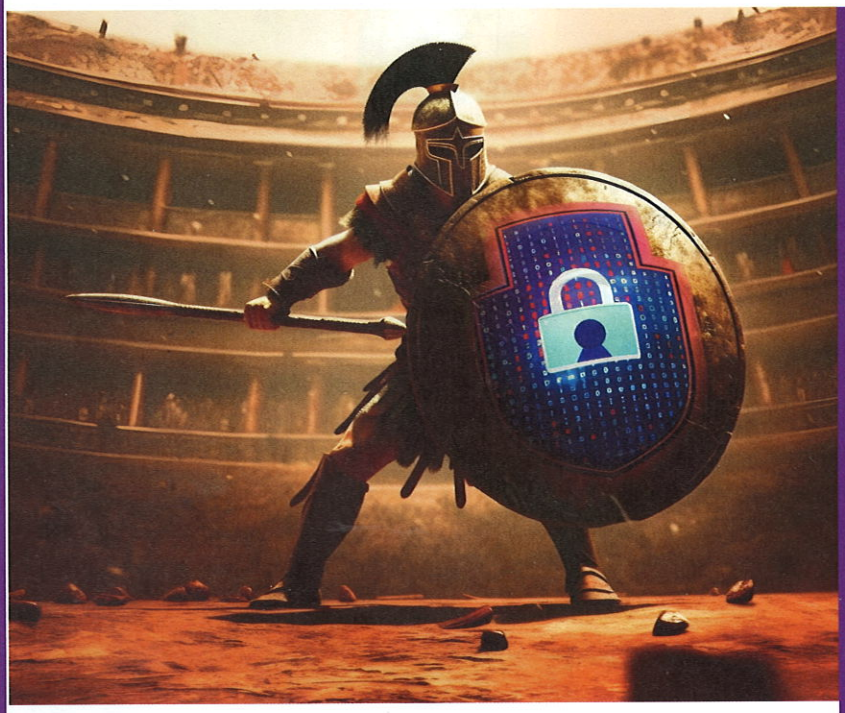
SÉCURISEZ VOTRE SERVEUR WEB AVEC LET'S ENCRYPT

JORDAN **SAMHI**
Chercheur en génie logiciel et sécurité logicielle – j.samhi@me.com



IL EST AUJOURD'HUI RARE DE VOIR UN SITE WEB SANS SON PETIT CADENAS LORS DE LA NAVIGATION WEB. EN EFFET, HTTPS S'EST DÉMOCRATISÉ CES DERNIÈRES ANNÉES, NOTAMMENT GRÂCE À L'INTRODUCTION D'UNE AUTORITÉ DE CERTIFICATION DÉLIVRANT DES CERTIFICATS GRATUITS ! DANS CET ARTICLE, NOUS ALLONS NOUS FOCALISER SUR LET'S ENCRYPT ET FOURNIR UN GUIDE PRATIQUE PAS À PAS POUR SÉCURISER LES ÉCHANGES ENTRE UN SITE WEB ET UN CLIENT SIMPLEMENT ET GRATUITEMENT.

1. INTRODUCTION

Let's Encrypt **[LENC]** est une autorité de certification gratuite et automatisée, qui a pour mission de rendre l'obtention et la gestion de certificats SSL/TLS accessibles à tous. Les certificats SSL/TLS sont essentiels pour sécuriser les communications entre un serveur web et ses utilisateurs : ils garantissent que les données transmises sont chiffrées et protégées contre les interceptions malveillantes (on évite ainsi la fameuse attaque *man-in-the-middle*).

Lancé en décembre 2015, Let's Encrypt est le fruit d'une collaboration entre plusieurs organisations, dont l'Electronic Frontier Foundation (EFF) **[EFF]**, l'Université du Michigan et Mozilla, avec le soutien de nombreux autres partenaires. Let's Encrypt s'est donné pour mission de créer un Web plus sécurisé et plus respectueux de la vie privée en facilitant l'adoption généralisée de HTTPS. Lorsque Let's Encrypt est sorti, il a eu l'effet d'une bombe sur le Web et révolutionné son industrie en démocratisant l'accès aux certificats SSL/TLS.

