



Tuto, LetsEncrypt: Installer un certificat de sécurité SSL sous Windows ou Linux

Commençons en préparant les dossiers de validation (/.well-known/acme-challenge)

Préparation pour Windows

Préparation pour Linux

Créer les clés et le certificat CSR

Obtenir le certificat SSL

Installer le certificat

Observer le contenu du certificat SSL

Exemple de script Bash pour renouvellement automatique du certificat Let's Encrypt sous Linux.

Exemple de script Batch pour renouvellement automatique du certificat Let's Encrypt sous

Windows (avec WAMP ou EasyPHP, par exemple)

Révoquer un certificat

Obtenir un certificat de sécurité wildcard (générique) ACME V2

(Haut de Page)

CE TUTO EST CONCU POUR LES SERVEURS APACHE!

Vous pouvez le suivre pour obtenir votre certificat, mais les paramétrages sont destinés à APACHE !

ATTENTION! Ce tuto est de niveau "geek à l'aise avec le SSL et la configuration d'un serveur Apache!" ☆ ☆ ☆

Plus vous en saurez, mieux ce sera. Si vous êtes un noob, il est possible d'y arriver, mais ce ne sera pas facile pour vous !

De toute façon, comme d'hab, pour les questions, insultes, menaces, vous pouvez répondre. C'est pas la peine d'être inscrit.

Salut tous,

Dans ce nouveau tuto, je vais essayer de vous aider à créer un certificat de requête (CSR) afin d'obtenir un vrai certificat de sécurité auprès de <u>Lets Encrypt</u>.

Ensuite, nous installerons ce certificat sur notre serveur pour pouvoir bénéficier d'une connexion sécurisée.

Lets Encrypt est aujourd'hui une autorité de certification autorisée à délivrer des certificats de sécurité gratuits reconnus par tous les navigateurs web.

Le seul vrai problème de **Lets Encrypt** et qu'ils délivrent des certificats valables 90 jours seulement. A la longue, quand vous serez familiarisé avec la façon de faire, ce ne sera plus gênant, d'autant plus qu'il est possible d'automatiser le renouvellement.

Il existe des outils gratuits entièrement automatisés (comme "certbot", par exemple) pour obtenir et renouveler les certificats de sécurité.

Ces outils ont un gros inconvénient. Ils fonctionnent sur des config "classiques".

Ici, vous le verrez, nous avons une configuration "exotique", avec des sous domaines, qui ne fonctionnerait pas avec **certbot** ou n'importe quel autre outil automatique.

Ce tuto est parfaitement valable pour Linux!

Il vous suffit d'avoir **Perl** sur votre serveur et d'avoir installé le module **Crypt::LE** qui vous donnera accès au script **le.pl**

C'est la version exécutable de ce script que nous utiliserons pour obtenir notre certificat de sécurité sous Windows.

Sous Linux, la démarche restera la même et la syntaxe sera rigoureusement identique. Il suffira juste de convertir **le.exe** en **le.pl**, c'est tout.

(Haut de Page)

ATTENTION! RISQUE DE SÉCURITÉ MAXIMAL!

Dans ce tuto, nous allons manipuler ce qu'on appelle des clés RSA.

Il existe deux types de clés RSA qui sont indissociables. La clé privée et la clé publique. Ces 2 clés sont liées, elles sont inséparables.

Sur votre certificat de sécurité, la clé publique apparaitra et c'est entièrement normal. La clé publique, comme son nom l'indique, est destinée à être rendue publique.

La clé privée est secrète et elle doit le rester! Elle ne doit être divulguée sous aucun prétexte! Seul votre site a besoin de la clé privée. Tous les autres sites de la planète ne doivent pas connaître cette clé! Même si c'est demandé gentiment, ne donnez jamais votre clé privée!

Une autorité de certification, l'autorité qui vous délivrera votre certificat de sécurité ne doit, elle non plus, pas connaître votre clé privée! Dans notre exemple, LetsEncrypt pourra vérifier que nous possédons la clé privée en analysant la signature de notre demande de certificat à l'aide de la clé publique et uniquement de la clé publique.

C'est cette clé privée qui fera qu'on sait que c'est vous et votre certificat sur un site sécurisé. C'est cette clé qui servira à établir une connexion sécurisée sur votre site.

Si quelqu'un connaît cette clé privée, il pourra créer un "vrai faux certificat" et/ou intercepter la clé générée pour établir la connexion sécurisée et ainsi lancer une <u>attaque de l'homme du milieu</u>. J'insiste et j'insiste lourdement! Votre clé privée doit rester secrète!

D'une façon générale, quand vous travaillez avec des clés RSA, la clé privée est secrète, elle est à vous seul.

La clé publique peut être divulguée sans aucun souci et elle doit quasiment toujours être divulguée.

(Haut de Page)

Commençons

La façon de faire est la même sous Linux que sous Windows. Quand une manipulation spéciale pour un système sera demandée, cela sera précisé.

Sur votre serveur, Rendez vous à la racine de celui ci (le point d'entrée où se trouve **index.php** ou **index.html**). La racine de votre site est le dossier où atterrissent chaque visiteur ou membre. Créez un dossier que vous appellerez .well-known/.

Ensuite, dans ce dossier .well-known/, créez un autre dossier qui s'appellera acme-challenge/. Voilà, votre site contient les dossiers .well-known/acme-challenge/, il est prêt pour répondre aux requêtes de vérification de Lets Encrypt.

(Haut de Page)

Préparation pour Windows

Maintenant, nous allons télécharger l'outil LE de la bibliothèque Crypt::LE qui sera indispensable pour générer vos clés et votre certificat de requête en vue d'obtenir un vrai certificat signé par Lets Encrypt.

Pour Windows, rendez vous ici et téléchargez la version qui correspond à votre machine :

- Windows 32 Bits
- Windows 64 bits

Ces archives contiennent un fichier qui ne nécessite aucune installation, vous pouvez l'utiliser directement. Aucune dépendance n'est nécessaire, y compris OpenSSL. Ce fichier embarque tout ce dont il a besoin.

