Configuration de base d'un serveur Apache

1. Apache et les serveurs web

Le célèbre Apache est le serveur Web le plus connu et depuis 1996 au moins le plus répandu sur Internet. Sa popularité vient de sa grande stabilité et de sa bonne tenue à la charge. Il est présent dans toutes les architectures à serveur de contenu dynamique LAMP et WAMP (Linux/Windows-Apache-MySQL-PHP), et sa structure modulaire le rend apte à la plupart des utilisations.

Le serveur web est composé d'un script de lancement de service qui, en fonction d'un fichier de configuration, chargera les démons apache et d'éventuels modules fonctionnels.

La richesse fonctionnelle d'Apache implique souvent un fichier de configuration impressionnant. Toutefois, la configuration d'un serveur basique est finalement assez simple à réaliser.

2. Fichier de configuration

a. Format du fichier de configuration

Le fichier de configuration d'Apache, selon versions et options de compilation **httpd.conf**, **apache.conf** ou **apache2.conf**, est composé de directives suivies de valeurs. Certaines directives sont intégrées dans des conteneurs afin que leur champ d'application soit limité.

Format type du fichier de configuration Apache

```
directive1 valeur1
directive2 valeur2
...
<directive_conteneur valeur>
  directive3 valeur3
  directive4 valeur4
  ...
</directive_conteneur>
```

On note ici des directives présentes directement dans le fichier de configuration, et d'autres intégrées dans une section. Cette section limitera le champ d'application des directives à un certain contexte de fonctionnement. La configuration d'Apache consistera à choisir les bonnes directives, leur affecter les bonnes valeurs, et construire les sections nécessaires.

Parmi les innombrables directives Apache, certaines sont fondamentales et devront se retrouver dans toute configuration Apache.

Fichier de configuration : directives courantes	
ServerRoot	Indique le répertoire racine des fichiers de configuration.
User	Désigne le compte de service propriétaire des processus Apache.
Group	Désigne le groupe de service propriétaire des processus Apache.
ErrorLog	Fichier de journalisation des erreurs.
Include	Indique un fichier de configuration annexe à intégrer dans le fichier apache2.conf.
Listen	Indique un port sur lequel le serveur sera à l'écoute.
DocumentRoot	Indique le répertoire contenant les fichiers html.

Exemple de fichier de configuration minimaliste

Notez que si cette configuration peut fonctionner, le serveur manquera un peu de confort, par exemple en ne reconnaissant pas les fichiers index.html ou index.htm. Il faudra donc préciser dans l'URL le nom des fichiers html à atteindre en tapant par exemple http://A.B.C.D/index.html si le répertoire /var/www contient un fichier index.html. De plus, les scripts de lancement de service pouvant être troublés par ce fichier de configuration minimaliste et en un seul morceau, il sera sans doute préférable de lancer l'exécutable ***apache*** *directement sans passer par le script.*

```
ServerRoot /etc/apache2
User www-data
Group www-data
ErrorLog /var/log/apache2/error.log
Listen 80
DocumentRoot /var/www
```

b. Les directives de conteneur

Nous avons vu que les directives servaient à appliquer un élément de configuration au serveur. Par exemple, la ligne de configuration **Listen 80** dans le fichier de configuration est composée de la directive **Listen** qui indique sur quel port le serveur doit attendre des requêtes, et de la valeur **80** qui est le port HTTP standard. Placée directement dans le fichier de configuration, cette directive s'appliquera à l'ensemble du serveur.

Il existe toutefois des éléments de configuration qui ne concernent qu'un aspect fonctionnel du serveur. Par exemple, des directives qui ne devraient s'appliquer qu'à une partie limitée d'un site web, comme des pages web protégées toutes situées dans une arborescence spécifique du système de fichiers. Pour ce type d'usage, Apache utilise des directives de conteneur.

Les directives de conteneur ont deux vocations : grouper un ensemble de directives de configuration, et les appliquer à une partie limitée du serveur Apache.

Syntaxe générique d'une directive de conteneur

```
<directive_conteneur valeur>
  directive3 valeur3
  directive4 valeur4
  ...
</directive_conteneur>
```

Notez que toute section définie par une directive de conteneur entoure la directive des caractères **< >**, et que la section est terminée par le nom de la directive, précédé d'un slash. Entre le début et la fin de section se trouvent toutes les directives ordinaires qui s'appliquent au contexte fonctionnel défini par la directive de conteneur.

Exemple de directive de conteneur

La directive Directory est sans doute la directive de conteneur la plus facile à appréhender. Elle admet comme argument un répertoire et définit des paramètres spécifiques au contenu web situé dans ce répertoire. Dans l'exemple ci-dessous, on indique que dans le répertoire /var/www/special et seulement dans ce répertoire, les liens symboliques peuvent être suivis par le serveur lors de la lecture des pages web.

```
<Directory /var/www/special>
  Options FollowSymLinks
</Directory>
```

c. Validation de la syntaxe

Il est possible (et même prudent) de valider la syntaxe d'un fichier de configuration avant de lancer le service.

