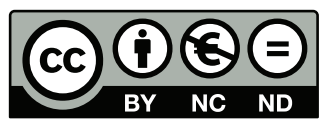
OpenSSL

Command Line Tool

Didier BERNAUDEAU

http://didier.bernaudeau.net/slide

3 février 2015



Sommaire

Présentation d'OpenSSL

Origine

- Edité par OpenSSL Software Foundation
- Ecrit en langage C par Eric Young de Cryptsoft
- "Fork" et successeur de SSLeay (Abandonné en 1998)

Licences "BSD-style Open Source licenses"

- OpenSSL License
- SSLeay license

Composants

- Outil en ligne de commande (OpenSSL)
- SSL API (Langage C)
- Crypto API (Langage C)

La suite de cette présentation concerne l'outil en ligne de commande.

Présentation du contexte

TestSign SA

Société Anonyme constituée

- Autorité d'Enregistrement chargée de recevoir et de vérifier les demandes de certificat
- Autorité de Certification chargée d'émettre les certificats

Root CA est une Autorité de Certification racine

- DN : CN=TestSign - Root CA ; OU=TestSign - CA ; O=TestSign SA ; C=FR
- Durée de validité : 30 ans
- usage : signe les certificats pour les AC intermédiaires de TestSign SA

Server CA est une Autorité de Certification intermédiaire

- DN : CN=TestSign - Server CA ; OU=TestSign - CA ; O=TestSign SA ; C=FR
- Durée de validité : 10 ans
- usage : signe les certificats pour des Serveurs SSL/TLS valable 2 ans

Sommaire

Créer une autorité de certification racine

Etape 1 - Créer les dossiers

Dossier pour conserver les certificats émis par l'autorité de certification

```
$ mkdir -p PKI/RootCA/certs
```

Dossier pour archiver les certificats émis par l'autorité de certification

```
$ mkdir -p PKI/RootCA/newcerts
```

Dossier pour stocker les demandes de certificats

```
$ mkdir -p PKI/RootCA/csr
```

Dossier pour stocker les listes de révocation

```
$ mkdir -p PKI/RootCA/crl
```

Dossier pour stocker la clé privée de l'autorité de certification

```
$ mkdir -m 700 -p PKI/RootCA/private
```

Créer une autorité de certification racine

Etape 2 - Créer les fichiers

Fichier de la base de données de l'autorité de certification

```
$ touch PKI/RootCA/index.txt
```

Fichier indiquant le numéro de la CRL

```
$ echo 01 > PKI/RootCA/crlnumber
```

Fichier indiquant le numéro de série du prochain certificat

```
$ openssl rand 16 -hex > PKI/RootCA/serial
```

Fichier de configuration de l'autorité de certification racine

```
$ cp /etc/ssl/openssl.cnf PKI/RootCA/RootCA.cnf
```

Créer une autorité de certification racine

Etape 3 - Configurer l'autorité de certification (1/3)

Editer le fichier PKI/RootCA/RootCA.cnf

```
[ CA_default ]

dir             = PKI/RootCA                    # Nom du dossier de l'AC
certs           = $dir/certs
crl_dir         = $dir/crl
database        = $dir/index.txt                # Nom du fichier de la base de données de l'AC
new_certs_dir   = $dir/newcerts

certificate     = $dir/RootCA.crt               # Nom du fichier du certificat de l'AC
serial          = $dir/serial
crlnumber       = $dir/crlnumber

crl             = $dir/crl.pem
private_key     = $dir/private/RootCA.key       # Nom du fichier de la clé privée de l'AC
RANDFILE        = $dir/private/.rand

x509_extensions = usr_cert                      # Nom de la section définissant les extensions

name_opt        = ca_default
cert_opt        = ca_default

default_days    = 3650                          # Durée de validité des certificats (10 ans)
default_crl_days= 30
default_md      = sha256                        # Fonction de hachage utilisée pour la signature
preserve        = no

policy          = policy_match                  # Nom de la section définissant les règles
```

Créer une autorité de certification racine

Etape 3 - Configurer l'autorité de certification (2/3)

