Solution LAMP Ubuntu > Dapper 6.06

Éditer

Description du tutoriel

Ce tutorial traite de la procédure à suivre pour installer une solution **LAMP (Linux, Apache, MySQL, Php/Perl/Python)** sur un système (machine) disposant d'**Ubuntu Dapper Drake (6.06 LTS)** ou **Edgy Eft (6.10)** en mode graphique.

Il s'adresse donc plus particulièrement aux nouveaux utilisateurs d'**Ubuntu** qui désirent rester en mode graphique à défaut de bien connaitre l'administration d'un serveur en lignes de commandes (*shell*).

Bien entendu, il s'adresse tout aussi bien aux développeurs professionnels désireux de travailler avec Ubuntu sur leur portable…

N.B. : Vous pouvez télécharger ce tutoriel au format PDF (*mise à jour le 27/12/2006*) en cliquant sur le lien suivant : tutoriel « Ubuntu Dapper Drake - Solution LAMP »

Éditer

Indispensable

La procédure traitée dans cetutoriel est à exploiter sur une nouvelle installation d'Ubuntu Dapper Drake ou sur un installation sur laquelle aucune solution LAMP n'a encore été installée.

Vous ne devez pas avoir choisi l'option LAMP à l'installation d'Ubuntu.

Éditer

Référence des logiciels installés via cette procédure

- **Apache2** (2.0.55-4ubuntu2) ;
- **Mysql-5.0** (5.0.21-3ubuntu1) ;
- **PHP5** (5.1.2-1ubuntu3) ;
- **PhpMyAdmin** (4:2.8.0.3-1) ;
- et divers modules pour PHP 5.

Étant précisé que la version des dits logiciels peut évoluer.

Éditer

I. Vérification des dépôts actifs et rechargement des informations relatives aux paquets

Éditer

A. Activation des dépôts nécessaires

Voici la liste des dépôts qui doivent être actifs pour l'installation de cette solution **LAMP**.

1. Ubuntu Dapper Drake

```
deb cdrom:[Ubuntu 6.06 _Dapper Drake_ - Alpha i386 (20060504)]/ dapper main
restricted

deb http://fr.archive.ubuntu.com/ubuntu/ dapper main restricted
deb-src http://fr.archive.ubuntu.com/ubuntu/ dapper main restricted

deb http://fr.archive.ubuntu.com/ubuntu/ dapper universe
deb-src http://fr.archive.ubuntu.com/ubuntu/ dapper universe
```

2. Ubuntu Edgy Eft

```
# deb cdrom:[Ubuntu 6.10 _Edgy Eft_ - Release i386 (20061025.1)]/ edgy main
restricted

deb cdrom:[Ubuntu 6.10 _Edgy Eft_ - Release i386 (20061025.1)]/ edgy main
restricted

deb http://fr.archive.ubuntu.com/ubuntu/ edgy main restricted
deb-src http://fr.archive.ubuntu.com/ubuntu/ edgy main restricted

deb http://fr.archive.ubuntu.com/ubuntu/ edgy-updates main restricted
deb-src http://fr.archive.ubuntu.com/ubuntu/ edgy-updates main restricted

deb http://fr.archive.ubuntu.com/ubuntu/ edgy universe
deb-src http://fr.archive.ubuntu.com/ubuntu/ edgy universe

deb http://security.ubuntu.com/ubuntu edgy-security main restricted
deb-src http://security.ubuntu.com/ubuntu edgy-security main restricted
deb http://security.ubuntu.com/ubuntu edgy-security universe
deb-src http://security.ubuntu.com/ubuntu edgy-security universe
```

Normalement, si vous n'avez pas modifié la configuration des dépôts depuis que vous avez installé Ubuntu, seuls les dépôts *universe* doivent être décommentés dans votre fichier « sources.list ».

De même, il est vivement recommandé de désactiver tous les autres dépôts afin d'éviter tous problèmes de dépendances, notamment, si les deux dépôts suivants s'y trouvent, il est fortement conseillé de les commenter (désactiver).

```
deb http://dotdeb.thefox.com.fr stable all
deb-src http://dotdeb.thefox.com.fr stable all
```

Le fichier « sources.list » se trouve dans le répertoire « /etc/apt ».

Pour décommenter (activer les dépôts) procédez de cette manière si vous utilisez Gedit :

```
gksudo gedit /etc/apt/sources.list
```

ou si vous voulez le faire directement dans la console :

```
sudo nano /etc/apt/sources.list
```

(Note : Si vous utilisez Kubuntu, utiliser *kdesu* au lieu de *gksudo* et *kate* au lieu de *gedit*)

Une fois que le fichier « sources.list » est ouvert, enlevez le signe « # » qui est situé devant les deux dépôts concernés et terminez en le sauvegardant.

Pour commenter (désactiver), ajoutez le signe « # » devant les dépôts concernés.

Éditer

B. Rechargement des informations relatives aux paquets

Après avoir activé les dépôts nécessaires à l'installation de la solution LAMP, il faut que vous rechargiez les informations relatives aux paquets disponibles.

Pour ce faire, tapez la commande suivante dans un terminal :

```
sudo apt-get update
```

Éditer

II. Vérification de la manière dont sont traitées les dépendances des paquets recommandés

Avant de commencer la procédure d'installation, il faut que vous vérifiez la manière dont sont traitées les dépendances des paquets recommandés :

Pour ce faire, ouvrez le gestionnaire de paquets Synaptic (**Système → Administration → Gestionnaire de paquets Synaptic**).

Une fois que Synaptic est ouvert, cliquez sur (**Configuration → Préférences**) et dans la nouvelle fenêtre qui s'ouvre, dans l'onglet **Général**, vérifiez que l'option « Traiter les paquets recommandés comme des dépendances » est bien activée.Si ce n'est pas le cas, activez-la et rechargez les informations relatives aux paquets comme vu précédemment ou *via* Synaptic directement.

Éditer

III. Installation d'Apache2

Pour installer le serveur Web Apache2, il vous suffit de taper la commande suivante dans un terminal :

```
sudo apt-get install apache2 apache2-doc apache2-mpm-prefork
```

Ceci aura pour effet d'installer Apache dans le répertoire « /etc/apache2 » ainsi que sa documentation.

