Ensimag 1ère année

TP n°4 — Serveur web : HTTP et HTTPS

Il est recommandé de prendre des notes.
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.

1 Exécution d'HTTP « à la main » — le retour

On se rappelle que, lors du TP1, on avait exécuté le protocole HTTP à la main, en lançant une requête HTTP vers le serveur `duda.imag.fr` sans utiliser de navigateur, mais en utilisant la commande `telnet`. À ce moment-là, vous vous comportiez comme le **client** HTTP (le navigateur).

Q 1 — *Pour vous rafraîchir la mémoire, essayez d'obtenir la page d'accueil de `duda.imag.fr` en lui envoyant une requête `GET`.*

Lorsqu'on est client, c'est nous qui faisons les requêtes et le serveur qui nous répond. Vous allez maintenant faire office de **serveur** HTTP à la main, en répondant à un client qui viendrait vous faire des requêtes. Pour cela, nous allons ouvrir une connexion TCP en mode *listen* : c'est une connexion côté serveur, qui attend qu'un client se connecte à elle.

Q 2 — *À l'aide du `man`, trouvez comment préciser à `netcat` de basculer en mode* listen *sur le port 8080. Attention, si vous n'êtes pas sur un ensipc ou une VM Ensimag, vous pouvez avoir des variantes de netcat telle que nc ou ncat.*

Info : on utilise le port 8080 plutôt que le port 80, port standard pour HTTP, parce que l'ouverture du port 80 nécessite les droits d'administrateurs, que vous n'avez pas ici. En fait, le port 8080 est même un port très utilisé pour avoir des mini-serveurs HTTP locaux : `/etc/services` le nomme même `http-alt`.

Lancez `netcat` de cette façon-là dans votre terminal. Votre terminal s'arrête et attend la connexion d'un client (c'est normal qu'il ne dise plus rien).

Q 3 — *Ouvrez l'URL* `http://localhost:8080` *dans un navigateur. Que se passe-t-il dans* `netcat` *?*

Q 4 — *Composez une réponse HTTP correcte minimale (vous devrez peut-être finir votre réponse par* `Control-C`*). Vous pouvez vous aider en regardant ce qu'un serveur web vous répond lorsque vous lui faites une requête (ou la diapo du cours AP_www donnant le format d'une réponse HTTP). Qu'avez-vous répondu ? Est-ce que le navigateur a correctement interprété votre réponse ?*

Voilà ! Vous venez de faire votre premier serveur HTTP. L'approche et l'implémentation sont très basiques, mais c'est quand même selon ce paradigme que les serveurs web et votre navigateur communiquent.

2 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 `apache2` 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
```

Il est impératif d'extraire l'archive à la racine du répertoire personnel, et sur un ensipc ou une VM. En effet, l'archive contient des fichiers de configuration qui partent du principe que l'on est dans cette condition exacte. **Si vous évitez d'utiliser noVNC pour des raisons de temps de latence :** connectez-vous par SSH (au lieu de noVNC) sur un ensipc, extrayez l'archive là-bas. Utilisez le navigateur sur votre machine, mais remplacez alors les occurences de `localhost` en `ensipcXXX` dans la suite du TP.
La configuration se trouve alors dans `~/www`. N'hésitez pas à regarder le contenu du fichier `~/www/conf/apache2.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 `apache2.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/` dans Firefox correspond au répertoire `~/www/htdocs` du système de fichier du serveur web. En d'autres termes, la page web `http://localhost/index` se trouve physiquement à l'emplacement `~/www/htdocs/index.html` sur le disque.

— Lancez le *daemon* (programme s'exécutant en arrière plan) apache2 :

```
export APACHE_RUN_DIR=~/www/run/
apache2 -f ~/www/conf/apache2.conf
```

(attention : le chemin doit **impérativement** être un chemin absolu)

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 `apache2.conf`, le plus simple est encore d'enchaîner ces deux commandes pour relancer le serveur avec la nouvelle configuration :

```
killall apache2
apache2 -f ~/www/conf/apache2.conf
```

Q 5 — *Vérifiez à l'aide de Mozilla Firefox que le serveur fonctionne correctement, en vous connectant à l'URL* `http://localhost:8080/index.html`.

Essayez d'accéder à l'URL `http://localhost:8080/toto.html`. Observez ensuite le contenu des fichiers de logs `~/www/logs` (vous pouvez par exemple le surveiller avec `tail -f <fichier>`, cf. le `man` pour la signification de cette commande).

Q 6 — *Quelles informations sont enregistrées dans ces fichiers ?*

3 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. Pour activer CGI, dans votre fichier `apache2.conf`, décommentez la ligne :

