Sécurisation des applications avec SSL

SSL/TLS

Parmi les outils susceptibles de garantir le cryptage, *SSL (Secure Socket Layer : Couche de socket sécurisé)*, défini par la société Netscape en 1994 (V.1) et déployé publiquement en version 2 (1995) s'est imposé comme la technique de sécurisation des échanges en environnement internet.

Il s'applique à tous les protocoles de la couche *application* (couche 7) puisqu'il se positionne plus bas dans le modèle *OSI* (couche 5 : Session).

Il supporte les *certificats X.509*.

TLS (Transport Layer Security) est le nom du protocole depuis 2001, qui remplace la version 3.1 de SSL. Il est maintenant défini par l'*IETF (Internet Engineering Task Force)* qui se charge de normaliser les techniques utilisées sur Internet.

La mise en place d'un service sécurisé par SSL/TLS permettra d'éviter que l'on puisse exploiter le contenu en cas d'interception des échanges entre le client et le serveur.

Procédure

La démarche est commune quel que soit le service que l'on souhaite sécuriser. Seuls les fichiers à modifier changent d'un service à l'autre.

1. Créer le répertoire pour stocker les clés et certificats
2. Créer la clé privée
3. Créer le certificat X509 à partir de la clé privée en renseignant toutes les informations demandées
4. Paramétrer le service concerné
    1. activer le module,
    2. configurer les fichiers pour qu'ils utilisent la clé et le certificat
    3. mettre en écoute sur les ports spécifiques
    4. redémarrer le service pour prendre en compte les modifications
5. Tester depuis un client en lui indiquant l'adresse et le port adéquats
6. Éventuellement, accepter le certificat auto-signé (**attention, cela peut être un danger si on ne connaît pas la source**).

Mode opératoire

Mise en place d'un certificat avec OpenSSL

OpenSSL est une transposition *Open Source* (http://www.openssl.org/) des préconisations de l'IETF pour la mise en place d'une couche sécurisée. Le nom est resté *OpenSSL* mais la bibliothèque supporte aussi la version TLS. *OpenSSL* gère de nombreux **algorithmes de cryptage** (AES, DES, RSA,...) et **algorithmes de hachage** (SHA, MD5, ...).

Installer la bibliothèque OpenSSL

```
apt install openssl
```

Cet outil propose notamment des commandes pour la création de clé secrète (*gendh*, *genrsa*, ...) ou la génération de certificat (*x509*).

1 : Création d'une clé privée

La base des techniques de cryptage asymétrique repose sur la présence, à un endroit unique, d'une **clé privée**. Elle permettra de créer une **clé publique** (dans un **certificat**) et de décrypter ce qui aura été chiffré par cette **clé publique**.

On utilisera (pour l'algorithme RSA) la commande ***genrsa*** de *OpenSSL*.

```
openssl genrsa -out <nom_fichier_cle> <taille_cle>
```

Exemple : crée une clé privée dans un dossier personnalisé

```
cd /etc/ssl
mkdir mesCles
cd mesCles
openssl genrsa -out cleGSB.key 2048
```

On pourra alors éditer la clé et en lire le contenu (évidemment inaccessible). Cette même clé peut servir à établir des certificats différents pour des usages distincts (serveur FTP, serveur Web, messagerie, etc).

2 : Création d'un certificat X509

La clé privée ne doit jamais être diffusée. On va donc générer la partie publique sous forme d'un certificat. Pour que ce dernier soit accessible aux navigateurs, on a recours au format standard X509.

La création va donc passer par deux étapes :

- Création d'un certificat générique (on devra renseigner les coordonnées de l'entreprise)

```
openssl req -new -key <nom_fichier_cle> -out <nom_certif_generic.csr>
```

- Transformation vers un format X509

```
openssl x509 -req -days <nb_jours> -in <nom_certif_generic.csr> -signkey
<nom_fichier_cle> -out <nom_certif_X509.crt>
```

Exemple : création d'un certificat dans un dossier spécifique à partir de la clé RSA

```
cd /etc/ssl
mkdir mesCertifs
cd mesCertifs
openssl req -new -key
../mesCles/cleGSB.key -out
GSBCertGen.csr
openssl x509 -req -days 365 -in
GSBCertGen.csr -signkey
../mesCles/cleGSB.key -out
GSBcertif.crt
```

Penser à adapter les valeurs à votre environnement

Le certificat est alors prêt.