Editer le fichier PKI/RootCA/RootCA.cnf

```
[ policy_match ]
# Cette section définit les contôles effectués sur le DN (Distinguished Name) présent dans la CSR

countryName             = match
stateOrProvinceName     = optional
organizationName        = match
organizationalUnitName  = match
commonName              = supplied
emailAddress            = optional

# match : la valeur du champs dans la CSR doit correspondre avec celle de l'AC
# supplied : le champs doit être obligatoirement définit dans la CSR
# optional : le champs n'est pas obligatoire
# Les champs non mentionnés dans la liste ci-dessus ne seront pas autorisés


[ v3_ca ]
# Cette section définit les extensions à ajouter dans un certificat d'une AC racine

# Ajouter un identifiant de la clé obtenu à partir d'une empreinte (SHA1) de la clé publique
subjectKeyIdentifier    = hash

# Retirer cette extension qui n'est pas nécessaire pour une AC racine
# authorityKeyIdentifier=keyid:always,issuer

# Définir les contraintes de base du certificat : "CA:true" spécifie que le certificat sera une AC (Racine)
basicConstraints        = critical,CA:true

# Limiter l'usage des certificats émis par l'AC racine (Signature de certificats et de CRL)
keyUsage                = critical, cRLSign, keyCertSign
```

Créer une autorité de certification racine

Etape 3 - Configurer l'autorité de certification (3/3)

Editer le fichier PKI/RootCA/RootCA.cnf

```
[ usr_cert ]
# Cette section définit les extensions à insérer dans les certificats qui seront signés par l'AC racine

# Définir les contraintes de base des certificats :
#   - CA:true : spécifie que le certificat sera une AC (Intermédiaire)
#   - pathlen:0 : spécifie que le certificat de l'AC (Intermétidiaire) signera des certificats finaux
basicConstraints       = critical,CA:true,pathlen:0

# Limiter l'usage des certificats émis par l'Autorité de Certification racine
# (Signature de certificats et de CRL)
keyUsage               = critical, cRLSign, keyCertSign

# Retirer les extensions NetScape devenues obsolètes
# nsComment            = ""

# Ajouter un identifiant de la clé obtenu à partir d'une empreinte (SHA1) de la clé publique
subjectKeyIdentifier   = hash

# Ajouter l'identifiant de lé clé de l'Autorité de Certification racine
authorityKeyIdentifier = keyid

# Spécifie l'adresse du service OCSP
authorityInfoAccess    = OCSP;URI:http://ocsp.testsign.fr/RootCA/

# Spécifie l'adresse de la CRL
crlDistributionPoints  = URI:http://www.testsign.fr/RootCA.crl
```

Drapeau "Critical"

Lorsque le drapeau "critical" est ajouté à une extension, celle-ci devra obligatoirement être contrôlée.

Créer une autorité de certification racine

Etape 4 - Créer la clé privée

```
$ openssl genpkey -out PKI/RootCA/private/RootCA.key -pass stdin \
  -aes-256-cbc -algorithm RSA -pkeyopt rsa_keygen_bits:8192
```

Paramètres

- -out : spécifie le fichier de la clé privée
- -pass stdin : spécifie la saisie d'un mot de passe pour chiffrer la clé privée
- -aes-256-cbc : spécifie l'algorithme de chiffrement de la clé privée
- -algorithm : spécifie l'algorithme pour générer la clé privée (RSA)
- -pkeyopt : spécifie les options de la clé privée (8192 bits)

stin (Standard Input)
Pour éviter toute comprimission du mot de passe, il ne faut pas l'écrire dans la ligne de commande.
Avec l'option "stdin", le mot de passe devra être saisie après validation de la ligne de commande. Ainsi, le mot de passe ne sera pas présent dans l'historique des commandes.

Créer une autorité de certification racine

Etape 5 - Créer la demande de certificat

```
$ openssl req -new -sha256 -key PKI/RootCA/private/RootCA.key \
  -out PKI/RootCA/RootCA.csr -passin stdin \
  -subj "/C=FR/O=TestSign SA/OU=TestSign - CA/CN=TestSign - Root CA"
```

Paramètres

- -new : spécifie la création d'une nouvelle demande de certificat
- -sha256 : spécifie l'algorithme de signature (sha256 With RSA Encryption)
- -key : spécifie le fichier de la clé privée
- -out : spécifie le fichier de la demande de certificat
- -passin stdin : spécifie la saisie du mot de passe de chiffrement de la clé privée
- -subj : spécifie le DN (Distinguished Name) du certificat