```
Include ${HOME}/www/conf/tp-cgi.conf
```

Enfin, redémarrez votre serveur HTTP.

Q 7 — *Décomposez les URL suivantes en protocole, machine, document et paramètres :*

```
http://localhost:8080/
http://localhost:8080/cgi-bin/test?parametre
http://localhost:8080/cgi-bin/test?plein+de+parametres
```

Q 8 — *Accédez à ces URL et remarquez les variables SHELL Unix dont le script nommé "test" hérite, et qu'il vous montre sur la page web générée.*

Ouvrez la page `http://localhost:8080/post-ping.html`. Elle contient un formulaire, que l'on peut soumettre. Regardez la source de ce fichier (Ctrl+U sous Firefox) et repérez le champ `<form>` qui définit le formulaire.

Q 9 — *À quel script sont envoyées les informations du formulaire ?*

Soumettez ce formulaire pour exécuter ce script.

Q 10 — *Comment ce script obtient-il l'adresse du site sur lequel il lance ping ?*

4 Attaques XSS

Un type d'attaque très courante sur le web est l'attaque XSS, pour Cross Site Scripting. L'aspect croisé vient du fait que la victime n'est pas connectée directement à la machine de l'attaquant, ni l'attaquant à celle de la victime, mais que l'attaquant se sert du serveur d'un site honnête pour y attaquer le client qui tourne sur la machine de la victime. C'est une attaque basée sur l'injection de code dans une page. Nous en voyons ici rapidement plusieurs facettes.

4.1 Injection d'éléments HTML

Q 11 — *Allez sur `http://localhost:8080/cgi-bin/livredor`, faites quelques essais et surtout regardez le fichier correspondant pour comprendre comment le livre d'or marche.*

Nous allons maintenant injecter du code HTML. Les titres en HTML sont délimités par la balise ouvrante `<h1>` en début de titre et la balise fermante `</h1>` en fin de titre.

Q 12 — *Utilisez le formulaire de `http://localhost:8080/cgi-bin/livredor` pour "injecter" un titre dans le livre d'or*

Vous trouvez que cette faille n'est pas très dangereuse ? L'exemple suivant devrait vous convaincre du contraire. Voici un élément malicieux :

```
<div style="position:absolute; top:0; left: 0; right: 0; text-align: center;
background-color:red; padding: 20px;">
Accédez à la nouvelle version du site <a href="http://url-malicieuse.html">ici</a>
</div>
```

Q 13 — *Utilisez le formulaire de `http://localhost:8080/cgi-bin/livredor` pour "injecter" le code malicieux (sans les retours à la ligne). Pourquoi est-ce dangereux ?*

4.2 Injection de code javascript

Le langage javascript est utilisé pour rendre les sites web plus dynamiques. Il est compris entre les balises `<script>` et `</script>`, par exemple `<script>alert("yo")</script>`. Le script est exécuté sur la machine du client par son navigateur.

Q 14 — *Utilisez à nouveau la faille XSS du livre d'or pour "injecter" du code javascript produisant une alerte.*

Javascript permet aussi de récupérer des cookies, de les envoyer par mail, de produire un "key-logger" sur la page...

5 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.

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 Digicert ou Comodo, et vous avez peut-être entendu parler de Let's Encrypt) ; qui a pour fonction de garantir l'association entre l'URL d'un site web 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).

5.1 Certificats et trousseau de confiance (trust store)

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

Visualisez le certificat qui vous est proposé en cliquant sur *Avancé* puis *Afficher le certificat.*

Q 15 — *Quelles informations importantes y figurent ? En particulier, quel est l'algorithme de signature utilisé ? Quelle est la clé publique du serveur* `www.anssi.gouv.fr` *? Quelle est la durée de validité de ce certificat ? Quel organisme a délivré ce certificat ? À quoi correspond la signature présente ? Par quelle entité a-t-elle été faite ? Pour quels nom(s) de serveur ce certificat est-il valide ? Pourquoi le certificat est-il refusé ?*

Maintenant que vous avez vérifié ces informations, acceptez ce certificat en cliquant sur *Accepter le risque et poursuivre.*

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 (utilisez le raccourci clavier `Ctrl + i`) puis accédez à l'onglet *Sécurité* pour finalement cliquer sur le bouton `Afficher le certificat`.

Q 16 — *Quel organisme a délivré ce certificat ?*

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 *Vie privée et sécurité* dans le menu de gauche. Cliquez enfin sur le bouton *Afficher les certificats*. Retrouvez les organismes ayant délivré les certificats des deux sites aux questions précédentes.

Supposons qu'une connexion HTTPS ait été établie avec succès (pas de message d'erreur) avec un serveur web.

Q 17 — *Quelles propriétés de sécurité sont assurées ? Quelles entités sont concernées par ces propriétés ?*

5.2 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 `apache2.conf`, décommentez la ligne :

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

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

