Ensimag 1ère année

TP n°3 – WWW : Serveurs HTTP et HTTPS

> Il est recommandé de prendre des notes. Se référer à la version numérique de ce document pour les liens (sur Chamilo).
> Une question sur l'effet ou l'utilisation d'une commande ?
> **man nom_de_la_commande** !

Ce TP s'intéresse au WWW et plus particulièrement au fonctionnement des serveurs HTTP (*HyperText Transfer Protocol*), des programmes exécutés par ces serveurs (scripts *CGI*) et de l'aspect sécurité sous-jacent.

Avertissement : L'installation de services tels qu'un serveur HTTP n'est autorisée que par l'administrateur du site. Cela vous est permis dans le cadre de ce TP, mais en aucun cas sur les ressources de l'école en dehors de ce contexte.

Attention : Lorsque vous effectuerez des modifications de configuration du serveur, il sera nécessaire de redémarrer le serveur, c'est-à-dire de le tuer puis de le relancer. À chaque modification du fichier `httpd.conf`, le plus simple est encore d'enchainer ces deux commandes pour relancer le serveur avec la nouvelle configuration :

```
killall httpd
httpd -f ~/www/conf/httpd.conf
```

1 Lancement d'Apache

Le logiciel *Apache* est le serveur WWW le plus répandu. Il permet de configurer un site WWW en spécifiant le répertoire racine des documents du site, les droits d'accès à ces documents, ainsi que le répertoire des scripts CGI (*Common Gateway Interface*).

Pour lancer un serveur Apache, il faut lancer le programme `httpd` en lui indiquant un fichier de configuration. Procédez de la manière suivante :

— Récupérez l'archive qui contient les fichiers nécessaires à ce TP sur Chamilo. L'archive est disponible sur la page du cours 3MMIRC, section *Documents*, répertoire *TP*. Son nom est `www.tar.gz`. Dans la racine de votre répertoire personnel `~/`, décompressez l'archive récupérée à l'aide de la commande :

```
tar xvzf www.tar.gz
```

La configuration se trouve alors dans `~/www`. N'hésitez pas à regarder le contenu du fichier `~/www/conf/httpd.conf` pour avoir un aperçu de la configuration fournie par défaut, en prenant le temps en particulier de lire les informations présentées sous forme de commentaires.

— Pour identifier votre page web de manière unique, éditez le fichier `~/www/htdocs/index.html` et ajoutez votre nom et prénom entre les balises `<title>` et `</title>`, et également dans `<h1>`... `</h1>` de cette page. Notez que le répertoire `~/www/htdocs` est le répertoire contenant les fichiers du site web. Nous avons configuré ce comportement en ajoutant la balise `DocumentRoot "${HOME}/www/htdocs"` dans le fichier de configuration `httpd.conf`. Cela signifie en pratique que *la racine de notre site web*, c'est-à-dire le répertoire de départ parcouru lorsqu'on tape l'url `http://localhost:[port]/` dans Firefox correspond au répertoire `~/www/htdocs` du système de fichier du serveur web. En d'autres termes, la page web `http://localhost:[port]/index.html` se trouve physiquement à l'emplacement `~/www/htdocs/index.html` sur le disque.

— Lancez le *daemon* (programme s'exécutant en arrière plan) httpd :
`httpd -f ~/www/conf/httpd.conf`
Vous remarquerez que vous ne possédez pas les droits nécessaires pour lancer le serveur sur le port 80 (il fait parti des ports réservés). Modifiez le port d'écoute (`Listen 80`) dans le fichier `~/www/conf/httpd.conf` par un port autorisé (8080 par exemple).