L'un des principaux atouts de Let's Encrypt est la gratuité de ses certificats SSL/TLS. En effet, contrairement à d'autres autorités

de certification qui facturent des frais souvent élevés pour émettre des certificats, Let's Encrypt permet à tout quidam, à toute organisation, grande ou petite, de sécuriser son site web sans coût financier.

Let's Encrypt facilite l'acquisition et le renouvellement des certificats grâce à des outils comme Certbot (que nous allons utiliser dans ce guide pratique), qui automatisent ces processus. Cette automatisation réduit le risque d'expiration des certificats et simplifie la gestion des certificats pour les administrateurs système, qui peuvent se concentrer sur d'autres tâches.

Quel est l'objectif de cet article ? Il s'agit de fournir un guide pratique et détaillé pour l'installation et la configuration de Let's Encrypt sur un serveur web. Nous allons couvrir chaque étape, de l'installation de l'outil Certbot à la configuration des certificats SSL/TLS pour Apache et Nginx, en passant par le renouvellement automatique et la résolution des problèmes courants. Ce guide pas à pas est conçu pour aider les administrateurs système, les développeurs web et tout autre professionnel à sécuriser leurs sites web avec des certificats Let's Encrypt de manière efficace et fiable, et surtout... gratuite !

2. PRÉREQUIS

Pour installer et configurer Let's Encrypt sur votre serveur, vous devez remplir certains prérequis :

- Accès serveur : Vous devez disposer d'un accès SSH au serveur sur lequel vous souhaitez installer Let's Encrypt. Cet accès SSH doit inclure des privilèges root ou sudo, car vous aurez besoin de permissions élevées pour installer des programmes et modifier la configuration du serveur web.
- Nom de domaine : Pour obtenir un certificat SSL/TLS, vous devez posséder un nom de domaine (par exemple, mon_domaine.com) qui pointe vers l'adresse IP de votre serveur. Il est important que ce nom de domaine soit correctement configuré dans le DNS pour que Let's Encrypt puisse vérifier la propriété du domaine : il doit vous appartenir.

Vous pouvez vérifier la configuration DNS de votre domaine en utilisant des outils en ligne comme MXToolbox **[MXTB]** ou en utilisant des commandes de terminal comme `dig` ou `nslookup`.

```
dig mon_domaine.com
```

Pour utiliser Let's Encrypt, vous devez avoir un serveur web installé et configuré sur votre machine. Let's Encrypt supporte principalement Apache et Nginx, qui sont les serveurs web les plus populaires.

Installation d'Apache :

```
sudo apt update
sudo apt install apache2
```

Installation de Nginx :

```
sudo apt install nginx
```

Assurez-vous que votre serveur web est en cours d'exécution et configuré. Vous pouvez vérifier l'état de votre serveur web avec les commandes suivantes.

Pour Apache :

```
sudo systemctl status apache2
```

Pour Nginx :

```
sudo systemctl status nginx
```

Si votre serveur web est correctement configuré et fonctionnel, vous êtes prêt à passer à l'installation de Certbot pour obtenir et gérer vos certificats SSL/TLS avec Let's Encrypt !

3. ÉTAPE 1 : INSTALLER CERTBOT

Pour utiliser les certificats SSL/TLS de Let's Encrypt, nous allons utiliser Certbot, un client développé par l'EFF pour automatiser le processus de demande et de renouvellement des certificats, vous verrez c'est facile ! En effet, cet outil simplifie grandement le processus en fournissant des commandes claires et des scripts.

3.1 Qu'est-ce que Certbot ?

Certbot est un outil en ligne de commandes qui permet de demander, renouveler et révoquer des certificats SSL/TLS émis par Let's Encrypt. Il prend en charge de nombreux serveurs web, dont Apache et Nginx, et peut configurer automatiquement les serveurs pour utiliser les certificats obtenus.

3.2 Installation de Certbot

L'installation de Certbot varie en fonction de la distribution Linux que vous utilisez. Voici comment l'installer sur les distributions les plus courantes.

Pour installer Certbot sur Debian ou Ubuntu, vous pouvez utiliser les dépôts officiels de la distribution :

```
sudo apt install certbot python3-
certbot-apache # pour apache
sudo apt install certbot python3-
certbot-nginx # pour nginx
```

Une fois Certbot installé, il faut vérifier que tout fonctionne correctement. Exécutez la commande suivante pour vérifier la version de Certbot installée :

```
certbot --version
```

La commande doit afficher la version de Certbot, confirmant ainsi que l'installation s'est bien déroulée.

