :
2 step ssh needs-renewal ./ssh_host_ed25519_key.pub --expires-in 1h15m
3 # Verifier si un certificat SSH a depasse 75 % de sa duree de vie :
4 step certificate needs-renewal ./ssh_host_ed25519_key.pub --expires-in 75%
```

Le renouvellement d'un certificat **SSH** se fait à l'aide de la commande ci-dessous. Elle permet le renouvellement d'un certificat **SSH** en utilisant l'**AC SSH**. Elle renouvelle un certificat hôte **SSH** à l'aide des certificats Step. Elle écrit le nouveau certificat sur le disque, soit en écrasant `ssh-cert`, soit en utilisant un nouveau fichier lorsque l'option `--out=file` est utilisée. Elle ne peut pas être utilisée pour renouveler les certificats utilisateur **SSH**.

```
1 step ssh renew -f id_ecdsa-cert.pub id_ecdsa
```

En ce qui concerne les certificats utilisateurs, il n'est pas possible de les renouveler, car ils ont été conçus pour être de courte durée. Par conséquent, il faut régénérer le certificat à son expiration.

## 5.6. RÉVOCATION DES CERTIFICATS ÉMIS

Lorsqu'un certificat est émis par l'**AC**, il ne peut pas être révoqué ultérieurement ; il reste valide jusqu'à sa date d'expiration. Cependant, cette longue période de validité peut poser un problème de sécurité, car si la clé privée associée au certificat est compromise, un attaquant peut l'utiliser pour se faire passer pour le propriétaire légitime pendant toute la durée de validité du certificat. Pour remédier à cela, Step-CA émet des certificats de courte durée.

Lorsqu'un certificat est révoqué dans Step-CA, l'**AC** bloque son renouvellement futur, ce qui est appelé révocation passive. Cette approche convient bien aux **PKI** internes, car elle évite la complexité liée à la vérification en temps réel de l'état de révocation par des tiers centralisés.

Il est important de noter que la révocation passive n'affecte pas la validité du certificat actuellement en circulation, qui reste valide jusqu'à sa date d'expiration. Cela signifie que les certificats révoqués passivement peuvent encore être utilisés pour l'authentification pendant leur période de validité. Step-CA ne prend pas en charge les mécanismes de révocation active tels que les **CRL** ou **OCSP**. Pour révoquer un certificat, il est nécessaire de transmettre son numéro de série unique à l'**AC**.

La révocation d'un certificat peut être effectuée uniquement une seule fois, et toute tentative répétée de révocation du même numéro de série échouera. Pour révoquer un certificat en utilisant son numéro de série, il faut s'authentifier, généralement en utilisant le mot de passe du provisionner de l'**AC**.

La commande qui inspecte le certificat retourne des informations, dont son numéro de série. Ensuite, pour révoquer le certificat, il suffit de saisir la commande `step ca revoke` avec le numéro de série correspondant au certificat.

```
1 step ca revoke serial_number
```

## 5.7. COMMANDES SSH

Il existe plusieurs commandes à connaître afin de manipuler / gérer **SSH**. Dans cette partie, seules les plus utilisées sont mentionnées. Pour obtenir plus de commandes, il suffit d'entrer :

```
1 man ssh
```



SSH(1)

BSD General Commands Manual

SSH(1)

**NAME****ssh** – OpenSSH remote login client**SYNOPSIS**

```
ssh [-46AaCfGgKkMnNqsTtVvXxYy] [-B bind_interface] [-b bind_address]
[-c cipher_spec] [-D [bind_address][:port] [-E log_file]
[-e escape_char] [-F configfile] [-I pkcs11] [-i identity_file]
[-J destination] [-L address] [-l login_name] [-m mac_spec]
[-O ctl_cmd] [-o option] [-p port] [-Q query_option] [-R address]
[-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]] destination
[command]
```

**DESCRIPTION**

**ssh** (SSH client) is a program for logging into a remote machine and for executing commands on a remote machine. It is intended to provide secure encrypted communications between two untrusted hosts over an insecure network. X11 connections, arbitrary TCP ports and UNIX-domain sockets can also be forwarded over the secure channel.

**ssh** connects and logs into the specified destination, which may be specified as either [user@]hostname or a URI of the form ssh://[user@]hostname[:port]. The user must prove his/her identity to the remote machine using one of several methods (see below).

If a command is specified, it is executed on the remote host instead of a login shell.

ILLUSTRATION 5.22 – Capture d’écran de la commande man ssh. Réalisé par Maya Inès

La partie cliente est fournie par le paquet openSSH-client. Il est possible de vérifier ce qui est déjà installé en tapant cette commande :

```
1 ssh -V
```

```
user@ssh:~$ ssh -V
OpenSSH_8.4p1 Debian-5+deb11u1, OpenSSL 1.1.1n 15 Mar 2022
```

ILLUSTRATION 5.23 – Capture d’écran de la commande ssh -V. Réalisé par Maya Inès

Le serveur SSH fonctionne en tant que service lancé automatiquement au démarrage de la machine. Il est possible notamment d’effectuer plusieurs actions dessus, comme par exemple, l’activer ou l’arrêter. Les commandes suivantes sont utilisées pour gérer le service SSH.