Ce tuto est maintenu à jour, mais si vous voulez être certain de télécharger une version toujours à jour, vous pouvez aller directement sur la <u>page de téléchargements Github du projet</u>. Vous n'avez plus qu'à choisir le téléchargement qui correspond à votre machine (le32.zip & le64.zip pour win 32 ou 64 bit).

Si vous n'êtes pas intéressé par la façon de faire sous Linux, vous pouvez passer directement à la <u>création</u> <u>du certificat de requête (CSR)</u>.

(Haut de Page)

Préparation pour Linux

Sous Linux avec Perl, utilisez CPAN. Installez le module Crypt::LE Code

cpan -i Crypt::LE

Si il vous manque des modules dont **Crypt::LE** a besoin, cela sera indiqué et l'installation stoppée. Installez les dépendances indiquées et reprenez l'installation de **Crypt::LE** Les dépendances PERL s'installent automatiquement avec **Crypt::LE**.

En cas de gros problème pour installer les dépendances, n'omettez pas le paquet **openssl-devel** . Vous êtes bien conscient que **OpenSSL** est absolument nécessaire, si ce n'est pas le cas, stoppez tout ici. Ce n'est pas la peine d'aller plus loin, ce sujet semble vous dépasser. Faites attention si vous continuez sans savoir ce que vous faites !

Pour CentOS/Fedora/Redhat

Code

yum install openssl-devel

Ubuntu/Debian:

Code

sudo apt-get install libssl-dev

Si vous avez Perl mais n'avez pas CPAN, installez le :

Pour CentOS/Fedora/Redhat

Code

yum install perl-CPAN -y

Ubuntu/Debian:

Code: bash

apt-get install build-essential

(Haut de Page)

Entrons dans le vif du sujet en créant les clés et le certificat de requête (CSR)

Vous pouvez vous aider avec le site <u>ZeroSSL.com</u> qui pourra même vous aider à obtenir des certificats "auto-signés" pour travailler en local et bien plus encore lorsque vous aurez votre CSR (ne donnez jamais votre clé privée! Un CSR, oui. Une clé, non!).

Pour Windows, dans chaque fichier .zip , vous trouverez un fichier le32.exe ou le64.exe selon la version choisie.

Renommez votre fichier en le.exe

Ca me permettra de donner les mêmes instructions pour Windows 32bits et 64bits. Rassurez vous, ça ne changera rien.

Sous Linux, vous utiliserez le.pl qui présente exactement la même syntaxe que pour Windows.

Imaginons que nous souhaitons un certificat pour le domaine example.com

Ce domaine possède les sous domaines suivants et nous voulons aussi un certificat pour ces sous domaines :

www.example.com static.example.com mail.example.com

Créez un dossier qui ne soit pas accessible au public et déplacez-y **le.exe** (**le.pl** sera accessible de partout).

Si vous souhaitez que ce fichier **le.exe** soit accessible de partout, peu importe le dossier dans lequel vous travaillez, je vous recommande de le placer dans le dossier **C:\windows**. De cette façon, vous n'aurez pas à copier/déplacer le fichier selon le dossier où vous vous trouvez pour travailler.

Vous pouvez également en placer une copie sur clé USB pour dépanner les copains qui auraient des soucis de SSL avec leur WAMP.

Sous Windows, avec WAMP, le dossier créé sera par exemple : C:\Wamp\ssl dans le cas où la racine du site est C:\Wamp\www

Sous Linux, ce dossier pourra être : /var/ssl dans le cas où la racine du site est /var/www

C'est vous qui choisissez comme vous en avez envie. Cependant, dites vous bien que le dossier devra être accessible pour le serveur.

Dans ce dossier seront stockés la clé privée et le certificat de sécurité. Le public ne doit pas pouvoir accéder à ce dossier.

Soyez vigilant! Pas de config du style example.com/ssl, ce serait la plus grosse boulette que vous ayez faite.

Ouvrez une invite de commande (la console sous Linux) et placez vous dans le dossier créé.

Sous Windows, appuyez sur la touche " *Windows* " et entrez " CMD " suivit de entrée. Sous Linux, ouvrez la console ou le shell, c'est pareil.

Rendez vous dans le dossier créé, pour Windows :

Code

cd \wamp\ssl

Ou, sous Linux : Code

cd /var/ssl

Sous Windows, nous allons utiliser le.exe et entrer ce qui suit :

Code: perl

```
le.exe --key account-key.key --email "hostmaster@example.com" --csr
example.com.csr --csr-key example.com.key --domains
"example.com, www.example.com, static.example.com, mail.example.com"
--generate-missing --generate-only
```

Sous Linux, vous entrerez:

Code: perl

```
le.pl --key account-key.key --email "hostmaster@example.com" --csr
example.com.csr --csr-key example.com.key -domains
"example.com, www.example.com, static.example.com, mail.example.com"
--generate-missing --generate-only
```

Bien entendu, vous modifierez le nom de domaine et le mail pour les remplacer par les votres. (Il n'existe aucun risque de SPAM avec le mail)

Suite à ceci, vous allez obtenir plusieurs fichiers :

account-key.key: Il s'agit de votre clé privée vous permettant de vous identifier chez Lets Encrypt. Il s'agit de la clé de votre compte, elle n'a rien à voir avec votre certificat mais elle doit aussi rester secrète. Cette clé peut être utilisée pour révoquer votre certificat.

Le mail fourni ne sera utilisé que pour vous prévenir en cas de problème avec votre certificat. Il est optionnel mais recommandé.

example.com.csr : C'est ce fichier qui contient votre demande de certificat. Il a été signé avec votre clé privée et il contient la clé publique.

example.com.key : Il s'agit de votre clé privée qui doit rester secrète. C'est cette clé qui servira sur votre serveur pour établir une connexion sécurisée. C'est également cette clé qui fera de vous le propriétaire officiel de votre certificat de sécurité.