Validation de la syntaxe du fichier apache2.conf

```
exe_apache -t
```

Où *exe_apache* représente l'exécutable de lancement du serveur Apache. Les valeurs généralement rencontrées sont **httpd**, **apache** et **apache2**.

d. Démarrage et arrêt du serveur

En situation de production, le démarrage et l'arrêt du serveur se fera par l'utilisation du script de gestion de service situé dans /etc/init.d. En phase de test toutefois, il est utile de démarrer ou d'arrêter le serveur ponctuellement.

Démarrage ponctuel du serveur Apache

```
exe_apache -k start
```

Arrêt du serveur Apache

```
exe_apache -k stop
```

Où *exe_apache* représente l'exécutable de lancement du serveur Apache. Les valeurs généralement rencontrées sont **httpd**, **apache** et **apache2**.

Il est aussi possible de piloter le démon apache par la commande de contrôle apache2ctl avec entre autres les mêmes paramètres start et stop.

3. Les modules Apache

a. Chargement des modules

Le serveur web Apache a une structure modulaire, c'est-à-dire que les fonctions fondamentales du serveur seront fournies par l'exécutable apache, alors que les fonctions accessoires seront assurées par des modules chargés à la demande depuis le fichier de configuration. Ces modules nécessitant souvent des éléments de configuration supplémentaires, les directives associées aux modules devront être ajoutées elles aussi au fichier de configuration.

Format standard de chargement de directive dans apache2.conf

```
LoadModule id_module fichier_module
directive valeur
```

Fichier de configuration : Chargement de module	
LoadModule	Directive de chargement des modules.
id_module	Identifiant de module. Valeur normalisée propre à chaque module.
fichier_module	Le chemin absolu du fichier de module fourni avec Apache.

Exemple de chargement de module

*Dans cet exemple le module chargé est **dir_module** dont la fonction est de simplifier l'écriture des URL par les utilisateurs et d'afficher un fichier html (en général index.html) même s'il n'est pas explicitement précisé. Le fichier exécutable de ce module est **mod_dir.so**. Une fois ce module chargé, il est possible d'appeler la directive **DirectoryIndex** qui demande de charger index.html si aucun fichier n'est précisé dans l'URL.*

```
LoadModule dir_module /usr/lib/apache2/modules/mod_dir.so
DirectoryIndex index.html
```

b. Visualisation des modules

La commande apache utilisée de façon interactive permet d'afficher les modules chargés. Les modules peuvent avoir deux origines : ils ont été appelés par la commande load depuis le fichier de configuration, ou ils ont été compilés dans le cœur du programme et sont chargés systématiquement.

Visualisation des modules compilés dans le programme

```
exe_apache -l
```

Visualisation des modules chargés

```
exe_apache -M
```

Exemple de visualisation des modules chargés

L'option -M affiche les modules statiques et ceux chargés depuis le fichier de configuration par la directive LoadModule.

