Installation et présentation du serveur Apache 2
Installation d'un serveur HTTP

Installation et présentation du serveur Apache 2

Ce chapitre donne un aperçu des fonctions et de l'environnement du serveur Apache 2. Vous pourrez retrouver tous les aspects développés dans la documentation (en partie en français) du produit à l'adresse suivante apache.org.

Installation du serveur Apache 2

Apache 2 a très certainement été installé par défaut lors de l'installation de votre Debian. Pour le vérifier : **dpkg -l | grep apache2**

```
ii apache2
ii apache2-common
ii apache2-mpm-prefork
ii apache2-utils
ii libapache2-mod-perl2
ii libapache2-mod-php4 (ou  libapache2-mod-php5)
```

Si Apache2 n'est pas installé, la commande : **apt-get install apache2** installera le serveur web avec ses dépendances.

Vous aurez certainement besoin par la suite du module php alors autant l'installer tout de suite : **apt-get install libapache2-mod-php4 (ou apt-get install libapache2-mod-php5)**

Si vous voulez installer la documentation en local : **apt-get install apache2-doc**

Présentation de l'environnement

Le serveur HTTP Apache2 est un programme modulaire. Mise à part quelques modules directement intégrés dans le programme binaire httpd, l'administrateur peut choisir les fonctionnalités qu'il souhaite en activant des modules.

De même, il existe plusieurs fichiers de configuration tous présents dans `/etc/apache2/`.

- Le fichier de configuration principal est `/etc/apache2/apache2.conf` Il contient les paramètres généraux et communs à tous les serveurs et plusieurs "Include" vers les autres fichiers.
- Le fichier de configuration `/etc/apache2/ports.conf` contient la liste des ports en écoute.
- On trouve tous les fichiers concernant les modules dans le répertoire `/etc/apache2/mods-available/`.

  On y trouve deux catégories de fichiers : `*.load` et `*.conf`

  Les fichiers avec *l'extension load* charge effectivement les modules dynamiques :

  ```
  cat /etc/apache2/mods-available/userdir.load

  LoadModule userdir_module /usr/lib/apache2/modules/mod_userdir.so
  ```

Les fichiers avec *l'extension conf* sont les fichiers de configuration des modules :

```
cat /etc/apache2/mods-available/userdir.conf

<IfModule mod_userdir.c>
 UserDir public_html
 UserDir disabled root

  <Directory /home/*/public_html>
    AllowOverride FileInfo AuthConfig Limit
    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
  </Directory>
</IfModule>
```

On trouve les fichiers concernant les modules **activés** dans le répertoire `/etc/apache2/mods-enabled/` : ce sont uniquement ces fichiers qui sont inclus dans le fichier de configuration principal par les directives :

```
Include /etc/apache2/mods-enabled/*.load
Include /etc/apache2/mods-enabled/*.conf
```

Et ces fichiers sont en fait des liens qui pointent vers les fichiers de `/etc/apache2/mods-available`

Pour activer un module (ce qui revient donc à créer le lien), il est pratique d'utiliser la commande suivante : **a2enmod mod_userdir**. Mais on peut bien évidemment créer le lien "à la main".

- On trouve tous les fichiers de configuration des serveurs web `/etc/apache2/sites-available/`

  On trouve les fichiers de configuration des sites web **activés** dans le répertoire `/etc/apache2/sites-enabled/` : ce sont uniquement ces fichiers qui sont inclus dans le fichier de configuration principal par la directive :

  ```
  Include /etc/apache2/sites-enabled/[^.#]*
  ```

  Et ces fichiers sont en fait des liens qui pointent vers les fichiers de `/etc/apache2/sites-available`

  De même que pour les modules, *pour activer un site*, il existe une commande : **a2ensite fichier_conf**. "fichier_conf" étant un fichier de configuration présent dans `/etc/apache2/sites-available/`

La documentation est dans `/usr/share/doc`.

Les journaux sont dans `/var/log/apache2/`.

Le script de lancement du service serveur est dans `/etc/init.d`

Pour que le serveur Web puisse répondre à une demande d'un client, il doit être démarré : **/etc/init.d/apache2 start**.

Apache2 ne fonctionne qu'en standalone (la directive *ServerType* n'est donc plus reconnu).

Installation d'un service minimum

Ce paragraphe décrit les principaux paramètres pour mettre en place un service HTTP minimum.