```
1 sudo systemctl status ssh
```

```

user@ssh:/etc/ssh$ sudo systemctl status ssh
[sudo] password for user:
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: e>
   Active: active (running) since Fri 2023-09-08 11:43:46 CEST; 30min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
   Process: 476 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
  Main PID: 493 (sshd)
    Tasks: 1 (limit: 2307)
   Memory: 3.5M
      CPU: 17ms
   CGroup: /system.slice/ssh.service
           └─493 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

Sep 08 11:43:46 ssh.ines.lan systemd[1]: Starting OpenBSD Secure Shell server...
Sep 08 11:43:46 ssh.ines.lan sshd[493]: Server listening on 0.0.0.0 port 22.
Sep 08 11:43:46 ssh.ines.lan sshd[493]: Server listening on :: port 22.
Sep 08 11:43:46 ssh.ines.lan systemd[1]: Started OpenBSD Secure Shell server.

```

ILLUSTRATION 5.24 – Capture d’écran de la commande `sudo systemctl status ssh`. Réalisé par Maya Inès

Cette commande affiche l’état actuel du service **SSH**. Elle indique si le service est en cours d’exécution (running) ou s’il a rencontré une erreur. Les informations affichées peuvent inclure le **PID** <sup>62</sup> du processus **SSH**, son temps d’exécution et d’autres détails pertinents.

```
1 sudo systemctl start ssh
```

Cette commande démarre le service **SSH**. Si le service n’est pas déjà en cours d’exécution, elle va lancer le processus **SSH** et permettre aux utilisateurs de se connecter au système à travers **SSH**. Après avoir exécuté cette commande, le service sera actif.

```
1 sudo systemctl stop ssh
```

Cette commande arrête le service **SSH**. Elle stoppe le processus **SSH** en cours d’exécution et ferme toutes les connexions **SSH** actives. Après avoir exécuté cette commande, le service **SSH** ne sera plus disponible pour les utilisateurs souhaitant s’y connecter.

```
1 sudo systemctl restart ssh
```

Cette commande redémarre le service **SSH**. Elle arrête le processus **SSH** en cours d’exécution et le relance immédiatement. Le redémarrage du service **SSH** peut être utile pour appliquer des modifications de configuration ou résoudre des problèmes liés au service **SSH**.

L’utilisation des commandes précitées nécessite des privilèges d’administration, d’où l’utilisation du préfixe `sudo`. Le préfixe `systemctl` fait référence à l’utilitaire de gestion des services

---

62. Process ID (PID)

système sur les distributions Linux utilisant systemd, telles que Debian, CentOS, Fedora, et d'autres.

Sinon en ce qui concerne les connexions, il existe plusieurs commandes qui permettent de se connecter de différentes manières.

Pour commencer, pour se connecter à un hôte distant.

```
1 ssh user@host
```

Il faut remplacer user par votre nom d'utilisateur et host par l'adresse IP ou le nom d'hôte de l'ordinateur distant.

Pour se connecter à un hôte distant avec une clé privée spécifique :

