# Certificats pour EzGED

**IMPORTANT** Dans tous les cas si vous êtes amenés à générer des CSR (demande de certificat numérique, préformatées). Il est impératif de faire une sauvegarde de la machine et de l'environnement qui a permis de générer cette CSR, car la plupart du temps, seule cette machine pourra à nouveau générer une CSR ou extraire la clé privée du certificat et la clé publique en vue de configurer EzGED.

## Certificat SSL (HTTPS dans apache)

Ce certificat est destiné à chiffrer la liaison, rendant ainsi les données qui y transitent illisibles, et à protéger votre client des hackers en tout genre. Il protège la communication entre le navigateur du client et votre serveur. Il est nécessaire voir vital, dès que vous ouvrez votre EzGED à l'extérieur.

### Certificats de confiance

Un certificat est dit de confiance si le navigateur (qui va vérifier la validité du certificat) connait l'autorité de certification qui l'a émis.

Nous vous conseillons Comodo comme fournisseur de certificat SSL pour environ 80 euros /an si vous le prenez pour 3 ans.

Pré-Requis :

```
          Disposer d'une IP FIXE
          CONNAITRE LE NOM COMPLET DE L'URL DONT VOUS VOUS SERVIREZ POUR
HEBERGER LA GED
          exemple ged.masociete.fr  où masociete.fr est le domaine internet
de votre société.
          Il faudra avoir la possibilité de lier sur votre serveur DNS
ged.masociete.fr à votre ip fixe.
```

Lien d'achat du certificat : https://ssl.comodo.com/comodo-ssl-certificate.php

cliquez sur le bouton orange comodo ssl get now

Remplissez le formulaire :




Puis suivez la procédure.

## Certificats auto-signé

Il faut installer la dernière version d'OpenSSL.

Un certificat auto-signé est un certificat que vous générez vous-même. Il est important de noter qu'il n'est en rien (à attributs équivalents) moins protecteur qu'un certificat émis par une autorité certifiée. MAIS il ne sera pas connu des navigateurs des clients. Ces derniers seront invités à reconnaitre explicitement votre certificat comme étant de confiance.

On va tout d'abord générer une clé privée:

```
openssl genrsa -des3 -out mydomain.key 1024
```

*mydomain* est à remplacer préférablement par l'url d'accès à EZGED (exemple: monserveurged.fr)
On nous demande de saisir une “pass phrase” (i.e un mot de passe). On choisit le mot de passe de notre choix.

On génère ensuite le fichier .csr (une demande de signature de certificat):

```
openssl req -new -key mydomain.key -out mydomain.csr -config
c:\openssl\share\openssl.cnf
```

*c:\openssl* est à remplacer avec le chemin vers lequel vous aurez installé OpenSSL

A partir de cette demande de signature de certificat nous générons le certificat:

```
openssl x509 -req -days 730 -in mydomain.csr -signkey mydomain.key -out
mydomain.crt
```

Nous avons presque terminé. Nous avons un fichier certificat et une clé privée. Cette dernière étant protégée par mot de passe nous allons la déchiffrée ainsi:

```
copy mydomain.key mydomain.key.org
openssl rsa -in mydomain.key.org -out mydomain.key
```

Remarques:

- La procédure pour générer et installer un certificat est également disponible ici: http://updates.nchp.net/doc/installhttps.pdf
- Cette procédure est valable si l'on souhaite générer un certificat auto-signé à usage de signature.