Il se peut que pendant l'installation, votre CD d'Ubuntu vous soit demandé. Pour éviter cela, rééditez le fichier « /etc/apt/sources.list » et commentez les lignes faisant références au CD-ROM. Ceci étant effectué, sauvegardez le fichier et rechargez les informations relatives aux paquets disponibles en tapant la commande suivante dans un terminal :

```
sudo apt-get update
```

Éditer

IV. Installation de MySQL 5.0

Pour installer le serveur **Mysql-5.0**, il vous suffit de taper la commande suivante dans un terminal :

```
sudo apt-get install mysql-server-5.0
```

Éditer

V. Installation de PHP 5 (avec modules pour Apache et MySQL)

Pour installer PHP 5 et quelques modules supplémentaires pour **Apache2** et **MySQL**, il vous suffit de

taper la commande suivante dans un terminal :

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

Pendant l'installation, vous pouvez voir apparaitre quelques erreurs, notamment lorsque Apache2 va être redémarré, vous allez surement rencontrer une erreur de ce genre :

```
apache2: Could not determine the server's fully qualified domain name...
```

Ceci n'est absolument pas gênant et peut être réglé très facilement. Il vous suffit de rajouter la directive **ServerName** dans le fichier « /etc/apache2/apache2.conf».

Exemple :

```
ServerName www.monsite.tld
```

Éditer

VI. Pré-vérifications

À ce stade de l'installation, il est conseillé de procéder à quelques vérifications, notamment, vérifier :

- le bon fonctionnement d'**Apache**
- le bon fonctionnement de **PHP5**

Avant tout, vous devriez redémarrer le serveur Web **Apache2** pour que les changements soient pris en compte.

Pour ce faire, il vous suffit de taper la commande suivante dans un terminal :

```
sudo /etc/init.d/apache2 reload
```

Éditer

A. Vérification du bon fonctionnement d'Apache

Dans la barre de votre navigateur internet (Firefox, etc.), tapez l'URL suivante :

```
http://127.0.0.1/
```

ou son équivalent :

```
http://localhost/
```

Précision : *localhost* est en réalité *un alias* de l'adresse IP 127.0.0.1 qui représente votre machine.

Si tout s'est bien passé, vous devriez voir une page Web dans laquelle l'index du répertoire Web apparait ainsi que le dossier « apache2-default ».

Éditer

B. Vérification du bon fonctionnement de PHP5

a. Création du fichier « phpinfo.php »

Pour vérifier que PHP 5 a bien été installé et est fonctionnel, vous allez devoir créer un fichier d'usage, le fameux « phpinfo.php ».

Pour ce faire, créez un nouveau document que vous nommerez « phpinfo.php » dans le répertoire « /var/www » grâce à la commande.

```
gksudo gedit /var/www/phpinfo.php
```

ou autre éditeur de texte (nano, kedit, vim, emacs, etc.)

Insérez-y le code suivant :

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

et finissez en enregistrant le document.

b. Exécution du fichier « phpinfo.php »

Le fichier « phpinfo.php » étant créé, exécutez-le.

Pour exécuter le fichier « phpinfo.php », il vous suffit de tapez l'URL suivante dans la barre d'adresse de votre navigateur internet :

```
http://127.0.0.1/phpinfo.php
```

ou celle-ci :

```
http://localhost/phpinfo.php
```

si vous voyez apparaitre :

```
Warning: Unknown: failed to open stream: Permission denied in Unknown on line 0

Warning: Unknown: Failed opening '/var/www/phpinfo.php' for inclusion
(include_path='.:/usr/share/php:/usr/share/pear') in Unknown on line 0
```

Il s'agit d'une erreur qui est simplement due aux droits actuels du fichier que vous venez de créer.

Pour résoudre cette erreur, il suffit donc de changer les droits du fichier.

Actuellement le fichier « phpinfo.php » peut seulement être lu, exécuté et modifié par l'utilisateur *root*. Vous comprendrez donc qu'il ne peut être exécuté par les autres utilisateurs.

Il faut donc que vous le rendiez accessible, en lecture et exécution par les utilisateurs autres que *root*. Pour ce faire, il vous suffit de lui appliquer un `chmod 605` en tapant la commande suivante dans un terminal :

```
sudo chmod 605 /var/www/phpinfo.php
```

Cette modification étant effectuée, réactualisez la page.

Si votre navigateur vous demande de télécharger le fichier « phpinfo.php », cela peut venir d'un problème de module mal chargé. Dans ce cas, vous pouvez tenter de résoudre cette « erreur » en tapant les commandes suivantes pour activer le module php5 :

```
sudo a2enmod php5
```

et

```
sudo /etc/init.d/apache2 force-reload
```

Si tout s'est bien passé, vous devriez voir apparaitre une page Web dans laquelle se trouvent plusieurs informations, notamment celles liées à la version de PHP utilisée sur votre système. Si ce n'est pas le cas, essayez de redémarrer votre ordinateur.

N.B. : Par la suite, lorsque vous allez créer des **pages HTML** ou des **scripts PHP**, pensez à appliquer les bons droits (permissions) sur les répertoires et fichiers de sorte qu'ils puissent être lus et exécutés. Ceci est une source d'erreurs très répandue chez les nouveaux utilisateurs Linux.

D'une manière générale, un `chmod 755` convient à la plupart des configurations.

Éditer

VII. Installation de quelques modules supplémentaires pour PHP 5

Dans la plupart des cas, des modules supplémentaires pour PHP 5 sont nécessaires au bon fonctionnement des applications Web l'utilisant.

Vous allez donc installer quelques modules qui sont souvent nécessaires.

Dans un terminal, tapez la commande suivante :

```
sudo apt-get install php-pear php5-cli php5-gd php5-sqlite php5-xsl php5-mcrypt
```

N.B. : D'autres modules pour PHP5 sont disponibles. Vous pouvez voir lesquels en exécutant le gestionnaire de paquets et en faisant une recherche sur PHP 5.

Éditer

VIII. Sécurisation de MySQL

Pour l'instant, MySQL est exécuté sous son utilisateur *root* sans mot de passe.

Il est donc nécessaire de sécuriser un tant soit peu l'installation du dit serveur.

Dans un terminal, tapez la commande suivante :

```
sudo mysql_secure_installation
```

Une fois cette commande entrée, le script de sécurisation du serveur MySQL démarre.