1. Vérifiez à l'aide de Mozilla Firefox que le serveur fonctionne correctement, en vous connectant à l'URL `http://localhost/index.html`. *Pourquoi la connection échoue-t-elle ? Que devez-vous changer pour avoir accès à la page demandée ?* (voir `https://tools.ietf.org/html/rfc2616#section-3.2` et `https://en.wikipedia.org/wiki/URI_scheme#Syntax`)
2. Essayez d'accéder à l'URL `http://localhost:[port]/toto.html`, en remplaçant `[port]` par le port d'écoute choisi dans la configuration. Observez ensuite le contenu des fichiers de logs (dans `~/www/logs`). *Quelles informations sont enregistrées dans ces fichiers ?*

2 Scripts CGI

Un script CGI (*Common Gateway Interface*) étend les fonctionnalités d'un serveur HTTP en lui permettant de lancer un programme exécutable externe. En effet, le serveur lance le script CGI avec un certain nombre de paramètres et le script peut ensuite lancer le programme externe. C'est également le script qui récupère le résultat et le formate. Le résultat retourné par le script contient le type MIME de l'objet généré (par exemple `text/html`) suivi du contenu binaire. Un script CGI peut être écrit en Perl, Python, PHP, script shell ou tout autre langage de programmation. La seule contrainte pour qu'il soit utilisable est de respecter l'interface CGI avec le serveur HTTP.

1. *Décomposer les URL suivantes en protocole, machine, document et paramètres :*

   ```
   http://localhost:[port]/cgi-bin/test?parametre
   http://localhost:[port]/cgi-bin/test?plein+de+parametres
   ```

   Accédez à ces URL et interprétez le plus de variables possibles affichées sur la page web générée.
2. Exécutez le script qui lance l'utilitaire ping : `http://localhost:[port]/post-ping.html`. *Comment ce script obtient-il l'adresse du site sur lequel il lance ping ?*
3. Exécutez le script qui lance l'utilitaire fping : `http://localhost:[port]/get-ping.html`. *Regardez le code source de cette page et expliquez ce qui se passe en appuyant sur le bouton « valider ». Quel script est exécuté ? Comment ce script récupère-t-il l'adresse du site sur lequel il lance ping ?*

3 Hôtes virtuels

Jusqu'à présent, vous avez observé le comportement du serveur lorsqu'il n'héberge qu'un unique site web. Il est néanmoins possible d'héberger différents sites web sur un unique serveur. L'intérêt est

évident : il n'y a alors pas besoin d'avoir plusieurs serveurs matériels, ni plusieurs serveurs logiciels, ce qui est un gain de ressources non négligeable, facilitant en outre l'administration du serveur.

Dans cette partie, vous allez utiliser des hôtes virtuels basés sur le nom. Il est aussi possible d'avoir des hôtes virtuels basés sur l'adresse IP, qui fonctionnent de manière similaire.

Il faut tout d'abord pouvoir accéder au serveur HTTP à l'aide de noms différents. Une solution simple pour cela est de modifier le fichier `/etc/hosts`. Malheureusement, la modification de ces fichiers nécessite d'avoir des droits root. Heureusement pour nous, les machines de l'ENSIMAG possèdent deux noms :

— `localhost`

— `ensipcXXX` (`XXX` étant le numéro de la machine)

Vous pouvez donc accéder à votre serveur HTTP en utilisant les adresses `http://localhost:[port]/` ou `http://ensipcXXX:[port]/`.

Éditez le fichier `~/www/conf/httpd.conf` et décommentez les lignes qui se trouvent en fin de fichier et précédées par la mention "`# TP - Partie 3 - à décommenter ==>`".

Attention : il vous faudra aussi ajouter une partie `<Directory>` pour `~/www/ensipcXXX` qui utilisera la directive `Require all granted`. N'hésitez pas à observer dans le reste du fichier la syntaxe précise à utiliser. Vous pouvez ajouter cette directive juste au-dessus des lignes que vous venez de décommenter, par exemple à l'emplacement indiqué par "`# TP - Partie 3 - à compléter...`".

Chaque partie `VirtualHost` définit un hôte virtuel en fonction de noms définis par `ServerName`. Le premier hôte virtuel est l'hôte par défaut, c'est-à-dire le site web qui est utilisé lorsque la requête effectuée par le client ne correspond à aucun des hôtes virtuels. Il remplace donc le site qui était auparavant défini dans le fichier de configuration. On peut constater que les racines des sites se situent dans deux répertoires distincts.

Créez le répertoire `~/www/ensipcXXX` et ajoutez quelques fichiers pour avoir un site minimal.

Comme le fichier de configuration a été modifié, il est nécessaire de redémarrer le serveur HTTP (comme indiqué en début de TP).