## Installation

Que ce soit un certificat émis par une autorité de certification ou un certificat auto-signé vous devez avoir en votre possession:

- Un fichier contenant le certificat (et sa clé publique) au format .crt
- Un fichier contenant la clé privée “déchiffrée” [1)]

Dans le fichier de configuration Apache (Ex: C:\nchp\Apache2\conf\httpd.conf), ajoutez les directives suivantes :

(Où 192.168.XXX.XXX est à remplacer par l'adresse IP de votre serveur EzGED)

En début de fichier, là où doit se trouver déjà une instruction *Listen*

```
Listen 192.168.XXX.XXX:443
```

Localisez la ligne suivante et décommentez-là (enlevez le signe # qui la précède)

```
LoadModule ssl_module modules/mod_ssl.so
```

En fin de fichier, après les définitions d'Alias déjà existante :

```
NameVirtualHost 192.168.XXX.XXX:443
<VirtualHost 192.168.XXX.XXX:443>
DocumentRoot c:/nchp/usr/local/nchp/ezged/www
ServerName 192.168.XXX.XXX
SSLEngine On
SSLOptions +FakeBasicAuth +ExportCertData +StrictRequire
SSLCertificateFile c:/nchp/etc/nchp/ezged/mydomain.crt
SSLCertificateKeyFile c:/nchp/etc/nchp/ezged/mydomain.key
</VirtualHost>
```

Sauvegardez le fichier et il ne vous reste plus qu'à relancer Apache:

```
net stop apache2.2
net start apache2.2
```

# Signature de masse

C'est le tampon numérique de la société au nom de la personne morale.

vous le trouverez chez Chambersign Negocio environ 800 euros pour 3 ans

http://www.chambersign.fr/certificat-cachet-serveur-negocio/

Il est demandé par le gérant de la société

Pièces nécessaires :

```
  Justificatif de la nomination du représentant légal ou de l'autorité
habilité (maire, président, directeur générale, procès-verbal ….) ou extrait
kbis
  Copie de la pièce d'identité en cours de validité du représentant légal ou
de l'autorité habilitée datée et signée de moins de trois mois
  Copie de la pièce d'identité en cours de validité du demandeur du
certificat (gestionnaire du certificat) datée et signée de moins de trois
```

```
mois
```

## Signature PDF intégrée

Avec cette méthode les fichiers PDF sont signés et la signature est incluse dans le PDF.
Elle fonctionne avec un fichier au format PKCS12 (extension .p12 ou .pfx). Ce type de fichiers embarque à la fois le certificat et la clé privée et il est protégé par mot de passe.

