tutoriel

SSL pour Nginx sur Raspberry Pi : mettre en place un certificat SSL auto-signé

Les protocoles Web **TLS** (et son prédécesseur **SSL**) englobent le trafic dans un contenant protégé et chiffré pour :

- échanger en toute sécurité sans que les messages soient interceptés par un tiers.
- permettre aux utilisateurs de vérifier l'identité des sites auxquels ils se connectent.

Nous allons configurer un certificat SSL auto-signé pour un serveur Web Nginx sur un Raspberry Pi.

Un certificat auto-signé ne valide pas l'identité du votre serveur pour les utilisateurs car il n'est pas signé par une autorité de certification de confiance de leur navigateur Web.

Il permet cependant de crypter les communications avec vos clients Web.

Au lieu d'un certificat auto-signé, vous pouvez utiliser **Let's Encrypt**, une autorité de certification qui émet des certificats SSL/TLS gratuits approuvés par la plupart des navigateurs Web.

Consultez le tutoriel **SSL pour Nginx : mettre en place un certificat SSL Let's Encrypt avec Certbot**

Pré-requis

- un serveur Web **Nginx** installé sur le Raspberry Pi :
  - **LEMP - un serveur avec Linux, Nginx, MariaDB, PHP**
  - ou **Nginx sur RaspBerry Pi : le serveur Web hautes performances (LEMP)**

Première étape : créer le dossier pour mettre les certificats SSL

Créez le répertoire **/etc/nginx/ssl** pour les certificats SSL et allez-y :

```
pi@framboise:~ $ sudo mkdir -p /etc/nginx/ssl
```

```
pi@framboise:~ $ cd /etc/nginx/ssl
pi@framboise:/etc/nginx/ssl $
```

Autres étapes

Créer la clé et le certificat

Créez en une seule commande la clé SSL **/etc/nginx/ssl/monsite.fr.key** et le fichier de certificat **/etc/nginx/ssl/monsite.fr.crt** :

```
pi@framboise:/etc/nginx/ssl $ sudo openssl req -x509 -nodes -days 365 -
newkey rsa:2048 -keyout monsite.fr.key -out monsite.fr.crt

Generating a RSA private key
<...>
writing new private key to 'monsite.fr.key'
<...>
-----
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:.
Locality Name (eg, city) []:.
Organization Name (eg, company) [Internet Widgits Pty Ltd]:.
Organizational Unit Name (eg, section) []:.
Common Name (e.g. server FQDN or YOUR name) []:monsite.fr
Email Address []:.
pi@framboise:/etc/nginx/ssl $ ll
<...>
-rw------- 1 root root 1,7K juil.  8 20:29 monsite.fr.key
-rw-r--r-- 1 root root 1,5K juil.  8 20:32 monsite.fr.crt
```

Réponses à fournir :

1. **Country Name (2 letter code) [AU]** : FR
2. **Common Name (e.g. server FQDN or YOUR name) []:** : monsite.fr (nom ou IP de votre site)
3. **Les autres lignes** : auxquelles vous répondez par un point (.) seront laissées vides.

Explication de la commande :

1. **openssl** : commande pour créer et gérer les certificats, clés et autres fichiers.
2. **req -x509** : le type de certificat à créer = certificat auto-signé
3. **-days 365** : durée de validité du certificat, ici un an
4. **-nodes** : sauter la sécurisation du certificat avec une phrase secrète. Nginx doit pouvoir lire le fichier sans intervention de l'utilisateur, au démarrage du serveur. Un mot de passe l'empêcherait car nous devrions le saisir après chaque redémarrage.
5. **-newkey rsa:2048** : générer un nouveau certificat et une nouvelle clé en même temps
6. **rsa:2048** : de créer une clé RSA de 2048 bits
7. **-keyout** : fichier de clé privée
8. **-out** : fichier du certificat

Configurer Nginx pour utiliser SSL

Il nous suffit maintenant de modifier les blocs **server** des fichiers de configuration de Nginx.

Nginx peut activer SSL dans le même bloc server que le trafic HTTP normal. Cela simplifie la configuration du site.