Ce script vous pose plusieurs questions :

```
Enter current password for root (enter for none):
```

Ici, le script vous demande d'entrer le mot de passe courant de l'utilisateur *root*. N'en ayant pas encore défini, vous devez simplement taper sur la touche « Entrée ».

```
Setting the root password ensures that nobody can log into the MySQL
root user without the proper authorisation.

Set root password? [Y/n]
```

Ici, il vous est demandé si vous voulez attribuer un mot de passe à l'utilisateur *root*. Vous devez donc taper la lettre « Y ».

```
New password:
```

Bien ici, c'est assez simple, vous devez taper le mot de passe que vous voulez attribuer à l'utilisateur *root* de MySQL. Attention à bien le choisir et de vous en rappeler. Il sera aussi utilisé pour la connexion à **PhpMyAdmin**.

```
By default, a MySQL installation has an anonymous user, allowing anyone
to log into MySQL without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n]
```

Ici, il vous est indiqué qu'à l'installation de MySQL, un utilisateur anonyme a été créé. Ceci étant

dangereux pour un environnement de production, tapez sur la lettre « Y » pour le supprimer.

```
Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n]
```

Si vous n'avez pas besoin d'administrer votre base de données à distance, ce qui devrait être le cas de la plupart des utilisateurs, tapez sur la lettre « Y ». Ainsi, l'utilisateur *root* ne pourra se connecter qu'en local (la machine sur laquelle MySQL est installé).

```
Remove test database and access to it? [Y/n]
```

Ici, contentez-vous de taper sur la touche « Entrée ».

```
Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n]
```

Entrez « Y » pour prendre en compte les changements que vous venez de faire et les appliquer immédiatement.

Voilà pour ce qui est de la sécurisation de MySQL.

Éditer

IX. Installation de PhpMyAdmin (PMA)

 Il est clairement rappelé que pour pouvoir installer le logiciel **PhpMyAdmin**, il faut que les **dépôts universe** soient décommentés dans votre fichier « sources.list ».

Pour installer **PhpMyAdmin**, il vous suffit de taper la commande suivante dans un terminal :

```
sudo apt-get install phpmyadmin
```

N.B. : Il est important d'installer **PhpMyAdmin** en dernier dans le cadre de ce tutorial et il est aussi important d'avoir défini un mot de passe pour l'utilisateur *root* de MySQL avant de l'installer. À défaut, vous risqueriez de rencontrer des problème avec **PhpMyAdmin**.

Éditer

X. Vérification du bon fonctionnement de PHPMyAdmin

Pour vérifier le bon fonctionnement de phpmyadmin, tapez ceci dans la barre de votre navigateur

```
http://127.0.0.1/
```

ou ceci :

```
http://localhost/
```

Et ensuite, dans la page web qui apparaît, cliquez sur le répertoire **phpmyadmin**.

Vous pouvez aussi taper l'url de **PhpMyAdmin** directement dans la barre de votre navigateur :

```
http://127.0.0.1/phpmyadmin/
```

ou ceci

```
http://localhost/phpmyadmin/
```

ce qui donnera le même résultat, à savoir, l'affichage d'une nouvelle page Web qui n'est autre que la page d'authentification de PhpMyAdmin.

Pour vous connecter à PhpMyAdmin, faites comme ceci :

```
Nom d'utilisateur
```

le nom de l'utilisateur MySQL, il s'agit de l'utilisateur *root*.

```
Mot de passe :
```

Le mot de passe que vous avez défini plus haut pour l'utilisateur *root* de MySQL.

Si tout s'est bien passé, vous devriez être connecté à **PhpMyAdmin** et pouvoir créer/gérer vos bases de données.

N.B. : Il existe déjà deux bases de données, n'y touchez pas !!!.

N.B. : Si votre navigateur vous propose de télécharger un fichier PHTML au lieu d'afficher PHPMyAdmin, cela vient sûrement d'un problème de module non chargé.

Dans ce cas, vous pouvez tenter de résoudre cette "erreur" en tapant les commandes suivantes dans un terminal :

1. On active le module PHP 5 :
   ```
   sudo a2enmod php5
   ```
2. On redémarre Apache :
   ```
   sudo /etc/init.d/apache2 force-reload
   ```

N.B. : N'oubliez pas de vider le cache de votre navigateur et de le fermer complètement avant de recommencer.

Éditer

POUR ALLER PLUS LOIN

Avant-propos

Toutes les sections suivantes pourraient êtres résumées en 5 ou 10 lignes tout au plus. Ce faisant, mon but n'est pas de donner des solutions exemptes de toutes explications. Mon but est de fournir des explications qui permettrons à l'utilisateur final de bien comprendre le fonctionnement du **Serveur Web Apache2** afin qu'il puisse, de lui-même, répéter les méthodes sans recourir à chaque ré-installation, à ce document.

Éditer

I. Les jeux de caractères (encodages) du Serveur Web Apache2

Par défaut, lorsqu'on installe le **Serveur Web Apache2**, c'est le jeu de caractères **UTF-8** qui est utilisé. Ceci peut être vérifié en éditant le fichier **charset** qui se trouve dans le dossier **/etc/apache2/conf.d** .

Ce fichier contient la ligne suivante :

```
AddDefaultCharset UTF-8
```

<RAPPEL>

La directive **AddDefaultCharset** spécifie le nom du jeu de caractères qui sera ajouté à toutes les réponses qui n'ont aucun paramètre sur le type de contenu dans l'en-tête HTTP. Elle remplace le jeu de caractères spécifié dans le corps du document Web par l'inclusion du marqueur **META** (ex :**<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">**.

Pour exemple, si la directive **AddDefaultCharset** est présente dans les fichiers de configuration du **Serveur Web Apache2** avec l'argument **UTF-8**, vous aurez beau inclure le marqueur **META** (ex : **<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">**) dans le corps de vos documents Web, cela ne changera rien au problème d'encodage puisque cette directive supplentera le dit marqueur.

</RAPPEL>

Il en résulte qu'à défaut d'avoir formaté les documents Web avec le jeu de caractères **UTF-8**, les accents ne pourront s'afficher correctement lors du traitement desdits documents par le **Serveur Web Apache2**.

Pour exemple, le **é** sera encodé de cette manière : **&#65533;**

<NOTE EXPLICATIVE>

Le contenu du fichier **charset** mentionné ci-dessus est inclu dans la configuration du **Serveur Web Apache2** grâce à la directive **Include** ⇒ **Include /etc/apache2/conf.d/[^.#]*** .