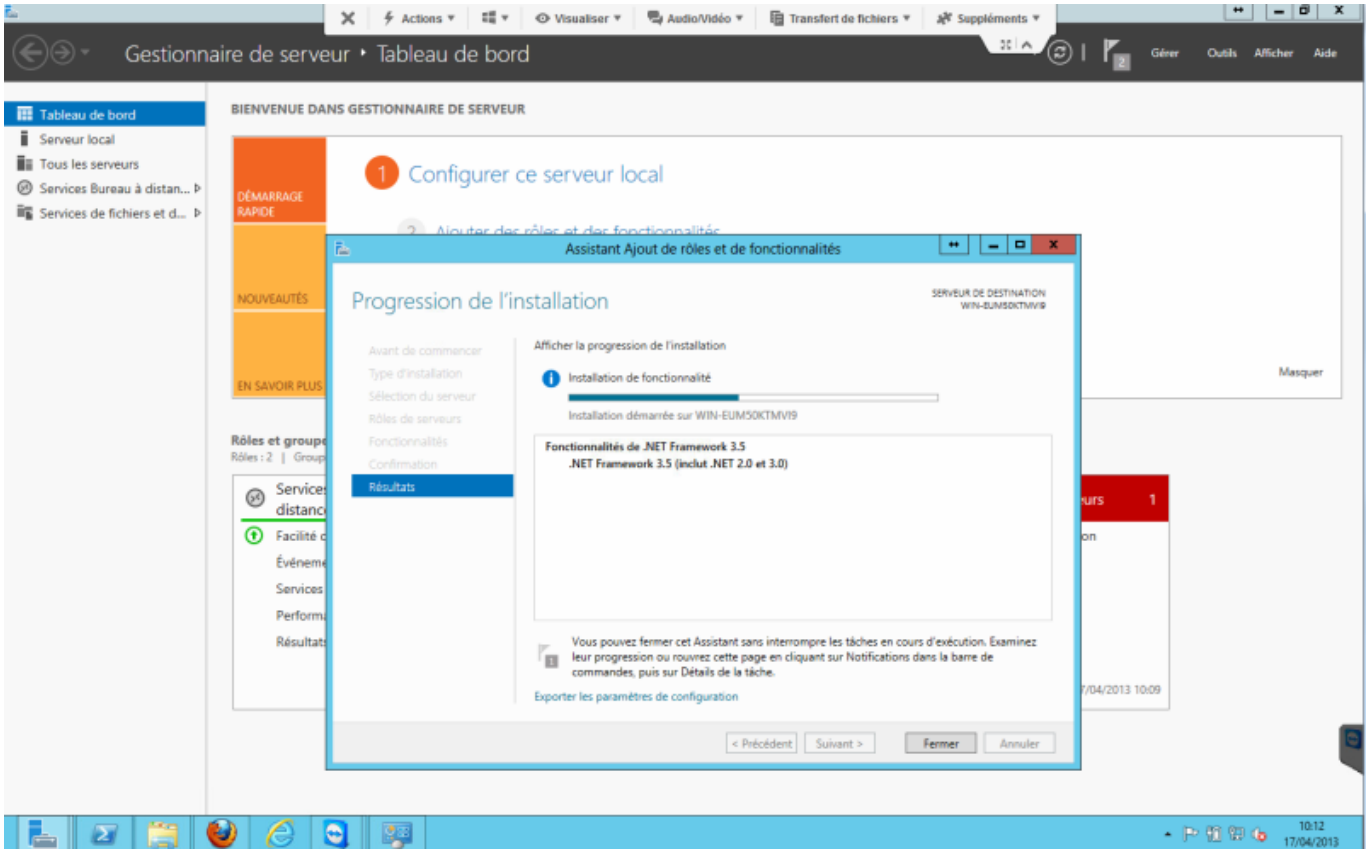
La configuration dans le fichier instance.conf s'effectue comme suit:

```
[ezged]
   pdfsignp12path = c:\nchp\etc\nchp\ezged\moncertificat.p12
   pdfsignp12pass = votre mot de passe
   pdfsignp12mask = 1
   digisign = 1
```

La paramètre ***pdfsignp12mask*** indique si oui (1) ou non (0) la signature doit être masquée, c'est-à-dire ne pas apparaître visuellement sur le fichier PDF sous la forme d'un tampon. Si elle n'apparaît pas visuellement la plupart des visionneur PDF notifierons toutefois de la présence d'une signature.

### Problème connu

Si après la mise en place de la signature vos travaux de COLD tombent en erreur à l'étape de stockage, qui est l'étape durant laquelle le fichier sera signé, vérifiez que le framework .Net 3.5, dont dépend l'outil de signature, est bien installé.



## Signature détachée

Elle concerne tout type de fichiers (les PDF aussi). Elle se présente sous la forme d'un fichier au format PKCS7 (extension .p7s) qui contient la signature du fichier.

Elle peut être conjointement utilisée avec la signature PDF intégrée (dans ce cas les PDF sont signés avec la signature intégrée, et les autres types de fichiers ont une signature détachée).

Pour la mettre en place il faut se munir:

- D'un fichier certificat (.crt)
- D'une clé privée déchiffrée (.key)

Vous pouvez les obtenir à partir du fichier PKCS12 (.p12 ou .pfx).