Créer une autorité de certification racine

Etape 6 - Signer le certificat

```
$ openssl ca -config PKI/RootCA/RootCA.cnf -extensions v3_ca -days 10950 \
  -md sha256 -keyfile PKI/RootCA/private/RootCA.key \
  -passin stdin -selfsign -out PKI/RootCA/RootCA.crt \
  -notext -infiles PKI/RootCA/RootCA.csr
```

Paramètres

- -config : spécifie le fichier de configuration de l'autorité de certification
- -extensions : spécifie les extensions nécessaires (v3_ca)
- -days : spécifie la durée de validité du certificat (30 ans)
- -md : spécifie l'algorithme de signature (sha256 With RSA Encryption)
- -keyfile : spécifie le fichier de la clé privée
- -passin stdin : spécifie le mot de passe de la clé privée
- -selfsign : le certificat sera signé par la clé privée ayant signé la demande de certificat
- -out : spécifie le fichier du certificat
- -notext : spécifie l'absence de texte dans le certificat afin d'obtenir un fichier conforme au format PEM
- -infiles : spécifie le fichier de la demande de certificat

Sommaire

Créer une autorité de certification intermédiaire - Serveur

Etape 1 - Créer les dossiers

Dossier pour conserver les certificats émis par l'autorité de certification

```
$ mkdir -p PKI/ServerCA/certs
```

Dossier pour archiver les certificats émis par l'autorité de certification

```
$ mkdir -p PKI/ServerCA/newcerts
```

Dossier pour stocker les demandes de certificats

```
$ mkdir -p PKI/ServerCA/csr
```

Dossier pour stocker les listes de révocation

```
$ mkdir -p PKI/ServerCA/crl
```

Dossier pour stocker la clé privée de l'autorité de certification

```
$ mkdir -m 700 -p PKI/ServerCA/private
```

Créer une autorité de certification intermédiaire - Serveur

Etape 2 - Créer les fichiers

Fichier de la base de données de l'autorité de certification

```
$ touch PKI/ServerCA/index.txt
```

Fichier indiquant le numéro de la CRL

```
$ echo 01 > PKI/ServerCA/crlnumber
```

Fichier indiquant le numéro de série du prochain certificat

```
$ openssl rand 16 -hex > PKI/ServerCA/serial
```

Fichier de configuration de l'autorité de certification

```
$ cp /etc/ssl/openssl.cnf PKI/ServerCA/ServerCA.cnf
```

Créer une autorité de certification intermédiaire - Serveur

Etape 3 - Configurer l'autorité de certification intermédiaire (1/2)

Editer le fichier PKI/ServerCA/ServerCA.cnf

```
[ CA_default ]

dir             = PKI/ServerCA                  # Nom du dossier de l'AC intermédiaire
certs           = $dir/certs
crl_dir         = $dir/crl
database        = $dir/index.txt                # Nom du fichier de la base de données de l'AC intermédiaire
new_certs_dir   = $dir/newcerts

certificate     = $dir/ServerCA.crt             # Nom du fichier du certificat de l'AC intermédiaire
serial          = $dir/serial
crlnumber       = $dir/crlnumber

crl             = $dir/crl.pem
private_key     = $dir/private/ServerCA.key     # Nom du fichier de la clé privée de l'AC intermédiaire
RANDFILE        = $dir/private/.rand

x509_extensions = usr_cert                      # Nom de la section définissant les extensions

name_opt        = ca_default
cert_opt        = ca_default

default_days    = 730                           # Durée de validité des certificats (2 ans)
default_crl_days= 30
default_md      = sha256                        # Fonction de hachage utilisée pour la signature
preserve        = no

policy          = policy_anything               # Nom de la section définissant les règles
```

Créer une autorité de certification intermédiaire - Serveur

Etape 3 - Configurer l'autorité de certification intermédiaire (2/2)

Editer le fichier PKI/ServerCA/ServerCA.cnf

```
[ usr_cert ]
# Cette section définit les extensions à insérer dans les certificats qui seront signés par l'AC intermédiaire


# Définir les contraintes de base des certificats : "CA:FALSE" spécifie que le certificat ne sera pas pour une AC
basicConstraints       = CA:FALSE

# Retirer les extensions NetScape devenues obsolètes
# nsComment            = ""

# Retirer cette extension qui n'est pas nécessaire pour des certificats finaux
#subjectKeyIdentifier  = hash

# Ajouter l'identifiant de lé clé de l'Autorité de Certification intermédiaire
authorityKeyIdentifier = keyid

# Limiter l'usage des certificats émis par l'Autorité de Certification racine (Serveur TLS)
keyUsage               = critical,digitalSignature,keyEncipherment
extendedKeyUsage       = critical,serverAuth

# Spécifie l'adresse du service OCSP
authorityInfoAccess    = OCSP;URI:http://ocsp.testsign.fr/ServerCA/

# Spécifie l'adresse de la CRL
crlDistributionPoints  = URI:http://www.testsign.fr/ServerCA.crl
```