Si vous perdez cette clé, votre certificat ne sera plus utilisable et vous êtes bon pour en demander un autre avec une nouvelle clé.

(Haut de Page)

Demandons notre certificat!

Attention! Chacun des domaines ou sous domaines indiqués dans le CSR, le certificat de requête, doit pointer sur votre serveur et avoir accès au dossier /.well-known/acme-challenge/Si ce n'est pas le cas, vous n'aurez pas votre certificat.

Tout est en règle ? Allons-y!

Pour Windows, la demande sera sous cette forme : Code: perl

```
le.exe --key account-key.key --email "hostmaster@example.com" --csr
example.com.csr --crt example.com.crt --generate-missing --unlink
    --path "C:\wamp\www\.well-known\acme-challenge" -live
```

Pour Linux, ce sera pareil mais avec **le.pl** : Code: perl

```
le.pl --key account-key.key --email "hostmaster@example.com" -csr
example.com.csr --crt example.com.crt --generate-missing -unlink
--path "/var/www/.well-known/acme-challenge" --live
```

Bien sûr, modifiez le chemin après le paramètre --path pour indiquer où se trouve /.well-known/acme-challenge selon votre configuration.

Le.exe ou le.pl créera des fichiers de signature dans le dossier afin de valider vos domaines et sous domaines qui seront lus par Lets Encrypt pour validation puis il détruira ces fichiers immédiatement après.

Si tout se passe bien, vous allez obtenir votre certificat. Dans notre exemple il s'appellera **example.com.crt**

Il ne reste plus qu'à l'installer sur le serveur.

Si vous souhaitez vous entrainer, retirez le paramètre "--live". Vous obtiendrez un "Fake certificat", un faux certificat. Vous replacerez ce paramètre "--live" quand vous serez prêt. Ca vous permettra d'étudier le certificat obtenu afin de vérifier que tout est bien là. Ces fakes certificats n'ont aucune valeur, vous pouvez en demander autant que vous le voulez pour vérifier que tout se passe bien.

Enfin, quand je dis "autant que vous le voulez", c'est presque ça mais pas tout à fait. Dans un environnement de test, la limite est fixée à 30.000 certificats par semaine et par domaine, 60 vérifications échouées par heure, 50 comptes par adresse IP et par période de 3 heures, etc. Alors doucement quand même.

Pour plus d'infos, lisez la documentation de test Lets Encrypt (Staging) (en anglais).

(Haut de Page)

Installer le certificat sur le serveur

Vous avez besoin de 3 choses.

- La clé privée.
- Le certificat de vos domaines et sous domaines.
- Le certificat intermédiaire (normalement, ce certificat est fourni avec le certificat des domaines, il est compris dedans) et est compris par Apache 2.4 avec la directive Apache SSLCertificateChainFile. On va faire plus compliqué, pour que Apache 2.2 soit utilisable, <u>téléchargez le certificat intermédiaire</u> <u>ou Bundle ici</u> et stockez le dans le même dossier que les autres.

Attention quand même. Si vous utilisez Apache 2.2, pensez à mettre à jour votre version de Apache. Il n'est jamais recommandé d'utiliser de vieilles versions de Apache/PHP.

Maintenant, rendez vous dans la configuration de Apache. Vous devriez trouver un fichier **httpd-ssl.conf** dans le dossier **extra**/

L'arborescence est comme ceci à peu de choses prêt :



Vous pouvez vous inspirer de **httpd.conf** pour avoir une idée des chemins.

Dans ce fichier, nous allons créer un Virtual Host puis nous réglerons les chiffrages autorisés. En tête de fichier, entrez ce qui suit en modifiant les chemins pour qu'ils pointent vers vos certificats (vous avez 2 certificats, le votre mais aussi le Bundle, celui de l'autorité) et votre clé privée. Code: apache

```
Listen 443
<VirtualHost *:443>
# General setup for the virtual host
 ServerName example.com
 ServerAlias www.example.com
 ServerAlias mail.example.com
 ServerAlias static.example.com
 DocumentRoot "C:/wamp/www"
 ServerAdmin noreply@example.com
# SSL Engine Switch:
# Enable/Disable SSL for this virtual host.
 SSLEngine on
 SSLCertificateFile "c:/wamp/ssl/example.com.crt"
 SSLCertificateKeyFile "c:/wamp/ssl/example.com.key"
 SSLCertificateChainFile "c:/wamp/ssl/lets-encrypt-x3-cross-
signed.pem"
</VirtualHost>
```

Sous Linux, vous n'avez qu'à simplement modifier les chemins des dossiers (/var/ssl dans notre exemple).

Ensuite, nous allons nous intéresser aux chiffrages autorisés ainsi qu'aux logs.

Code: apache

```
CustomLog "C:/wamp/logs/access_ssl.log" \"%t %h %{SSL_PROTOCOL}x % {SSL_CIPHER}x \"%r\" %b"

ErrorLog "C:/wamp/logs/apache_error.log"

TransferLog "C:/wamp/logs/access.log"

# modern configuration, tweak to your needs

SSLProtocol all -SSLv3 -SSLv2 -TLSv1 -TLSv1.1

SSLCipherSuite HIGH:!RSA:!RC4:!3DES:!DES:!IDEA:!MD5:!aNULL:!eNULL:!EXP

SSLHonorCipherOrder on

SSLCompression off

SSLSessionTickets on
```

Le protocole SSL n'est plus assez sûr, il présente des vulnérabilités, il est donc interdit sur notre serveur. Seul le TLS est à la hauteur.

Cependant, le TLS 1 est considéré comme plus assez sûr depuis le mois de juin 2018.

De même, vous remarquerez que le TLS 1.1 n'est pas pris en charge. Aucun appareil ni logiciel ne l'utilisent, ce protocole semble être le grand délaissé du monde SSL. Seul le TLS 1.2 est proposé. Ce protocole est utilisé par tous les appareils et navigateurs modernes voir un peu plus ancien.

