Installation complète d'un serveur web sous Debian

par Olivier Lange

Date de publication : 25 janvier 2007

Dernière mise à jour : 25 janvier 2007

Ce tutorial vous explique étape par étape la configuration d'un serveur dédié sous Debian pour l'utiliser comme serveur web. Il a été installé sur une Debian Sarge.

I - Pré-requis

I-A - Connaissances

Ce tutorial vous explique pas à pas comment installer un serveur dédié sous Debian. Chaque étape y est expliquée clairement, et dans l'absolu, il n'est pas nécessaire d'avoir des connaissances en informatique pour le suivre.

Néanmoins, et malgré la démocratisation des offres pour serveurs dédiés disponibles en francophonie (Suisse / France / Belgique) offert par OVH et FREE (par exemple), la gestion et la sécurisation d'un serveur dédié demande un minimum de connaissances du système Linux.

Pour bien pouvoir suivre ce tutorial, vous devrez savoir au moins :

1 Vous connectez en root au moyen de SSH sur votre machine
2 Editer un fichier de configuration en mode texte (VI, VIM, Nano, etc)
3 Avoir des bases en programmation (copier, c'est bien, comprendre, c'est mieux !)
4 Etre prêt à investir du temps, beaucoup de temps dans l'apprentissage de Linux !

S'il vous manque certaines informations, sachez qu'Internet en regorge. Ce tutorial ne prétend pas voir tous les aspects de l'hébergement web. Il reste actuellement pas mal de points qui pourraient nécessiter une explication (installation d'.htaccess, configuration du SSL, création de script de sauvegarde, ...). Ce tutorial sera donc amené à évoluer dans le futur. N'hésitez pas à me faire parvenir vos envies / idées.

I-B - Matériel

Qui dit serveur, dit obligatoirement ordinateur. L'avantage de Linux, c'est que vous pourrez faire tourner votre serveur web sur votre vieux 386 qui traîne dans un coin. Ce tutorial part du principe que vous disposez :

1 D'un serveur physique (hors VMWare, par exemple)
2 D'une adresse IP fixe
3 Ou simplement d'un serveur dédié loué chez un fournisseur (Sivit, OVH, Dedibox, etc)

I-C - Software

A l'heure où j'écris ce tutorial, Debian Etch (version 4.0) n'est pas encore sorti en version stable, malgré son retard d'un mois (pour le moment ...). Ce tutorial a été basé sur la distribution Debian fournie par OVH sur ses serveurs dédiés. Il s'agit donc d'une Sarge 3.1 patchée, fournie avec uniquement un accès SSH.

Un tutorial ne sera JAMAIS totalement complet ou totalement adapté à vos besoins. C'est une base de travail. A vous de le faire évoluer en fonction de vos besoins.

I-D - Remerciements

Il me sera difficile de remercier toutes les personnes qui ont contribué à ce tutorial. Et j'ai peur d'en oublier (n'hésitez pas à me le rappeler, le cas échéant !). Mais je souhaiterais tout de même remercier :

- Les membres de Developpez qui m'ont donné des idées ou des propositions d'évolutions.
- Les membres de Kimsufi.com qui m'ont supporté, et montré la nécessité d'un tel tutorial.

- Plus particulièrement Fred036 pour m'avoir autorisé à reprendre et adapter son tutorial sur Postfix.
- Akira, mon chat, pour lequel j'ai été "absent" plusieurs jours à cause de celui-ci ;)
- Un grand merci a Julp qui a effectué la correction orthographique de ce tutorial

I-E - Conventions typographiques

Pas grand chose de particulier à préciser, si ce n'est:

- # signifie que c'est une commande qui doit être écrite dans le shell
- > signifie que c'est une commande MySql qui doit être écrite après une connexion mysql en shell
- Chaque fichier $ édité est écrit avec son chemin absolu dans l'en-tête du code à modifier/écrire

I-F - Responsabillité

Ce tutorial est avant tout une base de travail pour installer un serveur dedié. Chaque personne l'utilisant le fait à ses risques et périls. Je ne pourrais en AUCUN cas être tenu pour responsable en cas de perte d'informations, de destruction de données ou d'indisponibillité de votre serveur. Ce tutorial est mis à la disposition sans AUCUNE garantie, et peut être amené à changer à tout moment! Si vous trouvez des erreurs, n'hésitez pas à me les signaler, que je puisse le mettre à jour !