Prenez la peine de faire une sauvegarde des fichiers de configuration avant de procéder à toutes modifications (par exemple : **cp -rp /etc/apache2 /etc/apache2.init**).

Les modifications ne seront prises en compte que si les fichiers de configurations sont relus **/etc/init.d/apache2 reload** ou si le service est redémarré **/etc/init.d/apache2 restart**.

Vous pouvez vérifier la syntaxe des fichiers de configuration par la commande : **apache2 -t**

Comment retrouver rapidement le fichier dans lequel se trouve une directive ? :

```
grep -ni "documentroot" /etc/apache2/*
ne renvoie rien...

grep -ni « documentroot » /etc/apache/*/*
sites-available/default:5:      DocumentRoot /var/www
sites-enabled/000-default:5:    DocumentRoot /var/www
```

L'option « ni » permet de chercher sans tenir compte de la casse et de renvoyer aussi le numéro de la ligne. L'astérisque peut être remplacé par un nom de fichier.

Dans le fichier */etc/apache2/ports.conf*

- *Listen 80*, indique quel est le port utilisé par le service (par défaut 80). Il est possible d'utiliser un autre port, par contre vous devrez spécifier au navigateur quel est le port utilisé par le serveur. Si vous configurez par exemple le port 8080 (*Listen 8080*) sur une machine `www.MonDomaine.edu`, vous devrez spécifier dans le navigateur `www.MonDomaine.edu:8080`, pour que le serveur reçoive et traite votre requête.

Dans le fichier */etc/apache2/apache2.conf*

- *user www-data* et *group www-data*, spécifient le compte anonyme utilisé par le serveur une fois qu'il est lancé. En effet, pour accéder aux ports inférieurs à 1024, le serveur utilise un compte administrateur, ce qui présente des dangers. Une fois le processus actif, il utilisera l'UID d'un autre compte (ici www-data). Ce compte doit pouvoir lire les fichiers de configuration et ceux de la racine du serveur HTTP (attention donc aux droits sur les pages web desservies). D'autres distributions utilisent le compte “nobody” ou “apache”
- ServerRoot `/etc/apache2`, indique l'adresse du répertoire racine du serveur, où sont stockés les fichiers de configuration du serveur HTTP. Cette adresse peut être modifiée.
- PidFile `/var/run/apache2.pid`, indique le fichier où le serveur en exécution stocke son numéro de processus (PID).
- ErrorLog, fichier `error_log`, journalisation des erreurs. L'adresse est calculée à partir de ServerRoot. Si ServerRoot est `/etc/httpd` et ErrorLog logs/error_log, le chemin complet est `/var/log/apache/logs/error_log`.

Il est fréquent d'héberger plusieurs serveurs web sur un même poste aussi la **déclaration et le paramétrage des différents serveurs est déportée dans des fichiers à placer dans /etc/apache2/sites-available/**. **Le fichier default** y est déjà présent pour assurer le paramétrage du site principal par défaut dont la racine se trouve, toujours par défaut à */var/www/*.

Le site par défaut est déjà activé ; on retrouve donc un lien vers ce fichier dans */etc/apache2/sites-enabled*.

Sinon, le principe est le suivant :

- On crée un fichier de configuration (appellé conf_site) pour un site web spécifique dans `/etc/apache2/sites-available/`.
- On active ce fichier de configuration par la commande : **a2ensite conf_site** ; cette commande a pour effet de créer un lien dans `/etc/apache2/sites-enabled/` qui pointe vers `/etc/apache2/sites-available/conf_site`.

De plus, **il est aussi possible de faire exécuter plusieurs instances d'Apache** en spécifiant un fichier de configuration particulier par une commande du style : **apache -f fichier_config** où fichier_config est le nom du fichier de configuration, en chemin absolu (sinon il est considéré comme situé relativement à la directive *ServerRoot*, c'est-à-dire */etc/apache2*, par défaut).

Les directives qui suivent correspondent à des serveurs spécifiques et sont donc incluses dans les fichiers de configuration présents dans `/etc/apache2/sites-available/` , notamment celui par défaut `/etc/apache2/sites-available/default`.