TLS 1.1 ne sert à rien du tout et TLS 1 n'est plus assez sûr. Vu qu'il ne seront pas utilisés, nous les interdisons donc. Ca ne sert à rien de laisser une porte ouverte et ce n'est pas prudent pour la sûreté de vos communications.

Ensuite, nous autorisons certaines suites de chiffrage parmi les plus solides (HIGH) et nous refusons les autres moins fiables (!RSA:!RC4!!3DES:!DES:!IDEA etc.). Si vous avez vu que RSA n'est pas utilisé, ne prenez pas peur. RSA ne peut être utilisé que pour l'échange de clés, ce qui n'est plus sûr depuis l'année 2017. RSA n'est jamais utilisé pour chiffrer une connexion, la longueur des clés utilisées est un handicap majeur en terme de performances.

Une fois redémarré votre serveur (le redémarrage du serveur est une obligation pour qu'il charge le certificat de sécurité) sous Windows ou le service HTTPD rechargé sous Linux, rendez vous sur ce site : https://www.ssllabs.com/ssltest/ pour tester votre installation SSL. En espérant que tout va pour le mieux.

Et voilà, désormais votre site est accessible en HTTPS!

ATTENTION!

Un certificat de sécurité n'a rien de secret et il ne faut pas y "cacher" des sous domaines que vous souhaitez garder pour vous.

Votre certificat va participer à la "transparence des certificats de sécurité" et il sera vérifié par tous les géants du web.

Vous pourrez le retrouver ici : https://crt.sh/. Ce site semble être victime de son succès et il peut arriver qu'il vous affiche des erreurs 404 qui n'en sont pas ou qu'il soit carrément inaccessible. Patientez, le site est surchargé.

La transparence des certificats est nécessaire afin qu'il n'y ait pas de "triche" comme ça a déjà été le cas avec **StartSSL** (**Startcom**) et **Woosign** qui a été gravement puni en voyant tous ses certificats et ceux de ses clients invalidés.

Ce site en a été victime et notre certificat délivré par Startcom et valable 3 ans n'a plus aujourd'hui aucune valeur. Nous avons donc été obligés de changer dans l'urgence.

(Haut de Page)

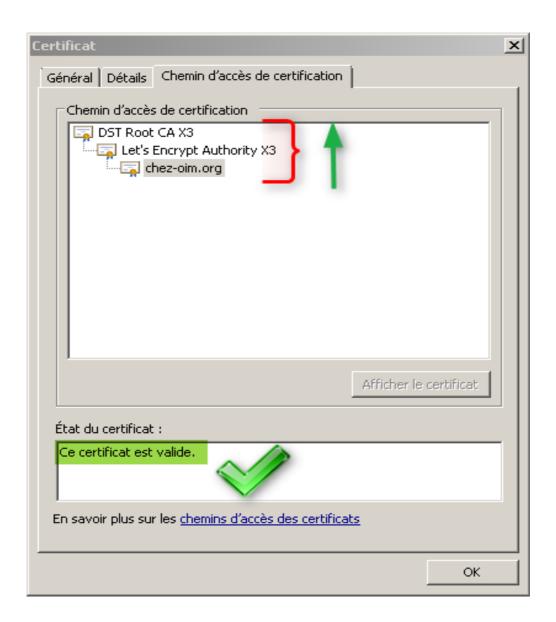
A quoi va ressembler notre certificat?

Prenons le certificat de ce site pour se simplifier la vie. Je vous le répète, un certificat de sécurité n'a rien de secret, vous pouvez récupérer le certificat de ce site ou d'un autre si ça vous chante.

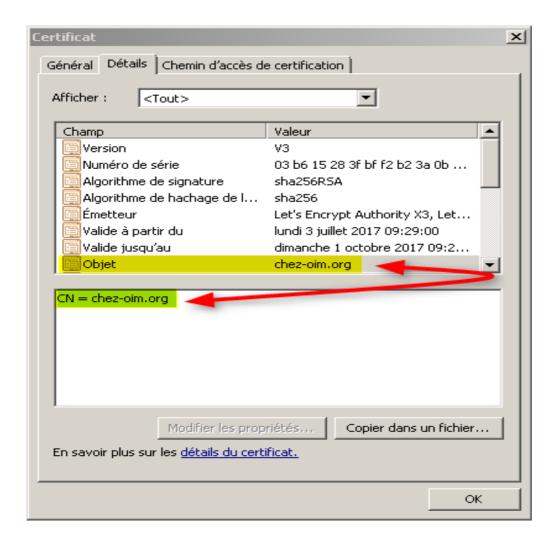
Par contre, si vous n'avez pas la clé privée du certificat, il pourra servir uniquement de décoration au dessus de la télé si vous trouvez ça joli...

Ouvrez votre certificat.

La première chose que vous remarquerez, c'est le chemin d'accès jusqu'à l'autorité de certification qui permettra à votre navigateur web de valider le certificat ou pas.

