TP N°2 Réseaux 2A - Mise en œuvre du protocole HTTPS
Module TR-C8

Les accès aux pages Web se font généralement à l'aide du protocole http, en empruntant le réseau Internet. Aucune garantie de confidentialité n'est assurée lors de ces accès. L'interception de vos requêtes et de la réponse faite par le serveur est relativement simple pour un pirate.

Afin de palier à ces inconvénients, le protocole HTTPS avec l'authentification SSL peut être mis en oeuvre. Grâce aux modèles de cryptographie qu'utilise SSL (cryptographie à clé publique), il est quasiment impossible à un pirate qui intercepterait des accès à des pages chargées via le protocole HTTPS de décrypter cet échange, et donc de récupérer des informations confidentielles.

Par ailleurs, l'utilisation des certificats permet de garantir une sécurité encore plus forte, par la signature des clés publiques via une entité de confiance, appelée Autorité de Certification (CA). Ces clés publiques signées deviennent alors des certificats. En peut utiliser les certificats pour chiffrer des communications dans des sessions SSL, ou encore authentifier des utilisateurs, signer et chiffrer des fichiers.

Les mécanismes de la norme X.509, qui permet de créer des certificats sont assez complexes, et la gestion des certificats est déléguée à une PKI (Public Key Infrastructure). Il s'agit d'un ensemble de règles nécessaire à une AC pour créer, gérer et distribuer des certificats.

Une PKI permet d'émettre des certificats, les révoquer (les déclarer comme nuls en cas de perte ou compromission de la clé privée) ; pour ces certificats elle émet des listes de révocation (Certificate Revocation List ou CRL) pour ses AC.

Afin de vérifier l'authenticité des certificats émis, Une PKI met à disposition la clé publique de son AC racine, sous forme d'un certificat **auto-signé** (la clé de voûte du système), les certificats des utilisateurs dans un annuaire, et les CRL des certificats révoqués.

Objectifs du TP

Dans ce TP, on se propose de réaliser :

1. Une autorité de certification (CA)
2. Une certification pour un serveur Apache
3. Un certificat pour un utilisateur
4. Une CRL

Pour cela on va créer l'architecture nécessaire pour générer les certificats du domaine *iutcaen.fr* en utilisant l'outil OpenSSL. C'est un outil de cryptographie très complet. Il permet de créer et gérer des bi-clés, des AC, des CRL. Il supporte également de nombreux algorithmes de chiffrement. Il est fourni sous la forme d'une bibliothèque pour être plus facile d'utilisation par le plus grand nombre d'applications.

1. Test du serveur Apache

Dans un premier temps, vous allez tester votre serveur Apache en se connectant via SSL en utilisant les certificats déjà installés. Pour cela,

- Il faut d'abord lancer votre serveur Apache (**# /etc/init.d/httpd start)**
- Se connecter à l'URL https://localhost.
- Accepter le certificat proposé.

Il est recommander d'examiner le contenu du certificat avant de l'accepter. Quelles informations contient-t-il ? Lancer Wireshark et analyser la capture de trame.

2. Mise en place d'une AC

La première chose à faire est de créer une arborescence pour les fichiers de l'autorité de certification, dans le répertoire */etc/pki/tls/* :

```
iutcaen.fr/
|-- certs/
|-- crl/
|-- index.txt
|-- newcerts/
|-- private/
|-- serial
|-- crlnumber
```

 # mkdir -p /etc/pki/tls/iutcaen.fr/{certs,crl,newcerts,private}
 # touch /etc/pki/tls/iutcaen.fr/index.txt
 # echo "01" > /etc/pki/tls/iutcaen.fr/serial

Ensuite il faut configurer OpenSSL pour qu'il reconnaisse cette arborescence comme celle d'une AC. Ce fichier de configuration est /etc/pki/tls/openssl.cnf.

Attention: Il faut d'abord sauvegarder la configuration par défaut,
`cp /etc/pki/tls/openssl.cnf. /etc/pki/tls/openssl.cnf.old`

Une AC est définie dans une section, le nom de l'AC est défini par le nom de cette section. La section particulière [ca] permet de définir l'AC par défault.

