Module 7 : Configuration du serveur WEB Apache

Introduction

Ce sont les gens du CERN (centre européen de recherche nucléaire) qui ont développé le concept de serveur et client HTTP. Une fois leur travail de recherche terminé, ils ont confié cela à une université américaine (NSCA). Apache est le nom d'un projet libre de serveur WEB. Le nom apache a une origine un peu cocasse , certains disent que cela vient de "**a patchy** server" à cause des nombreuses « patches du début », d'autres disent d'une manière plus sérieuse que les instigateurs de ce projet ont pris ce nom en mémoire des Apaches pour leur grande adaptabilité sur le terrain. Ce serveur est le plus utilisé sur Internet.

D'après netcraft (http://news.netcraft.com/archives/category/web-server-survey/) d'octobre 2014 , le serveur apache est utilisé à 37.45% suivi de près par le serveur IIS de Microsoft. En fait, en juillet 2014, le serveur IIS de Microsoft était le serveur le plus utilisé avec 1% de plus que Apache. C'était une première en 20 ans!!. Un serveur WEB est le programme "serveur" répondant aux requêtes de clients web comme par exemple un browser WEB (Internet explorer).

Installation

yum install httpd

Site WEB

Le meilleur site WEB concernant les dernières versions d'Apache se situe à l'adresse suivante :
http://www.apache.org

Emplacement des répertoires importants

- L'emplacement du répertoire qui contient, entre autre, le fichier de configuration « httpd.conf » :

  « /etc/httpd/conf »

- L'emplacement du répertoire qui contient les pages WEB qui ont trait au site WEB : « /var/www/html »

Fichier de configuration « httpd.conf »

Avant la version 2 du serveur Apache, il y avait 3 fichiers de configuration reliés au fonctionnement d'Apache. Avec l'arrivée de la version 2, toutes les directives de configurations ont été centralisées dans un seul et même fichier qui se nomme « **httpd.conf** ».

Voici un extrait d'un fichier httpd.conf :

```
#----------------- Name Space and Server Settings --------------------
# In this section, you define the name space that users see of your http
# server.  This file also defines server settings which affect how requests
are
# serviced, and how results should be formatted.
# This used to be a separate file. Now part of httpd.conf
# (srm.conf -- Apache HTTP server configuration file)
#

# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.

DocumentRoot /var/www/html

# UserDir: The name of the directory which is appended onto a user's home
# directory if a ~user request is recieved.

UserDir public_html

# DirectoryIndex: Name of the file or files to use as a pre-written HTML
# directory index.  Separate multiple entries with spaces.

DirectoryIndex index.html index.php index.htm index.shtml index.cgi
Default.htm default.htm index.php3

# FancyIndexing is whether you want fancy directory indexing or standard

FancyIndexing on

# AddIcon tells the server which icon to show for different files or filename
# extensions
```

Évidemment les lignes commençant par le caractère « # » sont des commentaires et ne sont pas traitées par le serveur.

Arrêt et démarrage du serveur

Pour arrêter le serveur :

systemctl stop httpd

Pour démarrer le serveur:

systemctl start httpd

Pour redémarrer le serveur après une modification au fichier httpd.conf:

systemctl restart httpd

Le fichier « httpd.conf »

Voici quelques directives qui peuvent être intéressantes d'utiliser dans le fichier de configuration du serveur WEB Apache.

Directive	Description
ServerType	Le type de serveur qui gèrera la réponse à la requête au serveur WEB. Il peut être soit : standalone (la plupart des cas) xinetd (C'est le démon xinetd qui s'occupera de répondre à la requête)
ServerRoot	Sert à configurer le chemin qui indique l'emplacement des fichiers de configuration du serveur. (typiquement : /etc/httpd)
Port	Le numéro du port sur lequel écoutera le serveur pour l'arrivée d'une requête. Le défaut est 80.
ServerAdmin	L'adresse de courier électronique de celui ou celle qui maintient le site WEB.
ServerName	C'est le nom que le serveur retournera au client. Cette valeur doit être un nom valide de domaine ou si vous n'avez pas de domaine, le nom de la machine. Ex : ServerName Kalimantan Kalimantan est le nom de la machine
DocumentRoot	Indique le répertoire qui contient les fichiers html pour le serveur.