Exemple de sortie :

```
certbot 1.22.0
```

Si la version de Certbot s'affiche correctement, vous êtes prêt à passer à l'étape suivante : obtenir un certificat SSL/TLS pour votre serveur web.

4. ÉTAPE 2 : OBTENIR UN CERTIFICAT SSL

Maintenant que Certbot est installé sur votre serveur, il est temps de l'utiliser pour obtenir et installer un certificat. Les étapes varient légèrement en fonction du serveur web que vous utilisez, nous donnerons les commandes pour à la fois Apache et Nginx.

4.1 Certificat pour Apache et Nginx

Pour obtenir et installer un certificat SSL/TLS sur un serveur Apache, suivez les étapes ci-dessous. Exécutez la commande suivante pour lancer Certbot. Cette commande demande à Certbot de gérer le processus d'obtention du certificat et de configurer Apache ou Nginx automatiquement.

```
sudo certbot --apache # pour apache
sudo certbot -nginx   # pour nginx
```

Certbot vous guidera à travers une série de questions interactives :

- Adresse e-mail : Fournissez une adresse e-mail valide. Cette adresse sera utilisée pour recevoir des notifications importantes concernant le renouvellement des certificats.
- Acceptation des conditions d'utilisation : Vous devez accepter les conditions d'utilisation de Let's Encrypt.
- Partage de votre adresse e-mail avec l'EFF : Optionnel, vous pouvez choisir de partager votre adresse e-mail avec l'EFF pour recevoir des informations de sécurité.
- Sélection du domaine : Certbot détectera les domaines configurés sur votre serveur Apache. Sélectionnez les domaines pour lesquels vous souhaitez obtenir un certificat.

Certbot configurera automatiquement votre serveur Apache/Nginx pour utiliser le nouveau certificat SSL/TLS. Il mettra à jour les fichiers de configuration pour inclure les directives nécessaires à l'utilisation des certificats.

Après avoir obtenu le certificat, il est important de vérifier que la configuration d'Apache ou de Nginx est correcte et qu'elle fonctionne avec le nouveau certificat. Utilisez les commandes suivantes :

```
sudo apachectl configtest
sudo systemctl reload apache2
```

La commande **`apachectl configtest`** vérifiera la syntaxe de votre configuration Apache, tandis que **`systemctl reload apache2`** rechargera Apache pour appliquer les nouvelles configurations.

Ou bien :

```
sudo nginx -t
sudo systemctl reload nginx
```

La commande **`nginx -t`** vérifiera la syntaxe de votre configuration Nginx, alors que **`systemctl reload nginx`** rechargera Nginx pour appliquer les nouvelles configurations.

Si vous voulez vérifier que les certificats pour le domaine mon_domaine.com ont bien été générés, il faut vérifier dans ce répertoire :

```
ls -larth /etc/letsencrypt/live/mon_
domaine.com/
```

Sortie :

```
total 12K
-rw-r--r-- 1 root root 1.8K Jun 23 14:00 privkey.pem
-rw-r--r-- 1 root root 1.7K Jun 23 14:00 cert.pem
-rw-r--r-- 1 root root 1.7K Jun 23 14:00 chain.pem
-rw-r--r-- 1 root root 3.4K Jun 23 14:00 fullchain.pem
drwxr-xr-x 3 root root 4.0K Jun 23 14:00 ..
drwxr-xr-x 2 root root 4.0K Jun 23 14:00 .
```

- **privkey.pem** : La clé privée associée au certificat.
- **cert.pem** : Le certificat émis pour votre domaine.
- **chain.pem** : Le certificat intermédiaire.
- **fullchain.pem** : Le certificat complet incluant le certificat intermédiaire et le certificat final.

4.2 Cas particuliers

Certbot permet également de gérer des cas particuliers tels que les certificats pour plusieurs domaines, les sous-domaines, et les certificats wildcard.

Pour obtenir un certificat couvrant plusieurs domaines, utilisez l'option **-d** pour spécifier les domaines. Par exemple :

```
sudo certbot --apache -d mon_domaine.com -d www.mon_
domaine.com -d blog.mon_domaine.com
```

Pour obtenir un certificat wildcard, vous devez utiliser la méthode de validation DNS. Par exemple :

```
sudo certbot -d "*.mon_domaine.com" --manual
--preferred-challenges dns certonly
```

Cette commande vous demandera de créer un enregistrement DNS spécifique pour valider la propriété du domaine.