Cette directive et l'argument qui l'accompagne font en sorte que tout le contenu des fichiers qui se trouve dans le répertoire **/etc/apache2/conf.d** soit inclu à la configuration du **Serveur Web Apache2**.

</NOTE EXPLICATIVE>

Pour résoudre ce problème plusieurs solutions s'offrent à vous :

Éditer

A. Laisser le navigateur Web choisir l'encodage approprié

Comme nous l'avons vu, par défaut, il est demandé aux **Serveur Web Apache2** d'utiliser le jeu de caractères **UTF-8** lors du traitement des documents Web. Nous pouvons aisément désactiver cette fonction ce qui aura pour effet de laisser le navigateur Web choisir le jeu de caractères à employer.

Pour désactiver cette fonction, deux solutions existent :

1. Vous commentez la directive d'inclusion **Include /etc/apache2/conf.d/[^.#]*** qui se trouve dans le fichier **/etc/apache2/apache2.conf**.

ou

2. Vous commentez la ligne **AddDefaultCharset UTF-8** qui se trouve dans le fichier **/etc/apache2/conf.d/charset** .

<RAPPEL>

Pour commenter il suffit simplement de rajouter le signe # devant le code. Pour décommenter, il suffit simplement de retirer le signe # qui se trouve devant le code.

</RAPPEL>

<ATTENTION>

Si vous optez pour la première solution, tout le contenu des fichiers qui se trouvent dans le dossier **/etc/apache2/conf.d** ne sera plus pris en compte par le **Serveur Web Apache2**. Il est donc préférable de choisir la deuxième solution.

</ATTENTION>

Éditer

B. Changer le jeu de caractères utilisé par défaut par le Serveur Web Apache2

Au lieu de désactiver la fonction de définition du jeu de caractères utilisé par défaut par le **Serveur Web Apache2**, nous pouvons définir celui que nous désirons employer.

Là encore, il existe plusieurs solutions pour le faire.

1. En modifiant le fichier **charset** du répertoire **/etc/apache2/conf.d**.

Comme nous l'avons vu, par défaut, le **Serveur Web apache2** utilise le jeu de caractères **UTF-8**, ce dernier étant défini par le contenu du fichier **charset** qui se trouve dans le répertoire **/etc/apache2/conf.d** lui-même intégré à la configuration du **Serveur Web Apache2** grâce à la directive Include ⇒ **Include /etc/apache2/conf.d/[^.#]*** .

Pour modifier le jeu de caractères utilisé par défaut par le **Serveur Web Apache2**, il vous suffit donc d'éditer le fichier **charset** et de remplacer la ligne :

```
AddDefaultCharset UTF-8
```

par celle de votre choix.

Exemple pour l'encodage **ISO-8859-1** :

```
AddDefaultCharset ISO-8859-1
```

2. En modifiant le fichier **apache2.conf**

Si vous éditez votre fichier **apache2.conf** qui se trouve dans le répertoire **/etc/apache2**, vous pourrez y trouver la directive et l'argument suivants :

```
#AddDefaultCharset      ISO-8859-1
```

Comme vous pourrez le constater, cette ligne est commentée par défaut. Si vous désirez que le jeu de caractères **ISO-8859-1** soit utilisé par défaut par le **Serveur Web Apache2**, il vous suffit donc de décommenter cette ligne.

Cette ligne deviendra donc :

```
AddDefaultCharset       ISO-8859-1
```

<NOTE EXPLICATIVE>

A ce stade, et si vous avez bien suivi mes explications depuis le début, vous devriez vous poser la question suivante :

Si je décommente la ligne **#AddDefaultCharset ISO-8859-1** qui se trouve dans le fichier **/etc/apache2/apache2.conf**, que deviendra la directive **AddDefaultCharset** qui se trouve dans le fichier **/etc/apache2/conf.d/charset** ?

Et bien la réponse est simple.

Lorsque vous demandez au **Serveur Web apache2** de relire sa configuration, il suit un ordre qui est celui du plus court chemin. Autrement dit, si deux directives ayant la même fonction sont déclarées, c'est la dernière qui sera prise en compte par le **Serveur Web Apache2**.

Dans notre cas, le contenu fichier **/etc/apache2/conf.d/charset** et inclut dans le fichier **/etc/apache2/apache2.conf** avant la déclaration de la directive **AddDefaultCharset ISO-8859-1**. C'est pour cette raison que ce sera cette dernière directive qui sera prise en compte par le **Serveur Web Apache2** si nous la décommentons.

Attention tout de même, car ceci n'est pas vérifié pour toutes les directives. Dans le cas présent, cette directive s'applique à la configuration générale du **Serveur Web Apache2**. Il en serait autrement s'il s'agissait d'une directive propre à un **Virtualhost** où encore, à un répertoire Web (<Directory > ... </Directory>)...

</NOTE EXPLICATIVE>

<RAPPEL>

Comme usuellement, il convient de demander au **Serveur Web Apache2** de relire sa configuration après avoir effectué des modifications dans ses fichiers de configuraton pour que ces dernières soient prises en comptes.

Pour ce faire, il vous suffit de taper la commande suivante dans un terminal :

```
sudo /etc/init.d/apache2 reload
```

</RAPPEL>

Éditer

II. Les Hôtes Virtuels (VirtualHost) - Méthodes

Introduction

Le Serveur Web Apache2 est capable de gérer simultanément plusieurs arborescences Web grâce à la notion d'hôtes Virtuels (Virtual Hosts). Dans ce tutorial, nous vous proposons de prendre connaissance des divers méthodes existantes pour mettre en place vos Sites Virtuels (VirtualHost).

Dans le cadre de ce tutorial, nous renseignons le fichier **/etc/hosts** afin d'assurer la résolution de nom (Nom –> Adresse Ip) des **hôtes Virtuels** que nous allons créer. Ceci est nécessaire que si votre serveur DNS (Serveur Bind) n'est pas installé et/ou configuré pour vos domaines ou encore, que les noms DNS attribués aux **hôtes Virtuels** sont purement fictifs ou ne vous appartiennent pas.

Si vous désirez tester toutes les méthodes proposées, pensez à effacer la configuration précédente pour ne pas créer des conflits entre les hôtes Virtuels. Il en va de même pour le fichier **/etc/hosts**.

Comme évoqué ci-dessus, le **Serveur Web Apache2** peut gérer plusieurs **arborescences Web** grâce à la notion d'**hôtes Virtuels** (*VirtualHost*).