Pour extraire le certificat du fichier p12:

```
openssl pkcs12 -in moncertificat.p12 -clcerts -nokeys -out moncertificat.crt
```

Pour extraire la clé privée du fichier p12:

```
openssl pkcs12 -in moncertificat.p12 -nocerts -out macleprivee.key
```

N'oubliez pas qu'il faut déchiffrer la clé:

```
openssl rsa -in macleprivee.key -out macleprivee-dechiffree.key
```

A présent nous pouvons renseigner le fichier instance.conf:

```
[ezged]
digisign = 1
digisigncert = C:\nchp\etc\nchp\ezged\moncertificat.crt
digisignkey = C:\nchp\etc\nchp\ezged\macleprivee-dechiffree.key
```

Les chemins sont à adapter selon votre cas.



Utilisez le mode `legacy` si vous rencontrez des erreurs de compatibilité entre votre version d'OpenSSL et celle du fichier p12.

Si par exemple vous obtenez une erreur semblable à celle ci-dessous au moment d'exporter le certificat:

```
Error outputting keys and certificates
083B0000:error:0308010C:digital envelope
routines:inner_evp_generic_fetch:unsupported:../openssl-3.1.2/crypto/evp/evp
_fetch.c:341:Global default library context, Algorithm (RC2-40-CBC : 0),
```

```
Properties ()
```

Dans ce cas ajoutez l'option `-legacy` à vos commandes. Exemple:

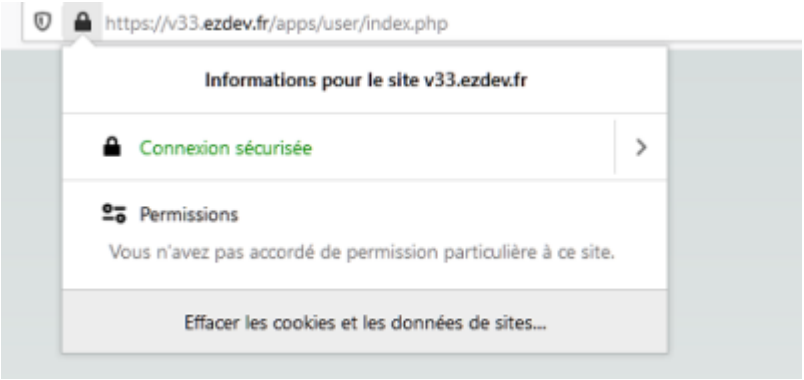
```
openssl pkcs12 -legacy -in moncertificat.p12 -clcerts -nokeys -out
moncertificat.crt
```

# Message d'erreur "the ssl connection could not be established see inner exception" sur l'application mobile Android.

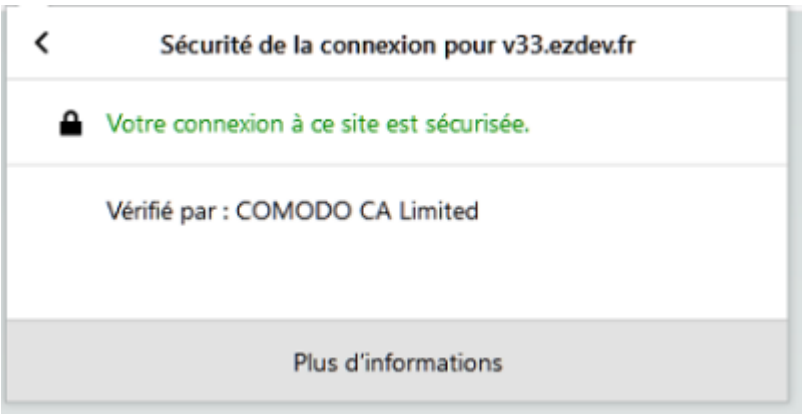
Ce message d'erreur est affiché lorsque que le certificat racine (ca-bundle) est expiré.