5. ÉTAPE 3 : VÉRIFICATION DE L'INSTALLATION

Après avoir obtenu et installé un certificat SSL/TLS sur votre serveur, il est nécessaire de vérifier que la configuration est correcte et que le certificat fonctionne comme prévu. Dans cette étape, nous allons vérifier la configuration du serveur web, la validation du certificat SSL, et la résolution des problèmes courants.

5.1 Tester la configuration du serveur web

Apache

Exécutez la commande suivante pour vérifier que la syntaxe de votre configuration Apache est correcte :

```
sudo apachectl configtest
```

Si la configuration est correcte, vous devriez voir un message indiquant « Syntax OK ». Si des erreurs sont détectées, examinez les messages d'erreur et corrigez les problèmes dans vos fichiers de configuration.

Après avoir vérifié la syntaxe, rechargez Apache pour appliquer les nouvelles configurations :

```
sudo systemctl reload
apache2
```

Cette commande rechargera Apache sans interrompre les connexions en cours.

Nginx

Exécutez la commande suivante pour vérifier que la syntaxe de votre configuration Nginx est correcte :

```
sudo nginx -t
```

Si la configuration est correcte, vous devriez voir un message indiquant « syntax is ok » et « test is successful ». De la même manière, si des erreurs sont détectées, examinez les messages d'erreur et corrigez les problèmes.

Après avoir vérifié la syntaxe, rechargez Nginx pour appliquer les nouvelles configurations :

```
sudo systemctl reload nginx
```

5.2 Vérification du certificat SSL

Ouvrez votre navigateur web et accédez à votre site en utilisant le protocole HTTPS. Par exemple, si votre domaine est `mon_domaine.com`, accédez à https://mon_domaine.com.

Assurez-vous que le navigateur affiche une icône de cadenas indiquant que la connexion est sécurisée.

Vous pouvez utiliser des outils en ligne comme SSL Labs **[SSLL]** pour vérifier la configuration de votre certificat SSL et obtenir un rapport détaillé. Entrez simplement votre nom de domaine et lancez le test.

Vous pouvez également utiliser la commande `openssl` pour vérifier les détails de votre certificat SSL. Par exemple, exécutez la commande suivante pour obtenir des informations sur le certificat associé au domaine `example.com` :

```
echo | openssl s_client -connect example.com:443 | openssl x509 -noout -text

depth=2 C = US, O = DigiCert Inc, OU = www.digicert.com, CN = DigiCert Global Root G2
verify return:1
depth=1 C = US, O = DigiCert Inc, CN = DigiCert Global G2 TLS RSA SHA256 2020 CA1
verify return:1
depth=0 C = US, ST = California, L = Los Angeles, O = Internet\C2\A0Corporation\C2\A0for\C2\
A0Assigned\C2\A0Names\C2\A0and\C2\A0Numbers, CN = www.example.org
verify return:1
DONE
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            07:5b:ce:f3:06:89:c8:ad:df:13:e5:1a:f4:af:e1:87
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = US, O = DigiCert Inc, CN = DigiCert Global G2 TLS RSA SHA256 2020 CA1
        Validity
            Not Before: Jan 30 00:00:00 2024 GMT
            Not After : Mar  1 23:59:59 2025 GMT
        Subject: C = US, ST = California, L = Los Angeles, O = Internet\C2\A0Corporation\C2\
A0for\C2\A0Assigned\C2\A0Names\C2\A0and\C2\A0Numbers, CN = www.example.org
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                [...]
Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Authority Key Identifier:
                keyid:74:85:80:C0:66:C7:DF:37:DE:CF:BD:29:37:AA:03:1D:BE:ED:CD:17

            X509v3 Subject Key Identifier:
                4C:FE:D0:12:4D:2E:21:CF:6B:FA:F2:F2:B8:4C:49:02:1D:31:91:8A
            X509v3 Subject Alternative Name:
                DNS:www.example.org, DNS:example.net, DNS:example.edu, DNS:example.com,
DNS:example.org, DNS:www.example.com, DNS:www.example.edu, DNS:www.example.net
            X509v3 Certificate Policies:
                Policy: 2.23.140.1.2.2
                  CPS: http://www.digicert.com/CPS
```

```
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication
            X509v3 CRL Distribution Points:

                Full Name:
                  URI:http://crl3.digicert.com/DigiCertGlobalG2TLSRSASHA2562020CA1-1.crl

                Full Name:
                  URI:http://crl4.digicert.com/DigiCertGlobalG2TLSRSASHA2562020CA1-1.crl

            Authority Information Access:
              OCSP - URI:http://ocsp.digicert.com
              CA Issuers - URI:http://cacerts.digicert.com/DigiCertGlobalG2TLSRSASHA2562020CA1-1.crt

            X509v3 Basic Constraints: critical
                CA:FALSE
            CT Precertificate SCTs:
[...]
```