- *ServerAdmin webmaster@localhost*, précise quel est le compte qui reçoit les messages. Par défaut un compte spécifique administrateur de site web (à modifier pour une adresse comme *`root@MonDomaine.edu`*).
- *ServerName www.MonDomaine.edu*, indique le nom ou l'alias avec lequel la machine est désignée. Par exemple, l'hôte `ns1.MonDomaine.edu`, peut avoir le nom d'alias `www.MonDomaine.edu`. Ce nom doit correspondre à une adresse IP, donc être renseigné dans un serveur DNS (ou dans un premier temps mais à éviter en production dans un fichier hosts sur le cient). S'il n'est pas défini, alors le serveur tentera de le résoudre à partir de sa propre adresse IP. Voir la résolution de nom avec un DNS.
- *DocumentRoot /var/www*, indique l'emplacement par défaut des pages HTML quand une requête accède au serveur. Exemple : la requête http://www.MonDomaine.edu/index.html pointe en fait sur le fichier local /var/ww/index.html sauf si le répertoire est pointé par une directive tel que Alias (voir ci-après).
- Alias /CheminVu/ /CheminRéel/, par exemple : "/icons/ /usr/share/apache/icons/", ce paramètre permet de renommer, à la manière d'un lien logique, un emplacement physique avec un nom logique.

  Exemple: vous voulez que www.MonDomaine.edu/test/index.html, ne corresponde pas physiquement à un répertoire sur la racine du serveur HTTP (défini par la directive précédente DocumentRoot) mais à un emplacement qui serait /usr/local/essai. Vous pouvez mettre dans le fichier de configuration d'Apache un alias de la forme: alias /test/ /usr/local/essai/
- *ScriptAlias /cgi-bin/ /usr/lib/cgi-bin*, de la forme "ScriptAlias FakeName RealName", indique où sont physiquement situés les scripts sur le disque, ainsi que l'alias utilisé par les développeurs pour le développement des scripts et des pages. Un développeur utilisera un lien (exemple : /cgi-bin/NomDuScript où /cgi-bin/ est un alias sur /home/httpd/cgi-bin/), et c'est le script /home/httpd/cgi-bin/NomDuScript qui sera effectivement exécuté. La mise en place d'alias permet de restructurer ou déplacer un serveur sans avoir à modifier toutes les pages développées.
- DirectoryIndex donne le ou les noms des fichiers que le serveur doit rechercher si le navigateur passe une requête sur un répertoire. Par exemple sur une requête http://www.MonDomaine.edu, le serveur va rechercher dans l'ordre s'il trouve un fichier index.html, index.stml, index.cgi... en fonction des paramètres de cette variable.

  On peut activer ou non (activée par défaut) l'option "Indexes" au niveau d'un répertoire (voir la partie suivante concernant la sécurisation des accès) de manière à ce que si une URL pointe sur un répertoire, et aucun fichier défini par DirectoryIndex n'existe dans ce dernier, alors le serveur retourne une liste du contenu du répertoire. Si l'indexation n'est pas activée (ce qui est plus sécurisé), on obtient une page d'erreur 403 ("You don't have permission to access /repertoire on this server").

Sécurisation des accès

- Chaque répertoire dont le contenu doit être géré par Apache2 peut être configuré spécifiquement.

  Pour chaque répertoire "UnRépertoire", sur lequel on désire avoir une action, on utilisera la procédure suivante:

  ```
  <Directory UnRépertoire>
  ...Ici mettre les actions...
  </Directory>
  ```

  Tout ce qui est entre les balises s'applique au répertoire "UnRépertoire".

- Quelques options :

  ```
  <Directory UnRépertoire>
   Options Indexes FollowSymLinks ExecCGI
   ...
  </Directory>
  ```

  Indexes : autorise l'affichage du contenu d'un répertoire (si un fichier par défaut n'y est pas trouvé).

  FollowSymLinks: le serveur est autorisé à suivre les liens symboliques dans ce répertoire.

  ExecCGI : l'exécution de scripts CGI est autorisé.

  Pour désactiver les options (par exemple Indexes)

  ```
  <Directory UnRépertoire>
   Options -Indexes FollowSymLinks ExecCGI
   ...
  </Directory>
  ```

- Sécuriser un répertoire en autorisant/refusant l'accès

  On peut réglementer pour chaque répertoire le droit d'accéder aux pages contenues dans ce répertoire, en fonction de la machine cliente et/ou de l'utilisateur qui se connecte. Le fichier dans lequel ce paramétrage s'effectue est un fichier de configuration présent dans `/etc/apache2/sites-available/`.