Le fichier de configuration contient un exemple ([CA_default]) qu'on va adapter à notre contexte. Pour cela, il faut modifier les paramètres suivants:
[ca]

```
default_ca      = iutcaen_fr
[ iutcaen_fr ]
dir             = /etc/pki/tls/iutcaen.fr
crlnumber       = $dir/crlnumber
```

La section [req_distinguished_name] peut également être modifiée pour configurer les paramètres par défaut pour la création de certificats :

```
[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = FR
countryName_min                 = 2
countryName_max                 = 2

stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = Calvados

localityName                    = Locality Name (eg, city)
localityName_default            = Caen

0.organizationName              = Organization Name (eg, company)
0.organizationName_default      = IUTCAEN

organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default  =
```

La seconde étape est de créer une paire de clés (public, privée) auto-signé pour l'AC :

```
# cd /etc/pki/tls/iutcaen.fr/
# openssl req -x509 -newkey rsa:1024 -days 3650 -keyout private/cakey.pem -out
cacert.pem
Generating a 1024 bit RSA private key
.......++++++
.....................++++++
writing new private key to 'private/cakey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
Country Name (2 letter code) [FR]:
State or Province Name (full name) [Calvados]:
Locality Name (eg, city) [Caen]:
Organization Name (eg, company) [IUTCAEN]:
Organizational Unit Name (eg, section) []:.
Common Name (eg, YOUR name) []: AC IUTCaen
Email Address []:ac@iutcaen.fr
```

Quelques explications sur les options passées à openssl:

req	On utilise la commande **req** qui permet de créer des demandes de certificats.
-x509	Permet de créer un certificat auto-signé de cette AC dite "racine"
-newkey rsa:1024	Permet de créer la clé privée en même temps que le certificat, on utilise l'algorithme de chiffrement RSA et une longueur de clé de 1024 bit
-days 3650	On fixe la validité de la signature à 3650 jours, soit environ 10 ans.
-keyout private/cakey.pem	Chemin vers le fichier où stocker la clé privée, en accord avec le fichier de configuration de openssl.cnf
-out cacert.pem	Chemin vers le fichier où stocker le certificat de l'AC, en accord avec le fichier de configuration de openssl.cnf

Pour assurer la sécurité de la clé privée de l'AC, on restreint les permissions sur le répertoire *private*,

```
# chmod -R 600 /etc/pki/tls/iutcaen.fr/private
```

L'AC est maintenant configurée, on peut commencer à signer des demandes de certificat.

2.1. Création d'un certificat pour le serveur Apache

Pour pourvoir utiliser des certificats avec Apache et donc accéder au serveur en utilisant HTTPS, on utilisera **`mod_ssl`**.

On va créer un certificat serveur pour le site **secure.iutcaen.fr**, pour cela il faut d'abord créer une demande de certificat et la clé privée pour le serveur :

```
# openssl req -newkey rsa:1024 -keyout secure.iutcaen.fr.key -out secure.iutcaen.fr.req
Generating a 1024 bit RSA private key
.............++++++
writing new private key to 'secure.iutcaen.fr.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
Country Name (2 letter code) [FR]:
```

State or Province Name (full name) [Calvados]:
Locality Name (eg, city) [Caen]:
Organization Name (eg, company) [IUTCAEN]:
Organizational Unit Name (eg, section) []:.
Common Name (eg, YOUR name) []:secure.iutcaen.fr
Email Address []:www@secure.iutcaen.fr

Attention :
Le Common Name doit **impérativement** être le nom DNS complet du serveur. En effet, les navigateurs se basent sur cette information pour vérifier le certificat du serveur.

La clé privée sera enregistrée dans le fichier **secure.iutcaen.fr.key** et la clé publique sous la forme d'une demande dans le fichier **secure.iutcaen.fr.req**

Pour signer la demande de certificat :

 # openssl ca -in secure.iutcaen.fr.req -out secure.iutcaen.fr.pem

2.2. Paramétrage du serveur Apache

Dans un premier temps, vérifier la présence de fichier de configuration de SSL (ssl.conf) dans le répertoire */etc/httpd/conf.d :*

Attention: Il faut d'abord sauvegarder la configuration par défaut,
cp /etc/httpd/conf.d/ssl.conf. /etc/httpd/conf.d/ssl.conf.old

- Vérifier que le serveur écoute bien sur le port 443 (commande **Listen**),
- Placer les fichiers générés dans les répertoires adéquats du serveur Apache

```
# Chemin vers le certificat du serveur
SSLCertificateFile /etc/pki/tls/certs/secure.iutcaen.fr.pem

# Chemin vers la clé privée associée au certificat
SSLCertificateKeyFile /etc/pki/tls/private/secure.iutcaen.fr.key
```