```
1 ssh -i chemin_vers_la_cle_privee user@host
```

Cette commande permet de spécifier le chemin vers la clé privée avec l'option -i. Cela permet de se connecter à l'hôte distant en utilisant une authentification par clé publique.

## CONCLUSION

Le but de ce travail était de mettre en place une infrastructure permettant, d'une part de délivrer des certificats [HTTPS](#) via le protocole [ACME](#), et d'une autre part de délivrer des certificats [SSH](#) pour des connexions à distance. Nous avons également décrit le fonctionnement des connexions [SSH](#) avec certificats ainsi que leur renouvellement et leur révocation.

### BILAN PERSONNEL

Ce travail a été très instructif. Il m'a permis non seulement d'enrichir mes connaissances mais aussi de mettre en pratique les connaissances acquises durant mon cursus.

Tout d'abord, j'ai appris à monter une [PKI](#) et à délivrer des certificats via le protocole [ACME](#) mais aussi des certificats [SSH](#) au travers d'une [PKI](#) afin de faciliter les connexions et éviter de le faire avec des clés. J'ai également pu voir une partie de l'étendu de l'outil Step-CA.

J'ai également appris à utiliser pfSense afin de gérer un serveur [DNS](#) et [DHCP](#) grâce à cet outil.

### DIFFICULTÉS

Pour la réalisation de ce travail, j'ai été confrontée à plusieurs difficultés. La première, a été la prise en main de plusieurs technologies avec lesquelles je n'avais jamais travaillé, j'ai dû me les approprier, ce qui m'a pris beaucoup de temps. Dans un deuxième temps, je devais sélectionner celles qui étaient le plus adaptées à mes besoins.

Au début, j'ai notamment rencontré des difficultés avec le montage du laboratoire. Je n'avais pas de machine pfSense et il m'était impossible de créer un réseau interne sans connexion avec l'Internet. De plus, pour le [DNS](#), d'autres technologies que pfSense ont été envisagées. Cependant, je n'arrivais pas à les configurer comme voulu. PfSense a été d'une grande aide et m'a permis de pallier aux différents problèmes.

### TRAVAIL FUTUR

Dans un travail futur, il pourrait être possible d'aborder la connexion [SSH](#) en utilisant Small-Step [SSH](#) et un fournisseur d'identité, afin de pouvoir effectuer la connexion de plusieurs utilisateurs et de faire des groupes d'utilisateurs selon certains critères et limiter les connexions à seulement certains services.

Nous pourrions également monter plusieurs serveurs **SSH** et plusieurs utilisateurs, afin que tous les utilisateurs puissent se connecter à l'ensemble des serveurs et ainsi limiter les connexions selon certains critères.

Pour finir, il pourrait être utile d'étudier la possibilité de faire un script qui redemande automatiquement un certificat utilisateur **SSH** avant l'expiration de l'autre, rendant ainsi le service **SSH** accessible en tout temps et évitant à l'utilisateur de faire des demandes à chaque fois qu'il souhaite se connecter.

Une autre possibilité est de refaire l'émission des certificats **SSH** sans l'outil de Step-CA et utilisant les commandes offertes par OpenSSH .

## RÉFÉRENCES DOCUMENTAIRES

- ADELIND, 2020. *Qu'est-ce qu'un fichier PEM (et comment en ouvrir un) – Commentouvrir Blog* [en ligne]. 2020-10. [visité le 2023-07-05]. Disp. à l'adr. : <https://commentouvrir.com/blog/quest-ce-quun-fichier-pem-et-comment-en-ouvrir-un/>.
- CERTBOT, [s. d.(a)]. *About Certbot* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://certbot.eff.org/pages/about>.
- CERTBOT, [s. d.(b)]. *Certbot Instructions* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://certbot.eff.org/instructions>.
- CERTBOT, [s. d.(c)]. *User Guide — Certbot 2.6.0 documentation* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://eff-certbot.readthedocs.io/en/stable/using.html>.
- CONTRIBUTOR, Techtarget, [s. d.]. *What is PKI (public key infrastructure)* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://www.techtarget.com/searchsecurity/definition/PKI>.
