Leçon 8 : Le serveur Web Apache

8.1 Introduction

Le serveur web *apache* est probablement un des serveurs web les plus déployés sur Internet. Il est présent sur toutes les plateformes (Windows / Unix / Mac OS X) et est un grand classique. Ainsi, les packages LAMP, XAMP ou WAMP utilisent abondamment Apache.

La popularité de PHP a également beaucoup aidé le déploiement d'*apache.* En effet, c'est une des plateformes les plus stables pour installer des sites web PHP. Il faut cependant être honnête, d'autre serveur web, plus léger qu'*apache* ont vu le jour ces dernières années. C'est ainsi que *nginx, lighttp* et bien d'autres viennent compléter le podium.

Enfin, n'oublions pas qu'actuellement, le déploiement d'application .NET nécessite l'utilisation du serveur Web IIS concurrent, proposé par Microsoft.

Dans cette leçon nous allons découvrir la configuration de base d'un serveur web Apache, la possibilité de sécurisation d'un site au moyen d'une authentification HTTP de base et finalement, les éléments à mettre en place pour activer SSL.

8.2 Configuration de Apache

La configuration du service *apache* est localisée dans le fichier `/etc/httpd/conf/httpd.conf` et dans le dossier `/etc/httpd/conf.d/`.

8.2.1 Fichier httpd.conf

Dans le fichier `/etc/httpd/conf/httpd.conf`, nous allons trouver les options *globales* du serveur *apache*.

Option	Explication
`ServerRoot`	Mentionne le chemin vers la configuration du serveur *apache*
`Listen`	Mentionne le port d'écoute utilisé. Le fichier de configuration propose le port 80. Il est possible de spécifier une adresse IP en plus du port pour *limiter* l'écoute du serveur uniquement sur cette IP.
`Include conf.modules.d/*.conf`	Cette directive va lire tous les fichiers portant l'extension `.conf` et les importer dans la configuration. Le dossier `conf.modules.d` contient principalement des fichiers comprenant les directives `LoadModule`. Ces directives permettent d'activer des extensions au niveau du serveur *apache*. Ainsi, si le langage PHP est installé, une directive `LoadModule` est nécessaire pour charger la librairie en question. Cette directive se trouve dans le fichier `/etc/httpd/conf.modules.d/10-php.conf`
`User apache` **`Group apache`**	Cette option détermine l'utilisateur et le groupe avec lequel le service *apache* est démarré. Cela implique que le service *apache* devra accéder aux fichiers HTML et dossiers web avec cet utilisateur. Il convient donc de fixer les droits correctement.

© Louis SWINNEN 2020, tous droits réservés

ServerAdmin	Mentionne l'adresse mail de l'administrateur du serveur. Cette adresse peut être présente sur les pages d'erreur.
ServerName	Mentionne le nom DNS du serveur *apache*. Si aucun service DNS n'est relié à ce serveur web, il est nécessaire de mentionner l'adresse IP du serveur.
DocumentRoot	Indique l'emplacement principal des pages web. Ainsi, les pages web sont placées dans le dossier `/var/www/html`.
<Directory ...> **...** **</Directory>**	L'option Directory permet de préciser des options, autorisations ou restrictions d'accès au dossier mentionné. On trouve, très souvent, la directive `Options` qui permet, notamment *de lister le contenu du dossier* (via `Indexes`) ou encore de *suivre les liens symboliques* (via `FollowSymLinks`). La directive `AllowOverride` permet d'indiquer si les fichiers cachés `.htaccess` qui peuvent être présent dans les dossiers web doivent être pris en compte ou pas. Les fichiers `.htaccess` permettent de modification la configuration locale du serveur et d'activer une authentification. Pour activer la prise en charge complète, il faut mentionner `AllowOverride All`. La directive `Require` qui permet de protéger l'accès à un dossier. Nous privilégierons l'utilisation des fichiers `.htaccess` pour arriver à cette configuration.
AddDefaultCharset	Permet de mentionner le codage de caractère par défaut utilisé par le serveur Web.

Il existe encore de nombreuses options qu'il est possible de définir dans ce fichier. Je vous renvoie à la page de manuel pour une présentation plus complète.

8.2.2 Le dossier conf.d

Le dossier `/etc/httpd/conf.d/` permet de configurer les sites web souhaités. En effet, tous les fichiers présents se terminant par `.conf` sont pris en charge par *apache* comme élément de configuration.

Une des options importantes est VirtualHost. Grâce à cette option, il est possible d'héberger plusieurs sites web sur le même serveur *apache*.

