

Apache +PHP + MySQL

(en utilisant les packages de la Mandrake)

Olivier Hoarau (olivier.hoarau@funix.org)

V1.3, 17 juin 2001

1	Historique du document	2
2	Préambule	2
3	Présentation	2
4	Installation	2
4.1	Installation Mandrake 7.2	2
4.2	Installation Mandrake 8.0	3
4.3	Présentation de l'arborescence	4
4.4	Configuration	5
4.4.1	Le fichier de configuration	5
4.4.2	Les pages webs utilisateurs	8
4.5	Lancement de l'application	8
4.6	Tests de fonctionnement	9
5	Configuration avancée	10
5.1	Les hôtes virtuels	10
5.2	Les alias	12
5.3	Protection d'une page	12
6	MySQL	13
6.1	Présentation	13
6.2	Installation	13
6.3	Mise en place des utilisateurs	15
6.4	Création de tables	16
6.5	Tests de fonctionnement MySQL	18
6.6	phpMyAdmin	20
7	Scripts CGI	22
8	PHP et LDAP	24

1 Historique du document

17.06.01 V1.3 Passage à Mandrake 8.0 pour la partie log apache voir le document Apache+PHP+MySQL avec les sources
3.12.00 V1.2 Rajout du paragraphe **PHP** et **LDAP** et mise à jour **webalizer** (v2.01.06)
22.10.00 V1.1 Passage à Mandrake 7.2 avec **Apache 1.3.14**, **PHP 4.0.3pl1**, **MySQL 3.23.23** et **phpMyAdmin 2.1.0**
25.7.00 V1.0 Création du document

2 Préambule

Ce document présente l'installation et la configuration d'**Apache** sur une distribution Linux Mandrake avec gestion **PHP4** et **MySQL** en utilisant les packages standards de la Mandrake. La dernière version de ce document est téléchargeable à l'URL <http://www.funix.org>. Ce document peut être reproduit et distribué librement dès lors qu'il n'est pas modifié et qu'il soit toujours fait mention de son origine et de son auteur, si vous avez l'intention de le modifier ou d'y apporter des rajouts, contactez l'auteur pour en faire profiter tout le monde. Ce document ne peut pas être utilisé dans un but commercial sans le consentement de son auteur. Ce document vous est fourni "dans l'état" sans aucune garantie de toute sorte, l'auteur ne saurait être tenu responsable des quelconques misères qui pourraient vous arriver lors des manipulations décrites dans ce document.

3 Présentation

Cette page n'a pas pour vocation de vous présenter l'installation d'un serveur Apache destiné à être vu du monde entier, mais plutôt l'installation d'un serveur **Apache** sur un réseau local en intranet, afin surtout de se familiariser au langage **PHP** et au SGBD **MySQL** et éventuellement à **LDAP**.

Si vous voulez avoir une version optimisée pour votre machine vous pouvez consulter le document "[Apache+PHP+MySQL avec les sources](http://www.funix.org)" disponible sur <http://www.funix.org>

NOTE: Pour plus d'info sur Apache www.apache.org

Pour l'analyser des logs d'Apache, reportez vous à la page "[Analyser les logs d'Apache avec Webalizer et Awstats](http://www.funix.org)" du site <http://www.funix.org>

Pour mettre en place un moteur de recherche, reportez vous à la page "[Ht://Dig](http://www.funix.org)" du site <http://www.funix.org>.

4 Installation

4.1 Installation Mandrake 7.2

La mise en place d'un serveur Apache est très simple, vous devez installer les packages suivants (sur une Mandrake 7.2) si ce n'est déjà fait :

rpm -ivh apache-common-1.3.14-2mdk
rpm -ivh apache-1.3.14-2mdk

Eventuellement (voir note)

rpm -ivh apache-suexec-1.3.14-2mdk.i586.rpm

NOTE Le package **suexec** permet de pouvoir lancer les scripts CGI avec les droits du propriétaire et non pas en tant que nobody (par défaut). Ceci a l'avantage qu'en tant que simple utilisateur on peut créer des scripts CGI pour modifier ses propres fichiers sans leur donner des droits en lecture écriture pour le monde entier (666). Faites gaffe quand même car si le script CGI est mal écrit, ça peut être un énorme trou de sécurité dans votre système.

Et pour le PHP4 (CD1)

rpm -ivh php-4.0.3pl1-1mdk
rpm -ivh mod_php-4.0.3pl1-1mdk

Pour MySQL (CD2)

rpm -ivh php-mysql-4.0.3pl1-1mdk.i586.rpm

Attention le serveur **MySQL** fourni avec la Mandrake 7.2 doit être préalablement installé, pour plus d'info voir la page le paragraphe MySQL.

Pour LDAP (CD2)

rpm -ivh php-ldap-4.0.3pl1-1mdk.i586.rpm

Attention le serveur **LDAP** fourni avec la Mandrake 7.2 doit être préalablement installé, pour plus d'info voir la page correspondante sur <http://www.funix.org>.