Il est bien entendu possible d'ajouter des directives spécifiques pour chaque site web dans la partie `VirtualHost` correspondante.

1. Testez le bon fonctionnement de vos hôtes virtuels. *Comment le serveur HTTP sait-il quel site web est demandé par le navigateur ?*
2. *À l'aide des scripts CGI disponibles, observez les variables d'environnement qui varient selon l'hôte virtuel.*

4 URL et système de fichiers

L'arborescence visible sur un site web ne correspond pas forcément à l'arborescence physique qui existe sur le disque dur. Ainsi, le fichier accessible via l'adresse `http://localhost:[port]/ensiwikiFAQ` peut très bien ne pas se situer dans `~/www/htdocs`. Cette possibilité est utile à la fois pour mieux organiser les fichiers sur le serveur, mais aussi pour cacher certains détails à l'utilisateur, ou tout simplement pour faciliter la vie de l'administrateur.

Cette partie consiste à utiliser trois méthodes différentes pour parvenir à ce résultat.

4.1 Redirection

Éditez le fichier de configuration `~/www/conf/httpd.conf` et ajoutez, dans la partie `<Directory "~/www/htdocs">` la ligne suivante (veillez à tout mettre sur la même ligne) :

```
Redirect /ensiwikiFAQ http://ensiwiki.ensimag.fr/index.php/FAQ
```

Redémarrez ensuite le serveur HTTP (comme indiqué en début de TP).

1. *Quel est l'effet de cette directive ?*
2. Observez le trafic généré lors de la requête de la page `http://localhost:[port]/ensiwikiFAQ`. *Qu'observez- vous ? Dans quels cas cette directive peut s'avérer utile ?*

4.2 Alias

Éditez le fichier de configuration `~/www/conf/httpd.conf` et ajoutez **AVANT** la partie `<Directory "${HOME}/www/htdocs">` les lignes :

```
Alias /logfiles "${HOME}/www/logs"
<Directory "${HOME}/www/logs">
    Require all granted
    Options Indexes
</Directory>
```

Redémarrez ensuite le serveur HTTP.

3. Allez à l'adresse `http://localhost:[port]/logfiles/`. *Qu'observez-vous ? Où se situent sur le disque dur les fichiers affichés dans le navigateur par rapport à la racine du site ?*

4.3 Réécriture

Éditez le fichier de configuration `~/www/conf/httpd.conf` et ajoutez, **A L'INTERIEUR DE** la partie `<Directory "${HOME}/www/htdocs">` les lignes :

```
RewriteEngine on
RewriteRule ^/?fping/([^/]*) /cgi-bin/fping?$1 [L,PT]
```

Redémarrez ensuite le serveur HTTP.

4. Chargez l'adresse `http://localhost:[port]/fping/google.fr/`. *Que se passe-t-il ? Est-ce spécifique à google.fr ? Quel est l'intérêt de ces directives ?*

5 Authentification

L'objectif de cette partie va être d'étudier les possibilités d'authentification qu'offre le serveur web Apache. Pour cela, on va créer et remplir un fichier de mots de passe. Prenez un peu de temps pour lire le document hébergé à l'adresse `http://www.coopernet.fr/infos/securite/gerer-mot-de-passe` qui explique les risques liés au stockage de mots de passes sur un serveur.

1. *Que recommandent les "best practices" en termes de stockage de mot de passe ? (chiffrement ? hash ? clair ? fonction utilisée ?)*

— La syntaxe pour créer le fichier de mot de passe est :
`htpasswd -c fichier-de-passwd nom-utilisateur`
Par exemple, la commande :
`htpasswd -c pass.txt toto` créé le fichier de mots de passe `pass.txt` en ajoutant une entrée pour l'utilisateur `toto`.
Cette commande vous demande un mot de passe pour l'utilisateur *nom-utilisateur*.

— Rajoutez d'autres utilisateurs avec :
`htpasswd fichier-de-passwd nom-utilisateur`
Utilisez le même fichier et mettez un autre nom d'utilisateur.

— Mettez en place le contrôle de sécurité pour les accès au répertoire `~/www/htdocs`. Pour cela, décommentez dans le fichier `httpd.conf` la partie qui débute par "`# TP - Partie 5`".

