SCI 460 – Cryptographie
TP2 : protection des sites web

IUT d'Arles – DUT SRC – 2010-2011

Objectifs : Étude du fichier .htaccess pour protéger un site web ou une partie seulement. Étude des protocoles SSL et HTTPS, grâce à XAMPP : savoir créer et manipuler une signature, un certificat et une clé de session.

Pour l'ensemble du TP, vous trouverez le sujet ainsi que les fichiers nécessaires pour répondre aux questions sur le site : http://x.heurtebise.free.fr. Vous utiliserez le logiciel **XAMPP**, qui contient le module **openssl**, afin de manipuler les protocoles sécurisés.

1 Le fichier *.htaccess*

Maintenant que nous savons où placer notre site, nous allons le sécuriser par l'ajout d'un fichier *.htaccess*. Les fichiers *.htaccess* sont des fichiers de configuration d'Apache, permettant de définir des règles dans un répertoire et dans tous ses sous-répertoires (qui n'ont pas de tel fichier à l'intérieur). On peut les utiliser pour protéger un répertoire par mot de passe, ou pour changer le nom ou l'extension de la page index, ou encore pour interdire l'accès au répertoire. Les fichiers ***.htaccess*** peuvent être utilisés dans n'importe quel répertoire virtuel ou sous-répertoire.

1.1 Intérêt des fichiers htaccess

Les principales raisons d'utilisation des fichiers .htaccess sont :

- Gérer l'accès à certains fichiers.
- Ajouter un mime-type.
- Protéger l'accès à un répertoire par un mot de passe.
- Protéger l'accès à un fichier par un mot de passe.
- Définir des pages d'erreurs personnalisées.

1.2 Principe des fichiers htaccess

Le fichier *.htaccess* est placé dans le répertoire dans lequel il doit agir. Il agit ainsi sur les permissions du répertoire qui le contient et de tous ses sous-répertoires. Vous pouvez placer un autre fichier *.htaccess* dans un sous-répertoire d'un répertoire déjà contrôlé par un fichier *.htaccess*. Le fichier *.htaccess* du répertoire parent reste en « activité » tant que les fonctionnalités n'ont pas été réécrites.

1.3 Empêcher l'accès à des ressources

Un fichier *.htaccess* est composé de deux sections :

- Une 1e section contient les chemins vers les fichiers contenant les définitions de groupes et d'utilisateurs :

```
ErrorDocument 403 http://www.commentcamarche.net/accesrefuse.php3
AuthUserFile /repertoire/de/votre/fichier/.FichierDeMotDePasse
AuthGroupFile /repertoire/de/votre/fichier/.FichierDeGroupe
AuthName "Accès protégé"
AuthType Basic
```

- Une 2e section contient la définition des conditions d'accès :

```
<Limit GET POST>
  Require valid-user
  Require user {username}
</Limit>
```

1.4 Protéger un répertoire par un mot de passe

Il s'agit d'une des applications les plus utiles du fichier ***.htaccess*** car elle permet de définir de façon sûre (à l'aide d'un login et d'un mot de passe) les droits d'accès à des fichiers par certains utilisateurs.

- La commande *AuthUserFile* permet de définir l'emplacement du fichier contenant les logins et les mots de passe des utilisateurs autorisés à accéder à une ressource donnée.
- La commande *AuthGroupFile* permet de définir l'emplacement du fichier contenant les groupes d'utilisateurs autorisés à s'identifier.

Voici un exemple de fichier ***.htaccess*** :

```
ErrorDocument 403 http://www.commentcamarche.net/accesrefuse.php3
AuthUserFile /repertoire/de/votre/fichier/.FichierDeMotDePasse
AuthGroupFile /dev/null
AuthName "Accès sécurisé au site CCM"
AuthType Basic
<LIMIT GET POST>
  Require valid-user
</LIMIT>
```

Le fichier de mot de passe est un fichier texte contenant sur chacune des ses lignes le nom de chaque utilisateur suivi des deux points (*:*), puis du mot de passe crypté (solution recommandée) ou en clair.
Voici un exemple de fichier de mot de passe non crypté (ici ***.FichierDeMotDePasse***)

```
JFPillou:Toto504
Damien:Robert(32)
Comma:Joe[leTaxi]
```