Ainsi, pour créer le site web par défaut du serveur, nous pourrions créer un fichier `/etc/httpd/conf.d/default-site.conf` (le nom doit juste se terminer par `.conf`). A titre d'exemple, voici la configuration pour celui de HELMo :

```
<VirtualHost 192.168.3.206:80 [2001:6a8:2cc0:8000::206]:80>
        ServerName      project.helmo.be
        ServerAlias     webmail.helmo.be
        DocumentRoot    /var/www/html/default
        <Directory "/var/www/html/default">
                AllowOverride All
                Options FollowSymLinks
                Require all granted
```

© Louis SWINNEN 2020, tous droits réservés

```
        </Directory>

        ErrorLog logs/error_log
        CustomLog logs/access_log combined
</VirtualHost>
```

Dans cet exemple, nous déclarons *un hôte virtuel* attaché à l'adresse IPv4 `192.168.3.206` et l'adresse IPv6 `2001:6a8:2cc0:8000::206` sur le port TCP `80`.

Le nom DNS associé au site web (via `ServerName`) est `project.helmo.be`. C'est de cette manière qu'il est possible de définir plusieurs site web sur le même serveur, en utilisant des noms DNS différents et donc, des `ServerName` distincts. La directive `ServerAlias` permet de mentionner d'autres noms pour le même site comme webmail.helmo.be dans cet exemple.

La directive `DocumentRoot` mentionne l'emplacement des fichiers HTML et/ou PHP de ce site.

Comme déjà abordé dans les options globale d'Apache, la directive `Directory` permet d'adapter, modifier, restreindre l'accès au dossier précisé (ici `/var/www/html/default`). A l'intérieur de cette directive, nous avons l'option `AllowOverride All` qui permet de prendre en compte les fichiers `.htaccess` qui se trouveraient avec les pages HTML (dans `/var/www/html/default` donc), la mention `FollowSymLinks` qui permet de suivre les liens symboliques s'ils sont présents et finalement, l'option `Require` qui permet à tout le monde d'accéder à ce site.

Enfin, les directives `ErrorLog` et `CustomLog` contrôlent les informations placées dans les fichiers journaux d'*apache*. Ils sont situés dans `/var/log/httpd`. Le premier indique les erreurs rencontrées (ce fichier est une mine d'or lorsqu'on développe son application web et que celle-ci ne fonctionne pas correctement) tandis que le second reprend la liste des accès au site web (qui est intéressant pour les statistiques).

8.2.3 Activation de SSL

La manière la plus simple d'activer SSL sur un site web *apache* est de modifier le fichier `/etc/httpd/conf.d/ssl.conf`. Avant de commencer, il est nécessaire de disposer du certificat SSL associé au nom DNS du site dans le dossier `/etc/pki/tls/certs` (ainsi que les éventuels certificats intermédiaires) et de la clé privée correspondante dans `/etc/pki/tls/private`.

Si l'on souhaite activer SSL pour le site web par défaut, on peut se contenter de modifier le `VirtualHost _default_:443` présent dans le fichier de configuration. On retrouve, sans surprise, les directives de configuration que nous venons de voir pour `VirtualHost`. Ainsi, on peut utiliser `DocumentRoot`, `ServerName`, Directory, `ErrorLog` et `CustomLog` comme ci-dessus.

A ces options, nous avons des directives propres à SSL.

Option	Utilisation
SSLEngine	Active le support SSL pour le site
SSLProtocol	Détermine les protocoles supportés. Il convient d'adapter la valeur par défaut afin de désactiver des protocoles complètement obsolètes : `SSLProtocol all -SSLv2 -SSLv3`

© Louis SWINNEN 2020, tous droits réservés

SSLCipherSuite[26]	Détermine les algorithmes de chiffrement qui peuvent être utilisés. Ici aussi, il vaut mieux modifier l'option par défaut : `SSLCipherSuite "EECDH+ECDSA+AESGCM EECDH+aRSA+AESGCM EECDH+ECDSA+SHA384 EECDH+ECDSA+SHA256 EECDH+aRSA+SHA384 EECDH+aRSA+SHA256 EECDH+AESGCM EECDH EDH+AESGCM EDH+aRSA HIGH !MEDIUM !LOW !aNULL !eNULL !LOW !RC4 !MD5 !EXP !PSK !SRP !DSS"`
SSLHonorCipherOrder	Cette option détermine s'il faut respecter l'ordre dans lequel les algorithmes sont précisés. Il vaut mieux activer cette option : `SSLHonorCipherOrder on`
SSLCertificateFile	Mentionne le chemin vers le certificat du serveur. Le nom inclut dans le certificat doit correspondre au nom DNS du serveur pour éviter une erreur de connexion depuis le navigateur. `SSLCertificateFile /etc/pki/tls/certs/swilabus.be.crt`
SSLCertificateKeyFile	Mentionne le chemin vers la clé privée associée à ce certificat. `SSLCertificateKeyFile /etc/pki/tls/private/swilabus.be.key`
SSLCertificateChainFile	Mentionne éventuellement le chemin vers les certificats intermédiaires fournis par l'autorité de certification. `SSLCertificateChainFile /etc/pki/tls/certs/alphaSSL.crt`

8.3 Sécurisation d'un site web avec une authentification simple

Il est facile d'activer une authentification simple avec *apache* sur un site ou une partie d'un site web existant. Pour ce faire, il faut :

1. Activer la prise en charge des fichiers `.htaccess` sur le site en question
2. Créer un fichier `.htaccess` à la racine du dossier à protéger avec un contenu déterminé
3. Créer le fichier des utilisateurs spécifiant les logins et mots de passe autorisés.