	Le défaut est : /var/www/html
UserDir	Indique le nom du répertoire qui servira ou usager ordinaire pour pouvoir faire des page WEB. Par exemple : Si la directive est : UserDir public_html alors je peux créer un répertoire nommé « public_html » dans mon répertoire maison et y mettre les fichiers html. Par exemple, si mon répertoire maison est « /home/stephane » alors je peux créer un répertoire « /home/stephane/public_html » et y mettre mes pages WEB. On accedera au pages WEB de mon compte en tapant : http://nom_machine/~stephane
DirectoryIndex	Indique quelle est la liste des fichiers qui seront utilisés comme fichier de page d'accueil par défaut. Le défaut est : index.html
Listen	Elle permet de mentionner au serveur Apache d'écouter sur une adresse IP et un port en particulier pour l'arrivée d'une requête HTTP. Par exemple, pour signifier au serveur que l'on veut que ce dernier écoute le port 8008 sur l'adresse IP 172.16.29.99, on inscrira : Listen 172.16.29.99:8008

Les blocs <Directory /> </Directory>

Chaque répertoire qui fait partie de l'arbre hiérarchique d'un site WEB peut se voir attribuer des options de configuration par défaut. Il s'agit de configurer le répertoire pour lequel vous voulez attribuer des options en ajoutant les options dans un bloc <Directory /> et </Directory>.

Par exemple, dans le fichier de configuration de « httpd.conf », vous avez un bloc définit par défaut comme ci-dessous :

```
<Directory />
        AllowOverride    None
        Require all denied
</Directory>
```

Ce sont les options par défaut qui vont être attribuées aux différents répertoires du site WEB.

Lorsque vous tapez : http://localhost dans un navigateur, ce dernier va chercher un fichier qui se nomme « index.html » qui se trouve dans le répertoire « /var/www/html ».

Directive « Options »

Ce sont les options qu'il peut être possible de faire dans le répertoire. Les paramètres possibles sont les suivants :

None	Aucune options n'est active pour ce répertoire.
All	Toutes les options seront actives pour ce répertoire.
Indexes	Cette option permet de faire afficher le contenu du répertoire si il n'y a pas de fichier « index.html » qui y est présent.
Includes	Ce répertoire admet les « server-side include » (SSI).
FollowSymLinks	Cette option permet de suivre les liens symboliques qui pourrait pointer sur d'autres répertoires.
ExecCGI	Les scripts CGI sont permit dans ce répertoire.

On peut restreindre l'accès à un répertoire en utilisant les options « Allow » et « Deny ». Ceux-ci permettent d'accéder à un répertoire ou de s'y voir refuser l'accès selon l'adresse IP du client.

Exemple :

J'ai un répertoire nommé « Films » qui se trouve dans le répertoire de base « /var/www/html ». J'aimerais protéger ce répertoire en permettant l'accès aux adresses IP commençant par 205.237.246. Voici comment y arriver :

<Location /Films>
  # Les options générales possibles ici avec la directives **Options**
  order deny,allow
  Deny from all
  Allow from 205.237.246
</Location>

La ligne « order deny,allow » permet d'exécuter le deny en premier suivit du Allow.

Ainsi, toutes les adresses IP se verront refuser l'accès puisque c'est le Deny avant qui sera exécuté en premier et seule les adresses IP commençant par 205.237.246.* (où l'étoile veut dire l'ensemble des chiffres de 0 à 255) pourront accéder au répertoire « Films ».

Remarquez la directive « Location » qui permet de définir l'emplacement d'un répertoire en particulier.

Autre exemple :

Admettre toutes les adresses IP sauf les adresses 172.17.113.75 et 172.17.113.77

<Location /Films>
  # Les options générales possibles ici avec la directives **Options**
  order allow,deny
  Allow from all
  Deny from 172.17.113.75 172.17.113.77
</Location>

Restreindre l'accès à un répertoire avec authentification par mot de passe.

Si on veut rendre les choses plus sécuritaire, on peut demander un à usager de donner un mot de passe avant de pouvoir accéder à un répertoire.

Vous aurez alors une boîte de dialogue qui vous demandera le nom d'usager et le mot de passe. Ces deux informations doivent être exacte pour pouvoir accéder au répertoire.