Pour que SSL fonctionne sur un bloc server, tout en autorisant les connexions HTTP régulières, éditez avec les droits d'administration le fichier **/etc/nginx/sites-available/monsite.fr** et ajoutez les lignes suivantes au bloc server (lignes 4,11 et 12 sur l'exemple) :

/etc/nginx/sites-available/monsite.fr

```
1. server {
2.     listen 80 defaultserver;
3.     listen [::]:80 defaultserver ipv6only=on;
4.     listen 443 ssl;
5.
6.     root /var/www/html/monsite;
7.     index index.html index.htm;
8.
9.     server_name monsite.fr;
10.
11.    ssl_certificate /etc/nginx/ssl/monsite.fr.crt;
12.    ssl_certificate_key /etc/nginx/ssl/monsite.fr.key;
13.
14.    location / {
```

```
15.         try_files $uri $uri/ =404;
16.     }
17. }
```

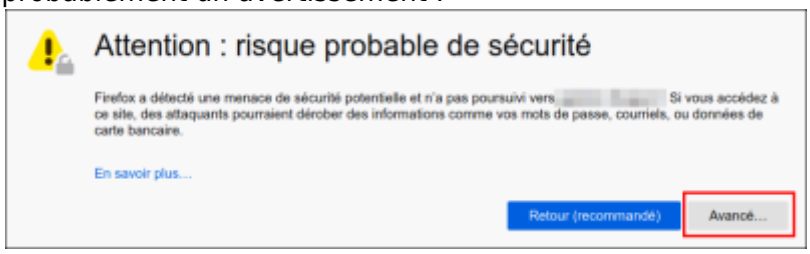
Redémarrez Nginx :

```
pi@framboise:~ $ sudo nginx -s reload
```

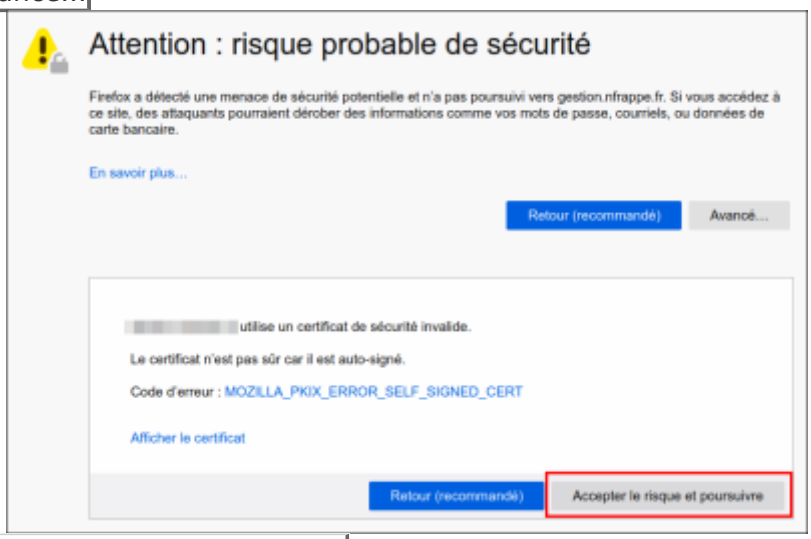
Votre site répond désormais aux demandes HTTP et HTTPS (SSL).

Testez votre configuration

1. **Ouvrez en hhtp** le nom de domaine http://monsite.fr ou l'adresse IP http://IP_du_serveur de votre serveur. Vous devriez voir votre site Web normal.
2. **Ouvrez en hhtps** (donc utilisez SSL) le nom de domaine https://monsite.fr ou l'adresse IP https://IP_du_serveur de votre serveur.
   Vous recevrez probablement un avertissement :



   C'est logique car vous avez créé un certificat auto-signé : le navigateur ne peut pas vérifier l'identité du serveur auquel vous essayez de vous connecter, car il n'est pas signé par une autorité de certification connue du navigateur.
   Cliquez sur Avancé...



   puis sur Accepter le risque et poursuivre.
   Vous devriez revoir votre site.

Conclusion

- Votre navigateur peut afficher le https barré dans la barre d'adresse ou un cadenas barré.
- Un clic sur le cadenas affiche des informations supplémentaires sur la connexion :

- le navigateur ne peut pas vérifier l'identité du serveur car il n'est pas signé par une autorité de certification qu'il connaît
- **la connexion est chiffrée** : nous avons atteint notre objectif.

Problèmes connus

Voir aussi

- **(fr)** https://admin-serv.net/blog/670/creer-et-installer-un-certificat-ssl-sous-nginx/
- **(en)** https://www.digitalocean.com/community/tutorials/how-to-create-an-ssl-certificate-on-nginx-for-ubuntu-14-04

Basé sur « How To Create an SSL Certificate on Nginx for Ubuntu 14.04 » par Justin Ellingwood.