Remplacez **example.com** par votre nom de domaine pour tester. Cette commande affichera les détails du certificat, y compris sa date d'expiration, l'autorité de certification émettrice, et plus encore. Nous avons raccourci la sortie pour des raisons évidentes de place, testez avec votre nom de domaine pour avoir un aperçu.

6. ÉTAPE 4 : CONFIGURATION DU RENOUVELLEMENT AUTOMATIQUE

Les certificats Let's Encrypt ont une durée de validité de 90 jours, après quoi ils doivent être renouvelés pour maintenir une connexion sécurisée. Pour éviter l'expiration des certificats et garantir une disponibilité continue, il suffit de configurer le renouvellement automatique.

6.1 Comprendre le renouvellement des certificats

Le renouvellement des certificats Let's Encrypt doit être effectué régulièrement pour garantir que votre site web reste sécurisé. Certbot, l'outil utilisé pour obtenir les certificats, simplifie également ce processus de renouvellement. En configurant correctement Certbot, vous pouvez automatiser le renouvellement des certificats, réduisant ainsi les risques d'erreur.

6.2 Configurer le renouvellement automatique

Avant de configurer le renouvellement automatique, il est recommandé de tester le processus de renouvellement pour s'assurer qu'il fonctionne correctement. Exécutez la commande suivante pour effectuer un test de renouvellement sans réellement renouveler le certificat (**dry run**) :

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

Cette commande simule le processus de renouvellement et vous permet de vérifier qu'il se déroule sans problème. Si la simulation réussit, vous pouvez configurer le renouvellement automatique. En général, tout devrait bien se passer et vous aurez un message du type « Congratulations, all simulated renewals succeeded ».

La méthode la plus courante pour automatiser le renouvellement des certificats est d'utiliser une tâche **cron**. Cron est un planificateur de tâches sous Linux qui permet d'exécuter des commandes à des intervalles de temps spécifiés.

Ouvrir le fichier **crontab** pour l'édition :

```
sudo crontab -e
```

Ajoutez la ligne suivante à votre fichier **crontab** pour configurer le renouvellement automatique une fois par semaine :

```
0 0 * * 0 /usr/bin/certbot renew --quiet
```

Cette ligne configure **cron** pour exécuter la commande **certbot renew** une fois par semaine. L'option **--quiet** réduit la sortie de la commande pour ne pas encombrer les logs système.

Pour vérifier que la tâche **cron** est correctement configurée, vous pouvez consulter les logs de Certbot pour vous assurer que le renouvellement se déroule comme prévu.

Consulter les logs de Certbot :

```
sudo tail -f /var/log/letsencrypt/
letsencrypt.log
```

Ces logs contiennent des informations détaillées sur les tentatives de renouvellement des certificats, y compris les réussites et les échecs.

Il est également possible d'utiliser systemd et ne pas passer par **cron**, mais pour des raisons de place, nous ne l'aborderons pas ici.

6.3 Vérification du renouvellement

Vous pouvez périodiquement exécuter la commande suivante pour simuler le processus de renouvellement et vérifier qu'il se déroule correctement :

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

En configurant le renouvellement automatique des certificats Let's Encrypt, vous vous assurez que vos certificats SSL/TLS restent valides et que votre site web reste sécurisé sans intervention manuelle régulière. La prochaine section couvrira la sécurisation et l'optimisation de votre configuration pour renforcer la sécurité de votre site web.