Voici le même fichier contenant des mots de passe cryptés :

```
JFPillou:$apr1$Si0.....$teyL5Y7BR4cHj0sX309Jj0
Damien:$apr1$TD1.....$sfPTHjoufoNsda4HsD1oL0
Comma:$apr1$zF1.....$wYKqRu2aVYlAEQnxVkly8
```

1.5 Crypter les mots de passe

Apache fournit un outil permettant de générer facilement des mots de passe cryptés (aussi bien sous Windows que sous Unix), il s'agit de l'utilitaire *htpasswd* accessible dans le sous-répertoire *bin* d'Apache.

La syntaxe de cet utilitaire est la suivante :

- Pour créer un nouveau fichier de mots de passe :

```
htpasswd -c {chemin du fichier de mot de passe} utilisateur
```

- Pour ajouter un nouvel utilisateur/mot de passe à un fichier existant :

```
htpasswd {chemin du fichier de mot de passe} utilisateur
```

Le mot de passe sera demandé en ligne de commande avec une confirmation.

1.6 Empêcher l'accès à un répertoire par un domaine

La syntaxe pour bloquer l'accès d'un répertoire par un domaine est la suivante :

```
allow from (all, [liste de domaine])
deny from (all, [liste de domaine])
order (Allow,Deny ou Deny,Allow)
```

Voici un exemple de restriction d'accès :

```
ErrorDocument 403 http://www.commentcamarche.net/accesrefuse.php3
AuthUserFile /repertoire/de/votre/fichier/.FichierDeMotDePasse
AuthGroupFile /dev/null
AuthName "Accès sécurisé au site CCM"
AuthType Basic
<LIMIT GET POST>
  order deny,allow
  deny from all
  allow from 193.48.172.2
  require user JFPillou
</LIMIT>
```

Dans ce cas, l'accès ne sera possible que pour l'utilisateur *JFPillou* à partir de l'adresse IP *193.48.172.2* et avec le bon mot de passe.

1.7 Empêcher l'accès à un répertoire autre que celui du fichier .htaccess

Par défaut, Apache applique les restrictions du fichier *.htaccess* à l'ensemble des fichiers du répertoire dans lequel il se trouve ainsi qu'à tous les fichiers contenus dans ses sous-répertoires. Il est également possible de restreindre l'accès à un sous-répertoire grâce à la balise <*Directory*>.

Voici un exemple de restriction aux sous-répertoires « documents » et « scripts » :

```
<Directory "./documents">
  AuthUserFile /repertoire/de/votre/fichier/.FichierDeMotDePasse
  AuthName "Accès privé au répertoire ‘documents’"
  AuthType Basic
  <LIMIT GET POST>
    require user JFPillou
  </LIMIT>
</Files>

<Directory "./script">
  AuthUserFile /repertoire/de/votre/fichier/.FichierDeMotDePasse
  AuthName "Accès privé au répertoire ‘scripts"
  AuthType Basic
  <LIMIT GET POST>
    require user ClaudeDupont
  </LIMIT>
</Files>
```

1.8 Empêcher l'accès à fichier ou un type de fichiers par un domaine

Par défaut, Apache applique les restrictions du fichier *.htaccess* à l'ensemble des fichiers du répertoire dans lequel il se trouve ainsi qu'à tous les fichiers contenus dans ses sous-répertoires. Il est également possible de restreindre l'accès pour un ou plusieurs fichiers du répertoire grâce à la balise *<Files>*.

Voici un exemple de restriction aux fichiers *admin.php3* et *admin2.php3* :

```
<Files admin.php3>
  AuthUserFile /repertoire/de/votre/fichier/.FichierDeMotDePasse
  AuthGroupFile /dev/null
  AuthName "Accès sécurisé au site CCM"
  AuthType Basic
  <LIMIT GET POST>
    require user JFPillou
  </LIMIT>
</Files>

<Files *.png>
  Order Deny, Allow
  Deny from .LeNomDuDomaine.com
</Files>
```

1.9 Autoriser l'accès à un groupe de fichiers par un domaine et un pays

```
<Files php*>
  Order Allow, Deny
  Deny from all
  Allow from .phpfrance.com
  Allow from .fr
</Files>
```

Toutes les personnes (requêtes) provenant du domaine *.phpfrance.com* ou des domaines ayant la terminaison *.fr* pourront avoir accès aux fichiers commençant par php (par exemple, les fichiers *phpbonjour.html*, *phpaurevoir.vxf*) compris dans le répertoire et ses sous-répertoires.

