OpenSSL

Présentation :

OpenSSL est une boîte à outils de chiffrement comportant deux bibliothèques (libcrypto et libssl). Il implémente le protocole Transport Layer Security (TLS) et Secure Sockets Layer (SSL)). On va utiliser l'interface en ligne de commande (openssl).

L'objectif de ce TP est la création d'une autorité de certification et la création de certificats SSL. Il sera ensuite déployé sur le serveur Web Apache.

Pré requis :

- ➔ Ordinateur sur Linux (ici, on utilisera la Debian 7.9).
- ➔ Avoir une connexion internet.
- ➔ On utilisera l'utilisateur root et pm
- ➔ L'adresse IP de la machine est 192.168.1.3/24 et s'appelle debian8

Sommaire :

I. Installation des paquets sur la machine Linux.

On met à jour les paquets :

```
root@debian8:/# apt-get update
```

Et on installe un serveur web et openssl :

```
root@debian8:/# apt-get install apache2 openssl
```

II. Configuration principale.

On va créer un répertoire avec l'user pm:

```
pm@debian8:/$ mkdir /home/pm/tpssl
```

Mais aussi d'autre répertoire :

```
pm@debian8:~/tpssl$ mkdir private certs crl newcerts
```

Puis, on va copier le fichier de configuration d'openssl dans ce répertoire :

```
root@debian8:/# cp /etc/ssl/openssl.cnf /home/pm/tpssl/
```

On va modifier cette ligne du fichier :

```
# These are used by the TSA reply generation only.
dir             = /home/pm/tpssl        # TSA root directory
```

Enfin, on crée 2 fichiers et on vérifie nos modifications :

```
pm@debian8:~/tpssl$ touch index.txt
pm@debian8:~/tpssl$ touch serial
pm@debian8:~/tpssl$ ls
certs  crl  index.txt  newcerts  openssl.cnf  private  serial
```

On ajoute cette valeur dans le fichier serial :

```
echo '01' > serial
```

III. Création des certificats.

On tape cette commande :

```
pm@debian8:~/tpssl$ openssl req -new -x509 -extensions v3_ca -keyout private/cak
ey.pem -out cacert.pem -days 3650 -config ./openssl.cnf
Generating a 2048 bit RSA private key
.........+++
...........+++
writing new private key to 'private/cakey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
```

Et remplir ou non les informations demandées :

```
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:14
Locality Name (eg, city) []:Caen
Organization Name (eg, company) [Internet Widgits Pty Ltd]:SIO
Organizational Unit Name (eg, section) []:Service Réseau
Common Name (e.g. server FQDN or YOUR name) []:BTS SIO
Email Address []:
```

On vérifie la présence des deux fichiers :

```
pm@debian8:~/tpssl$ ls
cacert.pem  certs  crl  index.txt  newcerts  openssl.cnf  private  serial
pm@debian8:~/tpssl$ ls private/
cakey.pem
```

On peut procéder à l'extraction du certificat :

```
pm@debian8:~/tpssl$ openssl x509 -text -in cacert.pem
```

IV. Création d'un certificat SSL pour un serveur web

On tape comme commande :

```
pm@debian8:~/tpssl$ openssl req -config ./openssl.cnf -new -keyout private/webke
y.pem -out certs/newreq.pem
Generating a 2048 bit RSA private key
..........+++
...............................+++
writing new private key to 'private/webkey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
```

Et remplir ou non les informations demandées :

```
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:14
Locality Name (eg, city) []:Caen
Organization Name (eg, company) [Internet Widgits Pty Ltd]:SIO
Organizational Unit Name (eg, section) []:Service Réseau
Common Name (e.g. server FQDN or YOUR name) []:BTS SIO
Email Address []:
```

On vérifie la présence des deux fichiers :

```
pm@debian8:~/tpssl$ ls certs/
newreq.pem
pm@debian8:~/tpssl$ ls private/
cakey.pem  webkey.pem
```

On va maintenant signer ce certificat :

```
pm@debian8:~/tpssl$ openssl ca -config ./openssl.cnf -policy policy_anything -ou
t certs/webcert.pem -infiles certs/newreq.pem _
```

Entrer la passphrase et répondre y aux questions, on arrive à ce résultat :

```
Certificate is to be certified until Nov  8 09:41:12 2016 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

On vérifie que la signature du certificat a bien été effectuée :

```
pm@debian8:~/tpssl$ openssl verify -CAfile cacert.pem certs/webcert.pem
certs/webcert.pem: OK
```

V. Installation du certificat SSL

On va générer un nouveau fichier contenant la clé privée non cryptée :

```
pm@debian8:~/tpssl$ openssl rsa -in private/webkey.pem -out private/webkey-clair
.pem _
```

On vérifie la présence :

```
pm@debian8:~/tpssl$ ls private/
cakey.pem  webkey-clair.pem  webkey.pem
```

On active le module ssl :

```
root@debian8:/# a2enmod ssl
```

On va maintenant s'occuper du serveur web :