II - Installation de base

II-B - Préparer son serveur

Se connecter en root sur le serveur puis effectuer une mise à jour :

```
apt-get update
apt-get upgrade
```

Une fois le serveur à jour, on peut commencer à installer les différents services nécessaires sur notre serveur. Attention, pensez à redémarrer les services après chaque installation ou reconfiguration :

```
# /etc/init.d/nom_service restart
```

II-C - Installer apache 2

Il nous faut installer un serveur HTTP qui va s'occuper d'afficher nos différentes pages. Pour cela, je vous propose d'installer l'un des serveurs les plus utilisés et les plus connus : Apache. Nous installerons ici sa version 2.

```
# apt-get install apache2
Tester l'installation d'apache : http://xxx.xxx.xxx.xxx/ (IP du serveur)
```

Vous pouvez supprimer la redirection sur /apache2-default/ :

```
# nano /etc/apache2/sites-available/default
#RedirectMatch ^/$ /apache2-default/
```

Pour accéder directement à la configuration des virtualhosts : ici

II-D - Installer php 5

Actuellement, notre serveur peut nous afficher des pages statiques au format HTML. La plupart des sites que vous voudrez installer disposeront d'une partie dynamique. C'est pourquoi nous poursuivons par l'installation de PHP 5 sur le serveur.

```
# apt-get install php5
```

S'il annonce que le package n'a pas été trouvé, éditez le fichier source d'Apt et ajoutez :

```
# nano /etc/apt/sources.list
deb http://packages.dotdeb.org stable all
```

On met à jour la liste des packages :

```
# apt-get update
```

Et on installe php5.

II-E - Installer La base de donnée (Mysql)

PHP est très très souvent couplé à un système de base de données : Mysql. Nous installons ici Mysql-server version 5. Vous verrez plus bas que nous allons également installer phpmyadmin. Il s'agit d'un script php qui permet de gérer ses bases de données Mysql de facon très simple.

```
# apt-get install mysql-server
```

Définir le mot de passe root de Mysql (« mysql » par exemple). Dans l'écran suivant, il demande s'il faut gérer les connexions d'hôtes qui utilisent Debian Sarge. On répond OUI (répondre non empêchera la configuration de Postfix par la suite !).

On vérifie que Mysql fonctionne bien :

```
# mysql -p
entrer le mot de passe
>Exit;
```

Installer les librairies php5-mysql :

```
# apt-get install php5-mysql
```

Installer PhpMyAdmin :

```
# apt-get install phpmyadmin
Choix du serveur a paramétrer : Apache2
On redémarre Apache quand proposé
```

On se connecte par l'adresse http://xxx.xxx.xxx.xxx/phpmyadmin.

II-F - Installation de BIND9 (serveur DNS)

Maintenant que nous avons notre serveur web installé, il faut que nos visiteurs puissent nous trouver. Et il n'est guère facile de leur faire retenir une adresse du style: 213.251.175.34/~nom_de_mon_site. Si vous y arrivez, vous êtes un champion ;). C'est donc le travail de Bind de traduire notre nom de domaine. Il vous suffira de définir votre nom de serveur chez votre registar, dans la partie NS1.

```
# apt-get install bind9
```

Pour accéder directement à la configuration de Bind 9 : ici

II-G - Installation du serveur mail

Un serveur web sans serveur mail ne vous sera pas très utile. Le serveur mail est nécessaire pour différentes tâches administratives (envois de rapport cron, information en cas de problèmes, etc), mais aussi simplement parce que vos sites hebergés souhaiteront disposer d'une ou plusieurs boîtes mails associées à leurs domaines !