Voici un exemple :

Je veux donner accès au répertoire Films en utilisant une boîte de dialogue pour l'accès par mot de passe. Le nom d'usager sera « user » et le mot de passe sera « yahoo ».

```
<Location  /Films>
  #Autres options ici
  AuthType   Basic   #Authentification  de base, c'est à dire sans encryption.
  AuthName   « Acces au repertoire Films »
  AuthUserFile   « FichierUsager »
  Require   Valid-user
</Location >
```

AuthType	C'est le type d'authentification. Il y en a deux. Le mode « Basic » envoi le mot de passe sans encryption alors que le mode « Digest » envoi les informations de façon encryptée.
AuthName	C'est la chaîne qui sera affichée comme titre du « realm » lors de la demande du mot de passe et du nom d'usager.
AuthUserFile	Cette directive permet de définir l'emplacement du fichier qui contient les usagers qui peuvent accéder à ce répertoire ainsi que leur mot de passe.
Require	Exige que les informations entrées (usager et mot de passe existe dans le fichier).

Création du fichier pour la directive AuthUserFile

Il est assez simple de créer le fichier qui reçoit les noms d'usager et les mots de passe. Il s'agit d'utiliser la commande « htpasswd »

Par exemple :

Je veux créer un fichier nommé « UserFile » avec comme usager « schasse » et comme mot de passe « yahoo ». Cette authentification doit être demandée lorsque j'accède au répertoire /var/www/html/personnel

```
<Location  /personnel>
AuthType   Basic
AuthName  « Repertoire personnel »
AuthUserFile « /etc/UserFile »
Require   Valid-user
</Location>
```

La création du fichier se réalise avec la commande suivante :

En étant dans le répertoire /etc

```
htpasswd  -c  UserFile  schasse   yahoo
```

Si je veux ensuite ajouter des usagers dans ce fichier, je n'ai pas besoin d'utiliser le paramètre –c. Ainsi, si j'ajoute l'usager « Meiliana » avec un mot de passe « MeiLian », la commande htpasswd à faire est :

```
htpasswd  Meiliana    MeiLian
```

Ceci permet d'ajouter l'usager dans le fichier déjà existant.

Combinaison des types de restriction

On peut combiner la restriction sur les adresses IP avec l'authentification par mot de passe. Voici un exemple :

```
<Location /un_Repertoire_a_proteger>
  order  deny,allow
  Deny from all
  Allow from  205.237.246
  AuthType  Basic
  AuthName  « Repertoire proteger »
  AuthFileName  « UserFile »
  Require   Valid-user
  Satisfy  Any
</Location>
```

Dans l'exemple précédent, la restriction sur les adresses IP est effectuée pour toutes les adresses sauf celles qui viennent des adresses commençant par « 205.237.246 ».

La directive « Satisfy » est la directive importante ici. Elle permet de procéder à une authentification par mot de passe à condition que l'adresse IP ne soit pas acceptée.

Satisfy Any Accepte la connexion si l'usager vient d'une adresse IP valide, c'est-à-dire, de clg.qc.ca OU que l'information pour l'authentification est entrée correctement.

Satisfy All Les deux types d'authentification (par adresse IP ET par mot de passe) doivent être satisfait.

Exemple :

Une requête arrive de l'adresse « 205.237.246.219 ».

Est-ce que l'adresse IP satisfait la directive Allow from ?

OUI	NON
On laisse entrer l'usager dans le répertoire sans demander de mot de passe.	On procède à authentifier l'usager par mot de passe.

Serveur WEB Virtuel

Il existe un moyen relativement simple pour simuler plusieurs serveurs WEB à même un seul serveur. C'est ce qui permet, entre autre, de faire de l'hébergement de site WEB. Il existe plusieurs méthodes pour y arriver dont une consiste à utiliser les « serveurs virtuels nommés » et l'autre à utiliser plusieurs adresses IP. Il y a aussi la possibilité de définir des serveurs virtuels et utilisant des ports différents sur lesquels Apache écoutera.