 # mv /etc/pki/tls/secure.iutcaen.fr.key /etc/pki/tls/private
 # mv /etc/pki/tls/secure.iutcaen.fr.pem /etc/pki/tls/certs

Dans un second temps, configurer Apache pour utiliser le certificat serveur. Pour cela, on crée un hôte virtuel (Virtual Host), dont la configuration est enregistrée dans */etc/httpd/conf.d/ssl.conf:*

```
NameVirtualHost a.b.c.d:443

<VirtualHost a.b.c.d:443>
  ServerAdmin www@secure.iutcaen.fr  # adresse mail de l'administrateur du site
  DocumentRoot /var/www/html         # répertoire des fichiers du site
  ServerName secure.iutcaen.fr       # Nom de domaine complet du site

# Configuration mod_ssl
  SSLEngine on
  SSLProtocol All             # On utilise SSLv2, SSLv3 et TLSv1
  SSLCipherSuite MEDIUM:HIGH  # Autorise seulement les algorithmes de cryptographie

  # Chemin vers le certificat du serveur
  SSLCertificateFile /etc/pki/tls/certs/secure.iutcaen.fr.pem

  # Chemin vers la clé privée associée au certificat
  SSLCertificateKeyFile "/etc/pki/tls/private/secure.iutcaen.fr.key"

</VirtualHost>
```

- *a.b.c.d* est l'adresse IP qui correspond au nom de domaine du site, **secure.iutcaen.fr.**
- *443* est le port d'écoute en mode SSL/TLS
- *NameVirtualHost* est une directive permettant d'indiquer l'adresse IP sur lequel le serveur recevra toutes les requêtes en provenance des hôtes virtuels par nom. Autrement dit, une seule adresse IP peut servir à héberger plusieurs sites d'où la notion de hôte virtuel par nom.

Pour finir, on crée une page html pour le site sécurisé :

 # echo "bienvenue sur secure.iutcaen.fr" > /var/www/html/index.html

Il faut maintenant relancer Apache pour que les modifications soient effectives :

 # /etc/init.d/httpd start
Starting web server: apache Enter pass phrase:

Il demande le mot de passe pour déverrouiller la clé privée.

On peut maintenant tester en allant à l'url `https://secure.iutcaen.fr`, on est accueilli par cette erreur : ***« impossible de vérifier l'identité de secure.iutcaen.fr comme site de confiance »***

Remarque importante: Le serveur DNS n'est pas configuré par conséquent la résolution du nom de domaine est impossible. Une solution serait de renseigner l'adresse associé au domaine dans le fichier **/etc/hosts**.

Le navigateur ne fait pas confiance au certificat du serveur car il ne possède pas le certificat de l'AC IUT CAEN. Il faut donc installer le certificat de l'AC dans le navigateur :

- Dans FireFox :
  - "Édition" > "Préférences" > "Avancé" > "Chiffrements" > "Afficher les certificats..."
  - cliquer sur l'onglet "Autorités", puis sur le bouton "Importer"
  - choisissez le fichier */etc/pki/tls/iutcaen.fr/cacert.pem* et valider.

Votre navigateur est à présent configuré avec le certificat de l'autorité de certification.

Quand on recommence, on a accès au site sans cet avertissement.

3. Utilisation des certificats clients

OpenSSL ne sert pas seulement à sécuriser les connexions vers certains serveurs, on peut aussi l'utiliser pour authentifier des utilisateurs auprès de ces serveurs.

Vous allez donc modifier la configuration du serveur Apache pour mettre en place une authentification cliente. Pour cela, il faut ajouter la directive ci-dessous à la configuration de l'hôte virtuel précédemment configuré :

```
NameVirtualHost a.b.c.d:443
<VirtualHost a.b.c.d:443>
…
# Demander l'authentification des visiteurs
  SSLVerifyClient require
</VirtualHost>
```

Ensuite, il faut générer un couple de clés pour tester l'authentification cliente :

$ openssl req -newkey rsa:1024 -keyout client.key -out client.req

```
Country Name (2 letter code) [FR]:
State or Province Name (full name) [Calvados]:
Locality Name (eg, city) [Caen]:
Organization Name (eg, company) [IUTCAEN]:
Organizational Unit Name (eg, section) []:.
Common Name (eg, YOUR name) []:Loudni Samir
Email Address []:loudni@iutcaen.fr
```