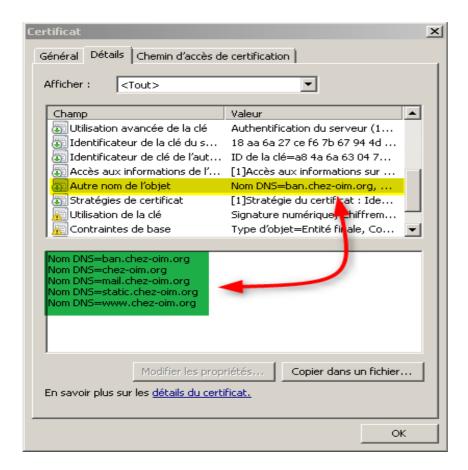


Ensuite, en regardant plus loin, vous verrez apparaître pour quel site le certificat a été délivré. Ca se passe à la ligne " **Objet** "



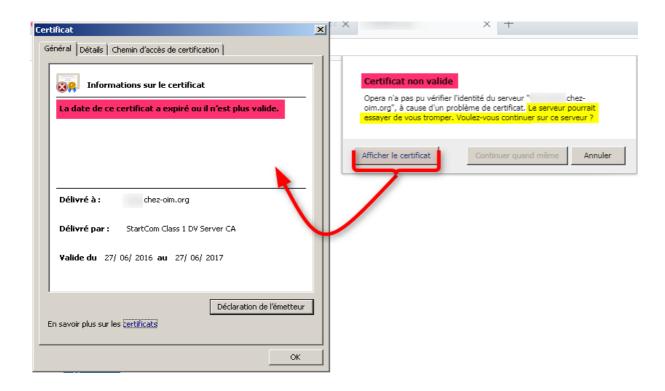
Et ensuite, vous aurez la liste des autre domaines ou sous domaines validés pour lesquels votre certificat fonctionnera.

Il s'agit de la ligne " Autre nom de l'objet " (San).



Faites très attention aux dates de validité du certificat ! Le certificat devra être renouvelé avant sa date de péremption.

Les navigateurs web refuseront d'utiliser un certificat qui n'est plus valide. Ils seront tellement alarmistes que n'importe quel internaute ne prendra pas le risque d'utiliser un certificat périmé et ne viendra plus sur votre site.



Exemple de script Bash pour renouvellement automatique du certificat sous Linux

Renouveler le certificat manuellement tous les 3 mois est une tâche qui devient vite pénible. Voilà donc pourquoi je vous propose un petit script pour Linux qui se chargera de renouveler le certificat automatiquement. Il vous suffit de créer une tâche **cron** pour que ce script se lance tous les jours. Ce script lancera le renouvellement de votre certificat Let's Encrypt si sa date de fin de validité est inférieure à 30 jours.

Pour Windows, vous avez un exemple de script un peu plus bas.

Dans votre dossier ./ssl , créez un sous dossier ssl-renew/ .

Dans ce sous dossier, placez votre clé privée de compte Let's Encrypt, votre clée privée SSL, votre certificat de requête CSR et aussi le certificat SSL (je vous expliquerai plus loin pourquoi tout ce bazard).

Voilà donc ce script : (vous modifierez les chemins et adresse mail selon votre configuration)

Code

```
#!/bin/bash
if ! openssl x509 -checkend 2592000 -in /var/ssl/ssl-
renew/example.com.crt
then
  echo "Renouvellement du certificat SSL." &>TMP SSL1
  /usr/local/bin/le.pl --key
  "/var/ssl/ssl-renew/account-key.key" --csr "/var/ssl/ssl-
renew/example.com.csr" --crt "/var/ssl/ssl-renew/example.com.crt"
--generate-missing --unlink --path "/var/www/html/.well-known/acme-
challenge" --live --renew 30 &>TMP SSL2
  cp -f /var/ssl/ssl-renew/example.com.crt /var/ssl/example.com.crt
  cp -f /var/ssl/ssl-renew/example.com.csr /var/ssl/example.com.csr
  cp -f /var/ssl/ssl-renew/example.com.key /var/ssl/example.com.key
  chown -R apache:apache /var/ssl/
  systemctl reload httpd &>TMP SSL3
  cat TMP SSL1 TMP SSL2 TMP SSL3 | mail -s "Renouvellement du
certificat SSL LetsEncrypt" webmaster@example.com
  rm -f TMP SSL*
else
  echo "Renouvellement du certificat SSL non nécessaire." # | mail -s
"Certificat SSL LetsEncrypt valide" webmaster@example.com
fi
```

La première ligne utilise openssl pour tester la fin de validité du cetificat, en secondes, afin de ne pas lancer le.pl et des requêtes inutiles vers Let's Encrypt, tous les jours.

Un peu de respect ne peut pas faire de mal. Inutile d'interroger les serveurs Let's Encrypt si ce n'est pas nécessaire.

La suite du script se charge de renouveler le certificat si cela est nécessaire et de recharger la nouvelle configuration si le certificat a été renouvelé. Dans la foulée, vous recevrez un mail vous donnant tous les détails du renouvellement afin de savoir si tout c'est bien passé.

Pourquoi renouveler le certificat dans un autre dossier pour le copier ensuite dans son dossier de "travail" ?

C'est la seule méthode que j'ai trouvée pour pouvoir changer de clé. Si vous avez une meilleure idée, je suis preneur!

Si vous décidez de changer de clé pour votre certificat, ce sera très simple. Il vous suffira de créer la nouvelle clé, de générer le nouveau certificat CSR et de copier le tout dans **ssl-renew**/. Au prochain renouvellement du certificat, c'est cette nouvelle clé qui sera utilisée puis la clé et le nouveau certificat seront copiés dans le dossier SSL de Apache.

Si tout se passe correctement, vous recevrez un mail ayant un contenu similaire à celui-ci lors du renouvellement :