Solution: Récupérer le ca-bundle: (ne fonctionne pas pour les certificats auto-signé) exemple sur un certificat comodo (setigo):

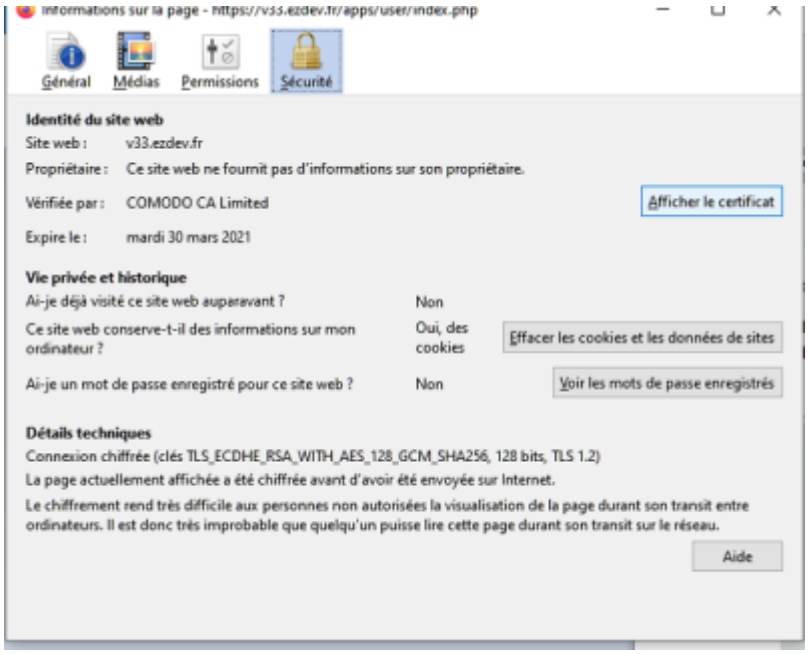
utilisez un navigateur à jour (ex: Firefox), allez sur l'url de l'application et cliquez sur le cadenas




cliquez sur connexion sécurisée, ensuite sur plus d'information




une fenêtre s'affiche contenant des informations, cliquez sur Afficher le certificat




Cette action ouvre une page web, cliquer sur télécharger: PEM (chain)




Enregistrer le fichier sous le nom de (name).ca-bundle.crt et édité le avec votre éditeur de texte préféré. Supprimer le premier bloc. Ce bloc correspond a votre certificat. Les autres blocs représentent la chain.




Placez ce fichier dans votre dossier apache où ce situe les autres *.crt

Éditez votre fichier httpd.conf

Cherchez votre bloc où vous déclarez les certificats. Ajoutez la ligne: SSLCertificateChainFile {votre path du ca-bundle}.ca-bundle.crt

```
NameVirtualHost 192.168.XXX.XXX:443
<VirtualHost 192.168.XXX.XXX:443>
DocumentRoot c:/nchp/usr/local/nchp/ezged/www
ServerName 192.168.XXX.XXX
SSLEngine On
SSLOptions +FakeBasicAuth +ExportCertData +StrictRequire
SSLCertificateFile c:/nchp/etc/nchp/ezged/mydomain.crt
SSLCertificateKeyFile c:/nchp/etc/nchp/ezged/mydomain.key
```

```
SSLCertificateChainFile c:/nchp/etc/nchp/ezged/myname.ca-bundle.crt
</VirtualHost>
```

pour conclure redémarrer apache.

# Certificats et applications mobiles

Lorsque EzGED doit être accessible depuis une application mobile il convient de s'assurer que le certificat TLS installé sur le serveur soit accepté par les OS mobiles donc principalement par iOS et Android.

Liste des certificats racines de confiance pour:

- Les OS Apple https://support.apple.com/fr-fr/HT209143
- Android : pas de liste officielle

# Utilisation de OpenSSL

## Convertir un DER en PEM

```
openssl x509 -inform der -in moncertificat.der -out moncertificat.crt
```

[1)]

une clé privée étant généralement protégée par mot de passe

From:
http://wiki.ezdev.fr/ - **EzGED Wiki**

Permanent link:
**http://wiki.ezdev.fr/doku.php?id=certificats&rev=1735297585**

Last update: **2024/12/27 11:06**