Créer une autorité de certification intermédiaire - Serveur

Etape 4 - Créer la clé privée

```
$ openssl genpkey -out PKI/ServerCA/private/ServerCA.key -pass stdin \
  -aes-256-cbc -algorithm RSA -pkeyopt rsa_keygen_bits:4096
```

Paramètres

- -out : spécifie le fichier de la clé privée
- -pass stdin : spécifie la saisie d'un mot de passe pour chiffrer la clé privée
- -aes-256-cbc : spécifie l'algorithme de chiffrement de la clé privée
- -algorithm : spécifie l'algorithme pour générer la clé privée (RSA)
- -pkeyopt : spécifie les options de la clé privée (4096 bits)

Créer une autorité de certification intermédiaire - Serveur

Etape 5 - Créer la demande de certificat

```
$ openssl req -new -sha256 -key PKI/ServerCA/private/ServerCA.key \
  -out PKI/ServerCA/ServerCA.csr -passin stdin \
  -subj "/C=FR/O=TestSign SA/OU=TestSign - CA/CN=TestSign - Server CA"
```

Paramètres

- -new : spécifie la création d'une nouvelle demande de certificat
- -sha256 : spécifie l'algorithme de signature (sha256 With RSA Encryption)
- -key : spécifie le fichier de la clé privée
- -out : spécifie le fichier de la demande de certificat
- -passin stdin : spécifie le mot de passe de chiffrement de la clé privée
- -subj : spécifie le DN (Distinguished Name) du certificat

Créer une autorité de certification intermédiaire - Serveur

Etape 6 - Envoyer la demande de certificat à l'AC racine

```
$ cp PKI/ServerCA/ServerCA.csr PKI/RootCA/csr/
```

Créer une autorité de certification intermédiaire - Serveur

Etape 7 - Signer le certificat par l'AC racine

```
$ openssl ca -config PKI/RootCA/RootCA.cnf -days 3650 \
  -md sha256 -keyfile PKI/RootCA/private/RootCA.key -passin stdin \
  -out PKI/RootCA/certs/ServerCA.crt -notext \
  -infiles PKI/RootCA/csr/ServerCA.csr
```

Paramètres

- -config : spécifie le fichier de configuration de l'AC racine
- -days : spécifie la durée de validité du certificat (10 ans)
- -md : spécifie l'algorithme de signature (sha256 With RSA Encryption)
- -keyfile : spécifie le fichier de la clé privée de l'AC racine
- -passin stdin : spécifie le mot de passe de chiffrement de la clé privée de l'AC racine
- -out : spécifie le fichier du certificat de l'AC intermédiaire
- -notext : spécifie l'absence de texte dans le certificat afin d'obtenir un fichier conforme au format PEM
- -infiles : spécifie le fichier de la demande de certificat de l'AC intermédiaire

Créer une autorité de certification intermédiaire - Serveur

Etape 8 - Envoyer le certificat à l'AC intermédiaire

```
$ cp PKI/RootCA/certs/ServerCA.crt PKI/ServerCA/ServerCA.crt
```

Créer une autorité de certification intermédiaire - Serveur

Etape 9 - Créer la chaîne de certification

```
$ cat PKI/ServerCA/ServerCA.crt PKI/RootCA/RootCA.crt \
  > PKI/ServerCA/ServerCA_Chain.crt
```

Chaine de certification (Bundle)

La chaine de certification est un fichier contenant, dans l'ordre, le certificat de l'AC intermédiaire puis le certificat de l'AC racine.