Code: bash

```
Renouvellement du certificat SSL.
2018/01/01 00:30:03 [ ZeroSSL Crypt::LE client v0.26 started. ]
2018/01/01 00:30:03 Loading an account key from /var/www/ssl/ssl-
renew/account-key.key
2018/01/01 00:30:03 Loading a CSR from /var/www/ssl/ssl-renew/chez-
oim.org.csr
2018/01/01 00:30:03 Checking certificate
expiration (
local
file).
2018/01/01 00:30:03 Expiration threshold
at 30 days, the certificate expires
in
29 days - will be renewing.
2018/01/01 00:30:04 Registering the account key
2018/01/01 00:30:09 The key is already registered. ID: XXXXXXXX
2018/01/01 00:30:09 Current contact details: dans ton cul@chez-oim.org
2018/01/01 00:30:30 Successfully saved a challenge file
'/var/www/html/.well-known/acme-
challenge/5FQo7DyvIOoPnfSun6yl61MpD9ER-2f1CGI5N6YqW4M'
for domain 'chez-oim.org'
2018/01/01 00:30:30 Successfully saved a challenge file
'/var/www/html/.well-known/acme-
challenge/55ehka3 yRyHYAHSRcNqAPe1gzzeBMSauwycc52HJCM'
for domain 'www.chez-oim.org'
2018/01/01 00:30:30 Successfully saved a challenge file
'/var/www/html/.well-known/acme-
challenge/r 8DpOcjaEES5ByPB RQdgToAl8DHumZFIVQaWL8RhI'
for domain 'static.chez-oim.org'
2018/01/01 00:30:30 Successfully saved a challenge file
'/var/www/html/.well-known/acme-challenge/d3GgjJ2-PbKiHl3C8gt-
ovZsHEzFlyMD7h99tCV0CV8'
for domain 'mail.chez-oim.org'
2018/01/01 00:30:30 Successfully saved a challenge file
'/var/www/html/.well-known/acme-
challenge/MS8ESOStnvOyukdg9BYMDr7LkkY34M8o4PI2nI-E D8'
for domain 'ban.chez-oim.org'
2018/01/01 00:30:35 Domain verification results
```

```
for 'chez-oim.org'
: success.
2018/01/01 00:30:35 Challenge file '/var/www/html/.well-known/acme-
challenge/5FQo7DyvIOoPnfSun6yl61MpD9ER-2f1CGI5N6YqW4M' has been
deleted.
2018/01/01 00:30:38 Domain verification results
for 'www.chez-oim.org'
: success.
2018/01/01 00:30:38 Challenge file '/var/www/html/.well-known/acme-
challenge/55ehka3 yRyHYAHSRcNqAPe1qzzeBMSauwycc52HJCM' has been
deleted.
2018/01/01 00:30:42 Domain verification results
for 'static.chez-oim.org'
: success.
2018/01/01 00:30:42 Challenge file '/var/www/html/.well-known/acme-
challenge/r 8DpOcjaEES5ByPB RQdgToAl8DHumZFIVQaWL8RhI' has been
deleted.
2018/01/01 00:30:46 Domain verification results
for 'mail.chez-oim.org'
: success.
2018/01/01 00:30:46 Challenge file '/var/www/html/.well-known/acme-
challenge/d3GqjJ2-PbKiH13C8qt-ovZsHEzFlyMD7h99tCV0CV8' has been
deleted.
2018/01/01 00:30:49 Domain verification results
for 'ban.chez-oim.org'
: success.
2018/01/01 00:30:49 Challenge file '/var/www/html/.well-known/acme-
challenge/MS8ESOStnvOyukdg9BYMDr7LkkY34M8o4PI2nI-E D8' has been
deleted.
2018/01/01 00:30:49 Requesting domain certificate.
2018/01/01 00:30:50 Requesting issuer's certificate.
2018/01/01 00:30:51 Saving the full certificate chain to
/var/www/ssl/ssl-renew/chez-oim.org.crt.
2018/01/01 00:30:51 The job is done, enjoy your certificate! For
feedback and bug reports contact us at [ https://ZeroSSL.com |
https://Do-Know.com ]
```

(Haut de Page)

Exemple de script Batch pour renouvellement automatique du certificat sous Windows (avec WAMP ou EasyPHP, par exemple)

Comme pour Linux, dans votre dossier ./ssl, créez un sous dossier ssl-renew/.

Dans ce sous dossier, placez votre clé privée de compte Let's Encrypt, votre clée privée SSL, votre certificat de requête CSR et aussi le certificat SSL. N'ommetez pas de copiez également l'exécutable le.exe

Sous Windows, vous utilisez l'exécutable le.exe que nous avons toujours utilisé jusqu'à présent.

C'est ce script qui renouvellera votre certificat (créez un fichier *.bat que vous appellerez lors du renouvellement)

Vous possédez OpenSSL pour Windows:

Code

```
@echo off
openssl x509 -checkend 2592000 -in c:\wamp\ssl\ssl-
renew\example.com.crt
if errorlevel 1 (
      c:\wamp\ssl\ssl-renew\le.exe --key "c:\wamp\ssl\ssl-
renew\account-key.key" --csr "c:\wamp\ssl\ssl-renew\example.com.csr"
--crt "c:\wamp\ssl\ssl-renew\example.com.crt" --generate-missing
--unlink --path "c:\wamp\www\.well-known\acme-challenge" -live --renew
30
      copy c:\wamp\ssl\ssl-renew\example.com.crt
c:\wamp\ssl\example.com.crt
      copy c:\wamp\ssl\ssl-renew\example.com.csr
c:\wamp\ssl\example.com.csr
      copy c:\wamp\ssl\ssl-renew\example.com.key
c:\wamp\ssl\example.com.key
      echo.
      net stop wampapache64
      net start wampapache64
      rem php -f c:\wamp\ssl\script annonçant le renouvellement.php
)
```

Vous ne possédez pas OpenSSL pour Windows :

Code

```
@echo off
c:\wamp\ssl\ssl-renew\le.exe --key "c:\wamp\ssl\ssl-renew\account-
key.key" --csr "c:\wamp\ssl\ssl-renew\example.com.csr" --crt
"c:\wamp\ssl\ssl-renew\example.com.crt" --generate-missing --unlink
--path "c:\wamp\www\.well-known\acme-challenge" --live --renew 30
--issue-code 99
if errorlevel 99 (
      copy c:\wamp\ssl\ssl-renew\example.com.crt
c:\wamp\ssl\example.com.crt
      copy c:\wamp\ssl\ssl-renew\example.com.csr
c:\wamp\ssl\example.com.csr
      copy c:\wamp\ssl\ssl-renew\example.com.key
c:\wamp\ssl\example.com.key
      echo.
      net stop wampapache64
      net start wampapache64
      rem php -f c:\wamp\ssl\script annonçant le renouvellement.php
```

ATTENTION!