1.10 Protéger un répertoire par un login

Cette méthode permet une authentification de bas niveau uniquement par le nom de l'utilisateur. La syntaxe est la suivante :

```
Require (user [liste des utilisateurs], group [liste des groupes], valid-user)
```

Voici un exemple de ligne du fichier *.htaccess* :

```
Require User Damien Comma PumpPHP Jeff Rastapaye
```

Tout utilisateur souhaitant rentrer dans le répertoire ou un de ses sous-répertoires sera refusé sauf s'il s'identifie en donnant un nom figurant dans la liste.

1.11 Obliger un utilisateur à satisfaire à au moins une des conditions

Voici la syntaxe à placer après les commandes order, deny, allow et require :

```
Satisfy (any, all)
```

1.12 Ajouter un Mime-Type à un répertoire

La syntaxe est la suivante :

```
AddType (mime/type [liste d'extension])
```

Voici un exemple de mise en œuvre du fichier ***.htaccess*** :

```
AddType image/x-photoshop PSD
AddType application/x-httpd-php .php3
```

Le serveur enverra au navigateur Internet le fichier en lui disant de lancer le programme PhotoShop (s'il est installé sur votre machine) et de lui donner le fichier. Habituellement, ceci est utilisé pour des fichiers nécessitant un Plug-In particulier non reconnu par votre navigateur. En pratique, on pourra utiliser cette commande pour ordonner à PHP de parser d'autres extensions de fichier (ex : *.php3*).

1.13 Forcer tous les fichiers d'un répertoire à un Mime-Type

Voici la syntaxe à adopter :

```
ForceType (mime/type)
```

Par exemple avec la ligne suivante, tous les fichiers du répertoire contenant le fichier ***.htaccess*** seront considérés comme étant des fichiers *.jpg* quelle que soit leur extension :

```
ForceType image/jpg
```

Ce type de commande ne peut être utilisé dans les bornes !

1.14 Définir les extensions de fichiers par défaut

La syntaxe à suivre est :

```
DefaultType (mime/type)
```

Par exemple

```
DefaultType text/html
```

Ici, il prendra tout fichier inconnu en tant que document HTML.

1.15 Personnalisation des messages d'erreurs

Il s'agit d'une fonctionnalité pratique car elle permet de définir une page par défaut pour un type d'erreur donné... Cela permet d'une part de guider l'utilisateur au lieu d'afficher la banale page d'erreur du navigateur, ainsi que d'égayer la navigation même en cas d'erreur.

```
ErrorDocument (code-à-3-chiffres [nom du fichier ou texte ou url])
```

Les deux lignes suivantes permettent de définir des pages d'erreurs personnalisées au cas où l'accès à un document serait interdit ou bien que le document n'existe pas :

```
ErrorDocument 403 /erreurs/403.php3
ErrorDocument 404 /erreurs/404.php3
```

Ceci vous permet de donner un message d'erreur personnalisé remplaçant les fichiers fournis avec le navigateur. Voici quelques-unes des erreurs les plus courantes à personnaliser :

- 401 Unauthorized : la personne n'a pas passé avec succès l'identification.
- 403 Forbidden : le serveur n'a pas le droit de répondre à votre requête.
- 404 Not Found : le serveur n'a pas trouvé le document souhaité.