7. ÉTAPE 5 : SÉCURISATION ET OPTIMISATION

Maintenant que vos certificats sont installés et que leur renouvellement automatique est configuré, il est temps de renforcer et d'optimiser votre configuration SSL/TLS. On verra notamment l'application des meilleures pratiques en matière de sécurité, la configuration de *HTTP Strict Transport Security* (HSTS), et la mise en place de l'OCSP Stapling.

7.1 Renforcement de la sécurité SSL/TLS

Pour garantir une sécurité optimale, il faut configurer correctement votre serveur web. Voici quelques meilleures pratiques à suivre pour Apache et Nginx.

Apache

Désactiver les protocoles obsolètes : les versions obsolètes de SSL et TLS, telles que SSLv3 et TLS 1.0, présentent des vulnérabilités connues et doivent être désactivées.

```
sudo nano /etc/apache2/sites-available/
mon_domaine.conf
```

Ajoutez ou modifiez les lignes suivantes :

```
SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1
```

Utilisez des suites de chiffrement modernes et sécurisées. Ajoutez ou modifiez la ligne suivante :

```
SSLCipherSuite HIGH:!aNULL:!MD5:!3DES
SSLHonorCipherOrder on
```

Nginx

Modifiez le fichier de configuration de votre site :

```
sudo nano /etc/nginx/sites-available/
mon_domaine
```

Ajoutez ou modifiez les lignes suivantes :

```
ssl_protocols TLSv1.2 TLSv1.3;
```

Ajoutez ou modifiez la ligne suivante :

```
ssl_ciphers 'HIGH:!aNULL:!MD5:!3DES';
ssl_prefer_server_ciphers on;
```

Évaluer la configuration SSL : comme précédemment, vous pouvez utiliser SSL Labs **[SSLL]** pour évaluer la configuration SSL de votre serveur. Suivez les recommandations pour améliorer votre score, si vous n'avez pas encore un A+.

7.2 HTTP Strict Transport Security (HSTS)

HSTS est un mécanisme de sécurité qui force les navigateurs à utiliser uniquement des connexions HTTPS pour accéder à votre site.

Apache

Ajouter l'en-tête HSTS à votre fichier de configuration, ajoutez la ligne suivante à l'intérieur du bloc `<VirtualHost>` :

```
Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"
```

Ensuite, redémarrez Apache :

```
sudo systemctl reload apache2
```

Nginx

Ajouter l'en-tête HSTS à votre fichier de configuration, ajoutez la ligne suivante à l'intérieur du bloc `server` :

```
add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;
```

Ensuite, redémarrez Nginx :

```
sudo systemctl reload nginx
```

7.3 OCSP Stapling

OCSP Stapling améliore la performance des requêtes SSL/TLS en permettant au serveur web de fournir la réponse OCSP (*Online Certificate Status Protocol*) directement au client, au lieu de le laisser interroger l'autorité de certification.

Apache

Ajoutez les lignes suivantes à l'intérieur du bloc `<VirtualHost>` de votre fichier de configuration :

```
SSLUseStapling on
SSLStaplingResponderTimeout 5
SSLStaplingReturnResponderErrors off
SSLStaplingCache "shmcb:logs/ssl_stapling(32768)"
```

Redémarrez Apache :

```
sudo systemctl reload apache2
```

Nginx

Ajoutez les lignes suivantes à l'intérieur du bloc `server` de votre fichier de configuration :

```
ssl_stapling on;
ssl_stapling_verify on;
resolver 8.8.8.8 8.8.4.4
valid=300s;
resolver_timeout 5s;
```

Redémarrez Nginx :

```
sudo systemctl reload nginx
```

8. ÉTAPE 6 : MAINTENANCE ET SURVEILLANCE

Une fois que vos certificats SSL/TLS sont correctement installés et configurés, il est essentiel de mettre en place des pratiques de maintenance et de surveillance pour garantir leur bon fonctionnement continu dans le temps.

8.1 Surveillance de l'état du certificat

Surveiller l'état de vos certificats SSL/TLS permet de détecter rapidement tout problème potentiel avant qu'il n'affecte la disponibilité ou la sécurité de votre site web (en sécurité, on parle de *proactivité*). Voici quelques méthodes pour surveiller vos certificats.