L'activation de la prise en charge des fichiers `.htaccess` a déjà été abordée précédemment avec l'option `AllowOverride`. Ainsi, si nous désirons protéger toutes les pages se trouvant dans le dossier `/admin`, par exemple, il faut créer un fichier `.htaccess` dans le dossier `admin` du site web. Ce fichier devra contenir les éléments suivants :

```
AuthType Basic
AuthName "Message affiché par le navigateur"
AuthBasicProvider file
AuthUserFile "/var/www/admin.passwd"
Require valid-user
```

Il faut encore créer le fichier `admin.passwd` mentionné dans le fichier `.htaccess`. Ce fichier peut être créé à l'aide de la commande `htpasswd` fournie par Apache :

```
$ htpasswd -B -c /var/www/admin.passwd admin
```

*Cette commande **crée <u>un nouveau fichier</u>** (`-c`) de mot de passe nommé `admin.passwd` en chiffrant le mot de passe de manière sûr (`-B`) et ajoute le login admin. Le mot de passe sera demandé pour compléter l'ajout de l'utilisateur.*

```
$ htpasswd -B /var/www/admin.passwd godswila
```

[26] Cette liste est extraite de nombreuses sources comme celle-ci (! elle est souvent mise à jour !): https://raymii.org/s/tutorials/Strong_SSL_Security_On_nginx.html

© Louis SWINNEN 2020, tous droits réservés

*Cette commande **ajoute** au fichier mentionné admin.passwd le login godswila en utilisant un chiffrement sûr (-B). Le mot de passe sera demandé pour compléter l'ajout de l'utilisateur.*

A l'inverse du service *samba* qui nécessitait qu'un compte UNIX du même nom existe, avec `htpasswd`, il est possible d'ajouter des logins n'existant pas sur le système.

Attention ! Les mots de passe circulent en clair entre le navigateur et le serveur. Sans SSL, cette option ne permet pas réellement de sécuriser un site puisqu'un espion peut facilement intercepter le mot de passe entré.

8.4 Site web personnel des utilisateurs

Une autre options intéressantes d'*apache* est de permettre, facilement, à des utilisateurs du système de disposer d'un site web personnel. Cette option est déployée à HELMo pour permettre aux étudiants d'avoir leurs sites web accessibles via `http://192.168.128.13/~login`.

Cette directive s'appelle `UserDir` et est configurable dans le fichier `/etc/httpd/conf.d/userdir.conf`. Ainsi à HELMo, la directive suivante est activée :

```
UserDir public_html
```

Cette directive mentionne que tous les utilisateurs peuvent créer un site web personnel. Ce site doit être hébergé dans le dossier `public_html` placé dans leur espace personnel (et donc dans le dossier suivant `/home/login/public_html`).

Attention ! Comme nous l'avons vu, le service *apache* est lancé par un utilisateur particulier, l'utilisateur *apache*. Il est **absolument nécessaire que l'utilisateur *apache* puisse atteindre le dossier `public_html` et lire tous les fichiers** qui s'y trouvent. Dans le cas contraire, une erreur 403 pourrait être retournée par le serveur.

8.5 Exercices

On vous demande de configurer *apache* :

1. Et placer une page web de votre création *sur le site web par défaut*
2. Et autoriser les utilisateurs à déployer leurs sites web personnels. Les fichiers seront placés dans un dossier `web` présent dans leur répertoire personnel.
3. Et créer un dossier `admin` dans le site web par défaut. Ajoutez-y une page web particulière. Protégez le site web au moyen d'une authentification simple et ajouter l'utilisateur `letmesee` et le mot de passe `yes1CanREAD`. Utilisez Wireshark pour capturer le mot de passe.
   !! Vérifier votre configuration précisément !!
4. Créer un script PERL permettant d'ajouter les utilisateurs dans le fichier `htpasswd` protégeant votre site web `admin`.
5. Configurer un site SSL `intranet.swilabus.be` en récupérant le certificat mis à votre disposition sur la page accompagnant la leçon. Vérifiez que la connexion sécurisée s'établit correctement en utilisant un navigateur depuis votre machine Client pour y accéder.

© Louis SWINNEN 2020, tous droits réservés