Comment ça marche ?

Pour demander un certificat SSL à une autorité de certification comme Verisign ou GoDaddy, vous leur envoyez une demande de signature de certificat (CSR) et ils vous fournissent en retour un certificat SSL qu'ils ont signé à l'aide de leur certificat racine et de leur clé privée. Tous les navigateurs disposent d'une copie (ou d'un accès à une copie à partir du système d'exploitation) du certificat racine des différentes autorités de certification, ce qui permet au navigateur de vérifier que votre certificat a été signé par une autorité de certification de confiance.

> C'est pourquoi, lorsque vous générez un certificat auto-signé, le navigateur ne lui fait pas confiance.

Il n'a pas été signé par une autorité de certification. Pour contourner ce problème, nous générons notre propre certificat racine et notre propre clé privée. Nous ajoutons ensuite le certificat racine à tous les appareils que nous possédons une seule fois, et tous les certificats auto-signés que nous générons seront alors intrinsèquement fiables.

Devenir une (petite) autorité de certification :

> Il suffit en réalité de deux commandes. Voyons comment nous pouvons procéder sous macOS et Linux, puis voyons comment cela fonctionne sous le système d'exploitation Windows.

1. Génération de la clé privée et du certificat racine sur macOS Monterey et Linux:

> Comme macOS et Linux sont tous deux des systèmes d’exploitation de type Unix, les processus de génération des fichiers requis sont identiques. □□

La seule différence réelle entre les deux est que sur macOS, vous devrez peut-être installer l' application de ligne de commande OpenSSL . Pour ce faire, si vous ne l'avez pas déjà, installez homebrew , qui vous permettra d'installer OpenSSL.

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
brew install openssl
```

Ensuite, nous pouvons créer un emplacement pour stocker nos fichiers de certificats locaux.

```
mkdir ~/certs
cd ~/certs
```

Avec cette configuration, nous sommes prêts à générer la clé privée pour devenir une autorité de certification locale :

```
openssl genrsa -des3 -out myCA.key 2048
```

> OpenSSL vous demandera une phrase de passe, que nous vous recommandons de **ne pas ignorer** et de conserver en lieu sûr. La phrase de passe **empêchera toute personne qui obtiendrait votre clé privée de générer son propre certificat racine**. Le résultat devrait ressembler à ceci :

```
Generating RSA private key, 2048 bit long modulus
.....................................................................+++
.....................................+++
e is 65537 (0x10001)
Enter pass phrase for myCA.key:
Verifying - Enter pass phrase for myCA.key:
```

Ensuite, nous générons un certificat racine :

```
openssl req -x509 -new -nodes -key myCA.key -sha256 -days 1825 -out myCA.pem
```

Vous serez invité à saisir la phrase secrète de la clé privée que vous venez de choisir et à répondre à une série de questions. Les réponses à ces questions ne sont pas si importantes. Elles apparaissent lorsque vous consultez le certificat, ce que vous ne ferez presque jamais. Je vous suggère de définir le nom commun comme quelque chose que vous reconnaîtrez comme votre certificat racine dans une liste d'autres certificats. **C'est la seule chose qui compte.**

Vous devriez maintenant avoir deux fichiers

>

> Félicitations, vous êtes désormais CA. En quelque sorte.

2. Génération de la clé privée et du certificat racine sous Windows :

Sous Windows, il est également possible de configurer votre environnement pour exécuter les `openssl` commandes. Il vous suffit de disposer de quelques outils supplémentaires.

> Si vous utilisez Windows Subsystem for Linux (WSL), c'est comme si vous utilisiez Linux et les commandes fonctionneront exactement de la même manière.

Le moyen le plus simple de le faire est d'installer Git pour Windows , qui est fourni avec OpenSSL et l'utilitaire Git Bash.

Une fois que vous avez ouvert une fenêtre **Git Bash**, vous pouvez **exécuter les mêmes commandes que pour macOS ou Linux**, avec **une petite différence**. En raison du fonctionnement de certaines applications de console (en particulier OpenSSL) dans Git Bash, **vous devez préfixer toutes `openssl` les commandes à l'aide de l' `winpty` utilitaire.**

Ainsi, par exemple, la commande suivante indique comment générer la clé privée pour devenir une autorité de certification locale dans Git Bash :

```
winpty openssl genrsa -des3 -out myCA.key 2048
```

> Les autres petites différences sont les chemins d'accès aux fichiers dans Git Bash. Lorsque vous ouvrez une instance Git Bash, le répertoire de base dans le terminal est mappé à votre répertoire utilisateur dans Windows, mais avec une structure de répertoire de type Linux. Ainsi, si votre répertoire utilisateur est situé dans `c:\Users\nl` Windows, votre répertoire de base Git Bash sera `c/Users/nl` .

Revision #3
Created 24 September 2024 21:47:33 by Admin
Updated 24 September 2024 22:18:26 by Admin