Utilisez des outils en ligne pour surveiller l'état de vos certificats SSL/TLS. Ces outils peuvent envoyer des notifications lorsqu'un certificat approche de son expiration ou si des problèmes sont détectés (par exemple SSLLABS).

Certbot Monitoring : Let's Encrypt propose des services de surveillance de certificat. Vous pouvez recevoir des notifications par e-mail lorsque votre certificat approche de son expiration.

Scripts de vérification : créez des scripts personnalisés pour surveiller l'état de vos certificats et intégrer ces scripts dans votre système de surveillance existant.

Voici un exemple de script Bash qui vérifie la date d'expiration de votre certificat et envoie une alerte si le certificat expire dans moins de 30 jours :

```
#!/bin/bash

DOMAINE="mon_domaine.com"
FICHIER_CERTIFICAT="/etc/letsencrypt/live/$DOMAINE/
fullchain.pem"
DATE_EXPIRATION=$(openssl x509 -enddate -noout -in
$FICHIER_CERTIFICAT | cut -d= -f2)
TIMESTAMP_EXPIRATION=$(date -d "$DATE_EXPIRATION" +%s)
TIMESTAMP_ACTUEL=$(date +%s)
JOURS_RESTANTS=$(( (TIMESTAMP_EXPIRATION -
TIMESTAMP_ACTUEL) / 86400 ))

if [ $JOURS_RESTANTS -le 30 ]; then
    echo "Le certificat pour $DOMAINE expire dans
    $JOURS_RESTANTS jours."
    # Envoyer un email ou une alerte via votre système
    # de surveillance
else
    echo "Le certificat pour $DOMAINE est valide pour
encore $JOURS_RESTANTS jours."
fi
```

Sauvegardez ce script dans un fichier, par exemple `check_ssl.sh`, rendez-le exécutable et planifiez son exécution régulière avec cron :

```
chmod +x check_ssl.sh
```

Ajoutez une tâche cron pour exécuter ce script quotidiennement :

```
sudo crontab -e
```

Ajoutez la ligne suivante pour exécuter le script chaque jour à minuit :

```
0 0 * * * /chemin/vers/check_ssl.sh
```

Parfois, le renouvellement automatique peut échouer en raison de problèmes de configuration, de connectivité ou d'autres facteurs. Dans ces cas, il est important de savoir comment renouveler manuellement vos certificats : rien de plus facile !

Pour renouveler manuellement vos certificats Let's Encrypt, utilisez la commande suivante :

```
sudo certbot renew
```

Cette commande tente de renouveler tous les certificats proches de leur expiration. Si le renouvellement réussit, Certbot mettra à jour les certificats et reconfigurera automatiquement votre serveur web.

Si le renouvellement manuel échoue, examinez les messages d'erreur pour déterminer la cause du problème. Les logs de Certbot peuvent fournir des informations détaillées sur les erreurs rencontrées :

```
sudo tail -f /var/log/letsencrypt/
letsencrypt.log
```

Identifiez les erreurs spécifiques et apportez les corrections nécessaires à votre configuration DNS, aux permissions de fichiers, ou à d'autres paramètres.

9. ÉTAPE 7 : RÉSOLUTION DES PROBLÈMES AVANCÉS

Malgré une installation et une configuration soigneuses, des problèmes peuvent survenir lors de la gestion des certificats SSL/TLS. Cette section traite des problèmes avancés que vous pourriez rencontrer et propose des solutions pour les résoudre.

- **Problèmes de renouvellement :** lorsqu'un renouvellement échoue, la première étape consiste à analyser les logs de Certbot pour identifier la cause du problème.

  Consulter les logs de Certbot :

```
sudo tail -f /var/log/letsencrypt/
letsencrypt.log
```

  Cherchez les erreurs spécifiques dans les logs.

- **Erreurs courantes et solutions :** si Certbot ne peut pas valider votre domaine via DNS, assurez-vous que les enregistrements DNS sont correctement configurés et propagés.

  Utilisez des outils comme `dig` pour vérifier la configuration DNS :

```
dig mon_domaine.com
```

- **Erreur de validation HTTP :** cette erreur se produit lorsque Certbot ne peut pas accéder au fichier de validation placé dans le répertoire `.well-known/acme-challenge`.

  Assurez-vous que votre serveur web est configuré pour servir ce répertoire et qu'il n'y a pas de redirections ou de règles de sécurité bloquant l'accès.