Il reste à configurer les services susceptibles de s'appuyer sur ce certificat.

Configuration d'Apache avec SSL/TLS

La configuration nécessite l'activation du module SSL pour Apache

```
a2enmod ssl
```

remarque : Si le module est déjà activé, un message l'indique.

On peut alors voir dans */etc/apache2/mods-enabled/ssl.load* que la ligne ci-dessous est dé-commentée

```
LoadModule ssl_module /usr/lib/apache2/modules/mod_ssl.so
```

Dans le fichier *ports.conf* de Apache, on vérifiera que le serveur écoute sur le port standard 443 (ou sur un autre port si on souhaite faire une configuration personnalisée).

Se placer dans le dossier */etc/apache2/sites-enabled* et ensuite éditer le fichier *000-default.conf* (ou dans */etc/apache2/httpd.conf*).

```
cd /etc/apache2/sites-enabled
nano 000-default.conf
```

On **ajoutera** un hôte virtuel pour cette écoute :

```
#on adaptera le numéro de port conformément à ce qui a été écrit dans
ports.conf
<VirtualHost *:443>
    DocumentRoot /var/www/html
    SSLEngine on # active le SSL
    SSLCertificateFile /etc/ssl/mesCertifs/GSBCertif.crt     # chemin du
certificat X509
    SSLCertificateKeyFile /etc/ssl/mesCles/cleGSB.key     # chemin de la clé
privée
</VirtualHost>
```

Il faudra redémarrer Apache, qui indiquera si une erreur éventuelle est rencontrée (dans le chemin, dans le nom du fichier, dans le contenu du certificat, etc).

Contrôle depuis un navigateur

Dans la barre de navigation du navigateur, on tapera https://adresseServeur.

Du fait que le certificat que nous avons créé n'est pas garanti par un organisme officiel, le navigateur met en garde sur le manque de confiance accordée à un certificat signé par son créateur. On doit vérifier qu'un cadenas fermé au bas du navigateur atteste d'une navigation sécurisée.

Configuration de ProFTP avec SSL/TLS

Pour commencer, on devra indiquer dans le fichier */etc/proftpd/modules.conf* qu'il faut activer TLS

```
LoadModule mod_tls.c
```

On ira ensuite préciser au fichier *proftpd.conf* d'inclure le fichier de configuration de TLS en dé-commentant la ligne suivante :

```
Include /etc/proftpd/tls.conf
```

Dans ce fichier *tls.conf* on devra trouver au minimum les éléments :

```
<IfModule mod_tls.c>
    TLSEngine on #active le TLS
    TLSLog /var/log/proftpd/tls.log # dossier pour enregistrer les journaux
tls
    TLSProtocol SSLv23 # versions supportées (2 et 3)
    TLSRSACertificateFile /etc/ssl/mesCertifs/GSBCertif.crt #chemin du
certif
    TLSRSACertificateKeyFile /etc/ssl/mesCles/cleGSB.key # chemin de la clé
    TLSVerifyClient off # n'oblige pas l'authentification des clients pour
TLS
    #TLSRequired on # peut obliger les clients à utiliser TLS
</IfModule>
```

Il faudra bien entendu redémarrer le service *proftpd* qui indiquera si une erreur est rencontrée (dans le chemin, dans le nom du fichier, dans le contenu du certificat, etc).

Test depuis le client

On accèdera depuis un client FTP en contactant le serveur par une connexion *FTP SSL/TLS Explicite* (ici sous FileZilla Client).

On rencontrera une mise en garde indiquant que le certificat n'étant pas garanti par un tiers (il est auto-signé).

Sources

- http://httpd.apache.org/docs/trunk/fr/ssl/ssl_intro.html : sur le site de Apache, une explication détaillée des principes du chiffrement, du rôle d'un certificat, des déclinaisons de SSL/TLS