1.16 Changer le fichier index par défaut

La syntaxe pour effectuer ce type d'opération est la suivante :

```
DirectoryIndex (fichiers)
```

Voici un exemple de mise en application :

```
DirectoryIndex index.php index.html index.phtml /erreurs/403.php
```

Lorsque vous essayez d'accéder au répertoire sans préciser la page à afficher, Apache va avoir recours à la directive *DirectoryIndex*. En général, par défaut, cette directive pointe vers *index.html* puis *index.htm*. Dans l'exemple ci-dessus, Apache va commencer par chercher *index.php*, puis *index.html*, et ensuite *index.phtml*. Si aucun de ces trois fichiers existent, la page *403.php* (se trouvant dans la racine) sera affichée pour éviter de lister le répertoire.

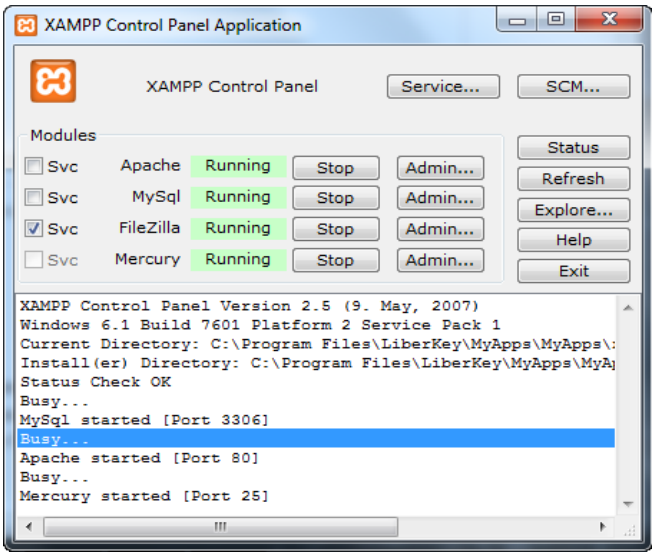
2 Logiciel XAMPP

XAMPP est un ensemble de logiciels permettant de mettre en place facilement un serveur Web, un serveur FTP et un serveur de messagerie électronique. Il s'agit d'une distribution de logiciels libres (X Apache MySQL Perl PHP) offrant une bonne souplesse d'utilisation, réputée pour son installation simple et rapide. Ainsi, il est à la portée d'un grand nombre de personnes puisqu'il ne requiert pas de connaissances particulières et fonctionne, de plus, sur les systèmes d'exploitation les plus répandus.

Il est distribué avec différentes bibliothèques logicielles qui élargissent la palette des services de façon notable : OpenSSL, Expat (parseur XML), PNG, SQLite, zlib, … ainsi que différents modules Perl et Tomcat.

2.1 Lancement des serveurs Apache et Mysql sous XAMPP

Tout d'abord, afin d'activer les serveurs Apache et Mysql, il est nécessaire d'ouvrir le panneau de contrôle de XAMPP. Vous allez pouvoir grâce à cette interface gérer à la fois votre serveur Apache et votre serveur Mysql. Ainsi, si vous voulez lancer le serveur Apache cliquez sur le bouton **Start** en face d'Apache. Si vous souhaitez lancer votre serveur Mysql, faites de même avec la ligne Mysql. Pour vérifier que vos serveurs sont correctement lancés, lorsque vous cliquerez sur les boutons **Start**, vous aurez le message **Running** en vert en face des serveurs lancés correctement.



2.2 Vérification de l'initialisation des serveurs Apache et Mysql

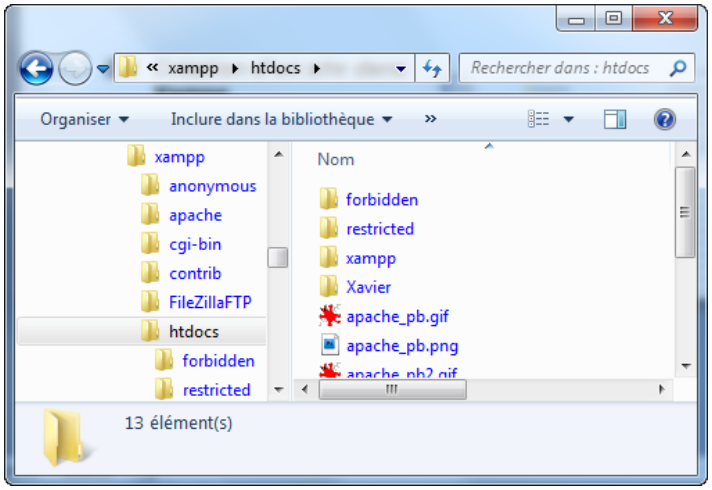
Une fois que vous aurez lancé votre serveur Apache, afin de tester son bon fonctionnement je vous invite à ouvrir votre navigateur et à taper l'adresse suivante : http://localhost Ainsi, vous devriez voir s'afficher une page contenant le logo de XAMPP.