Sous Windows, vous n'aurez pas la possibilité de recevoir un Email si le certificat est renouvelé. **Soyez vigilants lors du renouvellement!**

Vous remarquerez une ligne en commentaire (débutant par "rem") qui exécute un script PHP. Vous êtes libres de retirer ce commentaire, de renommer le script et de le créer pour qu'il vous envoie un mail ou toute autre information lors du renouvellement.

Pour lancez ce script Batch toutes les 24 heures, cette horreur appelée "Planificateur de Tâches" de Windows vous attend...

Cliquez sur **Démarrer** et Ouvrez le planificateur de tâches en tapant son nom puis cliquez sur **Créer une tâche** (pas Tâche de base !)

Dans la fenêtre qui s'ouvre, au premier onglet, donnez un nom à votre tâche et une description si le coeur vous en dit.

Cliquez sur **Utilisateurs ou groupe...** suivi de **Avancé** puis **Rechercher**. Là, sélectionnez l'utilisateur **Administrateur** et validez (Le script doit être exécuté avec les privilèges Administrateur sinon le service Apache ne pourra pas être redémmarré automatiquement).

Ensuite, passez à l'onglet **Déclencheurs** dans lequel vous allez définir l'heure d'exécution du script en cliquant sur **Nouveau** et vous cocherez la case **Chaque jour** et **Répéter chaque 1 jour** .

Et enfin, dans l'onglet **Actions** , vous n'avez plus qu'à cliquez sur **Nouveau** et indiquer l'emplacement de votre script.

Ouf! C'est fini! Ah oui, ca n'a rien à voir avec la cron de Nunux, hein?

Pourquoi renouveler le certificat dans un autre dossier pour le copier ensuite dans son dossier de "travail" ?

C'est la seule méthode que j'ai trouvée pour pouvoir changer de clé. Si vous avez une meilleure idée, je suis preneur!

Si vous décidez de changer de clé pour votre certificat, ce sera très simple. Il vous suffira de créer la nouvelle clé, de générer le nouveau certificat CSR et de copier le tout dans **ssl-renew**/. Au prochain renouvellement du certificat, c'est cette nouvelle clé qui sera utilisée puis la clé et le nouveau certificat seront copiés dans le dossier SSL de Apache.

(Haut de Page)

Mise à Jour : Obtenir un certificat Wildcard (Certificat générique)

On est d'accord que vous avez lu ce qui est ci-dessus, afin que vous sachiez obtenir un certificat avec les outils nécessaires installés.

Depuis l'adoption du protocole **ACMEv2** par Lets Encrypt, au début du mois de mars 2018, adopté par la bibliothèque **Crypt::LE** depuis la version v0.30, il est désormais possible d'obtenir un certificat de sécurité wildcard (un certificat générique) auprès de Lets Encrypt.

Un certificat Wildcard permet d'avoir un certificat de sécurité fonctionnant avec un nombre illimité de sous domaines, même si ceux-ci n'existent pas à la création du certificat.

Le seul vrai souci réside dans l'obtention de ce certificat de sécurité. Seule une validation DNS est possible, aucune vérification HTTP n'est proposée comme avec un certificat "normal". Cela viendra certainement, mais ce n'est pas encore proposé.

Ceci vous obligera à créer des enregistrements DNS afin de valider les requêtes de vérification de Lets Encrypt et ce, sans aucune API disponible pour automatiser tout cela.

Il serait donc plus pratique de se tourner vers un outil proposant beaucoup d'APIs DNS comme **acme.sh** (à ce sujet, un tuto sur **acme.sh** va voir le jour sous peu).

Pour obtenir notre certificat, il ne sera pas nécessaire d'indiquer les sous domaines utilisés. Seul le domaine et le sous domaine wildcard (générique en français) seront utilisés. **Par exemple : example.com** et *.example.com .

Pour Windows, il vous faudra entrer ceci:

Code: bash

```
le.exe --key account-key.key --email "hostmaster@example.com"
--csr example.com.csr --csr-key example.com.key --crt example.com.crt
--domains "example.com, *.example.com" --generate-missing --handle-as
dns --api 2 --live
```

Pour Linux, vous savez maintenant que la syntaxe est la même, il suffit simplement d'utiliser le.pl et pas le.exe :

Code: bash

```
le.pl --key account-key.key --email "hostmaster@example.com" --csr
example.com.csr --csr-key example.com.key --crt example.com.crt
--domains "example.com, *.example.com" --generate-missing --handle-as
dns --api 2 --live
```

Bien entendu, vous remplacerez **example.com** par votre nom de domaine. Une fois le script lancé, il vous sera demandé de créer un enregistrement DNS TXT et d'appuyer sur entrée. Appuyez donc sur entrée. Une seconde demande d'enregistrement DNS vous sera demandée et ensuite un appuis sur entrée. **ATTENTION!** Cette seconde fois, n'appuyez pas sur entrée tout de suite!

Votre écran ressemblera à cette image ci-dessous :

```
2018/04/11 21:11:34 [ZeroSSL Crypt::LE client v0.31 started.]
2018/04/11 21:11:34 Loading an account key from account.key
2018/04/11 21:11:34 Generating a new CSR for domains example.com,*.example.com
2018/04/11 21:11:34 New CSR will be based on a generated key
2018/04/11 21:11:39 Saving a new CSR into example.com.csr
2018/04/11 21:11:39 Saving a new CSR key into example.com.key
2018/04/11 21:11:44 Registering the account key
2018/04/11 21:11:44 The key has been successfully registered. ID:
2018/04/11 21:11:44 Make sure to check TOS at https://letsencrypt.org/documents/LE-SA-v1.2-November-15-2017.pdf
Challenge for 'example.com' requires the following DNS record to be created:
Host: _acme-challenge.example.com, type: TXT, value: DMuynZWGoaP@rFeoUg45Y-HuNn9Vkn4s5ZD9ZKXmcjI
Wait for DNS to update by checking it with the command: nslookup -q=TXT _acme-challenge.example.com
When you see a text record returned, press <Enter>
Challenge for '*.example.com' requires the following DNS record to be created:
Host: _acme-challenge.example.com, type: TXT, value: yzEA109GFGkVEGzu1fdh5YnDNLmcg2bivIZgaR5V1Hs
Wait for DNS to update by checking it with the command: nslookup -q=TXT _acme-challenge.example.com
When you see a text record returned, press <Enter>
```

Dans notre exemple, il nous sera demandé de créer 2 enregistrements DNS TXT pour le sous domaine **_acme-challenge.example.com** (en bleu).