Pour cela, plusieurs méthodes existent :

1. Les Hôtes Virtuels **basés sur l'adresse Ip** ;
2. Les Hôtes Virtuels **basés sur le numéro de port** ;
3. Les Hôtes Virtuels **basés sur le nom**.

Éditer

1. Les Hôtes Virtuels basés sur l'adresse Ip :

Dans le cadre de cette méthode, le Serveur est soit doté de plusieurs interfaces réseau (plusieurs cartes réseau), soit doté de plusieurs adresses Ip asociées à une seule interface réseau (une seule carte réseau). Dans ce dernier cas, on parlera d'**Ip aliasing**. Les systèmes Linux, notamment les distributions Ubuntu et Débian, permettent de mettre facilement en oeuvre cette dernière fonctionnalité (Ip aliasing).

Éditer

A : Association d'une deuxième adresse Ip à une carte Réseau (Ip aliasing)

Dans cet exemple, nous partirons du principe que nous possédons une seule interface réseau (une seule carte réseau) à laquelle nous voulons associer une seconde adresses Ip.

Dans un premier temps, nous tapons la commande suivante dans un terminal pour prendre connaissance de la configuration de notre interface réseau :

```
sudo ifconfig'
```

et voici le résultat que nous obtenons :

```
coucou@serveur:~$ ifconfig
eth0      Lien encap:Ethernet  HWaddr 00:13:D3:3C:58:84
          inet adr:192.168.0.2  Bcast:192.168.0.255  Masque:255.255.255.0
          adr inet6: fe80::213:d3ff:fe3c:5884/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Packets reçus:1257 erreurs:0 :0 overruns:0 frame:0
          TX packets:1247 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          Octets reçus:1037747 (1013.4 KiB) Octets transmis:423117 (413.2 KiB)
          Interruption:209 Adresse de base:0x8000

lo        Lien encap:Boucle locale
          inet adr:127.0.0.1  Masque:255.0.0.0
          adr inet6: ::1/128 Scope:Hôte
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          Packets reçus:429 erreurs:0 :0 overruns:0 frame:0
          TX packets:429 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          Octets reçus:117583 (114.8 KiB) Octets transmis:117583 (114.8 KiB)
```

Ce qui nous intéresse ici c'est la première série d'informations. Elle nous indique que nous disposons d'une interface réseau de type **Ethernet** d'adresse Ip **192.168.0.2**.

Nous disposons donc d'une seule et unique **adresse Ip** qui est **192.168.0.2**.

Nous devons donc associer une seconde adresse Ip à notre interface réseau. Pour ce faire, il nous suffit simplement de taper la commande suivante dans un terminal :

```
sudo ifconfig eth0:0 192.168.0.100
```

Grâce à la commande évoquée ci-dessus, nous avons associé une seconde adresse Ip =⇒ **192.168.0.100** à notre interface réseau (carte réseau).

Dès à présent, notre interface réseau dispose de deux adresse Ip bien distinctes : **192.168.0.2** (Adresse Ip)

et **192.168.0.100** (Alias Ip)

Ceci est vérifiable en tapant la commande suivante dans un terminal :

```
sudo ifconfig -a
```

Éditer

B : Configuration des Hôtes Virtuels

Ayant associé une nouvelle adresse Ip à notre unique carte réseau, nous pouvons désormais créer nos deux **Hôtes Virtuels** (sites virtuels) utilisant chacun une adresse Ip bien distincte.

Dans un premier temps, nous renseignons le fichier **/etc/hosts** avec deux noms de sites (ceux de nos hôtes Virtuels) associés respectivement aux deux adresses Ip disponibles afin de permettent la

résolution de nom (Nom –> Adresse Ip).

Exemple de fichier hosts :

```
127.0.0.1       localhost
192.168.0.2     virtual1.com
192.168.0.100   virtual2.com
```

Une fois notre fichier **/etc/hosts** correctement renseigné, il ne nous reste plus qu'à configurer nos **Hôtes Virtuels**.

Pour ce faire, il nous suffit de créer deux nouveaux fichiers dans le répertoire **/etc/apache2/sites-available** :

1er fichier de configuration :

Dans cet exemple, le premier fichier de configuration se nomme : **virtual1.conf**

Ce fichier contient le contenu minimal suivant :

```
<VirtualHost 192.168.0.2:80>
        DocumentRoot    /var/www/virtual1.com
        ServerName      virtual1.com
</virtualHost>
```

2ème fichier de configuration :

Dans cet exemple, le deuxième fichier de configuration se nomme : **virtual2.conf**

Ce fichier contient le contenu minimal suivant :

```
<VirtualHost 192.168.0.100:80>
        DocumentRoot /var/www/virtual2.com
        ServerName virtual2.com
</VirtualHost>
```

Note explicative

Dans chacun des fichiers de configuration, nous devons prévoir une strophe **<VirtualHost>** avec les deux directives de base : **DocumentRoot** et **ServerName**.

La directive **DocumentRoot**, à ne pas confondre avec la directive **ServerRoot**, permet de déclarer l'emplacement de l'arborescence Web du **hôte virtuel** qui sera accessible par les clients. Le nom du répertoire ne doit pas comporter le caractère / final.

La directive **ServerName** permet de définir le **nom DNS** du Serveur ainsi que le **port** utilisé par défaut. Dans le cadre d'**Hôtes Virtuels**, on parlera de **Serveurs Virtuels**.

Ps : Les répertoires **/var/www/virtual1.com** et **/var/www/virtual2.com** doivent exister et doivent bénéficier des bonnes permissions. A défaut, le **Serveur Web Apache2**, lors du **reload (re-lecture de ses fichiers de configurations)**, renverra des erreurs. Pour les tests, vous pouvez appliquer un **chmod 755** sur ces répertoires.

Nos deux **hôtes Virtuels** étant configurés, nous devons dès à présent les activer. Pour ce faire, nous tapons les commandes suivantes dans un terminal :

```
sudo a2ensite virtual1.conf
```

et

```
sudo a2ensite virtual2.conf
```

Note explicative : La commande **a2ensite** a pour effet de créer des liens symboliques des deux fichiers de configuration (VirtualHost) créés précédement, liens qui seront placés dans le répertoire **/etc/apache2/sites-enabled**, ce qui permettra au **Serveur Web Apache2** de les interpréter grâce à la

directive d'inclusion =⇒ **Include /etc/apache2/sites-enabled/[^.#]*** qui elle, se trouve dans le fichier de configuration principale (**apache2.conf**).