```
Listen 8443
```

Enfin, redémarrez votre serveur HTTP.

Q 18 — *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`).*

Q 19 — *Quel est le comportement de Mozilla Firefox pendant cette connexion ?*

Q 20 — *Pourquoi est-il important d'examiner soi-même les certificats que votre navigateur indique comme non vérifiés ?*

6 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 et représentées comme des entrées 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.

Examinez les cookies enregistrés par Firefox. Pour ce faire, naviguez dans les préférences de Firefox. Dans le menu de gauche, cliquez sur *Vie privée et sécurité*, puis sur *Gérer les données* dans la partie *Cookies et données de sites*. Comme cet onglet de Firefox ne donne pas tous les détails de chaque cookie (comme la date d'expiration), vous pouvez aussi ouvrir l'outil d'analyse de réseau de Firefox (Menu>Outils supplémentaires >Outils de développement Web>Réseau) pour voir ce qui circule exactement (analogue de Wireshark, mais dans Firefox).

Le formulaire `http://localhost:8080/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.

Q 21 — *Quelle est l'URL qui traite effectivement votre requête lorsqu'elle est soumise, dans la manipulation précédente ? Observez les échanges à l'aide de l'outil d'analyse réseau de Firefox. Quels champs permettent de lire et de mettre à jour l'état des cookies ?*

Q 22 — *Après combien de temps les cookies expirent-ils ?*

Firefox stocke vos données personnelles dans votre environnement utilisateur (`~/.mozilla/firefox/`). Vos cookies sont stockés dans un fichier `cookies.sqlite` contenant une base de données relationnelle exploitable par des requêtes en langage SQL.

Q 23 — *Trouvez l'emplacement de ce fichier `cookies.sqlite` à l'aide de la commande `find`.*

Fermez votre navigateur Firefox, puis affichez les informations contenues dans ce fichier en lançant la commande `sqlite3 cookies.sqlite` dans le bon dossier.
Dans l'interpréteur de commandes SQL, identifiez les cookies précédement créés à l'aide d'une requête SQL comme `SELECT * FROM moz_cookies where name like 'fruit';`

Q 24 — *Quel est le format utilisé pour stocker la date et l'heure d'expiration d'un cookie ?*

Partie OPTIONNELLE

Les parties suivantes sont à regarder si vous en avez le temps, en fin de TP, ou plus tard si vous voulez comprendre ces éléments du fonctionnement des sites Web.

7 Hôtes virtuels

Jusqu'à présent, vous avez observé le comportement du serveur lorsqu'il n'héberge qu'un unique site web. Vous savez néanmoins qu'il est possible d'héberger différents site sur un unique 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` et `ensipcXXX` (`XXX` étant le numéro de la machine).

Vous pouvez donc accéder à votre serveur HTTP en utilisant les adresses `http://localhost:8080/` ou `http://ensipcXXX:8080/`.

Éditez le fichier `~/www/conf/apache2.conf` et décommentez la ligne « `# Include ${HOME}/www/conf/tp-vhost.conf` ».

Attention : il vous faudra compléter dans le fichier `~/www/conf/tp-vhost.conf` une partie `<Directory>` pour `~/www/ensipcXXX` qui utilisera la directive `Require all granted`. N'hésitez pas à observer dans le fichier `~/www/conf/apache2.conf` la syntaxe précise à utiliser.

Chaque partie `VirtualHost` définit un hôte virtuel en fonction de noms définis par `ServerName`. Le premier hôte virtuel défini dans `~/www/conf/apache2.conf` 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. Le deuxième hôte virtuel est celui ajouté par `~/www/conf/tp-vhost.conf`. On peut constater que les racines des deux sites se situent dans deux répertoires distincts. Il est bien entendu possible d'ajouter des directives spécifiques pour chaque site web dans la partie `VirtualHost` correspondante.

Comme les fichiers de configuration ont été modifiés, il sera nécessaire de redémarrer le serveur HTTP (comme indiqué en début de section 2).

Créez le répertoire `~/www/ensipcXXX` et ajoutez quelques fichiers pour avoir un site minimal. Puis testez le bon fonctionnement de vos hôtes virtuels, sans oublier de redémarrer le serveur puisque sa configuration a été modifiée.

Q 25 — *Comment le serveur HTTP sait-il quel site web est demandé par le navigateur ?*

Q 26 — *À l'aide des scripts CGI disponibles, observez les variables d'environnement qui varient selon l'hôte virtuel.*

8 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:8080/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.

8.1 Redirection

Éditez le fichier de configuration `~/www/conf/apache2.conf` et ajoutez, dans la partie `<Directory "{$HOME}/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).