On commence par être sûr de disposer de la dernière version de postfix. Pour cela on ajoute :

nano /etc/apt/sources.list
```
deb http://debian.home-dn.net/sarge postfix-vda/
```

On recharge les paquets disponibles :

```
# apt-get update
```

Puis on installe Postfix :

```
# apt-get install postfix postfix-mysql
Choix de distribution : pas de configuration
```

II-H - Installation du FTP (VSFTPD)

Avoir un site disponible sur le net, c'est bien. Pouvoir y mettre des fichiers, c'est mieux ;). Et c'est le but de VSFTPD qui est un serveur FTP très sécurisé.

```
# apt-get install vsftpd
```

III - Configuration d'Apache 2

III-A - Introduction

Pour commencer, il faut installer un serveur web au sens premier. C'est lui qui va nous permettre d'interpréter nos pages HTML, PHP, etc. Apache 2 est le serveur http le plus utilisé sur les serveurs. Il dispose d'un bon niveau de sécurité et de beaucoup de documentations disponibles sur le net.

D'autre part, il permet de gérer des sites virtuels. Et c'est de cette manière que nous allons le configurer. En effet, le but étant de disposer de plusieurs sites sur notre serveur, il nous faut pouvoir les contacter directement avec une URL propre. Notre serveur ne dispose évidemment pas d'une adresse IP pour chaque site installé. Ce serait beaucoup trop complexe du point de vue physique à mettre en place. Et surtout, avec le système d'IPV4 utilisé actuellement, c'est impensable. Pour la petite histoire, le système IPV4 utilise 4 nombres de 0 à 255 sous la forme xxx.xxx.xxx.xxx. Cette série de nombres représente de manière unique chaque ordinateur connecté physiquement à Internet.

Certaines plages IP sont réservées pour des réseaux locaux (192.168.x.x ou 10.x.x.x par exemple). Mais ce souci va être prochainement résolu avec l'utilisation de l'IPV6 qui est en train de s'implémenter. Cette nouvelle notation permet de passer de 232 à 2128 IP différentes. Ce qui permettra d'en attribuer plusieurs à un même serveur, dans le futur.

Comme actuellement, c'est le protocole IPV4 qui est le plus utilisé, nous ne disposons en conséquence qu'une seule adresse IP pour 10 ou 100 sites à héberger sur notre serveur. Lorsqu'une requête HTTP est envoyée à notre serveur, le nom de domaine est transformé en adresse IP. La seule façon de différencier une demande venant de site1-cpnv.com ou de site2-cpnv.com, qui renvoient la même IP, passe par l'en-tête http, qui elle contiendra toujours le nom de domaine initialement demandé.

Lorsque la requête que nous désirons a atteint notre serveur http, celui-ci va regarder dans ses règles afin de trouver dans quel répertoire il doit se diriger. C'est là que la gestion des virtualhosts va intervenir. En conséquence, nous créerons une entrée pour chaque site hébergé sur notre serveur. Cette entrée contiendra le domaine prévu, et le répertoire de redirection.

III-B - Pré-requis

Une fois que le serveur est installé de base, nous allons créer et configurer nos espaces d'hébergements. Tout d'abord, ce tutorial part des principes suivants :

- Vous désirez pouvoir accéder à vos sites par ip_du_server/~nom_user
- Vous n'avez qu'une seule IP pour tous vos sites
- Vous allez configurer BIND

Dans cette première partie, nous allons modifier un fichier: /etc/apache2/apache2.conf, et créer des fichiers dans les répertoires /etc/apache2/sites-available et /etc/apache2/sites-enabled. Mais allons-y par étapes.