Enfin, pour que les modifications soient prises en comptespar le **Serveur Web Apache2**, nous demandons à ce dernier de relire ses fichiers de configuration. Pour ce faire, il nous suffit de taper la commande suivante dans un terminal :

```
sudo /etc/init.d/apache2 reload
```

Éditer

2. Hôtes Virtuels basés sur le numéro de port :

Par défaut, le **Serveur Web Apache2** est configuré pour écoutersur le **port 80**. Ce faisant, il est possible d'utiliser des ports bien distincts pour chaques **hôtes Virtuels**.

Exemple :

Dans cette exemple, les deux Sites Virtuels **virtual1.com** et **virtual2.com** se partagent la même adresse Ip : **192.168.0.2**. Seul le port d'écoute est différent.

Dans un premier temps, nous allons éditer le fichier **/etc/hosts** afin de donner deux noms bien distincts à la seule et unique adresse Ip de notre interface réseau. Ceci permettra la résolution de nom (Nom –> Adresse Ip).

Exemple de fichier hosts :

```
127.0.0.1       localhost
192.168.0.2     virtual1.com
192.168.0.2     virtual2.com
```

Ensuite, nous créons deux fichiers de configuration (**VirtualHost**) que nous plaçons dans le répertoire **/etc/apache2/sites-available**.

Voici ce que contiennent ces deux fichiers deconfiguration : **1er fichier de configuration** qui se nomme **virtual1.conf** :

```
<VirtualHost 192.168.0.2:80>
        DocumentRoot    /var/www/virtual1.com
        ServerName      virtual1.com
</VirtualHost>
```

2ème fichier de configuration qui se nomme **virtual2.conf** :

```
<VirtualHost 192.168.0.2:8080>
        DocumentRoot    /var/www/virtual2.com
        ServerName      virtual2.com
</VirtualHost>
```

Note explicative : Dans cet exemple, le site **virtual1.com** sera accessible via l'url : **http://virtual1.com** et le site **virtual2.com** sera accessible via l'url : **http://virtual2.com:8080** .

Ps : Les répertoires **/var/www/virtual1.com** et **/var/www/virtual2.com** doivent exister et doivent bénéficier des bonnes permissions. A défaut, le **Serveur Web Apache2**, lors du **reload**, renverra des erreurs. Pour les testes, vous pouvez appliquer **un chmod 755** sur ces répertoires.

Encore une fois, nous devons activer les deux **Hôtes Virtuels** avec la commande **a2ensite** (cf. ci-avant) et nous devons demanderau **Serveur Web Apache2** de relire ses fichiers de configuration pour que les changements soient pris en comptes =⇒ **/etc/init.d/apache2 reload**._

Éditer

3. Hôtes Virtuels basés sur le nom

Cette troisième méthode et celle qui est la plus utilisée et aussi la plus conseillée. Elle tend même à devenir un standard. Il s'agit simplement d'associer plusieurs **noms DNS** à une seule adresse IP.

Dans l'exemple suivant, nous allons configurer deux **Sites virtuels** qui utiliseront tout les deux l'adresse Ip **192.168.0.2**.

Dans un premier temps, nous allons éditer le fichier **/etc/hosts** afin de donner deux noms bien distincts à la seule et unique adresse Ip de notre interface réseau. Ceci permettra la résolution de nom (Nom –> Adresse Ip). Exemple de fichier **/etc/hosts** :

```
127.0.0.1       localhost
192.168.0.2     virtual1.com
192.168.0.2     virtual2.com
```

Notre fichier **/etc/hosts** étant correctement renseigné, nous allons rajouter la directive **NameVirtualHost** à la fin du fichier **/etc/apache2/apache2.conf** qui n'est autre que le fichier de configuration principale du **Serveur Web Apache2**.

Cette directive correspond à celle sur laquelle le **Serveur Web Apache2** acceptera uniquement les requêtes adressées aux Hôtes Virtuels que nous allons définir ci-après.

A la fin du fichier **/etc/apache2/apache2.conf** nous rajoutons donc :

```
NameVirtualHost 192.168.0.2:80
```

Ensuite, nous créons deux nouveaux fichiers de configurations (**VirtualHost**) que nous plaçons dans le répertoire **/etc/apache2/sites-available**.

Voici ce que contiennent ces deux fichiers de configuration :

1er fichier de configuration qui se nomme **virtual1.conf** :

```
<VirtualHost 192.168.0.2:80>
        DocumentRoot    /var/www/virtual1.com
        ServerName      virtual1.com
</VirtualHost>
```

2ème fichier de configuration qui se nomme **virtual2.conf** :

```
<VirtualHost 192.168.0.2:80>
        DocumentRoot    /var/www/virtual2.com
        ServerName      virtual2.com
</VirtualHost>
```

Ps : Les répertoires **/var/www/virtual1.com** et **/var/www/virtual2.com** doivent exister et doivent bénéficier des bonnes permissions. A défaut, le **Serveur Web Apache2**, lors du **reload**, renverra des erreurs. Pour les tests, vous pouvez appliquer un **chmod 755** sur ces répertoires.

Encore une fois, nous devons activer les deux **Hôtes Virtuels** avec la commande **a2ensite** et nous devons demander au **Serveur Web Apache2** de relire sa configuration pour que les modifications soient pris en comptes =⇒ **/etc/init.d/apache2 reload**.

ATTENTION : Dans les exemples ci-dessus, je n'ai pas fait état des directives qui d'usage, doivent êtres présentes dans les fichiers de configurations des **Hôtes Virtuels** (*VirtualHost*), notamment :

Les directives **ServerAdmin**, **ErrorLog** et **CustomLog** qui sont des directives qu'il est fortement conseillé d'inclure dans les fichiers de configuration de vos **Hotes Virtuels**.

Éditer

III. Perte du mot de passe mysql

Dans cette section, nous vous proposons deux méthodes bien distinctes pour que vous puissiez re-créer le mot de passe de l'utilisateur Root de Mysql si vous l'avez perdu.

Avant toute chose, il convient d'arrêter le Serveur Mysql :

```
sudo /etc/init.d/mysql stop
```

Éditer

a. Première méthode

Il faut commencer par créer un fichier contenant le nouveau mot de passe que vous voulez attribuer à l'utilisateur Root de Mysql.