Ensuite, il faut que l'AC signe la demande de certificat :

 # openssl ca -in client.req -out client.pem

Pour pouvoir importer la bi-clé de l'utilisateur dans un navigateur web, il faut que celui soit au format *PKCS#12*, il faut donc créer un tel fichier grâce à la commande :

 # openssl pkcs12 -export -in client.pem -inkey client.key -out loudni.p12 -name "Loudni Samir"

```
Enter pass phrase for client.key:
Enter Export Password:
Verifying - Enter Export Password:
```

Il faut entrer le mot de passe de la clé privée, puis un mot de passe qui sera demandé pour accéder aux informations du fichier PKCS#12. Par exemple, ce mot de passe sera demandé à l'utilisateur lors de l'importation dans le navigateur web.

Quelques détails sur cette commande :

pkcs12	Il s'agit de la commande `openssl` pour manipuler le format PKCS#12
-export	Permet de créer un nouveau fichier PKCS#12
-in client.pem	Spécifie le chemin du certificat
-inkey client.key	Spécifie le chemin de la clé privée correspondante
-out loudni.p12	Enregistre le résultat dans le fichier `loudni.p12`
-name	Ceci sera le nom affiché dans la boîte de sélection du navigateur

Enfin, on peut importer le fichier `.p12` dans le navigateur :

- Dans FireFox :
  - "Edition" > "Préférences" > > "Avancé" > "Chiffrements" > "Afficher les certificats..."
  - cliquer sur l'onglet "Autorités", puis sur le bouton "Importer"
  - choisissez le fichier *loudni.p12* et valider
  - Firefox demande alors de configurer un mot de passe qu'il faudra entrer pour utiliser les certificats installés. Il faut entrer le mot de passe du PKCS#12, pour terminer.
  - choisir "Demander à chaque fois" dans "Sélection des certificats clients".

On peut maintenant tester en allant sur le site sécurisé (https://secure.iutcaen.fr), le navigateur demande alors de choisir un certificat.

Firefox peut renvoyer un code d'erreur -12227. Ce code signifie que l'authentification à échoué. Cette erreur apparaît systématiquement lorsqu'aucun certificat client n'est pas installé dans le navigateur.

4. Gestion des listes de révocation (CRL)

Une CRL est une liste des numéros de série des certificats qui sont déclarés comme non valides. Une CRL est créée et signée par l'Autorité de Certification.

Pour révoquer un certificat, il est utile de connaître son numéro de série, car les certificats émis par l'AC sont stockés dans le répertoire */etc/pki/tls/iutcaen.fr/newcerts* sous la forme `<numéro>.pem`. La correspondance entre les informations nominatives du certificat et le numéro de série est possible grâce à la base de donnée sauvée dans le fichier */etc/pki/tls/iutcaen.fr/index.txt*.

Lorsqu'un utilisateur perd la clé privée de son certificat où qu'il se fait voler son fichier PKCS#12, le certificat doit être révoqué :

 # openssl ca -revoke /etc/pki/tls/iutcaen.fr/newcerts/"number".pem

1. "number" correspond au numéro de série associé au certificat révoqué
2. Un numéro de révocation est renvoyé
3. Mettre ce numéro dans /etc/pki/tls/iutcaen.fr/crlnumber

Il faut ensuite générer la CRL qui va permettre de signifier cette révocation au serveur Apache :

 # openssl ca -gencrl -out /etc/pki/tls/iutcaen.fr/crl.pem

Le serveur Apache doit connaitre le chemin vers le fichier de la CRL, c'est pourquoi on ajoute la directive `SSLCARevocationFile` dans le fichier de configuration de l'hôte virtuel :

```
...
 # Chemin vers la CRL
  SSLCARevocationFile /etc/pki/tls/iutcaen.fr/crl.pem
...
```

Après avoir relancé le serveur, L'utilisateur Loudni Samir obtient un message d'erreur lorsqu'il veut accéder à `https://secure.iutcaen.fr` :

5. Quelques liens

- Site officiel d'OpenSSL : http://www.openssl.org/
- Un document sur la création de PKI : http://ospkibook.sourceforge.net/
- Source du document (Nicolas Thauvin): http://www.andesi.org.