- ENCRYPT, LET'S, [s. d.]. *Pour commencer - Let's Encrypt - Certificats SSL/TLS gratuits* [en ligne]. [visité le 2023-06-20]. Disp. à l'adr. : <https://letsencrypt.org/fr/getting-started/>.
- G, Domantas, 2020. *What Is NGINX? How Does It Work?* [en ligne]. 2020-11. [visité le 2023-09-08]. Disp. à l'adr. : <https://www.hostinger.com/tutorials/what-is-nginx>.
- GENIORAMA, 2021. *Certificat X.509 : Qu'est-ce que c'est et à quoi sert-il? - Géniorama* [en ligne]. 2021-10. [visité le 2023-07-07]. Disp. à l'adr. : <https://geniorama.com/certificat-x-509-quest-ce-que-cest-et-a-quoi-sert-il/>. Section : Informatique.
- IONOS, 2022. *TLS vs. SSL : quelle est la différence?* [en ligne]. 2022-03. [visité le 2023-07-05]. Disp. à l'adr. : <https://www.ionos.fr/digitalguide/serveur/securite/tls-vs-ssl/>.
- JANKOV, Tonino, 2023. *Nginx vs Apache : La comparaison des serveurs Web* [en ligne]. 2023-08. [visité le 2023-09-08]. Disp. à l'adr. : <https://kinsta.com/fr/blog/nginx-vs-apache/>.

- KARIUKI, Collins, 2023. *Comment fonctionne un certificat X.509 ?* [en ligne]. 2023-01. [visité le 2023-09-08]. Disp. à l'adr. : <https://geekflare.com/fr/x509-certificate/>.
- KEYCDN, [s. d.]. *9 Popular Nginx Commands You Should Know - KeyCDN Support* [en ligne]. [visité le 2023-07-05]. Disp. à l'adr. : <https://www.keycdn.com/support/nginx-commands>.
- KRIMI, Roua, 2022. *Qu'est ce qu'un Serveur Web et Comment ça Marche ?* [Hostinger Tutoriels] [en ligne]. 2022-08-17. [visité le 2023-07-05]. Disp. à l'adr. : <https://www.hostinger.fr/tutoriels/serveur-web>.
- NGINX, [s. d.(a)]. *CommandLine NGINX* [en ligne]. [visité le 2023-07-05]. Disp. à l'adr. : <https://www.nginx.com/resources/wiki/start/topics/tutorials/commandline/>.
- NGINX, [s. d.(b)]. *What Is NGINX ?* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://www.nginx.com/resources/glossary/nginx/>.
- POINT, JAVA T, [s. d.]. *Apache vs. NGINX | Difference between Apache and NGINX - Javatpoint* [en ligne]. [visité le 2023-09-07]. Disp. à l'adr. : <https://www.javatpoint.com/difference-between-apache-and-nginx>.
- POSEY, Brien, 2005. *A beginner's guide to Public Key Infrastructure* [en ligne]. 2005-09. [visité le 2023-09-08]. Disp. à l'adr. : <https://www.techrepublic.com/article/a-beginners-guide-to-public-key-infrastructure/>.
- SCIENCE, ICY, 2023. *Qu'est-ce qu'une liste de révocation de certificats (crl) ? - définition de techopedia - Sécurité 2023* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://fr.theastrologypage.com/certificate-revocation-list>.
- SMALLSTEP, [s. d.(a)]. *ACME Basics* [en ligne]. [visité le 2023-09-07]. Disp. à l'adr. : <https://smallstep.com/docs/step-ca/acme-basics/>.
- SMALLSTEP, [s. d.(b)]. *Basic Certificate Authority Operations* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://smallstep.com/docs/step-ca/basic-certificate-authority-operations/>.
- SMALLSTEP, [s. d.(c)]. *Certificate Revocation* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://smallstep.com/docs/step-ca/revocation/>.

SMALLSTEP, [s. d.(d)]. *Configure popular ACME clients to use a private CA with the ACME protocol* [en ligne]. [visit   le 2023-09-08]. Disp.    l'adr. : <https://smallstep.com/docs/tutorials/acme-protocol-acme-clients/>.

SMALLSTEP, [s. d.(e)]. *Configuring 'step-ca'* [en ligne]. [visit   le 2023-09-08]. Disp.    l'adr. : <https://smallstep.com/docs/step-ca/configuration/>.

SMALLSTEP, [s. d.(f)]. *Configuring 'step-ca' Provisioners* [en ligne]. [visit   le 2023-07-07]. Disp.    l'adr. : <https://smallstep.com/docs/step-ca/provisioners/>.

SMALLSTEP, [s. d.(g)]. *Getting Started* [en ligne]. [visit   le 2023-07-07]. Disp.    l'adr. : <https://smallstep.com/docs/step-ca/getting-started/>.

SMALLSTEP, [s. d.(h)]. *Getting Started with Smallstep SSH* [en ligne]. [visit   le 2023-09-08]. Disp.    l'adr. : <https://smallstep.com/docs/ssh/>.

SMALLSTEP, [s. d.(i)]. *Install step-ca* [en ligne]. [visit   le 2023-09-08]. Disp.    l'adr. : <https://smallstep.com/docs/step-ca/installation/>.