  Exemple: On désire autoriser l'accès du répertoire "/intranet" aux machines du réseau d'adresse 192.168.1.0/24 et de nom de domaine MonDomaine.edu, et l'interdire à tous les autres.

  ```
  <Directory /intranet>
  #Ordre de lecture des règles
  order allow,deny
  deny from all
  allow from 192.168.1 #ou encore allow from .MonDomaine.edu
  </Directory>
  ```

  Il importe de préciser dans quel ordre les règles de restriction vont être appliquées. Cet ordre est indiqué par le mot réservé “order”, par exemple “order deny,allow” (On refuse puis on alloue l'accès à quelques adresses, c'est à dire que toutes les régles deny vont être lues d'abord, puis ce sera le tour de toutes les règles allow) ou “order allow,deny” (on accepte généralement les accès mais il sont refusés pour quelques adresses : ici, on prend en compte en premier lieu toutes les règles allow dans l'ordre trouvé, puis ensuite toutes les règles deny

).

Exemple: On désire que l'accès soit majoritairement accepté, sauf pour un ou quelques sites.

```
<directory /home/httpd/html>
AllowOverride none
Order deny,allow
deny from pirate.com badboy.com cochon.com
allow from all
</directory>
```

- Authentifier l'accès à un répertoire : ce procédé va permettre de sécuriser l'accès à un répertoire ou à des fichiers. L'accès sera autorisé à une ou plusieurs personnes ou encore à un ou plusieurs groupes de personnes.

  AuthName, définit ce qui sera affiché au client pour lui demander son nom et son mot de passe,

  AuthType, définit le mode d'authentification et d'encryptage “basic” avec HTTP/0 ou “MD5” par exemple avec HTTP/1.

  AuthUserFile, définit le fichier qui contient la liste des utilisateurs et des mots de passe. Ce fichier contient deux champs (Nom d'utilisateur, Mot de passe crypté). Vous pouvez créer ce fichier à partir du fichier /etc/passwd (attention ! faille de sécurité. Il n'est pas forcément avisé d'avoir le même mot de passe pour accéder à Linux et pour accéder à un dossier Web) ou avec la commande "htpasswd" d'Apache.

  Exemple du mode d'utilisation de la commande "htpasswd" :

```
root@mr:/home# htpasswd --help
        htpasswd [-cmdps] passwordfile username
 -c  Create a new file.

#> htpasswd -c /etc/apache/users mlx
Ici on crée le fichier /etc/apache/users et on ajoute un compte.
N'utiliser l'option "-c" que la première fois.
```

  AuthGroupFile définit le fichier qui contient la liste des groupes et la liste des membres de chaque groupe,

  Require permet de définir quelles personnes, groupes ou listes de groupes ont une permission d'accès.

  Exemple de fichier AuthUserFile :

```
doudou:zrag FmlkSsSjhaz
didon:PsddKfdqhgf.fLTER
```

  Exemple de fichier AuthGroupFile :

```
users: tintin milou haddock dupond dupont tournesol
tournesol dupont
```

  Exemple d'autorisation :

```
require user tintin dupond /* tintin et dupond ont un accès */
require group users /* le groupe users à un accès */
require valid-user /* toute personne existant dans AuthUserFile */
```

  Exemple d'accès sécurisé sur un répertoire :

```
<Directory /home/httpd/html/intranet/>
```

```
AuthName PatteBlanche
AuthType basic
AuthUserFile /etc/httpd/conf/users
AutGroupFile /etc/httpd/conf/group

<Limit GET POST>#Ici il faudra un mot de passe
 require valid-user
</Limit>

</Directory>
```

Voici la fenêtre sécurisée que propose Netscape sur l'URL http://localhost/essai:

Figure 16.1. Accés sécurisé sur un répertoire par Apache

La déclaration d'un accès authentifié sur un répertoire peut être faite dans le fichier de configuration correspondant, comme nous venons de le voir, ou en créant un fichier ".htaccess" dans le répertoire que l'on souhaite sécuriser. Le fichier ".htaccess" contient les mêmes directives que celles qui auraient été déclarées dans le fichier httpd.conf. Ainsi, il est possible de délocaliser la gestion de la configuration, au moyen de fichiers spéciaux appelés par défaut .htaccess. Ce dernier paramètre est modifiable.