Nous allons voir 2 méthodes qui permettent d'implanter des serveurs virtuels. La première méthode avec les serveurs « nommés » et la deuxième en définissant des ports. C'est la méthode des ports sur laquelle nous allons vous demander de vous concentrer. Non pas que la méthode des serveurs nommés n'est pas bonne mais la méthode par port est plutôt celle que vous allez implanter en laboratoire ☺

Serveur virtuel nommé

Pour implanter ce type de serveur virtuel, le fichier « httpd.conf » comportera les directives suivantes :

```
NameVirtualHost   Votre_adresse_IP

<VirtualHost   Votre_Adresse_IP>
  ServerName   Nom_qui_identifie_ce_serveur
  ServerRoot   /etc/httpd
  #Autres directives
  DocumentRoot   /var/www/Nom_serveur/html/
</VirtualHost>
```

Chaque directive VirtualHost comporte une directive ServerName qui identifie le nom du serveur. La directive DocumentRoot quant à elle identifie l'emplacement de base du fichier de la page d'accueil (index.html) de ce serveur. Dans l'exemple précédent, le fichier « index.html » est dans le répertoire « /var/www/Nom_serveur/html » où « Nom_serveur » est le nom réel de votre machine ☺

On peut aussi retrouver, dans un bloc <VirtualHost> … </VirtualHost>, les directives propres à un répertoire en particulier. On peut donc ajouter des options sur les répertoires de ce serveur spécifiquement.

Exemple :

J'ai un serveur virtuel nommé « Stephane » et j'aimerais protéger mon répertoire nommé « /var/www/Stephane/html/trucs » en acceptant seulement les adresses IP commençant par 205.237.246. Le bout de code pour réaliser cette astuce est en caractère gras.

À supposez que mon adresse IP est : 205.237.246.219

```
NameVirtualHost   205.237.246.219

<VirtualHost  205.237.246.219>
  serveradmin  root@localhost
  ServerRoot   /etc/httpd
  ServerName  Stephane
  DocumentRoot  /var/www/Stephane/html

  <location /trucs>     #ici, on ne met que /trucs car le caractère « / » est remplacé
                        par ce qui a été donné à la directive « DocumentRoot ».
     order Deny,Allow
     Deny from  all
     Allow from  205.237.246
  </location>
</VirtualHost>
```

Serveur virtuel avec adresse IP et port

Avec ce type de serveur virtuel, nous n'avons pas besoins de DNS pour pouvoir accéder correctement depuis une machine distante à votre serveur (En autant que la machine distante est dans le même réseau que votre serveur). Nous devons cependant utiliser l'adresse IP du serveur sur lequel vous voulez accéder à la page WEB. Par contre, la petite différence ici est qu'on ajoutera en plus, à la fin de l'adresse IP, la séquence « :#port » où #port est le numéro de port pour lequel on veut accéder à la page du serveur virtuel.

Voici un exemple d'une configuration de ce genre dans le fichier « httpd.conf » :

Supposons que la machine de cet exemple porte l'adresse IP 172.16.29.250 et que l'on veut créer deux serveurs virtuels sur deux ports différents(les numéros de port 80 et 7000)

Les deux « listen » ci-dessous vont habituellement dans la partie « globale » des définitions.

```
Listen  172.16.29.250:80
Listen  172.16.29.250:7000

.
.  La continuité des directives du fichier « httpd.conf » ici
.

# Vous n'avez pas besoin de la directive NameVirtualHost si vous utilisez les adresses IP avec
#ports

<VirtualHost   172.16.29.250:80>
  ServerName   Nom_qui_identifie_ce_serveur # Pas obligatoire
  ServerRoot   /etc/httpd
  DocumentRoot   /var/www/répertoire_pour_le_port_80/html/
  #Autres directives ici
</VirtualHost>

<VirtualHost   172.16.29.250:7000>
  ServerName   Nom_qui_identifie_ce_serveur # Pas obligatoire
  ServerRoot   /etc/httpd
  DocumentRoot   /var/www/répertoire_pour_le_port_7000/html/
  #Autres directives ici
</VirtualHost>
```

où :

répertoire_pour_le_port_# signifie que vous devez avoir créer un répertoire qui identifie le serveur en question. Ce nom n'a aucune espèce d'importance mais on lui donne un nom qui est habituellement relatif à ce qui se trouve dans ce site.

Ce qui est important, c'est que les deux répertoires portent des noms « **différents** » car peu importe l'adresse IP et le port qui sera entré, si les noms de répertoires sont identiques, c'est la même page WEB qui sera affichée!!

Comment accède-t-on au page WEB de nos serveurs avec AddresseIP et Port ?

Dans votre navigateur préféré, on tape

172.16.29.250:80
pour accéder à la page WEB du serveur virtuel sur le port 80 et on tape

172.16.29.250:7000
pour accéder à la page WEB du serveur virtuel sur le port 7000.

Documentation plus approfondie sur le sujet

À l'adresse : http://httpd.apache.org/docs/vhosts/index.html