4.2 Installation Mandrake 8.0

Voici les packages standards à installer pour **Apache**:

apache-1.3.19-3mdk
apache-common-1.3.19-3mdk
apache-conf-1.3.19-3mdk
apache-modules-1.3.19-3mdk

Les packages pour **PHP** avec option **GD**, **MySQL** et **LDAP**

php-common-4.0.4pl1-6mdk
mod_php-4.0.4pl1-6mdk
php-gd-4.0.4pl1-6mdk
php-mysql-4.0.4pl1-6mdk
php-4.0.4pl1-6mdk
php-ldap-4.0.4pl1-6mdk

ATTENTION - Pour **MySQL**, le serveur **MySQL** fourni avec la Mandrake 8.0 doit être préalablement installé, pour plus d'info voir le paragraphe [MySQL](#).

- pour **LDAP**, le serveur **LDAP** fourni avec la Mandrake 8.0 doit être préalablement installé, pour plus d'info voir la page [OpenLDAP](#) sur <http://www.funix.org>

4.3 Présentation de l'arborescence

L'installation va créer un répertoire **/etc/httpd** (droit 755) contenant:

- répertoire **conf** (droit 755)
- fichier **logs** lien vers répertoire **/var/log/httpd**
- fichier **modules** lien vers répertoire **/usr/lib/apache**

Sur une Mandrake 8.0 fichier **extramodules** lien vers **/usr/lib/apache-extramodules/**

Le répertoire de log **/var/log/httpd** contient essentiellement deux fichiers:

- **access_log** listant les accès au serveur
- **error_log** listant les erreurs en tout genre

Le répertoire de modules **/usr/lib/apache** contient tous les modules utilisables par **Apache**, pour info un module est une extension logicielle à **Apache**, lui permettant par exemple d'interpréter le **PHP4** (module **libphp4.so**), ou alors d'interpréter les extensions **FrontPage** (module **mod_frontpage**). Les modules ont 755 comme droits (proprio root, groupe root).

NOTE Sur une Mandrake 8.0 **libphp4.so** se trouve sous **/usr/lib/apache-extramodules/**

Le répertoire **/etc/http/conf** contient:

- le fichier de configuration d'Apache **httpd.conf**
- **apache-mime.types** fixe le type de fichier suivant l'extension du dit fichier (**.doc**=mword, **.ps**=postscript, ...), ça permet au client qui se connecte sur le serveur, de savoir comment interpréter le fichier suivant son extension.
- **magic** sert pour le module **mod_mime_magic** qui est désactivé par défaut
- un répertoire **vhosts** pour gérer les hôtes virtuels, en gros pour un même serveur vous pouvez définir plusieurs noms, voir paragraphe plus bas
- un répertoire **addon-modules** qui contient des fichiers de configuration de certaines modules

NOTE Sur une Mandrake 8.0 on trouve aussi le fichier **commonhttpd.conf** qui est appelé dans **httpd.conf**, il contient des paramètres de configuration.

Maintenant vous avez d'autres fichiers sous **/var/www/html** avec :

- répertoire **cgi-bin** qui contiendra vos scripts CGI
- répertoire **html** contenant la page d'accueil d'**Apache** par défaut
- répertoire **icons**, qui comme son nom l'indique contient des icônes, notamment celles pour identifier le type de fichier

4.4 Configuration

4.4.1 Le fichier de configuration

Voici un extrait du fichier de conf d'Apache `/etc/httpd/conf/httpd.conf` (et `commonhttpd.conf` sur une Mandrake 8.0) voici les points que je juge important dans ce(s) fichier (s):

(...)

----- Server Configuration -----

ServerType is either inetd, or standalone.

**# vous pouvez soit lancer Apache par le super daemon inetd (config dans /etc/inetd.conf) soit tout seul avec un script dans /etc/rc.d/init.d, à noter qu'avec la première méthode vous pouvez contrôler l'accès avec les TCP_Wrappers (hosts.deny, hosts.allow)
voir la note en bas du paragraphe pour les avantages de l'un ou de l'autre
moi j'ai choisi lancement en standalone (voir paragraphe suivant)**

ServerType standalone

(...)

**# chemin pour les fichiers de conf
ServerRoot /etc/httpd**

(...)

**# nom du serveur
ServerName obelix**

(...)

**# Port: The port the standalone listens to. For ports < 1023, you will
need httpd to be run as root initially.
numéro de port pour le serveur, traditionnellement à 80**

Port 80

(...)

**# Limit on total number of servers running, i.e., limit on the number
of clients who can simultaneously connect --- if this limit is ever
reached, clients will be LOCKED OUT, so it should NOT BE SET TOO LOW.
It is intended mainly as a brake to keep a runaway server from taking
Unix with it as it spirals down...
ici on fixe le nombre de clients max qui peuvent se connecter à un moment donné**

MaxClients 150

(...) suit ensuite une liste de modules, ceux avec un # devant ne seront pas inclus

```
# LoadModule mmap_static_module modules/mod_mmap_static.so
# LoadModule vhost_alias_module modules/mod_vhost_alias.so
LoadModule env_module      modules/mod_env.so
LoadModule config_log_module modules/mod_log_config.so
LoadModule agent_log_module modules/mod_log_agent.so
LoadModule referer_log_module modules/mod_log_referer.so
# LoadModule mime_magic_module modules/mod_mime_magic.so
LoadModule mime_module     modules/mod_mime.so
LoadModule negotiation_module modules/mod_negotiation.so
....
```