Maintenant nous allons tester que notre serveur Mysql est correctement lancé, pour cela je vous invite à ouvrir votre navigateur et à taper l'adresse suivante : http://localhost/phpmyadmin. Vous devriez voir s'afficher une interface appelée **PhpMyAdmin** permettant de gérer son serveur de BDD Mysql

Maintenant que nous disposons d'un serveur Apache et d'un serveur Mysql, nous pouvons bénéficier d'un serveur Web complet afin de développer notre propre site Internet Local.

2.3 Création d'un site dans XAMPP

Maintenant que nous avons lancé nos 2 serveurs, nous pouvons créer des sites à volonté sur notre serveur local. Ce qu'il faut savoir, c'est que la racine du site est situé à l'endroit où vous avez installé XAMPP dans le dossier **HTDOCS**. Ainsi, lorsque vous voulez créer un site sur votre serveur, il vous suffit de créer un dossier dans le dossier **HTDOCS**.



3 Exercice

3.1 Sécuriser à l'aide d'un fichier .htaccess

Maintenant nous allons sécuriser notre site web grâce au fichier *.htaccess*. Dans cet exercice, vous prendrez le site web de votre choix, ou bien un simple site html, que vous allez placer dans le dossier **HTDOCS** de **XAMPP**, puis vous répondrez aux questions suivantes.

Questions :

Q1. Créez un fichier *.htpassword* dans lequel vous placerez tous les identifiants et mots de passe pour votre site web. N'oubliez pas de crypter les mots de passe pour les rendre illisible, grâce à la commande **htpasswd**.

Q2. Créez maintenant un fichier *.htaccess* dans lequel vous allez restreindre l'accès à différents répertoires de votre site :

- Restreignez l'accès au répertoire principal aux utilisateurs dont les identifiants sont dans le fichier *.htpassword*. Testez l'accès avec le navigateur.
- Rajoutez maintenant une page 'erreur' si un utilisateur non désiré tente d'accéder au répertoire protégé. Testez l'accès avec le navigateur.
- Dans le même fichier *.htaccess*, vous allez restreindre un sous-répertoire à un utilisateur donné et une adresse IP donnée. Testez l'accès avec le navigateur, en spécifiant votre adresse IP dans un premier, puis une autre adresse dans un second temps.
- Dans le même fichier *.htaccess*, vous allez restreindre l'accès à un fichier ou type de fichier à un utilisateur donné. Testez l'accès avec le navigateur.
- Utilisez votre fichier *.htaccess* à changer le fichier index par défaut.

Q3. Le fichier *.htaccess* vous semble t-il suffisamment protégé ?
Q4. A quel endroit convient-il le mieux de placer le fichier contenant les mots de passe ? Que faut-il faire pour être sûr que le fichier contenant les mots de passe soit inaccessible avec un navigateur internet ?
Q5. Comment faire pour inhiber totalement l'action de *.htaccess* ? Comment faire pour rendre l'action de *.htaccess* prioritaire par rapport aux droits des répertoires définis par défaut ?

3.2 Utilisation du protocole SSL

Si vous n'avez pas activé le chiffrement sur un répertoire protégé par un mot de passe, ce dernier sera envoyé en clair, ce qui signifie qu'il peut être vu par n'importe qui, notamment un hacker. C'est une bonne idée de crypter la transmission de ces mots de passe, lors de la connexion d'un utilisateur à un site web. Pour cela, nous allons sécuriser notre site web grâce à un certificat SSL (en utilisant le service OpenSSL de XAMPP), puis nous aurons besoin de nous assurer que les pages protégées par mot de passe sont seulement accessibles par cryptage.