  Pour Apache :

```
<Directory "/var/www/html/.well-known/acme-
challenge">
    Options None
    AllowOverride None
    Require all granted
</Directory>
```

  Pour Nginx :

```
location ~ /.well-known/acme-challenge {
    allow all;
}
```

- **Problèmes de permissions :** si Certbot ne peut pas accéder aux fichiers de certificat ou de clé, vérifiez les permissions et assurez-vous que les fichiers sont accessibles par l'utilisateur sous lequel fonctionne votre serveur web.

  Modifiez les permissions des fichiers :

```
sudo chown www-data:www-data /etc/
letsencrypt/live/mon_domaine/fullchain.pem
sudo chown www-data:www-data /etc/
letsencrypt/live/mon_domaine/privkey.pem
```

- **Problèmes de configuration SSL/TLS :** les erreurs de configuration SSL/TLS peuvent empêcher votre serveur web de démarrer ou d'accepter des connexions sécurisées.

Consulter les logs du serveur web :

```
sudo tail -f /var/log/apache2/error.log     # pour apache
sudo tail -f /var/log/nginx/error.log       # pour nginx
```

- **Erreur de démarrage du serveur web :** si votre serveur web ne démarre pas après une modification de configuration, il est probable qu'il y ait une erreur de syntaxe dans vos fichiers de configuration.

```
sudo apachectl configtest
sudo nginx -t
```

- **Problèmes spécifiques aux environnements complexes :** dans des environnements avec plusieurs serveurs (par exemple, des clusters ou des configurations de haute disponibilité), la gestion des certificats peut être plus complexe.

  Utilisez des outils comme rsync ou des systèmes de fichiers distribués pour synchroniser les certificats entre les serveurs.

  Exemple avec rsync :

```
rsync -avz /etc/letsencrypt/live/ mon_serveur:/etc/
letsencrypt/live/
rsync -avz /etc/letsencrypt/archive/ mon_serveur:/etc/
letsencrypt/archive/
rsync -avz /etc/letsencrypt/renewal/ mon_serveur:/etc/
letsencrypt/renewal/
```

- **Automatisation :** utilisez Ansible pour automatiser la gestion des certificats dans des environnements complexes.

  Exemple de playbook Ansible :

```
- name: Synchroniser les certificats Let's Encrypt
  hosts: serveurs_web
  tasks:
    - name: Synchroniser les certificats actifs
      synchronize:
        src: /etc/letsencrypt/live/
        dest: /etc/letsencrypt/live/
        recursive: yes
    - name: Synchroniser les certificats archivés
      synchronize:
        src: /etc/letsencrypt/archive/
        dest: /etc/letsencrypt/archive/
        recursive: yes
    - name: Synchroniser les fichiers de
renouvellement
      synchronize:
        src: /etc/letsencrypt/renewal/
        dest: /etc/letsencrypt/renewal/
        recursive: yes
```

En suivant ces étapes, vous pouvez résoudre les problèmes les plus courants de gestion des certificats et garantir la sécurité et la disponibilité continues de votre site web. La gestion proactive des certificats et la résolution rapide des problèmes sont essentielles pour maintenir un environnement sécurisé et fiable.

CONCLUSION

La mise en place de certificats SSL/TLS avec Let's Encrypt constitue une avancée essentielle pour sécuriser votre site web et protéger les données de vos utilisateurs, et ceci totalement gratuitement grâce aux initiatives de certains acteurs. En automatisant l'obtention et le renouvellement des certificats, Let's Encrypt simplifie un processus autrefois complexe et coûteux, ce qui rend la sécurité web accessible à tous.

Il faut toutefois rester prudent et maintenir une veille active pour rester informé des évolutions dans le domaine de la sécurité SSL/TLS, telles que les nouvelles versions de protocoles et les recommandations de l'industrie. L'automatisation et la gestion proactive de la sécurité deviendront de plus en plus importantes, surtout à mesure que les cybermenaces évoluent et se diversifient. ■

RÉFÉRENCES

[LENC] Let's Ecrypt : https://letsencrypt.org

[EFF] EFF : https://www.eff.org

[MXTB] MXToolBox : https://mxtoolbox.com/DNSLookup.aspx

[SSLL] SSLLABS : https://www.ssllabs.com/ssltest/