III-C - Configuration d'apache2.conf

On édite le fichier de configuration principal d'apache2 :

```
# nano /etc/apache2/apache2.conf
```

On vérifie les utilisateur et groupe d'apache (autour de la ligne 100 environ) :

```
User www-data
Group www-data
```

On modifie les fichiers que l'on désire par défaut (ligne 210 environ. A vous de choisir ce que vous désirez obtenir) :

```
DirectoryIndex index.html index.php index.xhtml
```

On décommente cette ligne pour autoriser les connexions des utilisateurs :

```
UserDir public_html
```

On vérifie la présence de ces lignes à la fin du fichier:

```
# Include the virtual host configurations:
Include /etc/apache2/sites-enabled/[^.#]*
```

On sauve les modifications, et on ferme le fichier.

A ce niveau-là, un /etc/init.d/apache2 restart permet déjà d'accéder aux répertoires privés de chacun de ses utilisateurs (penser à rajouter un dossier public_html dans ceux-ci pour voir quelque chose !).

Pour accéder directement à la configuration de VSFTPD : ici

III-D - Installation des virtualhosts

On va maintenant créer nos hôtes virtuels. Par défaut, je les appellerai test1.com et test2.com. A vous de mettre les noms que vous désirez. Mais avant de s'attaquer aux utilisateurs, on commence par modifier le squelette de la création des nouveaux users. L'avantage ? Ne pas avoir besoin à chaque fois de devoir créer le répertoire public_html et logs quand on crée un nouvel utilisateur, mais aussi d'avoir directement une page d'accueil.

```
# mkdir /etc/skel/public_html
# mkdir /etc/skel/logs
# echo " <h1>Nouvel espace web crée</h1> " > /etc/skel/public_html/index.html
```

Une fois le squelette créé, on peut créer un nouvel utilisateur :

```
# useradd -g www-data -m test1
```

On copie le contenu ci-dessous :

```
# nano /etc/apache2/sites-available/test1.com
<VirtualHost *>
        ServerAdmin postmaster@test1.com
        ServerName www.test1.com
        ServerAlias test1.com *.test1.com
        DocumentRoot /home/test1/public_html/
        <Directory /home/test1/public_html/>
                Options -Indexes FollowSymLinks MultiViews
                AllowOverride All
        </Directory>
        ErrorLog /home/test1/logs/error.log
```

```
# nano /etc/apache2/sites-available/test1.com
        LogLevel warn
        CustomLog /home/test1/logs/access.log combined
        ServerSignature Off
</VirtualHost>
```

On valide et on ferme le fichier. On rend le domaine créé disponible.

```
# ln -s /etc/apache2/sites-available/test1.com /etc/apache2/sites-enabled/test1.com
```

On redémare apache2 :

```
/etc/init.d/apache2 restart
```

Et on peut accéder à notre répertoire :

```
http://xxx.xxx.xxx.xxx/~test1
```

Et on devrait voir une page web ! Il ne reste plus qu'à informer les visiteurs de la présence de ce site sur ce serveur. Et cela, c'est Bind qui s'en charge !

IV - Configuration de Bind 9

IV-A - Introduction

Un des points les plus problématiques de l'installation, la configuration de Bind9. Bind9 (Berkeley Internet Name Domain) est le serveur DNS le plus utilisé sur Internet. C'est lui qui va autoriser notre URL www.site1.com à pointer sur notre serveur (dans le cas d'une utilisation locale). En effet, il permet de transformer les différents alias en adresse IP, et donc de rediriger au bon endroit (sur le même serveur, en l'occurrence).

La configuration de Bind est un peu plus complexe que ce que nous avons vu jusqu'à maintenant. D'une part, parce que les différents tests a effectué prennent du temps. En effet, la propagation des DNS prend de 6 à 48 heures, en fonction des FAI, et des registars. Ensuite, c'est la première étape qui aura une réelle incidence du point de vue extérieur sur notre serveur. Un Bind mal configuré, et nos sites sont inaccessibles !

L'installation de Bind en elle-même se fait comme pour les autres, vu que nous utilisons ici aussi les binaires disponibles. Une fois installé, nous devons effectuer quelques ajustements. En effet, de base, le serveur Bind est fourni en open DNS. Ce qui signifie que n'importe qui peut utiliser notre serveur DNS. Cela peut poser des problèmes de sécurité, et surtout de serveur de relay DNS à des pirates, ou des spammeurs. Nous devons donc n'autoriser l'utilisation du serveur que depuis le serveur lui-même (localhost).