SMALLSTEP, [s. d.(j)]. *Run your own private CA & ACME server using step-ca* [en ligne]. [visit   le 2023-07-05]. Disp.    l'adr. : <https://www.smallstep.com/>.

SMALLSTEP, [s. d.(k)]. *Smallstep Certificate Manager Getting Started* [en ligne]. [visit   le 2023-09-08]. Disp.    l'adr. : <https://smallstep.com/docs/certificate-manager/getting-started/>.

SMALLSTEP, [s. d.(l)]. *Smallstep SSH Host Quickstart* [en ligne]. [visit   le 2023-09-08]. Disp.    l'adr. : [https://smallstep.com/docs/ssh/hosts/?utm\\_campaign=Trial%20Conversion%20Campaign&utm\\_medium=email&\\_hsmi=90316247&utm\\_content=90316247&utm\\_source=hs\\_automation](https://smallstep.com/docs/ssh/hosts/?utm_campaign=Trial%20Conversion%20Campaign&utm_medium=email&_hsmi=90316247&utm_content=90316247&utm_source=hs_automation).

SMALLSTEP, [s. d.(m)]. *Step ca provisioner add* [en ligne]. [visit   le 2023-09-08]. Disp.    l'adr. : <https://smallstep.com/docs/step-cli/reference/ca/provisioner/add/>.

SMALLSTEP, [s. d.(n)]. *Step certificate install* [en ligne]. [visit   le 2023-09-08]. Disp.    l'adr. : <https://smallstep.com/docs/step-cli/reference/certificate/install/>.

SMALLSTEP, [s. d.(o)]. *Step ssh* [en ligne]. [visit   le 2023-07-22]. Disp.    l'adr. : <https://smallstep.com/docs/step-cli/reference/ssh/>.



- SMALLSTEP, [s. d.(p)]. *Step ssh certificate* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://smallstep.com/docs/step-cli/reference/ssh/certificate/>.
- SMALLSTEP, [s. d.(q)]. *Step ssh config* [en ligne]. [visité le 2023-08-24]. Disp. à l'adr. : <https://smallstep.com/docs/step-cli/reference/ssh/config/>.
- SMALLSTEP, [s. d.(r)]. *Step ssh inspect* [en ligne]. [visité le 2023-07-25]. Disp. à l'adr. : <https://smallstep.com/docs/step-cli/reference/ssh/inspect/>.
- SMALLSTEP, [s. d.(s)]. *Step ssh needs-renewal* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://smallstep.com/docs/step-cli/reference/ssh/needs-renewal/>.
- SMALLSTEP, [s. d.(t)]. *Step ssh renew* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://smallstep.com/docs/step-cli/reference/ssh/renew/>.
- SMALLSTEP, [s. d.(u)]. *Step ssh revoke* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://smallstep.com/docs/step-cli/reference/ssh/revoke/>.
- SMALLSTEP, [s. d.(v)]. *Step-ca server* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://smallstep.com/docs/step-ca/>.
- TEAM, SSL.com Support, 2020. *Which ACME Challenge Type Should I Use ? HTTP-01 or DNS-01 ?* [en ligne]. [visité le 2023-07-07]. Disp. à l'adr. : <https://www.ssl.com/faqs/which-acme-challenge-type-should-i-use-http-01-or-dns-01/>.
- WEB, CERTIFY THE, [s. d.]. *HTTP Validation (http-01) | Certify The Web Docs* [en ligne]. [visité le 2023-06-20]. Disp. à l'adr. : <https://docs.certifytheweb.com/docs/http-validation/>.
- WIKIPEDIA, 2020. *Liste de révocation de certificats* [en ligne]. 2020-06. [visité le 2023-07-05]. Disp. à l'adr. : [https://fr.wikipedia.org/w/index.php?title=Liste\\_de\\_r%C3%A9vocation\\_de\\_certificats&oldid=172052467](https://fr.wikipedia.org/w/index.php?title=Liste_de_r%C3%A9vocation_de_certificats&oldid=172052467). Page Version ID : 172052467.
- WIKIPEDIA, 2022a. *ACME (protocole)* [en ligne]. 2022-06. [visité le 2023-09-08]. Disp. à l'adr. : [https://fr.wikipedia.org/w/index.php?title=ACME\\_\(protocole\)&oldid=194233405](https://fr.wikipedia.org/w/index.php?title=ACME_(protocole)&oldid=194233405). Page Version ID : 194233405.
- WIKIPEDIA, 2022b. *pfsense* [en ligne]. 2022-02. [visité le 2023-07-05]. Disp. à l'adr. : <https://fr.wikipedia.org/w/index.php?title=Pfsense&oldid=190885037>. Page Version ID : 190885037.

WIKIPEDIA, 2022c. *X.509* [en ligne]. 2022-01. [visité le 2023-07-07]. Disp. à l'adr. : <https://fr.wikipedia.org/w/index.php?title=X.509&oldid=189716709>. Page Version ID : 189716709.

ZONER, [s. d.]. *Protocole ACME pour émettre et installer un certificat sur Linux* [en ligne]. [visité le 2023-09-08]. Disp. à l'adr. : <https://www.sslmarket.fr/>.