Ps : Dans la mesure ou ce mot de passe est stocké en clair, il est vivement recommandé de le mettre dans un répertoire suffisamment sûr. N'oubliez pas de supprimer le fichier une fois la procédure terminée.

```
echo "SET PASSWORD FOR 'root'@'localhost' = PASSWORD('Nouveau mot de passe');" > mdp.txt
```

On redémarre le server mysql en root :

```
sudo mysqld --user=root --init-file=/chemin/vers/le/fichier/mdp.txt
```

où

```
sudo mysqld_safe --user=root --init-file=/chemin/vers/le/fichier/mdp.txt
```

Et voilà, votre nouveau mot de passe est pris en compte. On stoppe à nouveau le serveur lancé :

```
/etc/init.d/mysql stop
```

Si `ps auxww|grep mysql` vous renvoie quelque chose, vous pouvez faire

```
killall mysqld
```

Où sinon, rebootez votre machine.

Et voilà, il ne reste plus qu'a relancez le service mysql proprement :

```
/etc/init.d/mysql start
```

Ceci est inutile si vous venez de rebooter votre machine.

Éditer

b. Deuxième méthode

Vous pouvez trouver une méthode alternative ici : http://www.nuxwin.com/articles/view.php/13

Si mysql vous indique qu'il ne veut pas démarrer en root, Changez le II. en :

```
/usr/sbin/mysqld --user=root --skip-grant-tables &
```

Éditer

IV. Sécurisation d'un Site Web via Protocol SSL (Secure Socket Layer)

Dans cette section, je vous propose de prendre connaissance d'une procédure permettant de sécuriser un **hôtes virtuel** grâce au **Protocol SSL** (Secure Socket Layer).

Éditer

1. Le Protocol SSL en quelques mots

SSL (Secure Sockets Layer) est un protocole qui a été développé par la société Netscape.

Ce protocol permet à deux machines de communiquer de manière sécurisée. Les informations échangées entre les deux machines sont de ce fait inviolables.

Le Protocol SSL se traduit par la combinaison de deux protocols bien distincts (*Handshake & Record*) qui permettent la négociation entre les deux machines et le chiffrement des données échangées.

Pour obtenir plus d'information concernant le fonctionnement du Protocole SSL, vous pouvez vous rendre sur cette page –> http://sebsauvage.net/comprendre/ssl

Éditer

2. Mise en application avec le Serveur Http Apache2

A. Installation de la librairie OpenSSL :

Pour pouvoir utiliser le protocole SSL avec le Serveur http Apache2, la librairie **openssl** doit être préalablement installée sur votre système. Pour installer cette librairie, il vous suffit de taper la commande suivante dans un terminal :

```
sudo apt-get install openssl
```

Ps : Il se peut que cette librairie soit déjà installée sur votre système.

B. Activation du module SSL du Serveur Http Apache2 :

Pour que le protocol SSL puisse fonctionner avec le **Serveur Http Apache2**, il faut activer un module spécifique nommé **SSL**.

Pour activer ce module, il vous suffit de taper les commandes suivantes dans un terminal :

1. On active le module SSL :

```
sudo a2enmod ssl
```

2. On demande au **Serveur Http Apache2** de relire ses fichiers de configuration pour que les changements soient pris en comptes :

```
sudo /etc/init.d/apache2 force-reload
```

C. Les Certificats :

Les certificats permettent de fournir divers informations concernant l'identité de son détenteur, de la personne qui publie les données. Ce certificat s'accompagne d'une clé publique qui est indispensable pour que la communication entre les machines soit chiffrée.

De même, afin de garantir l'authenticité du certificat, ce dernier est signé numériquement par le biais

d'une clé dite privée provenant soit d'un organisme officiel (Société spécialisée dans la certification) soit par le détenteur du Certificat lui même. Dans ce dernier cas, on parlera de certificat auto-signé.

Dans la plupart des cas, l'obtention d'un Certificat certifié par une autorité officielle ayant un prix assez élevé, les webmasters auront tendance a vouloir signé eux-même leur certificat. Ce faisant, il est à noter que dans ce cas, le certificat ne sera pas reconnu par les navigateurs internet comme étant certifié.

<Note Importante>

Dans le cadre de ce tutorial, je présente la procédure à suivre pour mettre en place un Site Web sécurisé via protocole SSL en utilisant un Certificat auto-signé.

Je part du principe que nous avons déjà mis en place un Hôtes virtuel basé sur le nom nommé **nuxwin.com** (cf. ci-avant), ce dernier étant accessible sur le **port 80** (http://nuxwin.com)

</Note Importante>

Pour générer un certificat auto-signé avec **Ubuntu**, il nous suffit de taper les commandes suivantes dans un terminal :

1. On se place dans le répertoire dans lequel le certificat doit être généré :

```
cd /etc/apache2/ssl
```

2. On lance la commande de génération :

```
sudo apache2-ssl-certificate
```

(Note: par défaut, apache2-ssl-certificate génère un certificat valable 1 mois. Il est possible d'utiliser le paramètre -days pour modifier la durée de validité. Par exemple: **sudo apache2-ssl-certificate -days 365** génèrera un certificat valable 1 an.)

et ensuite, on répond aux questions posées :

a. Première question :

```
Country Name (2 letter code) [GB]:
```

Il s'agit ici d'entrer les deux lettres correspondant à notre pays. Dans la mesure ou nous sommes situé en France, nous indiquons : FR et on valide par la touche Enter.

b. Deuxième question :

```
State or Province Name (full name) [Some-State]:
```

Ici, nous devons indiquer le nom de notre pays. Etant situé en France, Nous indiquons : FRANCE et nous validons par la touche Enter.

c. Troisième question :

```
Locality Name (eg, city) []:
```

Ici, nous devons indiquer le nom de la ville où nous nous situons. Comme nous sommes basé à CAEN, nous indiquons : CAEN et nous validons par la touche Enter.

d. Quatrième question :

```
Organization Name (eg, company; recommended) []:
```

Ici, nous devons indiquer le nom de notre organisation, de notre société. Nous indiquons donc france-hosting et nous validons par la touche Enter. Bien entendu, si nous n'avions pas de société nous aurions pu mettre un nom fictif, le nom de notre site Web par exemple.

e. Cinquième question :

```
Organizational Unit Name (eg, section) []:
```

Ici, Nous devons indiquer le nom de la section de notre organisation, de notre société. N'en ayant pas, nous avons indiqué : France-hosting.

f. Sixième question :

```
server name (eg. ssl.domain.tld; required!!!) []:
```

Ici, il convient de faire particulièrement attention à ce que nous allons entrer. Nous devons indiquer le nom de domaine que nous désirons sécuriser. En ce qui nous concerne, il s'agit du domaine : **nuxwin.com**. Nous indiquons donc nuxwin.com et nous validons par la touche Enter.

g. Septième question :

```
Email Address []:
```

Ici, il s'agit d'indiquer l'adresse E-mail de l'administrateur. En ce qui nous concerne, il s'agit de : admin@nuxwin.com. Nous terminons bien entendu en validant par la touche Enter.