(...) Pour chaque module on doit maintenant la routine **AddModule**

```
# AddModule mod_mmap_static.c
# AddModule mod_vhost_alias
AddModule mod_env.c
AddModule mod_log_config.c
AddModule mod_log_agent.c
AddModule mod_log_referer.c
# AddModule mod_mime_magic.c
AddModule mod_mime.c
AddModule mod_negotiation.c
AddModule mod_status.c
AddModule mod_info.c
AddModule mod_include.c
...
(...)
```

```
# On lance initialement httpd en tant que root, puis immédiatement
# c'est l'utilisateur apache (groupe apache) qui en devient le proprio
# ainsi s'il y a une faille dans Apache, le hacker au lieu de devenir root
# devient apache avec les droits qui vont avec
```

```
User apache
Group apache
(...)
```

```
# Apache peut se comporter comme un proxy comme squid
# par défaut cette fonction n'est pas activée
# Proxy Server directives. Uncomment the following line to
# enable the proxy server:
```

```
# ProxyRequests On
```

```
# To enable the cache as well, edit and uncomment the following lines:
```

```
# CacheRoot /var/cache/httpd
# CacheSize 5
# CacheGcInterval 4
# CacheMaxExpire 24
```

```
# CacheLastModifiedFactor 0.1
# CacheDefaultExpire 1
# NoCache a_domain.com another_domain.edu joes.garage_sale.com
```

(...)

```
# UserDir: The name of the directory which is appended onto a user's home
# directory if a ~user request is recieved.
# on pourra accéder aux pages HTML de vos utilisateurs si ceux-ci créent un répertoire
public_html (droit 755) sous leur homedirectory (droit 755)
# on y accès en tapant dans un navigateur http://serveur-apache/~login-utilisateur
```

```
UserDir public_html
```

```
# DirectoryIndex: Name of the file or files to use as a pre-written HTML
# directory index. Separate multiple entries with spaces.
# fichier par défaut qui sera appelé en premier (dans l'ordre index.html testé en
premier)
```

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

(...)

```
# LanguagePriority allows you to give precedence to some languages
# in case of a tie during content negotiation.
# Just list the languages in decreasing order of preference.
# on peut éventuellement donner un ordre de préférence pour la langue à utiliser si on a
le choix
```

```
LanguagePriority fr en de
```

(...)

```
# AddType allows you to tweak mime.types without actually editing it, or to
# make certain files to be certain types.
# Format: AddType type/subtype ext1
```

```
# For example, the PHP3 module (not part of the Apache distribution)
# will typically use:
# ici on définit les extensions des fichiers pour que le module PHP soit appelé
AddType application/x-httpd-php3 .php3 .phtml .php
AddType application/x-httpd-php3-source .phps
```

(...)

tout à la fin du fichier, on trouve l'inclusion du fichier de conf de PHP

```
Include conf/addon-modules/php.conf
```

Ce fichier contient:

```
LoadModule php4_module    /usr/lib/apache/libphp4.so  
AddModule mod_php4.c
```

```
AddType application/x-httpd-php    .php .php4 .php3 .phtml  
AddType application/x-httpd-php-source .phps
```

Vous constaterez que vous pouvez très bien l'intégrer très facilement au fichier **http.conf** pour simplifier.

NOTE "Anciennement" on trouvait un fichier **srm.conf** et **access.conf**, ils sont maintenant complètement intégrés dans **httpd.conf**

4.4.2 Les pages webs utilisateurs

Le problème avec le répertoire **public_html** des utilisateurs et qu'il faut mettre 755 au niveau de la home directory, ce qui est particulièrement gênant au niveau sécurité. Vous pouvez spécifier que chaque utilisateur doit créer ses pages sous **/home/http/login-utilisateur** en écrivant pour la variable **UserDir**

```
UserDir /home/httpd
```

Ainsi pour l'utilisateur toto quand vous taperez comme URL **http://serveur-apache/~toto**, apache ira chercher le fichier **index.htm** sous **/home/httpd/toto**. On peut aller plus loin en spécifiant un répertoire particulier, **/home/httpd/toto/html** par exemple, en écrivant:

```
UserDir /home/httpd/*/html
```

4.5 Lancement de l'application

Vous allez trouver un fichier de lancement/arrêt du nom de **httpd** dans **/etc/rc.d/init.d** avec les liens suivants:

Pour un lancement automatique à l'état de marche 345 et à un arrêt pour les autres états de marche, on tapera :

```
chkconfig --level 345 httpd on  
chkconfig --level 0126 httpd off
```

Pour lancer

```
/etc/rc.d/init.d/httpd start
```

Pour relancer

```
/etc/rc.d/init.d/httpd start
```

Pour connaître l'état

```
/etc/rc.d/init.d/httpd status
```

Pour stopper

```
/etc/rc.d/init.d/httpd stop
```

4.6 Tests de fonctionnement

Tout d'abord en tant que simple utilisateur, **toto** par exemple, vous allez créer un répertoire **public_html** dans votre répertoire contenant un fichier **index.html** basique du style:

```
<html>
<body>
<p> hello world </p>
</body>
</html>
```

ATTENTION répertoire **/home/toto** à 755 de même que **/home/toto/public_html**

Maintenant d'un poste client tapez dans le champ URL de votre navigateur préféré, en admettant que votre serveur se nomme **obelix**:

```
http://obelix
```