Sommaire

Créer une autorité de certification intermédiaire - Client

Etape 1 - Créer les dossiers

Dossier pour conserver les certificats émis par l'autorité de certification

```
$ mkdir -p PKI/ClientCA/certs
```

Dossier pour archiver les certificats émis par l'autorité de certification

```
$ mkdir -p PKI/ClientCA/newcerts
```

Dossier pour stocker les demandes de certificats

```
$ mkdir -p PKI/ClientCA/csr
```

Dossier pour stocker les listes de révocation

```
$ mkdir -p PKI/ClientCA/crl
```

Dossier pour stocker la clé privée de l'autorité de certification

```
$ mkdir -m 700 -p PKI/ClientCA/private
```

Créer une autorité de certification intermédiaire - Client

Etape 2 - Créer les fichiers

Fichier de la base de données de l'autorité de certification

```
$ touch PKI/ClientCA/index.txt
```

Fichier indiquant le numéro de la CRL

```
$ echo 01 > PKI/ClientCA/crlnumber
```

Fichier indiquant le numéro de série du prochain certificat

```
$ openssl rand 16 -hex > PKI/ClientCA/serial
```

Fichier de configuration de l'autorité de certification

```
$ cp /etc/ssl/openssl.cnf PKI/ClientCA/ClientCA.cnf
```

Créer une autorité de certification intermédiaire - Client

Etape 3 - Configurer l'autorité de certification intermédiaire (1/2)

Editer le fichier PKI/ClientCA/ClientCA.cnf

```
[ CA_default ]

dir               = PKI/ClientCA                  # Nom du dossier de l'AC intermédiaire
certs             = $dir/certs
crl_dir           = $dir/crl
database          = $dir/index.txt                # Nom du fichier de la base de données de l'AC intermédiaire
new_certs_dir     = $dir/newcerts

certificate       = $dir/ClientCA.crt             # Nom du fichier du certificat de l'AC intermédiaire
serial            = $dir/serial
crlnumber         = $dir/crlnumber

crl               = $dir/crl.pem
private_key       = $dir/private/ClientCA.key     # Nom du fichier de la clé privée de l'AC intermédiaire
RANDFILE          = $dir/private/.rand

x509_extensions   = usr_cert                      # Nom de la section définissant les extensions

name_opt          = ca_default
cert_opt          = ca_default

default_days      = 730                           # Durée de validité des certificats (2 ans)
default_crl_days= 30
default_md        = sha256                        # Fonction de hachage utilisée pour la signature
preserve          = no

policy            = policy_anything               # Nom de la section définissant les règles
```

Créer une autorité de certification intermédiaire - Client

Etape 3 - Configurer l'autorité de certification intermédiaire (2/2)

Editer le fichier PKI/ClientCA/ClientCA.cnf

```
[ usr_cert ]
# Cette section définit les extensions à insérer dans les certificats qui seront signés par l'AC intermédiaire


# Définir les contraintes de base des certificats : "CA:FALSE" spécifie que le certificat ne sera pas pour une AC
basicConstraints       = CA:FALSE

# Retirer les extensions NetScape devenues obsolètes
# nsComment            = ""

# Retirer cette extension qui n'est pas nécessaire pour des certificats finaux
#subjectKeyIdentifier  = hash

# Ajouter l'identifiant de lé clé de l'Autorité de Certification intermédiaire
authorityKeyIdentifier = keyid

# Limiter l'usage des certificats émis par l'Autorité de Certification racine (Serveur TLS)
keyUsage               = critical,digitalSignature,keyEncipherment
extendedKeyUsage       = critical,clientAuth

# Spécifie l'adresse du service OCSP
authorityInfoAccess    = OCSP;URI:http://ocsp.testsign.fr/ClientCA/

# Spécifie l'adresse de la CRL
crlDistributionPoints  = URI:http://www.testsign.fr/ClientCA.crl
```

Créer une autorité de certification intermédiaire - Client

Etape 4 - Créer la clé privée

```
$ openssl genpkey -out PKI/ClientCA/private/ClientCA.key -pass stdin \
  -aes-256-cbc -algorithm RSA -pkeyopt rsa_keygen_bits:4096
```

Paramètres

- -out : spécifie le fichier de la clé privée
- -pass stdin : spécifie la saisie d'un mot de passe pour chiffrer la clé privée
- -aes-256-cbc : spécifie l'algorithme de chiffrement de la clé privée
- -algorithm : spécifie l'algorithme pour générer la clé privée (RSA)
- -pkeyopt : spécifie les options de la clé privée (4096 bits)