Une fois effectué, nous devons créer une entrée dans la zone pour chaque domaine que nous hébergeons. Il s'agit en fait d'un simple fichier, que nous nommons : db.nom_domaine.tld par principe. Il est important de décortiquer ce fichier, qui contient de nombreux paramètres à configurer, et surtout à veiller à bien le faire.

Pour avoir des détails sur Bind, et les DNS ; je vous invite à vous rendre sur Wikipedia, où vous trouverez toutes les informations adéquates : BIND DNS

IV-B - Named.conf

Pour configurer Bind, nous allons modifier un ficher /etc/bind/named.conf et en créer un pour chaque domaine que nous désirons héberger.

```
# nano /etc/bind/named.conf
```

On ne touche pas aux données par défaut, mais on rajoute, après la dernière zone (1 entrée pour chaque domaine, évidemment. Et test1.com est à changer par votre nom de domaine, cela va de soit !). Nous avons 2 solutions ici. Soit nous modifions directement le fichier named.conf, soit nous pouvons ajouter nos déclarations de zones dans named.conf.local. A vous de voir comment vous désirez le faire :

```
zone "test1.com" {
        type master;
        file "/etc/bind/db.test1.com";
};
```

Une petite modification à effectuer, pour éviter que notre serveur ne serve de relay DNS ouvert. On ajoute ces lignes dans le fichier (entre les {}):

```
# nano /etc/bind/named.conf.options
        allow-recursion { localhost; };
```

Dans le cas où le fichier /etc/bind/named.conf.options n'existe pas (possible sur certains systèmes), il suffit d'ajouter le code suivant dans le fichier /etc/bind/named.conf:

```
# nano /etc/bind/named.conf
   options {
           allow-recursion { localhost; };
   };
```

On valide et on sort du fichier.

IV-C - Création du fichier de configuration de zone

On crée le fichier de description de notre zone :

```
   # nano  /etc/bind/db.test1.com
```

On entre les valeurs de notre domaine (ceci est un exemple. A vous de l'adapter à vos besoins/envies) :

```
   $ttl 86400
   test1.com.            IN      SOA     ksXXXXX.kimsufi.com. webmaster.test1.com. (
                                                 2006121903
                                                  21600
                                             3600
                                             604800
                                             86400 )
   test1.com. IN      NS                    ksXXXXX.kimsufi.com.
   test1.com. IN      NS                    ns.kimsufi.com.
   test1.com. IN      MX                10 mail.test1.com.
   test1.com. IN A               xxx.xxx.xxx.xxx
   Server          IN A          xxx.xxx.xxx.xxx
   www          IN A                xxx.xxx.xxx.xxx
   mail          IN A                xxx.xxx.xxx.xxx
   smtp          IN A                xxx.xxx.xxx.xxx
   pop                 IN A                xxx.xxx.xxx.xxx
   pop3          IN CNAME          Server
   imap          IN A                xxx.xxx.xxx.xxx
   sql          IN A                xxx.xxx.xxx.xxx
   mysql          IN A                xxx.xxx.xxx.xxx
```

Quelques explications:

Champ	Description
2006121903	Est à modifier à chaque édition du fichier. Par convention, on l'écrit: année-mois-jour-numéro à 2 chiffres
21600	Temps que le serveur esclave doit attendre avant de questionner à nouveau le serveur maître. Unité de temps: [s]
3600	Temps à attendre avant d'effectuer une nouvelle demande au serveur maître en cas de non réponse. Unité de temps: [s]
604800	Temps d'expiration du serveur principal en cas de non réponse. Unité de temps: [s]
86400	Temps de mise en cache minimum par d'autres serveurs DNS. Unité de temps: [s]
CNAME	Enregistrement de nom canonique qui dit au serveur de noms qu'un nom donné est aussi connu qu'un autre (alias)