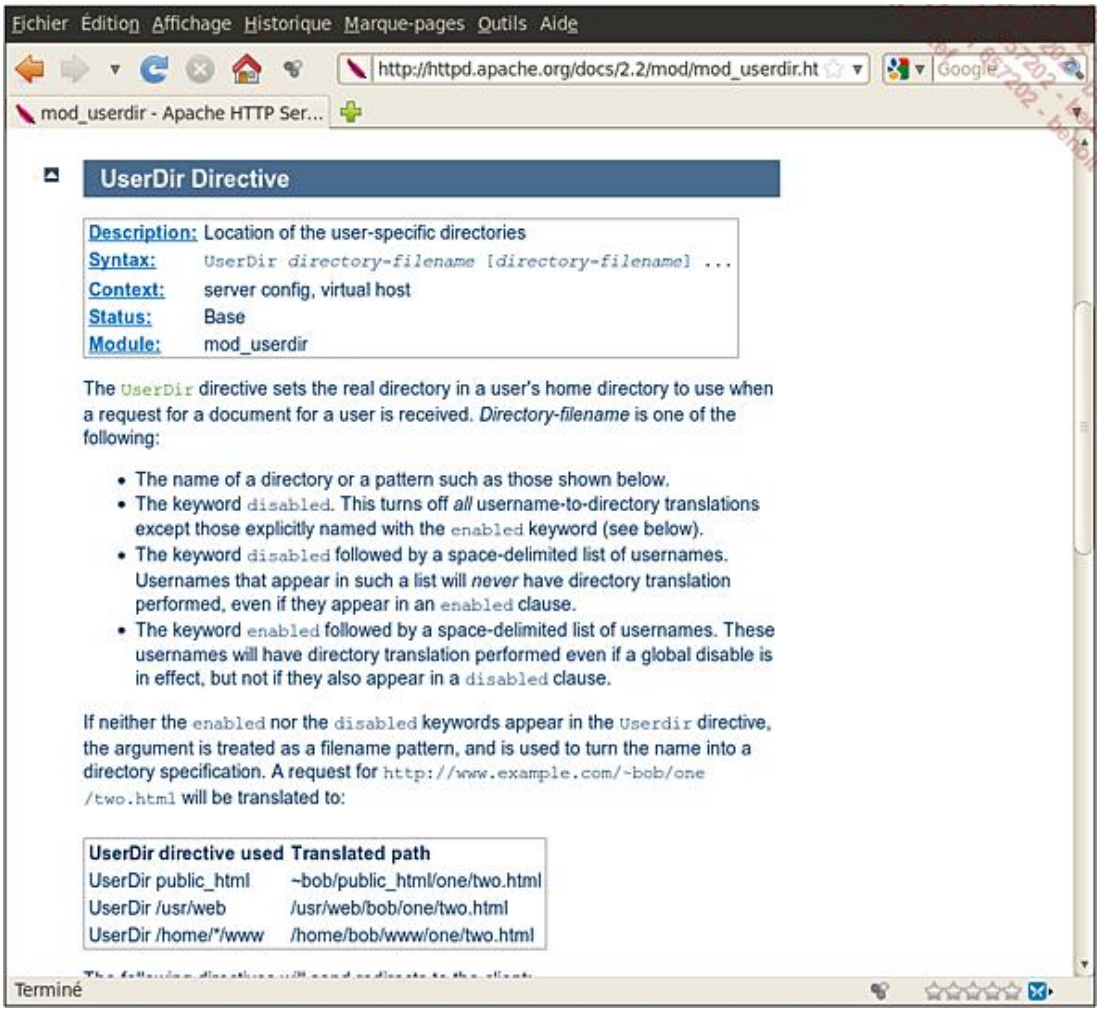
```
alpha:/etc/apache2# apache2 -l
Compiled in modules:
  core.c
  mod_log_config.c
  mod_logio.c
  worker.c
  http_core.c
  mod_so.c
alpha:/etc/apache2# apache2 -M
Loaded Modules:
 core_module (static)
 log_config_module (static)
 logio_module (static)
 mpm_worker_module (static)
 http_module (static)
 so_module (static)
 alias_module (shared)
 auth_basic_module (shared)
 authn_file_module (shared)
 authz_default_module (shared)
 authz_groupfile_module (shared)
 authz_host_module (shared)
 authz_user_module (shared)
 autoindex_module (shared)
 cgid_module (shared)
 deflate_module (shared)
 dir_module (shared)
 env_module (shared)
 mime_module (shared)
 negotiation_module (shared)
 setenvif_module (shared)
 status_module (shared)
Syntax OK
alpha:/etc/apache2#
```

c. Choix des modules

Les modules dépendent naturellement de l'usage qui est fait du serveur. L'usage consiste à identifier les besoins, déterminer les directives associées à ces besoins, et vérifier sur la documentation en ligne Apache (section Directive de configuration à l'exécution) quels sont les modules impliqués.

Supposons qu'on veuille donner aux utilisateurs l'accès à des pages personnelles situées dans leur répertoire personnel. La directive **UserDir** est justement faite pour cela. Elle admet comme paramètre un chemin relatif qui indique l'emplacement du répertoire personnel de l'utilisateur où placer les ressources html de l'utilisateur. Les documents seront alors accessibles par l'url http://serveur/~utilisateur/.

Si on consulte cette directive dans la documentation en ligne, on constate qu'elle est dépendante du module **mod_userdir**. Il faudra donc configurer cette directive et s'assurer que le module sera chargé.



Exemple de configuration de l'accès utilisateur

Avec la configuration de l'accès utilisateur, l'url http://serveur/~toto/doc.html donnera accès au fichier /home/toto/web/doc.html sur le serveur.

```
LoadModule userdir_module modules/mod_userdir.so
UserDir web
```

4. Gestion des ressources

Contrairement à une idée reçue, le serveur Apache n'est pas forcément le plus rapide du marché et des produits au code plus simple permettent des temps de réponses plus rapides à matériel équivalent. Toutefois Apache, par sa gestion intelligente des ressources, et la pré-allocation de processus permet une bien meilleure adaptation à la charge.

Regardons les processus lancés par Apache sur un serveur inactif :

```
alpha:~# ps -ef | grep apache
root      3244     1  0 Feb05 ?        00:00:04 /usr/sbin/apache2 -k start
www-data  3245  3244  0 Feb05 ?        00:00:00 /usr/sbin/apache2 -k start
www-data  3247  3244  0 Feb05 ?        00:00:00 /usr/sbin/apache2 -k start
www-data  3252  3244  0 Feb05 ?        00:00:00 /usr/sbin/apache2 -k start
alpha:~#
```

On constate la présence d'un processus exécuté par root, et de trois autres par le compte de service www-data. Le premier processus ne sert qu'à lancer les autres, qui eux traiteront les requêtes des clients. Le serveur observé ici ne gère aucune connexion cliente, et pourtant, trois processus ont été pré-alloués pour être prêts à gérer toute demande de clients. La gestion du premier service par le compte root est obligatoire, car c'est le seul apte à ouvrir le port 80 sur un système Linux.