Et là magique, apparaît la page d'accueil d'Apache, pour info celle-ci se trouve sous **/home/httpd/html**. Maintenant tapez:

```
http://obelix/~toto
```

et vous voyez apparaître sur un fond bien gris (ou d'une autre couleur ça dépend de la config de votre navigateur).

hello world

C'est bon ça marche. Pour tester le PHP3, en tant que **toto**, écrivez un fichier **info.php3** sous votre répertoire **public_html** contenant:

```
<?
phpinfo();
?>
```

D'un poste client, tapez dans le champ adresse (ou URL):

```
http://obelix/~toto/info.php3
```

Voilà le résultat :



C'est bon le module **PHP** fonctionne.

5 Configuration avancée

5.1 Les hôtes virtuels

On peut mettre en place des hôtes virtuels, en d'autres termes un utilisateur pour un même serveur Apache croira en voir plusieurs. Exemple, soit votre serveur Apache **obelix** (adresse IP **192.168.13.11**), votre domaine **breizland.bz**, on va créer les hôtes virtuels **www.asterix.breizland.bz** et **www.idefix.breizland.bz** qui vont pointer chacun vers un endroit différent du disque (respectivement **/usr/local/asterix** et **/usr/local/idefix** chacun contenant des pages html).

Editez le fichier **/etc/httpd/conf/httpd.conf** et décommentez (si nécessaire) les lignes suivantes tout à la fin du fichier (enlevez le **#** devant la ligne) :

```
# For DynamicVhosts and VirtualHomePages, uncomment those lines:
LoadModule vhost_alias_module modules/mod_vhost_alias.so
AddModule mod_vhost_alias.c
```

Maintenant vous allez éditer le fichier **/etc/httpd/conf/vhosts/Vhosts.conf** et rajouter:

```
NameVirtualHost 192.168.13.11

<VirtualHost 192.168.13.11>
ServerName obelix.breizland.bz
DocumentRoot /home/httpd/html
ErrorLog logs/obelix-error_log
```

```
TransferLog logs/obelix-access_log
</VirtualHost>
```

```
<VirtualHost 192.168.13.11>
ServerName www.asterix.breizland.bz
DocumentRoot /usr/local/asterix
ErrorLog logs/asterix-error_log
TransferLog logs/asterix-access_log
</VirtualHost>
```

```
<VirtualHost 192.168.13.11>
ServerName www.idefix.breizland.bz
DocumentRoot /usr/local/idefix
ErrorLog logs/idefix-error_log
TransferLog logs/idefix-access_log
</VirtualHost>
```

Pour info, ce fichier est appelé dans `/etc/http/conf/httpd.conf` à la ligne

```
# We also put virtual hosts "Includes" in this section. This will be used by
# administration scripts. Instead of parsing all httpd.conf to modify a
# vhost, we just include the config file. Each type of vhosting has its own
# config file.
#
Include conf/vhosts/Vhosts.conf
```

Relancez Apache en tapant:

```
/etc/rc.d/init.d/httpd restart
```

Maintenant nous allons créer nos hôtes **asterix** et **idefix**, pour cela vous avez deux méthodes:
- rajouter **www.asterix.breizland.bz** et **www.idefix.breizland.bz** dans `/etc/hosts` sur la même ligne que votre serveur Apache (**obelix** dans notre exemple).

```
192.168.13.11      obelix      obelix.breizland.bz      www.asterix.breizland.bz
www.idefix.breizland.bz
```

Normalement si vous faites un ping sur **www.idefix.breizland.bz** ça devrait marcher, pour les postes clients il faudra rajouter la même ligne dans le fichier **hosts** (non nécessaire).

- si vous disposez d'un serveur DNS sur votre machine , au niveau de votre config DNS dans votre fichier **breizland.bz** qui se trouve sous `/var/named` vous devez rajouter tout à la fin:

```
www.asterix  A  192.168.13.11
www.idefix   A  192.168.13.11
```

Relancez le DNS en tapant:

```
/etc/rc.d/init.d/named restart
```

Pour tester tapez dans un shell:

```
ping www.asterix.breizland.bz
```

Maintenant dans le champ URL de votre navigateur préféré:

```
http://www.asterix.breizland.bz
```

Et là, normalement vous devriez voir s'afficher la page que vous avez placé sous `/usr/local/asterix`

5.2 Les alias

Si vous ne voulez pas mettre en place un serveur DNS, vous avez un moyen plus simple, les alias. Concrètement, votre serveur s'appelle **obelix**, vous voulez rendre accessible les fichiers html se trouvant sous `/usr/doc/html`, les utilisateurs devront taper dans leur navigateur préféré: **http://obelix/doc/**. Pour cela dans votre fichier `/etc/httpd/conf/httpd.conf`, vous allez rajouter:

```
Alias /icons/ /var/www/icons/  
Alias /doc/ /usr/doc/html/
```

La première ligne est déjà existante sur une config standard, vous n'avez qu'à essayer de taper l'URL **http://obelix/icons/** pour s'en convaincre.