Champ	Description
A	Enregistrement d'adresse qui spécifie une adresse IP à assigner à un nom
NS	Enregistrement de serveur de noms (NameServer) qui annonce les serveurs de noms faisant autorité pour une zone particulière
MX	Enregistrement Mail eXchange, qui dit où doit se diriger le courrier envoyé à un nom d'espace particulier contrôlé par cette zone
SOA	Enregistrement "Start Of Authority", qui proclame des informations importantes faisant autorité à propos des espaces de nom pour les serveurs de noms
PTR	Enregistrement PoinTeR record, conçu pour orienter vers une autre partie de l'espace de nom

Si vous désirez avoir plus d'exemples sur la création de ce fichier, vous pouvez vous référer à l'adresse suivante : http://www.linux-kheops.com/doc/redhat72/rhl-rg-fr-7.2/s1-bind-configuration.html

On sauvegarde, on redémarre bind9.

```
# /etc/init.d/bind9 restart
```

Et voila, il ne nous reste plus qu'à tester notre redirection, avec nos noms de domaine. Il est également possible d'avoir une description complète des différentes options de bind sous google. Une simple recherche vous donnera une foule de sites pour cela !

Il y a 2 possibillités pour tester notre fichier de configuration de zone :

- http://www.afnic.fr/outils/zonecheck
- http://www.dnsstuff.com/

Si vous désirez installer un .fr, vous ne devrez plus avoir aucune erreur (fatal chez zonecheck ou case rouge chez dnsstuff). En effet, l'afnic refuse d'enregistrer vos domaines avec une erreur. Pour les autres domaines, c'est à tester directement avec votre registar. Mais quoi qu'il en soit, idéalement, il ne faudrait pas avoir une seule erreur, c'est évident !

V - Configuration de VSFTPD en mode "utilisateur virtuel"

V-A - Introduction

Maintenant que nous avons un site serveur web opérationnel, et la possibilité de créer nos bases de donnée, il devient nécessaire de mettre nos fichiers sur le serveur. Pour cela, nous installons VSFTPD. C'est un serveur FTP très sécurisé (Very Secure File Transfert Protocol Daemon).

VSFTPD dispose de plusieurs styles de paramétrage de base. Là encore, étant donné que nous souhaitons pouvoir disposer de plusieurs comptes FTP par domaine, et notamment d'avoir des comptes FTP qui pointent sur des sous domaines, nous utiliserons le paramétrage par utilisateur virtuel.

Pour ce faire, nous allons utiliser une base de données de type Berkeley. Il s'agit d'une base de type non-sql. Elle n'est pas prévue pour être interrogée comme Mysql ou SQL server. En fait, il s'agit d'une table de hachage. Chaque enregistrement ne sera constitué que d'un login et d'un mot de passe. Ce type de base de données est indexé, extrêmement rapide et simple à mettre en #uvre. L'utilisation de ce type de base de données est obligatoire pour l'utilisation d'une identification de type PAM.

Pour le principe, nous ne définissons qu'un seul utilisateur UNIX à notre serveur FTP. Lorsque l'on se connecte avec un utilisateur, le programme vérifie dans notre base de données si celui-ci existe, et si le mot de passe correspond. A partir de là, il va chercher les paramètres concernés (chroot, droits spécifiques) et renvoit le répertoire concerné.

Grâce au chroot, il n'y a aucun souci de sécurité, car le répertoire est considéré comme étant un répertoire racine, il n'est donc pas possible de remonter la hiérarchie. Ce point est important du point de vue sécuritaire, car chaque connexion FTP utilise exactement le même utilisateur Unix : www-data.

Pour créer un nouvel utilisateur, il suffit de lui créer une entrée dans la base Berkeley, et un fichier de configuration personnel.