D. Ajout de la directive Listen 443 :

Par défaut, le **Serveur Http Apache2** est configuré pour écouter sur le **port 80**. Il s'agit là de la configuration usuelle d'un Serveur Web. Cependant, le protocol SSL a besoin d'un port spécifique pour pouvoir fonctionner. Il s'agit du **port 443**.

Nous allons donc rajouter une directive de configuration nommée **Listen** qui permettra d'indiquer au **Serveur Web Apache2** qu'il doit aussi écouter sur le **port 443**.

Pour demander au **Serveur Web Apache2** d'écouter sur le **port 443**, il nous suffit d'éditer le fichier **ports.conf** qui se trouve dans le répertoire **/etc/apache2** et de rajouter la ligne suivante :

```
Listen 443
```

Ensuite, il suffit de demander au **Serveur Web Apache2** de relire ses fichiers de configuration pour que les changements soient pris en comptes :

```
sudo /etc/init.d/apache2 reload
```

E. Création du fichier de configuration

Comme indiqué ci-avant, dans le cadre de ce tutorial, je suis parti du principe que nous avions déjà configuré un hôte virtuel basé sur le nom accessible sur le **port 80**. Ceci implique donc l'existence d'un fichier de configuration nommé **nuxwin.com.conf** situé dans le répertoire **/etc/apache2/sites-available**.

Voici le contenu de ce fichier :

```
<VirtualHost 192.198.0.2:80>
        DocumentRoot /var/www/nuxwin.com
        ServerName nuxwin.com
</VirtualHost>
```

Ps : Je rappelle qu'il s'agit du contenu minimal d'un virtualhost.

Pour sécuriser cet Hôte Virtuel, nous allons donc devoir modifier ce fichier en y ajoutant un hôte virtuel accessible sur le **port 443**, ce dernier contenant des directives particulières qui sont les suivantes :

1. Directive **SSLEngine** :

Cette directive permet d'activer le moteur SSL au sein d'un hôte virtuel, Elle peut prendre deux

arguments –> **on/off**

2. Directive **SSLCertificateFile** :

Cette directive définit le certificat authentifiant le Serveur auprès des clients. L'arguement est le chemin d'accès au certificat. En ce qui nous concerne, le certificat se trouve dans le répertoire **/etc/apache2/ssl**

3. Directive **SSLCertificateKeyFile** :

Cette directive définit la clé privée du Serveur utilisée pour signer l'échange de clé entre le client et le serveur. Elle prend en argument le chemin d'accès à la clé (fichier). Dans notre cas, la clé se trouve dans le même fichier que le certificat. Elle se trouve donc dans le répertoire **/etc/apache2/ssl**.

Toutefois, il convient de noter que le chemin ne sera pas directement celui du fichier. En effet, il s'agira d'un lien symbolique de la forme **f9b34192.0** se trouvant lui aussi dans le répertoire **/etc/apache2/ssl**.

Par ailleurs, comme nous l'avons déjà fait pour notre hôte virtuel accessible sur le **port 80**, nous allons devoir rajouter une directive **NameVirtualHost** qui permettra que l'adresse nommée par le nom de notre hôte virtuel accessible sur le **port 443** soit résolue correctement. Nous rajouterons donc cette directive (*NameVirtualHost 192.168.0.2:443*) au début de notre fichier de configuration.

Enfin, afin que les clients puissent continuer d'accéder au site Web en tapant une url de type **http** et non **https**, nous allons modifier l'hôte virtuel accessible sur le **port 80** en remplaçant la directive **DocumentRoot** par une directive de redirection.

Voici donc le contenu de notre fichier une fois modifié :

```
NameVirtualHost  192.168.0.2:443

<VirtualHost 192.168.0.2:80>
    ServerName  nuxwin.com
    Redirect / https://nuxwin.com
</VirtualHost>

<VirtualHost 192.168.0.2:443>
        ServerName nuxwin.com
        DocumentRoot /var/www/nuxwin.com

        SSLEngine on
        SSLCertificateFile /etc/apache2/ssl/apache.pem
        SSLCertificateKeyFile /etc/apache2/ssl/<nom du fichier.key>
</VirtualHost>
```

Ps : <*nom du fichier.key*> doit être remplaçé par le nom du lien symbolique de la forme **f9b34192.0**. Je rappelle que ce lien symbolique se situe dans le répertoire **/etc/apache2/ssl**.

F. Reload du Serveur Http Apache2 :

Afin que les modifications que nous venons d'effectuer soient prises en compte, nous devons demander au **Serveur Http Apache2** de relire ses fichiers de configuration.

Pour ce faire, il suffit de taper la commande suivante dans un terminal :

```
sudo /etc/init.d/apache2 reload
```

Normalement, si tout s'est bien passé, vous devriez désormais avoir accès à votre site Web de manière sécurisée.

Éditer

Quelques Précisions concernant ce tutorial

Ce tutorial a été rédigé pendant l'installation de la solution LAMP proposée sur un serveur de développement. Il est précisé que l'auteur est parti d'une nouvelle installation d'**Ubuntu Dapper Drake (6.06 LTS)** et donc, qu'aucun des **logiciels** sus-mentionnés n'avaient été installés sur son système auparavant.

Enfin, Il est précisé que l'auteur du tutorial n'a rencontré aucune erreur particulière pendant l'installation de cette solution LAMP et que tout fonctionne correctement.

Date de création par l'auteur : *06/06/2006 02:02*

Dernière édition par l'auteur : *29/12/2006 22:13*

Dernière tâche effectuée : Rajout d'une section – Sécurisation d'un Site Web via protocole SSL.