ATTENTION N'oubliez pas de rajouter le `/` à la fin

5.3 Protection d'une page

La protection d'une page pour l'utilisateur **olivier** se fait de manière très simple, tous les fichiers à accès limité doivent être concentré dans un même répertoire `/home/olivier/public_html/reserve` par exemple, il suffit de créer dans celui-ci un fichier qu'on devra nommer `.htaccess` contenant:

```
AuthUserFile auth/olivier.users  
AuthName "Acces Restreint"  
AuthType Basic
```

```
<Limit GET POST>  
require valid-user  
</Limit>
```

Le fichier **olivier.users** va contenir la liste des utilisateurs habilités à accéder au répertoire où se trouve `.htaccess`, il va se trouver sous le répertoire `/etc/httpd/auth`, pour info vous pouvez changer le chemin `/etc/httpd` en modifiant la valeur de la variable **ServerRoot** qu'on trouve dans le fichier `httpd.conf`. Pour créer ce fichier il suffit d'une part de créer le répertoire `/etc/httpd/auth` si celui-ci n'existe pas, puis de taper:

```
htpasswd -c /etc/httpd/auth/olivier.users olivier
```

L'option **-c** correspond à la création du fichier. On va alors avoir à rentrer un mot de passe pour l'utilisateur **olivier** habilité à accéder au répertoire protégé.

New password:

On confirme

Re-type new password:

Pour que l'utilisateur **veronique** puisse accéder aussi au répertoire **reserve d'olivier**, vous taperez alors:

```
htpasswd /etc/httpd/auth/olivier.users veronique
```

Maintenant quand à partir de votre navigateur préféré quand vous allez rentrer comme URL **http://obelix/~olivier/reserve**, vous aurez une fenêtre popup qui va s'ouvrir vous demandant de rentrer votre nom d'utilisateur et le mot de passe préalablement rentré.

Si vous ne voulez pas que quelqu'un jette un coup d'oeil dans les fichiers **.htaccess** de vos utilisateurs, rajoutez (déjà fait sur Mandrake 8.0) dans le fichier **httpd.conf**, la directive suivante:

```
<files ~ "\.ht">  
order deny,allow  
deny from all  
</files>
```

6 MySQL

6.1 Présentation

Cette n'a pas pour but de vous présenter ce qu'est un **SGBD** et encore moins de vous expliquer le langage **SQL** mais de vous présenter l'installation et la configuration de **MySQL** afin de pouvoir l'utiliser avec **Apache+PHP**

Cette page concerne la Mandrake 7.2 et la 8.0.

Le chapitre [Installation](#) concerne l'installation des packages binaires, pour une installation en utilisant les sources (version plus récente) voir ma page installation le document téléchargeable « Apache+PHP+MySQL avec les sources » sur <http://www.funix.org>. Les autres chapitres concernent les deux types d'installation.

6.2 Installation

MySQL 3.23.23 est livré avec le CD n°2 de la Mandrake 7.2, il n'est pas installé par défaut, cependant pour voir si **MySQL** est installée sur votre système, vous pouvez taper:

```
rpm -aq | grep -i mysql
```

Sur une Mandrake 8.0 vous trouverez la version stable 3.23.36 (dernière version MySQL 3.23.38).

Voici les différentes étapes de l'installation du packages binaire sur la machine **obelix** avec le package livré avec la Mandrake 7.2 (idem sous Mandrake 8.0 à quelques détails près).

```
[root@obelix linux]# rpm -ivh MySQL-3.23.23-1mdk.i586.rpm
MySQL #####
Creating db table
Creating host table
Creating user table
Creating func table
Creating tables_priv table
Creating columns_priv table
```

PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !

This is done with:

```
/usr/bin/mysqladmin -u root -p password 'new-password'
```

```
/usr/bin/mysqladmin -u root -h asterix.kervao.fr -p password 'new-password'
```

See the manual for more instructions.

Please report any problems with the `/usr/bin/mysqlbug` script!

The latest information about MySQL is available on the web at

<http://www.mysql.com>

Support MySQL by buying support/licenses at <http://www.mysql.com/license.htm>.

Starting `mysqld` daemon with databases from `/var/lib/mysql`

Suivi de

```
[root@obelix linux]# rpm -ivh MySQL-client-3.23.23-1mdk.i586.rpm
MySQL-client #####
```

Et de

```
[root@obelix linux]# rpm -ivh MySQL-shared-3.23.23-1mdk.i586.rpm
MySQL-shared #####
```

Cela va vous créer un ensemble d'exécutables sous `/usr/sbin` (ceux réservés à root) et sous `/usr/bin` (pour tous les utilisateurs), par ailleurs un répertoire `/var/lib/mysql` va être créé. Le fichier de démarrage du daemon **MySQL**, [mysql](#) (droit 755) se trouve sous `/etc/rc.d/init.d`, par défaut le serveur est lancé à l'état de marche 2, 3, 4 et 5.

L'installation par **rpm** a lancé le daemon.

NOTE Vous noterez que si vous faites un:

```
ps aux | grep mysqld
```

L'utilisateur **mysql** est proprio du daemon **mysqld**. Vous vous rendrez compte que même si c'est **root** qui lance le **daemon** aussitôt après c'est l'utilisateur **mysql** qui en devient le proprio, ainsi si jamais il y avait une faille de sécurité dans **MySQL**, le hacker ne se retrouverait pas **root** mais **mysql** avec les droits qui vont avec.