— Redémarrez ensuite le serveur HTTP.

2. *Qu'avez-vous activé exactement dans `httpd.conf` ? Que se passe-t-il maintenant quand vous accédez à vos pages web ?*
3. *Examinez la requête de votre navigateur ainsi que la réponse du serveur avec wireshark en capture sur l'interface `lo0`. Avez-vous vraiment sécurisé l'accès à vos pages ?*

6 Serveur HTTPS

Toute communication utilisant le protocole HTTP se fait de manière non sécurisée. Pour s'en convaincre, utilisez wireshark pour capter et visualiser le contenu des paquets émis et reçus par le serveur ou le client : les pages, données et résultats de formulaires passent sous forme non chiffrée sur le réseau, interceptable par n'importe quelle machine sur les réseaux concernés. Imaginez que vous remplissiez sur un site un formulaire avec votre numéro de carte bleue : vous comprendrez qu'il s'agit d'un système non fiable pour effectuer des achats ou transmettre toute information sensible sur le web.

Pour pallier ces problèmes de sécurité, un protocole d'échange de pages web sécurisé a été créé : il s'agit d'HTTPS (Secure HTTP). Ce protocole repose sur le concept de certificat. Un certificat est un fichier d'authentification d'un site web. Il lui est associé une paire de clés de chiffrement SSL (de nature identique à celles manipulées avec ssh) qui permet d'échanger des données chiffrées avec le site à protéger. Ce certificat est validé par un organisme dédié (il en existe plusieurs, comme Comodo, Thawte, Verisign, etc.), qui a pour fonction de garantir l'association entre l'adresse IP d'un site web, son URL et la clé publique de chiffrement qui lui est associée. L'organisme est donc garant du fait que l'on communique bien avec l'interlocuteur attendu (mais pas de la bonne foi de cet interlocuteur !).

6.1 Certificats et trousseau de confiance (trust store)

À l'aide de Mozilla Firefox, rendez-vous sur le site `https://igc.services.cnrs.fr/CNRS2-Standard`. Analysez l'avertissement de votre navigateur qui s'affiche lors de la première connexion à ce site web.

1. Visualisez le certificat qui vous est proposé en cliquant sur *Je comprends les risques* puis sur *Ajouter une exception...*. Dans la fenêtre qui s'ouvre, cliquez ensuite sur *Voir...* pour afficher les informations relatives au certificat. *Quelles informations importantes y figurent ? (en particulier, quel est l'algorithme de signature utilisé ? quelle est la clé publique du serveur www.cnrs.fr ? quelle*

est la durée de validité de ce certificat ?) Quel organisme a délivré ce certificat ? A quoi correspond la signature présente ? Par quelle entité a-t-elle été faite ? Maintenant que vous avez vérifié ces informations, acceptez définitivement ce certificat.

2. Toujours avec Mozilla Firefox, rendez-vous sur le site `https://particuliers.societegenerale.fr/`. Visualisez le certificat du site : pour ce faire, ouvrez les informations de la page (clic droit sur la page, puis `Informations sur la page`) puis accédez à l'onglet `Sécurité` pour finalement cliquer sur le bouton `Afficher le certificat`. *Quel organisme a délivré ce certificat ?*
3. Ouvrez la fenêtre de gestion de certificats de votre navigateur. Pour ce faire, cliquez sur le bouton *Menu* de Firefox, représenté par trois barres horizontales en haut à droite de la fenêtre. Cliquez ensuite sur *Préférences*, puis choisissez *Avancé* dans le menu de gauche. Cliquez enfin sur l'onglet *Certificats* puis sur le bouton *Afficher les certificats*. Retrouvez les organismes ayant délivré les certificats des deux sites aux questions précédentes. *Pourquoi selon vous Firefox vous fait examiner le certificat de force pour le site de la question 1 et pas pour celui de la question 2 ?*
4. Supposons qu'une connexion HTTPS ait été établie avec succès (pas de message d'erreur) avec un serveur web. *Quelles propriétés de sécurité sont assurées et pour chaque sur quelles entités ?*

6.2 Génération d'un certificat

La validation d'un certificat par un organisme d'autorité est généralement payante. Cependant n'importe quelle personne peut créer pour ses besoins un certificat non vérifié avec la clé de chiffrement correspondante. Il ne manquerait alors plus à la personne que d'envoyer les fichiers correspondants à un organisme de vérification, qui se chargerait de valider le certificat.

