NFA083 – Réseau et Administration Web
OpenSSL

Partie 1: Génération de Certificats

Afin d'activer le support SSL/TLS sur un serveur web, il faut lui fournir un certificat. Ce certificat lui permettra de s'authentifier auprès de ses clients. Afin que les clients puissent valider le certificat, celui-ci doit être signé par une autorité de certification jugé de confiance par le client.

1) Génération des clés du Serveur

La première étape dans la génération d'un certificat est de générer les clés associées (la taille de clé précisée pour RSA doit être au moins 2048) :

```
$ openssl genrsa 2048 > serveur.key
```

Ce fichier correspond aux clés *privée et publique* de votre serveur. On génère une clé sans mot de passe pour éviter de devoir fournir le mots de passe lors du démarrage du serveur.

Remarque. Ces clés sont importantes lors de la procédure d'authentification du serveur. Si une personne prend connaissance de la clé privée, elle peut s'en servir pour usurper l'identité de votre serveur. Il faut donc protéger cette clé en lecture afin d'en interdire l'accès :

```
$ chmod go-r serveur.key
```

2) Génération de la Demande de Signature de Certificat

Une fois la clé générée, nous pouvons générer une demande de signature de certificat (*CSR: Certificate Signing Request*).

```
$ openssl req -new -key serveur.key > serveur.csr
```

Cette demande sera transmise à une autorité de certification pour signature qui fournira en retour un certificat signé.

Remarque. Ces deux premières étapes peuvent être réalisée directement via webmin, dans « Webmin Configuration », choisir « SSL Encryption » puis l'onglet « Certificate Signing Request ».

Partie 2: Autorité de Certification (CA)

Afin de faire signer votre certificat, vous pouvez faire appelle à une autorité de certification (CA) ou bien créer votre propre autorité de certification.

1) Création d'une Autorité de Certification

Afin de créer une autorité de certification, nous devons générer une clé et un certificat permettant d'authentifier cette CA. Ce certificat sera auto-signé.

1. Création des clés privée et publique de l'autorité de certification protégé par mot de passe :

   ```
   $ openssl genrsa -des3 1024 > ca.key
   ```

Remarque. L'option `-des3` permet d'activer la protection par mot de passe de la clé de votre CA. Il est **très** important de garder cette clé confidentielle sous peine de compromettre votre CA.

2. Création d'un certificat x509 pour une durée de validité d'un an auto-signé:

   ```
   $ openssl req -new -x509 -days 365 -key ca.key > ca.crt
   ```

   Remarque. Le mot de passe demandé est celui de la clé privée du CA.

2) Publication du Certificat de la CA

Pour que les certificats issues par cette CA soient validés automatiquement par votre navigateur, celui-ci doit connaître la CA. Pour cela, on va importer dans le navigateur le certificat de notre nouvelle CA. Pour importer le certificat de notre CA dans Firefox, cliquez sur:

- Menu : `Edit/Preferences/Advanced`
- Onglet : `Certificates`
- Bouton : `View Certificates`
- Onglet : `Authorities`
- Bouton : `Import`

Une autre façon de publier le certificat est de le placer dans le `DocumentRoot` du serveur `default` puis de le récupérer directement depuis un navigateur web.

3) Signature de Certificats

Pour signer un certificat d'un serveur, il suffit d'utiliser la commande suivante:

```
$ openssl x509 -req -in serveur.csr -out serveur.crt -CA ca.crt -CAkey ca.key \
-CAcreateserial -CAserial ca.srl
```

Remarque. Lors de la signature du premier certificat, il faut rajouter l'option `-CAcreateserial` pour créer le fichier `ca.srl` (le fichier `ca.srl` n'existe pas avant la première signature).

Le fichier `serveur.crt` créé correspond au certificat de votre serveur.

Partie 3: Configuration du Serveur Apache

1) Activation du Module ssl

Activez le module avec la commande `a2enmod` et redémarrez le serveur apache:

```
$ sudo a2enmod ssl
$ sudo systemctl restart apache2
```

2) Configuration

1. Ouvrez le fichier `/etc/apache2/sites-available/default-ssl.conf`
2. Identifiez les différences avec le fichier `/etc/apache2/sites-available/000-default.conf`

3. Modifiez le fichier en conséquence pour indiquer l'emplacement de la clé privée et du certificat du serveur. Vous pouvez aussi utiliser webmin pour cela (dans « Virtual Server Options », choisir « SSL Options ».

4. Limiter TLS à la version 1.3 pour ce serveur virtuel, et comparer la configuration obtenue à celle recommandée par l'outil SSL Configuration Generator :

   https://ssl-config.mozilla.org

5. Activer le site `default-ssl` et redémarrez le serveur.

6. Tester votre hôte virtuel sécurisé en visitant le site :

   `https://localhost:2443`

Partie 4: Authentification Apache

1. Utiliser webmin pour activer l'authentification apache pour hôte virtuel sécurisé. L'activavion se fait pour un repertoire donné (« Per-Directory Options »). Dans l'onglet « Access Control », on choisira le mode « Basic » avec un fichier utilisateurs et mots de passe textuel (par exemple `/var/www/htpasswd`).

2. Sauvegarder et revenir sur dans l'onglet « Access Control » pour cliquer sur « Edit users », et ajouter de nouveaux utilisateurs.

3. Relancer Apache, et tester l'accès au site. Vérifier avec Firefox le contenu du champ « Authorization » de la requète `get` (le décoder avec la commande en ligne `base64`).

4. Créer une page php avec le contenu suivant :

```
<?php
if (!isset($_SERVER['PHP_AUTH_USER'])) {
    header('WWW-Authenticate: Basic realm="Restricted"');
    header('HTTP/1.0 401 Unauthorized');
    echo 'Authentification cancelled';
    exit;
} else {
    echo "<p>Hello {$_SERVER['PHP_AUTH_USER']}.</p>";
    echo "<p>Your password is {$_SERVER['PHP_AUTH_PW']}.</p>";
}
?>
```

   Tester cette page.

5. Modifier la page `sakila.php` du TP précédent pour authentifier les utilisateurs de la base de données en utilisant l'authentification Apache.