6.3 Mise en place des utilisateurs

La première chose à faire est de mettre un mot de passe pour **root** pour l'accès à l'administration des bases de données. La commande à taper est:

```
mysqladmin -u root password 'mot-de-passe'
```

Le mot de passe peut être différent de celui du login. Maintenant on va créer un compte utilisateur (**olivier** dans mon exemple), pour cela on doit se connecter en tant que **root** à la base de donnée **mysql** contenant les infos sur les utilisateurs et leurs droits.

```
[olivier@obelix olivier]$ mysql -u root -p mysql
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
```

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2 to server version: 3.23.37
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the buffer
```

```
mysql>
```

A présent on va entrer l'utilisateur **olivier** qui sera un super utilisateur avec les mêmes droits que **root**:

```
mysql> INSERT INTO user
-> VALUES('localhost','olivier',PASSWORD('mot-de-passe'),
-> 'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y');
Query OK, 1 row affected (0.00 sec)
```

NOTE Il n'est pas obligatoire de rentrer le login pour le nom d'utilisateur et le mot de passe de login.

Pour voir si la saisie s'est bien passée:

```
mysql> SELECT * FROM user;
+-----+-----+-----+-----+-----+-----+-----+
|Host    |User    |Password    | Select_priv|Insert_priv|Update_priv|Delete_priv|
+-----+-----+-----+-----+-----+-----+-----+
|localhost|root    |626f04681159| Y          | Y          | Y          | Y          |
|localhost|olivier|588c54bd00a | Y          | Y          | Y          | Y          |
+-----+-----+-----+-----+-----+-----+-----+
```

```

-----+-----+-----+-----+-----+-----+-----+-----+-----+
Create_priv|Drop_priv|Reload_priv|Shutdown_priv|Process_priv|File_priv|Grant_priv|
References_priv|Index_priv|Alter_priv |
-----+-----+-----+-----+-----+-----+-----+-----+-----+
Y          | Y          | Y          | Y          | Y          | Y          | Y          |
Y          | Y          | Y          |           |           |           |           |
Y          | Y          | Y          | Y          | Y          | Y          | Y          |
Y          | Y          | Y          |           |           |           |           |
-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0.01 sec)

```

Maintenant pour prendre tout ça en compte.

```
mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.08 sec)
```

Pour quitter

```
mysql>quit
```

6.4 Création de tables

Maintenant notre utilisateur olivier va créer une table qui nous servira plus tard pour nos expérimentations avec Apache. Il doit d'abord se connecter:

```
[olivier@obelix olivier]$ mysql -u olivier -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3 to server version: 3.23.37
```

Type 'help;' or '\h' for help. Type '\c' to clear the buffer

```
mysql>
```

Pour voir la liste des bases de données disponibles, on tapera:

```
mysql> SHOW DATABASES;
+-----+
| Database |
+----+----+
| mysql    |
| test     |
+-----+
4 rows in set (0.00 sec)
```

On va maintenant créer une base de données **essai**:

```
mysql> CREATE DATABASE essai;
Query OK, 1 row affected (0.00 sec)
```

On va utiliser maintenant cette base de donnée

```
mysql> USE essai
Database changed
```

Comme la base vient d'être créée, elle ne contient aucune table, pour s'en convaincre il suffit de taper:

```
mysql> SHOW TABLES;
Empty set (0.00 sec)
```

Pour notre première exemple **Apache+PHP+MySQL**, on va créer la table suivante:

```
mysql> CREATE TABLE coord (
  -> nom VARCHAR(20),
  -> prenom VARCHAR(20),
  -> email VARCHAR(30)
  -> );
Query OK, 0 rows affected (0.03 sec)
```

Jetons un coup d'œil maintenant sur les tables disponibles:

```
mysql> SHOW TABLES;
+-----+
| Tables in essai          |
+-----+
| coord                    |
+-----+
1 row in set (0.00 sec)
```

La table nouvellement créée apparaît bien. Pour avoir le détail de cette table, on tapera:

```
mysql> DESCRIBE coord;
+-----+-----+-----+-----+-----+-----+
| Field  | Type      | Null  | Key  | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nom    | varchar(20) | YES  |     | NULL    |      |
| prenom | varchar(20) | YES  |     | NULL    |      |
| email  | varchar(30) | YES  |     | NULL    |      |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

Pour notre deuxième exemple **Apache+PHP+MySQL**, on créera la table suivante:

```
mysql> CREATE TABLE ref (
  -> date VARCHAR(20),
  -> host VARCHAR(20),
  -> ip VARCHAR(15),
  -> os VARCHAR(20),
  -> page VARCHAR(30)
```

```
-> );  
Query OK, 0 rows affected (0.05 sec)
```

Elle contiendra les informations sur les visiteurs du site. A présent pour quitter tapez simplement quit.

6.5 Tests de fonctionnement MySQL

Comme prérequis on suppose que vous avez installé, configuré **MySQL** et créé les deux exemples du paragraphe MySQL. On suppose que le serveur s'appelle **obelix** et l'utilisateur **olivier**.

A priori il suffit de mettre en place **PHP** avec **Apache** pour que les routines **MySQL** soient prises en compte.