3.2.1 Création d'un certificat SSL and de la clef privée du serveur

Afin d'activer le cryptage de vos mots de passe, vous allez créer un certificat SSL (contenant votre clef publique) et une clef privée du serveur. XAMPP fournit un couple (certificat / clef) par défaut, mais il est préférable de créer son propre certificat, car la clef par défaut est disponible à tous les utilisateurs possédant XAMPP. En effet, si quelqu'un connait votre clef, il peut décrypter tous vos fichiers et transmissions.

Pour créer un couple (certificat / clef), XAMPP fournit un fichier batch qui créée un tel couple avec des clefs de cryptage aléatoires. Pour accéder à ce fichier batch, vous allez exécuter les commandes suivantes :

1. Ouvrez une console en allant dans le menu « Démarrer > Exécuter… », puis tapez « cmd ».
2. Dans la console, allez dans le répertoire « XAMPP\apache » grâce à la commande :
   `cd <chemin relatif/absolu>\XAMPP\apache`
3. Puis tapez la commande suivante :
   `makecert`

Vous verrez les instructions suivantes, à condition que le fichier « `openssl.cnf` » soit présent dans le répertoire « `XAMPP\apache\bin` » :

```
Loading 'screen' into random state - done
Generating a 1024 bit RSA private key
.........................++++++
.....................................++++++
writing new private key to 'privkey.pem'
```

Entrez une phrase clef pour le décryptage de votre clef privée de serveur. Confirmez la phrase clef.

```
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
```

Maintenant vous allez entrer les informations du demander de certificate, soit :

- Les 2 lettres du code du pays (ex : FR pour France)
- L'État ou la Province : optionnel
- Le nom de la ville (ex : Arles)
- Le nom de l'organisation (ex : IUT de Provence)
- Le nom de l'unité organisationnelle : optionnel
- Le nom commun : adresse DNS ou adresse IP du site web (ex : 127.0.0.1 pour localhost) : il est important de choisir l'adresse qui sera indiqué dans le navigateur pour l'accès au site web avec le protocole SSL, sinon un message d'erreur vous avertira lors de la navigation.

```
Country Name (2 letter code) []: FR
State or Province Name (full name) []:
Locality Name (eg, city) []: Arles
Organization Name (eg, company) []: IUT de Provence
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []: 127.0.0.1
```

Éventuellement, il vous sera demandé des informations supplémentaires telles que votre adresse email, un mot de passe, un nom de compagnie (optionnel) et à nouveau votre phrase clef.

```
Email Address []:
Please enter the following 'extra' attributes to be sent with your certificate request
A challenge password []:
An optional company name []:
Enter pass phrase for privkey.pem:
```

Puis vous verrez le message suivant, annonçant la création de la clef RSA. Le fichier « `makecert.bat` » déplacera votre clef privée serveur « `ssl.key` » et le certificat « `ssl.crt` » dans un répertoire approprié « `XAMPP\apache\conf` » :

```
writing RSA key
Loading 'screen' into random state - done
Signature ok
subject=/C=xx/ST=xx/L=xxxx/O=xxx/CN=commonname
Getting Private key
…
The certificate was provided.
Press any key to continue . . .
```

Exercice :

Q6. Créez votre propre certificat pour votre site web.

3.2.2 Importation du certificat dans les navigateurs webs coté client

Puisque le certificat ainsi créé n'est pas signé par l'Autorité de Certification, votre navigateur affichera un message d'attention lorsqu'il entrera sur les pages protégées. Pour éviter cela, il suffit de faire reconnaitre à votre navigateur le certificat que vous avez créé.

Exemple avec Firefox :

1. Menu « `outils > options` » ou « `options > options` »
2. Onglet « `avancé > chiffrement > afficher les certificats` »
3. Onglet « `autorités` » puis le bouton « `importer…` »
4. Sélectionnez le fichier « `ssl.crt\server.crt` » pour l'ouvrir
5. Sélectionnez « Confirmer cette AC pour identifier des sites web » puis cliquez sur « OK »
6. Vous pouvez fermer le gestionnaire de certificats et le panneau d'option.