Créer une autorité de certification intermédiaire - Client

Etape 5 - Créer la demande de certificat

```
$ openssl req -new -sha256 -key PKI/ClientCA/private/ClientCA.key \
  -out PKI/ClientCA/ClientCA.csr -passin stdin \
  -subj "/C=FR/O=TestSign SA/OU=TestSign - CA/CN=TestSign - Client CA"
```

Paramètres

- -new : spécifie la création d'une nouvelle demande de certificat
- -sha256 : spécifie l'algorithme de signature (sha256 With RSA Encryption)
- -key : spécifie le fichier de la clé privée
- -out : spécifie le fichier de la demande de certificat
- -passin stdin : spécifie le mot de passe de chiffrement de la clé privée
- -subj : spécifie le DN (Distinguished Name) du certificat

Créer une autorité de certification intermédiaire - Client

Etape 6 - Envoyer la demande de certificat à l'AC racine

```
$ cp PKI/ClientCA/ClientCA.csr PKI/RootCA/csr/
```

Créer une autorité de certification intermédiaire - Client

Etape 7 - Signer le certificat par l'AC racine

```
$ openssl ca -config PKI/RootCA/RootCA.cnf -days 3650 \
  -md sha256 -keyfile PKI/RootCA/private/RootCA.key -passin stdin \
  -out PKI/RootCA/certs/ClientCA.crt -notext \
  -infiles PKI/RootCA/csr/ClientCA.csr
```

Paramètres

- -config : spécifie le fichier de configuration de l'AC racine
- -days : spécifie la durée de validité du certificat (10 ans)
- -md : spécifie l'algorithme de signature (sha256 With RSA Encryption)
- -keyfile : spécifie le fichier de la clé privée de l'AC racine
- -passin stdin : spécifie le mot de passe de chiffrement de la clé privée de l'AC racine
- -out : spécifie le fichier du certificat de l'AC intermédiaire
- -notext : spécifie l'absence de texte dans le certificat afin d'obtenir un fichier conforme au format PEM
- -infiles : spécifie le fichier de la demande de certificat de l'AC intermédiaire

Créer une autorité de certification intermédiaire - Client

Etape 8 - Envoyer le certificat à l'AC intermédiaire

```
$ cp PKI/RootCA/certs/ClientCA.crt PKI/ClientCA/ClientCA.crt
```

Créer une autorité de certification intermédiaire - Client

Etape 9 - Créer la chaîne de certification

```
$ cat PKI/ClientCA/ClientCA.crt PKI/RootCA/RootCA.crt \
  > PKI/ClientCA/ClientCA_Chain.crt
```

Chaine de certification (Bundle)

La chaine de certification est un fichier contenant, dans l'ordre, le certificat de l'AC intermédiaire puis le certificat de l'AC racine.

Sommaire

Afficher un certificat

```
$ openssl x509 -in certificat.crt -text -noout
```

Paramètres

- -in : spécifie le fichier du certificat
- -text : spécifie l'affichage au format text
- -noout : spécifie l'absence d'affichage au format base64

Sommaire

Afficher une demande de certificat

```
$ openssl req -in certificat.csr -text -noout
```

Paramètres

- -in : spécifie le fichier de la demande de certificat
- -text : spécifie l'affichage au format text
- -noout : spécifie l'absence d'affichage au format base64

Sommaire

Créer un certificat serveur

Etape 1 - Créer le dossier

```
$ mkdir -p PKI/CertificatServer
```

En pratique

Le dossier "PKI/CertificatServer" est utilisé uniquement dans le cadre de cet exemple. Dans la pratique, ce dossier sera celui de votre serveur (web, mail, ...).

Créer un certificat serveur

Etape 2 - Créer la clé privée

```
$ openssl genpkey -out PKI/CertificatServer/CertificatServer.key -pass stdin \
  -aes-256-cbc -algorithm RSA -pkeyopt rsa_keygen_bits:2048
```

Paramètres

- -out : spécifie le fichier de la clé privée
- -pass stdin : spécifie la saisie d'un mot de passe pour chiffrer la clé privée
- -aes-256-cbc : spécifie l'algorithme de chiffrement de la clé privée
- -algorithm : spécifie l'algorithme pour générer la clé privée (RSA)
- -pkeyopt : spécifie les options de la clé privée (2048 bits)