Voici une page écrite en **PHP** qui va accéder à la base de donnée **essai** et à sa table **coord**.

```
<?  
$serveur="localhost";  
$login="olivier";  
$pass="mot-de-passe";  
$base="essai";  
$table="coord";  
  
$id=MYSQL_CONNECT($serveur,$login,$pass);  
mysql_select_db($base);  
$nom="hoarau";  
$prenom="olivier";  
$email="olivier.hoarau@fnac.net";  
$query="INSERT INTO $table VALUES('$nom','$prenom','$email')";  
$result=mysql_query($query,$id);  
echo "Saisie terminée";  
?>
```

Placer ce script dans `~/public_html` et appeler le **bd1.php3**

Dans votre navigateur préféré, dans le champ URL saisissez

http://obelix/~olivier/bd1.php3

A priori y a pas grand chose qui s'est passé, maintenant connecter vous à votre base **essai** dans un shell

```
[olivier@obelix olivier]$ mysql -u olivier -p essai  
Enter password:  
Welcome to the MySQL monitor. Commands end with ; or \g.  
Your MySQL connection id is 10 to server version: 3.22.32
```

Type **'help'** for help.

```
mysql> SELECT * FROM coord;  
+-----+-----+-----+-----+
```

```
| nom | prenom | email |
+-----+-----+-----+
| hoarau | olivier | olivier.hoarau@fnac.net |
+-----+-----+-----+
1 row in set (0.00 sec)
```

C'est bon ça fonctionne. Passons à un exemple plus pointu, on va entrer les informations concernant vos visiteurs dans une base **MySQL**, créer la table telle que décrite dans l'exemple 2 de la page **MySQL**, créer maintenant le script **PHP3**.

<?

```
$page=getenv("HTTP_REFERER");
$ip=getenv( "REMOTE_ADDR");
$host=gethostbyaddr($ip);
$d = date("d/m/Y H:i:s");
$expl=getenv("HTTP_USER_AGENT");

$serveur="localhost";
$login="olivier";
$pass="mot-de-passe";
$base="essai";
$table="ref";

$id=MYSQL_CONNECT($serveur,$login,$pass);
mysql_select_db($base);

$query="INSERT INTO $table VALUES('$d','$host','$ip','$expl','$page)";
$result=mysql_query($query,$id);

echo "$d $host($ip) $expl $page";

?>
```

Nommez ce script **bd2.php3** et placez le dans **~/public_html**. Dans votre navigateur préféré tapez dans le champ URL

http://obelix/~olivier/bd2.php3

Vous devriez voir la date, le nom de votre machine avec son adresse IP et des infos sur votre OS et votre navigateur. A présent connectons nous à la base:

```
[olivier@obelix olivier]$ mysql -u olivier -p essai
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 10 to server version: 3.22.32

Type 'help' for help.
```

```
mysql> SELECT * FROM ref;
```

```
+-----+-----+-----+-----+-----+
| date           | host           | ip           | os           | page |
+-----+-----+-----+-----+-----+
| 24/04/2000 08:34:05 | asterix.armoric.bz | 192.168.13.11 | Mozilla/4.61 [en] (X |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

C'est bon le visiteur a bien été pris en compte.

Maintenant que vous savez comment **Apache** fonctionne avec **MySQL** et **PHP**, laissez libre cours à votre imagination.

6.6 phpMyAdmin

phpMyAdmin est un ensemble de scripts **PHP** qui permet d'administrer des bases **MySQL** à partir d'un navigateur. Vous pouvez le récupérer à l'URL <http://www.phpwizard.net/phpMyAdmin>. En détail **phpMyAdmin** permet de

- créer et supprimer des bases de données,
- éditer, ajouter ou supprimer des champs,
- taper des commandes **SQL**,
- gérer les clés de champs,
- ...

L'archive se présente sous la forme d'un tarball **phpMyAdmin_2_1_0_tar.tar.gz**, pour décompresser:

```
tar xvfz phpMyAdmin_2_1_0_tar.tar.gz
```

Cela va créer dans le répertoire de travail un répertoire **phpMyAdmin**. Dans ce répertoire on va éditer le fichier **config.inc.php3**, on va utiliser la méthode d'authentification "basique", pour cela au lieu de **root**, saisissez l'utilisateur que vous avez précédemment créé en installant

MySQL.

```
$cfgServers[1]['adv_auth'] = false;// Use advanced authentication?
$cfgServers[1]['stduser'] = 'root';// MySQL standard user (only needed with advanced a
uth)
$cfgServers[1]['stdpass'] = '';// MySQL standard password (only needed with advanced
auth)
$cfgServers[1]['user'] = 'olivier';// MySQL user(only needed with basic auth)
$cfgServers[1]['password'] = 'mot-de-passe';//
MySQL password (only needed with basic auth)
```

Et maintenant pour avoir la version française, dans le même fichier au lieu de:

```
require("english.inc.php3");
```

On va mettre

```
require("french.inc.php3");
```

Maintenant on doit rendre accessible le répertoire **phpMyAdmin** d'une page web, pour cela deux solutions:

- (solution simple) placer **phpMyAdmin** dans **/usr/local/apache/htdocs** et au niveau de la page d'accueil d'**Apache** faire un lien vers **/home/httpd/phpMyAdmin/index.php3**
- (solution préconisée), créer un hôte virtuel pointant vers **./phpMyAdmin** qu'on appellera **www.sql.breizland.bz**.

NOTE Si ça vous gêne que n'importe qui d'un navigateur puisse aller dans le répertoire **phpMyAdmin**, mettez y des restrictions d'accès avec un fichier **.htaccess**.

Avec la solution hôte virtuel, à partir d'un navigateur quand on sélectionne **www.sql.breizland.bz** on tombe sur une fenêtre avec frame avec à gauche la liste des bases de données disponibles (on retrouve les deux bases de données créées précédemment dans nos exemples) et à droite un menu d'administration.



Si on sélectionne **essai** par exemple et plus particulièrement la table **coord**, le frame de droite disparaît pour laisser la place à une autre page permettant de créer des nouvelles tables, faire des requêtes **SQL**, etc.

Accueil
 essai
 coord
 ref
 mysql
 mysql.orgine
 test

Base de données essai – table coord

| Champ | Type | Attributes | Null | Defaut | Extra | Action | | | | |
|--------|-------------|------------|------|--------|-------|----------|---------|----------|-------|--------|
| nom | varchar(20) | | Oui | | | Modifier | Effacer | Primaire | Index | Unique |
| prenom | varchar(20) | | Oui | | | Modifier | Effacer | Primaire | Index | Unique |
| email | varchar(30) | | Oui | | | Modifier | Effacer | Primaire | Index | Unique |

- Afficher
- Sélectionner
- Insérer

Ajouter un champ:

- Insérer un fichier texte dans la table
- Afficher le schéma de la table
 - Structure seule Ajouter des énoncés 'drop table'
 - Structure et données transmettre
 - Données CSV terminé(e)s par

Changer le nom de la table pour:

Copier la table vers:

Structure seule Structure et données

7 Scripts CGI

Pour activer les scripts CGI, veuillez vous référer au fichier <httpd.conf>, je n'ai seulement que rajouté la ligne suivante:

```
ScriptAlias /cgi-bin/ /var/www/cgi-bin/
```

On crée donc un alias pour accéder au répertoire des scripts CGI. On relance **Apache** pour prendre en compte cet alias.

```
/etc/rc.d/init.d/httpd restart
```

Le but de l'exercice est de créer un script perl CGI qui va traiter un formulaire quelconque d'une page HTML. Vous allez créer votre script perl sous **/var/www/cgi-bin/**, et le nommer **form.pl**, voici son contenu:

```
#!/usr/bin/perl
use CGI;
$html=new CGI;
print $html->header;
```

```

print "<HTML>\n";
print "<HEAD>\n";
print "<TITLE>Premier script CGI perl</TITLE>\n";
print "</HEAD>\n";
print "<BODY>\n";
print "<H1>Traitement du formulaire</H1>\n";

```

```

print "Nom :";
print $html->param('nom');
print "<p>\n";
print "Email :";
print $html->param('email');
print "<p>\n";
print "Commentaire:";
print $html->param('comment');

```

```

print "</BODY>\n";
print "</HTML>\n";

```

Donner les droits qui vont bien avec ce fichier:

chmod 755 form.pl

En tant qu'utilisateur standard (**olivier** dans notre exmple), créer maintenant le fichier HTML suivant que vous appellerez **formulaire.htm**

```

<html>
<body>
<h2>Formulaire</h2>
<form action="http://obelix/cgi-bin/form.pl" METHOD=GET>
Nom: <input type="text" name=nom size=20><br>
Email: <input type="text" name=email size=30><br>
Commentaire: <input type="text" name=comment size=100><br>
<input type=submit value="Envoyer"> <input type=reset value="remettre à zéro">
</form>
</body>
</html>

```

Voilà maintenant quand vous allez accéder à **http://obelix/~olivier/formulaire.htm**, vous allez avoir une page du style:

Nom:

Email:

Commentaire:

En appuyant sur **Envoyer** ça va déclencher l'exécution du script CGI perl, qui va provoquer l'affichage des valeurs précédemment saisies.

8 PHP et LDAP

Voilà un petit programme qui va nous permettre de rajouter une entrée dans la base **LDAP**, libre à vous maintenant de créer des formulaires de saisie, de destruction, et de recherche:

```
<?
// nom du serveur LDAP
$server="asterix";

// identification de l'administrateur de la base
$rootdn="cn=Manager, dc=breizland, dc=bz";

//mot de passe administrateur
$rootpw="secret";

//connexion à la base
$result=ldap_connect($server);
if ($result==1)
{
    ldap_bind($result,$rootdn,$rootpw);
}

//Enregistrement d'une entrée dans la base
echo"Enregistrement de Benjamin Hoarau\n";

$dn="dc=breizland, dc=bz";
$nom="Hoarau";
$prenom="Benjamin";

$info["cn"]=$nom." ".$prenom;
$info["sn"]=$nom;
$info["objectclass"]="person";

ldap_add($result,"cn=$nom $prenom,$dn",$info);
ldap_close($result);

echo "L'enregistrement a réussi";
?>
```

Appelez ce fichier **ldap.php3**, vous pouvez le tester et vérifier que l'entrée a bien été saisie dans la base.