TP: Sécurisation d'un serveur WEB (HTTPS)

N.B: ***Ce Tp a été testé sous ubuntu 14.04 LTS***

I. Outils utilisés:

- Machine ubuntu 14.04 LTS qui va jouer le rôle du client et du serveur.
- Installerz le serveur web apache2, le serveur d'application php, l'analyseur de trafic wireshark et l'outil de chiffrement openssl. **# sudo apt-get install apache2 php ssl wireshark**

II. travail à faire

1) Mise en place d'une application Client/Serveur WEB non sécurisée

a. Lancer le serveur web apache2: **# sudo /etc/init.d/apache start** ou **sudo service apache2 start**
b. Ajouter dans le fichier /etc/hosts du client une ligne qui va faire correspondre l'adresse ip de votre serveur (on travailler en local: 127.0.0.1) et le nom www.<votre nom>.com
c. Accéder au site par défaut du serveur à travers le navigateur (client web): **http://www.<votre nom>.com**
d. Dans /var/www/, créer le répertoire monsite (à travers la commande mkdir).
Dans ce répertoire, créer deux fichiers accueil.html et bienvenue.php.

==> accueil.html

```
    <html>
    <head>
    <title>
    Login page
    </title>
    </head>
    <body>
    <h1 style="font-family:Comic Sans Ms;text-align="center";font-
size:20pt;color:#00FF00;>
    Simple Login Page
    </h1>
    <form name="login" method="post" action="bienvenue.php">
    Username<input type="text" name="userid"/>
    Password<input type="password" name="passwd"/>
 <input type="submit" name="Submit" value="Login">
    <input type="reset" value="Cancel"/>
    </form>
    </body>
    </html>
```

==> bienvenue.php.

```
<html>
      <head>
      <title>
      Bienvevue
      </title>
      </head>
      <body>
      <?php
      $myusername=$_POST['userid'];
```

```
      $mypassword=$_POST['passwd'];
      if($myusername=='myuserid'&&$mypassword=='mypasswd'){
      echo "bienvenue";
      }
      else {
      echo "Wrong Username or Password";
      }
      ?>
      </body>
</html>
```

Ajouter la configuration suivante dans /etc/apache2/sites-available/000-default.conf pour que le site **monsite** soit accessible puis redémarrer le serveur apache2 (pour considérer la nouvelle configuration)

```
ServerName <votre nom>.com
ServerAlias www.<votre nom>.com
DocumentRoot /var/www/monsite
```

8. Accéder au site accueil.html. Analyser le trafic Wireshark en distinguant :
- La Phase de connexion (Three way handshake)
- La requête GET de la page accueil.html
- Les accusés de réception TCP/Ack.
- La requête POST dans laquelle le login et le mot de passe sont transférés.

2) HTTPS (HTTP Secure)

L'objectif de cette activité est de sécuriser le trafic entre client et serveur WEB par le biais du protocole TLS (postérieur à SSLv3). On utilisera l'outil openssl pour générer essentiellement le certificat garantissant, auprès du client, l'authenticité de la clé publique du serveur.

Dans la machine serveur,
a Activer le module SSL à travers la commande **#a2enmod ssl** et redémarrer le service apache2.
b. Créer un répertoire pour stocker la clé privée et le certificat :**#sudo mkdir /etc/apache2/ssl**
d. A partir de la commande openssl, créer la clé privée /etc/apache2/ssl/apache.key et le certificat électronique /etc/apache2/ssl/apache.crt. Fixer La validité du certificat à 1 an (365 jours) et La taille de la clé privée à 1024 bits. Le certificat est auto-signé (pas de recours à une autorité de certification). Dans ce cas, l'option -nodes est obligatoire.
Une fois la commande lancée, certains champs devraient être remplis : Le nom commun (CN) doit être : <votre nom>.com
#sudo openssl req -x509 -nodes -days 365 -newkey rsa:1024 -keyout /etc/apache2/ssl/apache.key -out /etc/apache2/ssl/apache.crt
e. Visualiser la clé privée avec la commande cat.

f. Visualiser le contenu du certificat électronique avec la commande :

#openssl x509 -in /etc/apache2/ssl/apache.crt -purpose -text.

g. Le principe des serveurs virtuels (virtual host) consiste à cohabiter un ou plusieurs serveurs Web. sur une même machine. Dans notre cas, créer un autre répertoire **monsite2.** Dans ce répertoire, copier les deux fichiers **accueil.html** et **bienvenue.php** du répertoire monsite.

h. Modifier acceuil.html en remplacant le mot « `Simple Login Page` » par « `Simple Secured Login Page` ».

i. Ajouter la cofiguration suivante dans le fichier /etc/apache2/sites-available/default-ssl.conf :

ServerName <votre nom>.com
ServerAlias www.<votre nom>.com
DocumentRoot /var/www/siteisi2
SSLEngine on
SSLCertificateFile /etc/apache2/ssl/apache.crt
SSLCertificateKeyFile /etc/apache2/ssl/apache.key

h. Vérifier que le port 443 est en écoute dans le fichier /etc/apache/ports.conf. Redémarrer apache2 (Pour visualiser et identifier les erreurs, accéder au journal : /var/log/apache2/error.log à travers la commande tail -n pour afficher les n derniers messages d'erreur.)
i. Lancer wireshark puis accéder au site acceuil.html par https.
j. Analyser le trafic en identifiant :
- La Phase de connexion (Three way handshake)
- Les étapes d'établissement d'un tunnel TLS
- Chiffrement des données.
k. Accéder au site à travers http. Vérifier qu'il s'agit d'un autre site.

Remarques
1) pour considérer le fichier default-ssl.conf (seulement un lien symb vers 000-default... existe) créer un lien symbolique de ce dernier dans sites enabled
/etc/apache2/sites-enabled# sudo ln -s ../sites-available/default-ssl.conf default-ssl.conf
1) Ajouter dans le fichier (à creer sil n'existe pas) /etc/apache2/conf-available/fqdn.conf la ligne suiante (pour corriger le warning lors du redemarrage du serveur apache2)
ServerName localhost
puis apliquer les modifications en tapant la commande suivante **#sudo a2enconf fqdn**