À titre d'exemple, vous allez maintenant générer un certificat non vérifié par un organisme d'autorité (auto-vérifié) pour votre site :

Dans `~/www/conf/`, générez la clé de chiffrement privée de votre site à l'aide de la commande suivante :
`/usr/bin/openssl genrsa > privkey.pem`

Générez une requête de certificat à l'aide de la commande suivante :
`/usr/bin/openssl req -new -key privkey.pem -out cert.csr`

Il vous est demandé de remplir des champs, comme le pays où est basé votre site, le nom de l'organisation/compagnie du site, le nom du site, le courriel de l'administrateur/responsable du site. N'hésitez pas à mettre les noms qui vous semblent les plus officiels dans tous les champs administratifs (Ministère de la Défense, Elysée...). C'est cette requête qui serait examinée par un organisme de vérification : l'entreprise existe-t-elle ? L'entreprise est-elle bien celle qu'elle prétend être ? Le contact administratif est-il bien joignable ? Cela explique pourquoi ces organismes sont payants, car ils gèrent aussi tous les aspects légaux de cette authentification.

Signez cette requête vous-même à l'aide de la commande suivante :
`openssl x509 -req -in cert.csr -extensions v3_ca -signkey privkey.pem -out cacert.pem -trustout`

Cette opération est celle normalement effectuée par l'organisme ayant autorité de vérification, qui vous renverrait alors le certificat (après vérification de vos informations et paiement de votre part) sous la forme du fichier `cacert.pem`. Pour activer HTTPS, dans votre fichier `httpd.conf`, décommentez la ligne :

```
Include ${HOME}/www/conf/00-ssl.conf
```

et ajoutez un port d'écoute pour les connections SSL en ajoutant la ligne :

```
Listen 8443
```

Redémarrez votre serveur HTTP.

5. *Observez à l'aide de wireshark et décrivez les échanges avec votre serveur HTTP sécurisé (`https://localhost:8443/`) lors de votre première connexion à celui-ci (faites la capture sur l'interface `lo0`).*
6. *Quel est le comportement de Mozilla Firefox pendant cette connexion ? Notez bien le message d'erreur.*
7. *Pourquoi est-il important d'examiner soi-même les certificats que votre navigateur indique comme non vérifiés ?*

7 Cookies

Un cookie est un enregistrement d'informations par le serveur dans un fichier situé sur l'ordinateur client (le vôtre), informations que ce même serveur peut aller relire et modifier ultérieurement. Un cookie se compose d'un ensemble de variables (ou de champs) que le client et le serveur s'échangent lors de transactions HTTP, stockées sur la machine cliente soit dans un simple fichier texte, soit représenté comme une entrée d'une base de données. Un cookie est obligatoirement rattaché à un nom de domaine et un ensemble d'URL de telle sorte que seule une requête provenant du même serveur pourra y accéder. Grâce à un programme CGI, le serveur a la possibilité de mettre à jour ou d'effacer un cookie. Un même "client" peut stocker environ un maximum de 300 cookies, dont 20 maximum pour un même serveur, chaque cookie pouvant environ atteindre jusqu'à 4000 octets (`https://tools.ietf.org/rfc/rfc6265.txt`).

1. Examinez les cookies enregistrés par Firefox. Pour ce faire, naviguez dans les préférences de Firefox. Dans le menu de gauche, cliquez sur *Vie privée*, puis sur le lien hypertexte *supprimer des cookies spécifiques*.

   Le formulaire `http://localhost:[port]/cookie.html` traite deux cookies « nom » et « fruit » et permet de les mettre à jour. Après avoir soumis le formulaire, constatez que son effet a bien été pris en compte par Firefox. *Après combien de temps les cookies expirent-ils ?*
2. *Quelle est l'URL qui traite effectivement votre requête lorsqu'elle est soumise, dans la manipulation précédente ?* Observez les échanges à l'aide de wireshark. En déduire quels champs permettent de lire et de mettre à jour l'état des cookies.