Admin Linux Fiche 6 : Apache

Apache reste un Serveur Web incontournable. Nous allons voir ici comment l'installer et le configurer pour un environnement de production.

1 Apache2

1.1 Fonctionnement

Le principe de fonctionnement d'Apache2 repose sur l'utilisation de modules. En effet, il suffit d'installer et/ou d'activer des modules suivant nos besoins. Il existe donc un module pour PHP, pour activer SSL, pour une authentification LDAP, … .

Apache2 est un service qui est initialisé/démarré par systemd. Pour redémarrer le service :

```
/etc/init.d/apache2 restart
OU
service apache2 restart
OU
systemctl restart apache2
```

En production, un serveur apache s'occupe de servir plusieurs sites Web et/ou sert de serveur HTTP frontal. Ces 2 points seront abordés ci-dessous.

1.2 Installation

```
apt-get install apache2 apache2-doc
```

L'installation créée un compte et un groupe www-data. Apache 2 fonctionne par défaut sur ce compte et groupe pour des raisons de sécurité et tourne sur le port 80. Un site de base (page HTML) est placée dans /var/www ce qui permet de tester directement Apache2 après son installation : http://adresseip. Il est à noter que si vous voulez tester apache sur un serveur ne disposant pas d'interface graphique (et donc pas de navigateur classique), vous pouvez installer lynx qui est un navigateur en mode texte (c'est moche mais cela permet de tester !).

1.3 Configuration

1.3.1 Modules

Apache dispose de nombreux modules. Nous n'en ferons pas l'inventaire ici. Nous nous contenterons d'en citer 3, très utilisés : rewrite(frameworks MVC), proxy_http, ssl

Pour activer un module il suffit d'utiliser la commande « a2enmod » (apache2 enable module). Il existe évidemment la commande réciproque « a2dismod ». N'oubliez pas de redémarrer le service apache2 après activation du module.

```
a2enmod <<module>>
```

1.3.2 Configuration PHP

Apache peut être configuré pour servir des pages PHP. Il suffit d'installer PHP ainsi que le module PHP pour apache et de redémarrer le service apache2.

```
apt-get install php5 php5-mysql libapache2-mod-php5
```

1.3.3 Virtualhosts

Les virtualhosts permettent de déployer plusieurs sites Web sur un même serveur (même adresse IP). La distinction se fait en général sur le nom du site, apache doit en effet savoir suivant l'url quel site il doit présenter.

L'ajouter d'un vhost se fait en créant un fichier dans */etc/apache2/sites-available/ « monsite.conf »* et puis d'y insérer :

```
<VirtualHost *:80>
        ServerName monsite.be

        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/htdocs/monsite
        ErrorLog ${APACHE_LOG_DIR}/monsite_error.log
        CustomLog ${APACHE_LOG_DIR}/monsite_access.log combined

        <Directory /var/www/htdocs/monsite>
        Require all granted
        AllowOverride All
        </Directory>
</VirtualHost>
```

Directives:

La directive « ServerName » est nécessaire pour qu'apache fasse une distinction sur le nom du site. Toute URL comportant « monsite.be » utilisera ce vhost.

La directive « ServerAdmin » permet de préciser le responsable du site.

La directive « DocumenRoot » permet de préciser l'endroit où se trouve l'arborescence du site.

La directive « ErrorLog » et « CustomLog » permettent de préciser où les logs seront stockés.

On peut ensuite appliquer des règles/restrictions sur le site via la directive « Directory ». Ici on autorise tout le monde à voir le site « Require all granted » et on autorise les utilisateurs à redéfinir ces règles « AllowOverride All ». Ceci permet par exemple de définir des .htacess.

La directive « Require » peut autoriser ou interdire des adresses IPs, des utilisateurs, …. . Exemples :

```
# n'autoriser l'accès au site que depuis localhost
require ip localhost
# accès uniquement au site pour l'utilisateur admin
require user admin
# pas de redéfinition possible (pas de .htaccess)
AllowOverride None
```

Pour activer le vhost, il suffit d'utiliser la commande « a2ensite » (apache2 enable site). Il existe la commande réciproque « a2dissite ». Ne pas oublier de redémarrer le service apache2 après activation.

```
a2ensite monsite
```

1.3.4 Reverse proxy

Un proxy inverse est un serveur frontal c'est-à-dire un serveur exposé sur Internet et par lequel toutes les requêtes passeront. Ce serveur ne traitera pas les requêtes mais se contentera des les rediriger vers d'autres serveurs internes à l'entreprise. Les intérêts de ce mécanisme sont multiples. Vu qu'il n'y a qu'un seul point d'accès, la sécurité est plus facile à gérer. Cela permet également de mettre en œuvre du « load balancing » entre des serveurs internes. C'est également un moyen simple de rendre disponible un serveur interne sur le Web (pas besoin de configuration réseau).

Pour mettre en place un reverse proxy, il faut activer le module apache « proxy_http » et « proxy ».

```
a2enmod proxy proxy_http
```

Ensuite dans le fichier VirtualHost :

```
<VirtualHost *:80
        ServerName siteReverseProxy
        ServerAdmin webmaster@localhost

        ProxyPass /  http://www.example.com/
         ProxyPassReverse / http://www.example.com/

        ErrorLog ${APACHE_LOG_DIR}/siteReverse_error.log
        CustomLog ${APACHE_LOG_DIR}/siteReverse_access.log combined
</VirtualHost>
```

1.4 Sécurité

Un serveur Web doit être sécurisé en particulier les échanges entre le client et le serveur doivent être cryptés. Ceci se fait aisément grâce au paquet openssl. Le port par défaut pour les communications https est le 443.

1.4.1 Installation

```
apt-get install openssl
a2enmod ssl
systemctl restart apache2
```

1.4.2 Création d'un certificat auto-signé

La commande openssl permet de créer un certificat ainsi qu'un clé associée à ce certificat.

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout
/etc/apache2/ssl/apache.key -out /etc/apache2/ssl/apache.crt
```

Ici la clé et le certificat seront déposés dans le répertoire /etc/apache2/ssl créé au préalable.

Le VirtualHost sera modifié de la sorte :

```
<VirtualHost *:443>
        ServerName monsite.be

        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/htdocs/monsite
        ErrorLog ${APACHE_LOG_DIR}/monsite_error.log
        CustomLog ${APACHE_LOG_DIR}/monsite_access.log combined

        SSLEngine on
        SSLCertificateFile /etc/apache2/server.crt
        SSLCertificateKeyFile /etc/apache2/server.key

        <Directory /var/www/htdocs/monsite>
        Require all granted
        AllowOverride All
        </Directory>
</VirtualHost>
```

1.4.3 Let’s encrypt

Let’s encrypt est une autorité de certification libre, gratuite et automatisée. Ceci permet d’obtenir un certificat valide pour son site Web sans trop d’effort. Cependant, la machine servant le site Web doit être « publiquement » accessible ainsi que le nom du domaine. Cela veut dire qu’en test ce procédé n’est pas applicable.

https://letsencrypt.org/