Exercice :

Q7.Importer votre propre certificat dans le(s) navigateur(s) de votre choix.

3.2.3 Edition des fichiers de configuration d'Apache

Nous allons maintenant expliquer à Apache comment accéder aux répertoires protégés par mots de passe avec le cryptage SSL exclusivement :

1. Mise à jour des fichiers de configuration Apache afin de signaler les répertoires qui doivent être accéder avec le cryptage SSL.
2. Redirection du trafic « http » vers « https » uniquement pour ces pages cryptées en SSL.

Tout d'abord, nous allons informer Apache des répertoires à protéger en SLL. Pour cela, nous allons ajout une directive **SSLRequireSLL** à la fin de chaque bloc `<Directory>` (juste avant `</Directory>`) des fichiers de configuration Apache suivant :

- Config File: `c:\XAMPP\apache\conf\extra\httpd-XAMPP.conf`
  - `c:\directory…\XAMPP\phpmyadmin`
  - `c:\XAMPP\htdocs\XAMPP`
  - `c:\XAMPP\webalizer`
  - `c:\XAMPP\security\htdocs`
- Config File: `c:\XAMPP\webdav`
  - `c:\XAMPP\webdav`

Exemple du fichier de configuration `c:\XAMPP\apache\conf\extra\httpd-XAMPP.conf` :

```
Alias /webalizer "C:/directory…/XAMPP/webalizer/"
<Directory "C:/directory…/XAMPP/webalizer">
    <IfModule php5_module>
    …
    </IfModule>
    AllowOverride AuthConfig
    SSLRequireSSL
</Directory>
```

Exercice :

Q8.Modifiez les fichiers de configuration Apache pour utiliser le protocole SSL.

Ensuite, l'étape suivante est optionnelle, mais elle permet de rediriger les requêtes « http » vers les requêtes « https » pour les pages à sécuriser. Cela permet de mettre automatiquement le mode « https » même si l'utilisateur a entré une adresse web commençant par « http » dans le navigateur. Sans cela, vous ne pourrez pas accéder aux pages si vous n'utilisez pas les requêtes « https ». Pour cela, nous allons activer la commande « `mod_rewrite` » dans le fichier « `XAMPP\apache\conf\httpd.conf` » dans Apache. Pour cela, il suffit de décommenter (en retirant le caractère #.) la ligne suivante :

```
LoadModule rewrite_module modules/mod_rewrite.so
```

Puis, coller le texte suivant au début du fichier « `XAMPP\apache\conf\extra\httpd-XAMPP.conf` »:

```
<IfModule mod_rewrite.c>
    RewriteEngine On

    # Redirect /XAMPP folder to https
    RewriteCond %{HTTPS} !=on
    RewriteCond %{REQUEST_URI} XAMPP
    RewriteRule ^(.*) https://%{SERVER_NAME}$1 [R,L]

    # Redirect /phpMyAdmin folder to https
    RewriteCond %{HTTPS} !=on
    RewriteCond %{REQUEST_URI} phpmyadmin
    RewriteRule ^(.*) https://%{SERVER_NAME}$1 [R,L]

    # Redirect /security folder to https
    RewriteCond %{HTTPS} !=on
    RewriteCond %{REQUEST_URI} security
    RewriteRule ^(.*) https://%{SERVER_NAME}$1 [R,L]

    # Redirect /webalizer folder to https
    RewriteCond %{HTTPS} !=on
    RewriteCond %{REQUEST_URI} webalizer
    RewriteRule ^(.*) https://%{SERVER_NAME}$1 [R,L]
</IfModule>
```

Si vous avez d'autres répertoires où faire cette redirection, ajouter le texte suivant pour chaque répertoire (en modifiant « `folder_name` » par le répertoire) :

```
# Redirect /folder_name folder to https
RewriteCond %{HTTPS} !=on
RewriteCond %{REQUEST_URI} folder_name
RewriteRule ^(.*) https://%{SERVER_NAME}$1 [R,L]
```

Exercice :

Q9. Réalisez les modifications à apporter pour rediriger automatiquement les pages web protégées grâce au protocole SSL, des requêtes « http » vers « https ».