Créer un certificat serveur

Etape 3 - Créer la demande de certificat

```
$ openssl req -new -sha256 -key PKI/CertificatServer/CertificatServer.key \
  -out PKI/CertificatServer/CertificatServer.csr -passin stdin \
  -subj "/C=FR/O=Exemple SA/OU=Exemple SA/CN=www.exemple.fr"
```

Paramètres

- -new : spécifie la création d'une nouvelle demande de certificat
- -sha256 : spécifie l'algorithme de signature (sha256 With RSA Encryption)
- -key : spécifie le fichier de la clé privée
- -out : spécifie le fichier de la demande de certificat
- -passin stdin : spécifie la saisie du mot de passe de chiffrement de la clé privée
- -subj : spécifie le DN (Distinguished Name) du certificat

Créer un certificat serveur

Etape 4 - Envoyer la demande de certificat à l'AC TestSign

```
$ cp PKI/CertificatServer/CertificatServer.csr PKI/ServerCA/csr/
```

En pratique

Pour cet exemple, il suffit de copier la demande de certificat dans le dossier de l'autorité de certification intermédiaire.

Dans la pratique, il faudra envoyer la demande par email ou par un site internet. Le fichier de la demande de certificat ne contient aucune donnée sensible. Par conséquent, il peut être envoyé sans protection supplémentaire.

Créer un certificat serveur

Etape 4 - Signer le certificat par l'AC TestSign

```
$ openssl ca -config PKI/ServerCA/ServerCA.cnf -days 730 \
  -md sha256 -keyfile PKI/ServerCA/private/ServerCA.key -passin stdin \
  -out PKI/ServerCA/certs/CertificatServer.crt -notext \
  -infiles PKI/ServerCA/csr/CertificatServer.csr
```

Paramètres

- -config : spécifie le fichier de configuration de l'autorité de certification intermédiaire
- -days : spécifie la durée de validité du certificat (2 ans)
- -md : spécifie l'algorithme de signature (sha256 With RSA Encryption)
- -keyfile : spécifie le fichier de la clé privée de l'AC
- -pass stdin : spécifie la saisie du mot de passe de chiffrement de la clé privée de l'autorité de certification intermédiaire
- -out : spécifie le fichier du certificat du site internet
- -notext : spécifie l'absence de texte dans le certificat afin d'obtenir un fichier conforme au format PEM
- -infiles : spécifie le fichier de la demande de certificat du site internet

Créer un certificat serveur

Etape 5 - Envoyer les fichiers au propriétaire

Envoyer le certificat

```
$ cp PKI/ServerCA/certs/CertificatServer.crt PKI/CertificatServer/
```

Envoyer la chaîne de certification

```
$ cp PKI/ServerCA/ServerCA_Chain.crt PKI/CertificatServer/
```

En pratique
Dans la pratique, le certificat sera mis à disposition du propriétaire via un site internet ou envoyé par email.

Sommaire

Créer un certificat client

Etape 1 - Créer le dossier

```
$ mkdir -p PKI/CertificatClient
```

Créer un certificat client

Etape 2 - Créer la clé privée

```
$ openssl genpkey -out PKI/CertificatClient/CertificatClient.key -pass stdin \
  -aes-256-cbc -algorithm RSA -pkeyopt rsa_keygen_bits:2048
```

Paramètres

- -out : spécifie le fichier de la clé privée
- -pass stdin : spécifie la saisie d'un mot de passe pour chiffrer la clé privée
- -aes-256-cbc : spécifie l'algorithme de chiffrement de la clé privée
- -algorithm : spécifie l'algorithme pour générer la clé privée (RSA)
- -pkeyopt : spécifie les options de la clé privée (2048 bits)

Créer un certificat client

Etape 3 - Créer la demande de certificat

```
$ openssl req -new -sha256 -key PKI/CertificatClient/CertificatClient.key \
  -out PKI/CertificatClient/CertificatClient.csr -passin stdin \
  -subj "/C=FR/O=Exemple SA/OU=MonDépartement/CN=MonNom"
```

Paramètres

- -new : spécifie la création d'une nouvelle demande de certificat
- -sha256 : spécifie l'algorithme de signature (sha256 With RSA Encryption)
- -key : spécifie le fichier de la clé privée
- -out : spécifie le fichier de la demande de certificat
- -passin stdin : spécifie la saisie du mot de passe de chiffrement de la clé privée
- -subj : spécifie le DN (Distinguished Name) du certificat

Créer un certificat client

Etape 4 - Envoyer la demande de certificat à l'AC TestSign

```
$ cp PKI/CertificatClient/CertificatClient.csr PKI/ClientCA/csr/
```

En pratique

Pour cet exemple, il suffit de copier la demande de certificat dans le dossier de l'autorité de certification intermédiaire.