V-B - Préparation

On commence par préparer la configuration en créant le répertoire qui contiendra tous nos fichiers :

```
# mkdir /etc/vsftpd
```

On va devoir modifier la configuration de VSFTPD. Pour cela, on effectue une sauvegarde. Cela permettra de revenir en arrière en cas de souci :

```
# cp /etc/vsftpd.conf /etc/vsftpd.conf.bak
# cp /etc/pam.d/vsftpd /etc/pam.d/vsftpd.bak
```

V-C - La base de donnée

Il nous faut maintenant installer la base de données. On utilise pour cela la commande :

```
# apt-get install libdb3-util
```

Ce type de base de données est extrêmement simple, c'est pourquoi nous l'utilisons. Il se base sur un fichier de

type texte contenant nos différentes informations, entrées une à une. En fait, il n'y a pas de tables, ni de champs à configurer ! On va juste convertir un fichier contenant nos données sous cette forme :

```
login 1
password 1
login 2
password 2
...
login n
password n
```

On crée donc un fichier login.txt (le .txt n'est là que pour indiquer que ce sera nos données brutes ! Vous pouvez le nommer comme bon vous semble !). Puis on va le convertir en base de données. Actuellement, je n'ai pas encore trouvé le moyen de rajouter des données dans la base Berkeley directement. En conséquence, je garde le fichier login.txt, je lui donne des droits minimums (600), pour pouvoir régénérer la base par la suite.

```
# nano /etc/vsftpd/login.txt
        user1
        pass1
        user2
        pass2
```

```
# db3_load -T -t hash -f /etc/vsftpd/login.txt /etc/vsftpd/login.db
# chmod 600 /etc/vsftpd/login.db
# chmod 600 /etc/vsftpd/login.txt
```

On va maintenant informer le module PAM d'utiliser notre base de données nouvellement créée. Pour cela :

```
nano /etc/vsftpd/vsftpd.pam
auth required /lib/security/pam_userdb.so db=/etc/vsftpd/login
account required /lib/security/pam_userdb.so db=/etc/vsftpd/login
```

```
# cp /etc/vsftpd/vsftpd.pam /etc/pam.d/vsftpd
```

Si le système nous informe que le fichier existe déjà, on l'écrase.

V-D - Modification de vsftpd.conf

On va maintenant modifier notre fichier vsftpd.conf. Pour cela, je vous en propose un, que vous pourrez modifier à votre convenance (les commentaires sont assez clairs, je pense) :

```
# nano /etc/vsftpd.conf
# Ceci configure VSFTPd en mode "standalone"
listen=YES

# On désactive les connexions anonymes
# et on active les non-anonymes (c'est le cas des utilisateurs virtuels) :
anonymous_enable=NO
local_enable=YES

# Pour des raisons de sécurité on interdit toute action d'écriture :
write_enable=NO
anon_upload_enable=NO
anon_mkdir_write_enable=NO
anon_other_write_enable=NO

# 'guest_enable' est très important: cela active les utilisateurs virtuels !
# 'guest_username' fait correspondre tous les utilisateurs virtuels à
# l'utilisateur 'virtual' que nous avons défini plus haut, et au home
# correspondant : '~virtual/'.
guest_enable=YES
```

nano /etc/vsftpd.conf
```
guest_username=www-data

# On définit les droits par défaut des fichiers uploadés
anon_umask=022

# On veut que les utilisateurs virtuels restent chez eux : '~virtual/'
# (attends, on leur a fait un toit, c'est pas pour rien !)
chroot_local_user=YES

# On défini le nombre maximum de sessions à 200 (les nouveaux clients recevront
# un message du genre: "erreur : serveur occupé").
# On défini le nombre maximum de sessions par IP à 4
max_clients=200
max_per_ip=4

#####################################
# Debian customization              #
# (ou adoptons la Debian attitude) #
#####################################
# Some of vsftpd's settings don't fit the Debian filesystem layout by
# default.  These settings are more Debian-friendly.
#
# This option should be the name of a directory which is empty.  Also, the
# directory should not be writable by the ftp user. This directory is used
# as a secure chroot() jail at times vsftpd does not require filesystem
# access.
secure_chroot_dir=/var/run/vsftpd
#
# This string is the name of the PAM service vsftpd will use.
pam_service_name=vsftpd
#
# This option specifies the location of the RSA certificate to use for SSL
# encrypted connections.
rsa_cert_file=/etc/ssl/certs/vsftpd.pem

# Permet d'utiliser les configurations individuelles pour chaque utilisateur
user_config_dir=/etc/vsftpd/vsftpd_user_conf
```