```
root@debian8:/# nano /etc/apache2/sites-available/default-ssl.conf
```

On modifie le chemin des certificats :

```
SSLCertificateFile /home/pm/tpssl/certs/webcert.pem
SSLCertificateKeyFile /home/pm/tpssl/private/webkey.pem
```

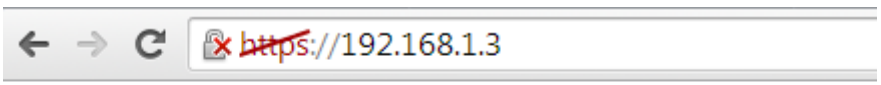
On active l'hôte virtuel :

```
a2ensite default-ssl.conf
```

Puis on redémarre le serveur web :

```
root@debian8:/etc/apache2/sites-available# service apache2 restart
Enter passphrase for SSL/TLS keys for 127.0.1.1:443 (RSA): ******
```

On vérifie à l'aide d'un navigateur et on n'accepte pas ce certificat :

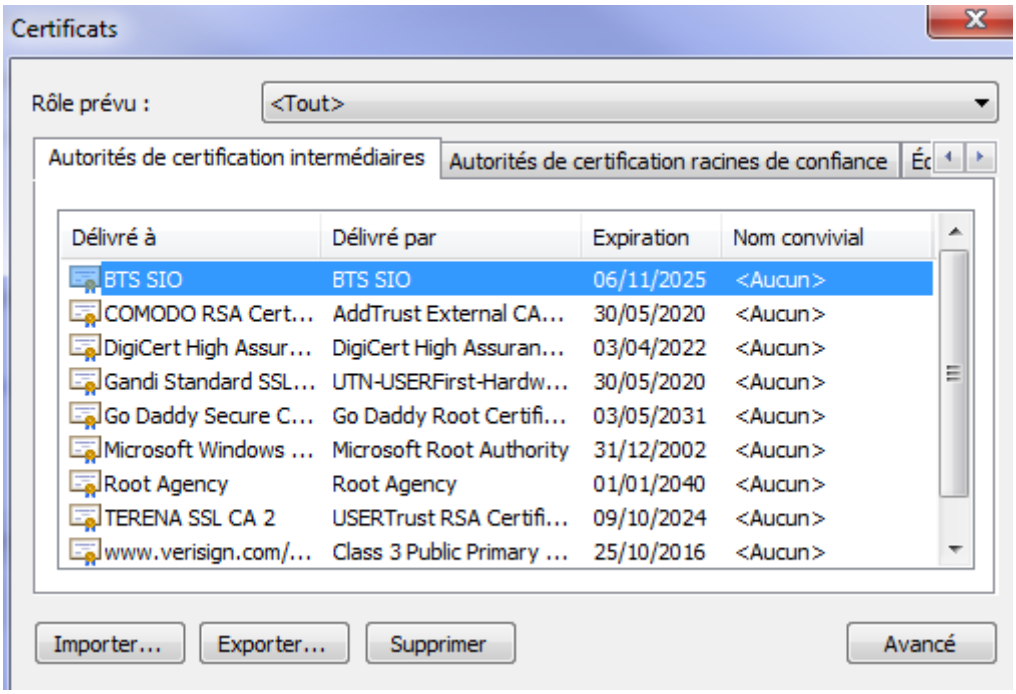


Serveur Web de Pierre-Marie

Pour éviter le message du certificat, on ajoute cette ligne dans /etc/hosts :

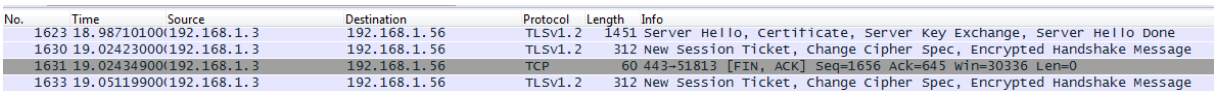
```
  GNU nano 2.2.6               Fichier : /etc/hosts

127.0.0.1       localhost
127.0.1.1       debian8
192.168.1.3     BTS SIO
```

Ensuite on récupère le cacert.pem à l'aide d'un client ftp et on l'importe dans les paramètres du navigateur. On remarque que « BTS SIO » est apparu :



Ainsi, en se connectant au serveur web, on n'a plus de message du certificat.

De plus, avec Wireshark, on observe bien la demande du client (192.168.1.56) au serveur web (192.168.1.3) :



On voit bien la méthode DH et la signature RSA réalisée précédemment :

```
⊟ EC Diffie-Hellman Server Params
    Curve Type: named_curve (0x03)
    Named Curve: secp256r1 (0x0017)
    Pubkey Length: 65
    Pubkey: 0432cb8f8bd9500f776afdf3bdfd3dee2fb71c143a8f5151...
  ⊟ Signature Hash Algorithm: 0x0601
      Signature Hash Algorithm Hash: SHA512 (6)
      Signature Hash Algorithm Signature: RSA (1)
    Signature Length: 256
    Signature: 6193dc3d12c61e6b76623030e2767edbd3e553abef418916...
```