Q 27 — *Quel devrait être l'effet de cette directive ?*

Observez le trafic généré lors de la requête de la page `http://localhost:8080/ensiwikiFAQ`. Capturez sur toutes les interfaces simultanément.

Q 28 — *Qu'observez-vous ?*

8.2 Alias

Éditez le fichier de configuration `~/www/conf/apache2.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. Allez à l'adresse `http://localhost:8080/logfiles/`.

Q 29 — *Qu'observez-vous ? Où se situent sur le disque dur les fichiers affichés dans le navigateur par rapport à la racine du site ?*

8.3 Réécriture

Éditez le fichier de configuration `~/www/conf/apache2.conf` et ajoutez, **à l'intérieur de** la partie `<Directory "${HOME}/www/htdocs">` les lignes :

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

Redémarrez ensuite le serveur HTTP. Chargez l'adresse `http://localhost:8080/fping/google.fr/`.

Q 30 — *Que se passe-t-il ? Est-ce spécifique à google.fr ? Quel est l'intérêt de ces directives ?*

9 Authentification par mot de passe

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 passe sur un serveur.

Q 31 — *Quelles sont les pratiques recommandées en termes de stockage de mot de passe (chiffrement, hash, clair, fonction utilisée...) ?*

La syntaxe pour créer le fichier de mot de passe avec Apache 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 demandera le mot de passe pour l'utilisateur. Pour les utilisateurs suivants, on omettra l'option -c pour ne pas écraser le fichier.

Créez un fichier de mots de passe, et ajoutez des utilisateurs. Utilisez le même fichier pour plusieurs utilisateurs.

Q 32 — *Regardez (par cat) le contenu du fichier de mots de passe que vous avez créé. Comment ont été stockés les mots de passe ? Sur quelle longueur ? Cherchez les explications par* `man htpasswd`.

Mettez en place le contrôle de sécurité pour les accès au répertoire `~/www/htdocs`. Pour cela, éditez le fichier `~/www/conf/apache2.conf` et décommentez la ligne
« `# Include ${HOME}/www/conf/tp-auth.conf` ».

Q 33 — *Qu'avez-vous activé exactement dans* `~/www/conf/tp-auth.conf` *? Que se passe-t-il maintenant quand vous accédez à vos pages web ?*

Q 34 — *Examinez la requête de votre navigateur ainsi que la réponse du serveur avec wireshark. Avez-vous vraiment sécurisé l'accès à vos pages ?*

10 Envoi de paramètres par GET au lieu de POST

Ouvrez la page `http://localhost:8080/get-ping.html`. Regardez le code source de cette page et expliquez ce qui se passe en appuyant sur le bouton « valider » :

Q 35 — *Quel script est exécuté ? Comment ce script récupère-t-il l'adresse du site sur lequel il lance ping ?*

11 Faille XSS, ingénierie sociale et requête GET

Q 36 — *Depuis l'ordinateur hébergeant le serveur apache, cliquez ici pour avoir accès à la question suivante. En quoi vous êtes-vous fait avoir, en particulier si le livre d'or était fait sous authentification ?*

L'ingénierie sociale (en anglais "social engineering") consiste à "hacker" un système en profitant de la faiblesse humaine et non technologique. C'est ce que nous avons fait en vous faisant cliquer sur un lien sans regarder vers où il menait. Ces techniques sont souvent utilisées, par exemple par l'envoi de mails se faisant passer pour quelqu'un d'autre (hameçonnage, "phishing") et appelant l'utilisateur à donner ses identifiants, mais des techniques bien plus complexes et ciblées peuvent être mises en oeuvre.

Les deux questions suivantes sortent du cadre du livre d'or et sont plus théoriques.

Q 37 — *Imaginez une attaque XSS du type de la question précédente permettant de faire qu'un administrateur d'un forum supprime un message que vous n'avez les droits de supprimer*

Q 38 — *Imaginez une attaque XSS, combinant l'injection de code javascript et les requêtes GET du style des questions précédentes, permettant de faire qu'un administrateur d'un forum supprime un message sans s'en rendre compte (et sans avoir recours à l'ingéniérie sociale).*

12 Révocation de certificats

Q 39 — *Qu'est-ce que la révocation d'un certificat ? Quels sont les deux protocoles les plus communément utilisés pour vérifier la révocation d'un certificat ?*

Fréquemment, des CA sont compromises et des certificats frauduleux délivrés (eg : Cf. Diginotar 2011). Apple a mis 40 jours avant de pousser un update CRL trivial dans son système d'exploitation mobile iOS.

Q 40 — *Pourquoi de nombreux professionnels de la sécurité considéraient que le risque pour les utilisateurs d'iOS était important pendant cette durée ?*