Dans la pratique, il faudra envoyer la demande par email ou par un site internet. Le fichier de la demande de certificat ne contient aucune donnée sensible. Par conséquent, il peut être envoyé sans protection supplémentaire.

Créer un certificat client

Etape 4 - Signer le certificat par l'AC TestSign

```
$ openssl ca -config PKI/ClientCA/ClientCA.cnf -days 730 \
  -md sha256 -keyfile PKI/ClientCA/private/ClientCA.key -passin stdin \
  -out PKI/ClientCA/certs/CertificatClient.crt -notext \
  -infiles PKI/ClientCA/csr/CertificatClient.csr
```

Paramètres

- -config : spécifie le fichier de configuration de l'autorité de certification intermédiaire
- -days : spécifie la durée de validité du certificat (2 ans)
- -md : spécifie l'algorithme de signature (sha256 With RSA Encryption)
- -keyfile : spécifie le fichier de la clé privée de l'AC
- -pass stdin : spécifie la saisie du mot de passe de chiffrement de la clé privée de l'autorité de certification intermédiaire
- -out : spécifie le fichier du certificat client
- -notext : spécifie l'absence de texte dans le certificat afin d'obtenir un fichier conforme au format PEM
- -infiles : spécifie le fichier de la demande de certificat client

Créer un certificat client

Etape 5 - Envoyer les fichiers au propriétaire

Envoyer le certificat

```
$ cp PKI/ClientCA/certs/CertificatClient.crt PKI/CertificatClient/
```

Envoyer la chaîne de certification

```
$ cp PKI/ClientCA/ClientCA_Chain.crt PKI/CertificatClient/
```

En pratique
Dans la pratique, le certificat sera mis à disposition du propriétaire via un site internet ou envoyé par email.

Sommaire

Révoquer un certificat serveur

```
$ openssl ca -config PKI/ServerCA/ServerCA.cnf \
  -revoke PKI/ServerCA/certs/CertificatServer.crt
```

Paramètres

- -config : spécifie le fichier de configuration de l'autorité de certification
- -revoke : spécifie le fichier du certificat à révoquer

Sommaire

Créer la liste des certificats révoqués

```
$ openssl ca -config PKI/ServerCA/ServerCA.cnf -gencrl -crldays 30 \
  -out PKI/ServerCA/crl/ServerCA.crl
```

Paramètres

- -config : spécifie le fichier de configuration de l'autorité de certification
- -gencrl : spécifie la création de la liste des certificats révoqués
- -crldays : sépcifie la durée de validité de la liste des certificats révoqués
- -out : spécifie le fichier de la liste des certificats révoqués

Sommaire

Afficher une liste de certificats révoqués

```
$ openssl crl -text -in PKI/ServerCA/crl/ServerCA.crl -noout \
  -CAfile PKI/ServerCA/ServerCA.crt
```

Paramètres

- -text : spécifie l'affichage au format text
- -in : spécifie le fichier de la liste de certificats révoqués
- -noout : spécifie l'absence d'affichage au format base64
- -CAfile : spécifie le fichier du certificat de l'Autorité de Certification pour vérifier la signature de la CRL

Sommaire

Tester la couche SSL d'un serveur

```
$ openssl s_client -connect www.exemple.fr :443 \
  -no_ssl3 -no_tls1 -no_tls1_1
```

Paramètres

- -connect : spécifie le nom de domaine et le port du serveur
- -no_ssl3 : spécifie de ne pas utiliser le protocole SSL v3
- -no_tls1 : spécifie de ne pas utiliser le protocole TLS v1
- -no_tls1_1 : spécifie de ne pas utiliser le protocole TLS v1.1


fin

Sources

Liens

- Manual page documenting the openssl command line tool
- Utilisation d'Openssl pour les applications SSL/TLS par Franck Davy
- Maitriser les certificats avec openssl sur N30Sec's Blog
- Creating a CA by Phil Dibowitz
- OpenSSL Software Foundation
- Cryptsoft
- RFC 2459 - Public Key Infrastructure