En rouge, une instruction vous est donnée afin de vérifier que les enregistrements DNS se sont propagés (la propagation est très rapide, Lets Encrypt interroge les DNS les plus proches du domaine, les DNS autoritaires).

Vous pouvez copier/coller l'instruction donnée pour l'utiliser dans une nouvelle console (fenêtre). N'interrompez pas le.pl/le.exe pour vérifier les enregistrements DNS! Faites le dans une nouvelle console (fenêtre). Sinon, vous seriez contraint de tout recommencer.

N'essayez pas de créer d'enregistrement TXT pour le sous domaine *._acme-challenge.example.com . Le caractère wildcard n'est pas autorisé dans cette situation.

Une fois que vos enregistrements sont créés et propagés (2 minutes d'attente suffisent amplement), vous pouvez enfin appuyer un dernière fois sur la touche entrée et laisser faire.

Lets Encrypt va vérifier vos enregistrements DNS et il créera le certificat immédiatement après. Sinon, il vous sera mentionné ce qui ne va pas et il faudra tout recommencer.

Vous avez votre certificat wildcard? Vous pouvez supprimer les enregistrements DNS et installer le certificat sur votre serveur.

(Haut de Page)

Révoquer un certificat

Révoquer un certificat revient à l'invalider pour qu'il ne puisse plus être utilisé. Les raisons de la révocation d'un certificat sont nombreuses :

- Votre clé privée a été divulguée. Par sécurité, la révocation du certificat est une obligation !
- Vous avez perdu votre clé privée.

- Vous voulez ajouter un domaine/sous domaine sur un nouveau certificat. Il est plus prudent de révoquer l'ancien.
- Vous avez fait des essais sans passer par des "fakes certificats" et il est nécessaire de faire du ménage.
- Vous vendez votre domaine. Si ce n'est pas fait, le nouvel acquéreur vous demandera très certainement de révoquer tous les certificats concernant ce domaine.
- Etc, etc, etc...

La révocation ne sera pas immédiate, ne vous impatientez pas. Il faut plusieurs heures et chaque navigateur vérifie comme il veut les certificats.

Pour les geeks, il s'agira d'une révocation **OCSP** . Cela ira donc un peu plus vite que pour les listes **CRL** . Quelques jours après une révocation, ne vous étonnez pas de voir un certificat inutilisable sur un navigateur mais parfaitement valable sur un autre.

La révocation d'un certificat est très simple. Vous aurez besoin de votre clé privée de compte Lets Encrypt et du certificat à révoquer.

Dans notre exemple, nous allons révoquer le certificat example.com.crt

Sous Linux, il faudra simplement entrer:

Code: bash

le.pl --key account.key --crt example.com.crt --revoke

Et pour Windows:

Code: batch

le.exe --key account.key --crt example.com.crt --revoke

Et voilà, c'est fini! Vous pourrez vérifier que la révocation a bien été prise en compte sur le site <u>crt.sh</u>. Recherchez votre certificat et affichez le. Sur la page qui s'affiche, repérez la ligne **Revocation** et la colonne **Status**.

Là, cliquez sur **Check** . Le statut révoqué de votre certificat devrait apparaître après un petit temps d'attente.

crt.sh ID	<u>338555168</u>									
Summary	Leaf certificate									
Certificate	Timestamp		Entry #	Log Operator	Log URL					
Transparency	2018-02-22 23:11:13 UTC		3665563	Cloudflare	https://ct.cloudflare.com/logs/nimbus201					
	2018-02-22 23:11:13	3 UTC 83	3675607	Google	https://cf	t.googleapis.com/logs/ar	gon2018			
	2018-02-22 23:11:13	3 UTC 20	03016048	Google	https://cf	t.googleapis.com/icarus				
Revocation	Mechanism P	rovide	r Status	Revocati	on Date	Last Observed in CRL	Last Checked (Error)			
	OCSP T	he CA	Check	?		n/a	?			
Report a problem with this certificate to the CA		he CA	Unknown	11/0		n/a				
	CRLSet/Blacklist G	Google	Not Revo	ked n/a		n/a	n/a			
	disallowedcert.stl M	/licrosof	t Not Revo	ked n/a		n/a	n/a			
	OneCRL M	/lozilla	Not Revo	ked n/a		n/a	n/a			

Vous pouvez vérifier ce statut sur un certificat que j'ai déjà révoqué par le passé : https://crt.sh/?id=338555168

Ce site semble être victime de son succès et il peut arriver qu'il vous affiche des erreurs 404 qui n'en sont pas ou qu'il soit carrément inaccessible. Patientez, le site est surchargé.

Vous devriez obtenir quelque chose de similaire à ceci :

Revocation	Mechanism	Provider	Status	Revocation Date	Last Observed in CRL	Last Checked (Error)
	OCSP	The CA	Revoked	2018-02-22 23:22:38 UTC	n/a	2018-04-17 10:12:50 UTC
Report a problem with this certificate to the CA	CRL	The CA	Unknown	n/a	n/a	
	CRLSet/Blacklist	Google	Not Revoked	n/a	n/a	n/a
	disallowedcert.st	Microsoft	Not Revoked	n/a	n/a	n/a
	<u>OneCRL</u>	Mozilla	Not Revoked	n/a	n/a	n/a