Le ".htaccess" correspondant à l'exemple précédent contient les directives suivantes :

```
AuthName PatteBlanche
AuthType basic
AuthUserFile /etc/httpd/conf/users
AutGroupFile /etc/httpd/conf/group

<Limit GET POST>#Ici il faudra un mot de passe
 require valid-user
</Limit>
```

Attention :

```
Si vous mettez les clauses d'accès restreint pour un
répertoire dans le fichier de configuration d'Apache, les clauses
seront incluses entre 2 balises :
<Directory ...>
</Directory ...>

Si vous mettez les clauses de restriction dans un fichier ".htaccess",
vous n'avez pas besoin de mettre ces balises.
```

Quelques remarques :

- Les fichiers .htaccess sont lus dynamiquement au moment de chaque requête qui concerne son répertoire : toute modification de ces fichiers prend donc effet immédiatement sans qu'il

soit nécessaire au service de relire la configuration.

- La directive AllowOverride None, permet de désactiver l'utilisation des fichiers .htaccess (attention, elle est à "None" par défaut). Pour qu'Apache lise ce fichier il lui faut mettre la directive : *AllowOverride AuthConfig*
- Limitations de la sécurité par répertoire: ce procédé alourdit la charge du serveur. En effet, si une requête est passée sur www.MonDomaine.edu/rep1/rep2/index.html, le serveur va vérifier dans chaque répertoire rep1, rep2... l'existence d'un fichier .htaccess. Ce sont les règles du dernier fichier qui seront appliquées. Ce processus est mis en oeuvre pour chaque accès. Cette directive est donc à utiliser avec beaucoup de parcimonie car elle crée une surcharge pour le serveur.

Un serveur WEB personnel pour chaque utilisateur

Une directive particulière Userdir public_html permet de gérer pour chaque utilisateur (c'est à dire chaque possesseur d'un compte) des pages personnelles.

Dans apache2, cette directive est en fait associé à un module non activé par défaut. Il suffit donc d'activer le module correspondant par la commande **a2enmod mod_userdir**, ce qui aura pour effet de créer deux liens vers les fichiers correspondants userdir.conf et userdir.load.

Dans userdir.conf, on trouve par défaut les directives suivantes :

- *UserDir public_html*, ce paramètre décrit le processus utilisé pour accéder aux pages personnelles d'une personne, si ces pages sont stockées dans son répertoire personnel.

  Par défaut :

  ```
  <Directory /home/*/public_html>
  AllowOverride FileInfo AuthConfig Limit
  Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
  </Directory>
  ```

  Supposons que vous êtes l'utilisateur "bestof" du réseau et que vous ayez des pages personnelles. Il sera possible d'accéder à vos pages, avec l'adresse suivante: www.MonDomaine.edu/~bestof/index.html. Le (tilde "~") permet d'accéder à votre répertoire personnel. La requête sera réellement exécutée sur "/home/bestof/public_html/index.html.

  Attention, vérifier que le répertoire personnel ne soit pas en mode 700, car personne ne pourrait accéder aux pages personnelles.

- *UserDir disabled root*, ce paramètre, par mesure de sécurité ne permet pas à l'utilisateur "root" de mettre en ligne un web personnel.
- La configuration par défaut n'est pas très sécurisée car elle oblige à mettre les droits de lecture à tout le monde (755) sur le répertoire personnel. Il vous est possible de changer la variable Userdir par exemple :

  ```
  UserDir /home/web
  ```

  Pensez à changer la valeur de Directory.

Activation du serveur (Rappel)

Utilisez les commandes suivantes pour activer, désactiver le serveur:

/etc/init.d/apache start

/etc/init.d/apache stop

Pour relire le fichier de configuration alors qu'apache est déjà lancé, utilisez :

/etc/init.d/apache reload

Pour activer et desactiver un module, utilisez :

a2enmod nomModule

a2dismod nomModule

nomModule est le nom d'un module présent dans `/etc/apaches2/mods-available`

Pour activer et desactiver un site WEB, utilisez

a2ensite nomSite

a2dissite nomSite

nomSite est le nom d'un fichier de configuration du site présent dans `/etc/apaches2/sites-available`

Pensez dans tous les cas à consulter les journaux afin de détecter les dysfonctionnements.

Test de la configuration

Lancez le navigateur et tapez l'url http://localhost. Vous devriez pouvoir utiliser indifféremment l'adresse IP ou le nom d'hôte de votre machine. Vous devez également pouvoir accéder à partir des autres machines de la salle.