V-E - Paramétrer les utilisateurs

On va maintenant chrooter nos utilisateurs dans leur répertoire respectif. Pour cela, on crée le répertoire qui sera utilisé pour contenir nos différents fichiers par utilisateur. Et on crée le fichier pour chaque utilisateur dans notre base de données.

```
# mkdir /etc/vsftpd/vsftpd_user_conf/
```

nano /etc/vsftpd/vsftpd_user_conf/user1
```
anon_world_readable_only=NO
local_root=/home/user1
write_enable=YES
anon_upload_enable=YES
anon_mkdir_write_enable=YES
anon_other_write_enable=YES
```

V-F - Redémarrage et test

On redémarre notre service FTP :

```
# /etc/init.d/vsftpd restart
```

Et il ne nous reste plus qu'à nous connecter à notre compte ftp en utilisant :

```
IP: ip du serveur ou nom de domaine
```

```
login: user1
password: password1
```

Et voilà. Un petit test, que tout fonctionne, et c'est terminé ;). Dans le cas oà vous ne pouvez pas uploader de fichiers, c'est que les droits ne sont pas bons. Vérifiez bien que le répertoire /home/user1 ait les droits www-data:www-data pour propriétaire et groupe (attention, récursif, donc valable pour tous les fichiers et répertoires contenus dans /home/user1) !

V-G - Ajouter des users ultérieurement

Une fois votre serveur FTP paramétré, vous voudrez sûrement rajouter des utilisateurs dans vos dossiers. Cela se fait de cette façon :

```
# echo "login" > /etc/vsftpd/login.txt
# echo "password" >> /etc/vsftpd/login.txt
# db3_load -T -t hash -f /etc/vsftpd/login.txt /etc/vsftpd/login.db
# /etc/init.d/vsftpd restart
```

Il est parfaitement possible d'éditer le fichier login.txt et d'y rajouter les 2 lignes, plutôt que d'utiliser la commande echo !

VI - Configuration de Postfix

VI-A - Introduction

Que manque-t-il de crucial sur notre serveur web ? Un MTA. En effet, actuellement, nous pouvons accéder à notre site, le mettre a jour, et y exécuter des pages web. Il faut maintenant pouvoir envoyer et recevoir des mails.

C'est le rôle de Postfix. Un MTA (Mail Transfert Agent) très connu, très performant, et assez simple à mettre en #uvre. Une fois de plus, son choix a été décisif vis-à-vis de sa possibilité à gérer les utilisateurs virtuels. Chacun de nos sites sera désireux d'avoir un ou plusieurs mails. Ce système nous permettra de proposer un nombre infini de mails pour chaque site, sans risque de conflit entre utilisateurs.

VI-B - Création de l'utilisateur

On commence par rajouter l'utilisateur et le groupe qui gérera postfix :

```
# groupadd -g 5000 vmail
# useradd -g vmail -u 5000 vmail -d /var/spool/vmail/ -m
```

VI-C - Configuration de Mysql

Pour commencer, il faut autoriser mysql à se connecter en local (pour cela, rechercher et décommenter bind-adress) :

```
# nano /etc/mysql/my.cnf
bind-address            = 127.0.0.1
# skip-networking
```

On se connecte à mysql en root, afin de le configurer :

```
# mysql -p
```

On entre le mot de passe lorsque le shell nous le demande. On va maintenant ajouter nos paramètres Mysql :

```
> create database postfix;
```

```
> use postfix;
```

```
> CREATE TABLE `domain` (
  `domain` varchar(255) NOT NULL default '',
  `actif` tinyint(1) NOT NULL default '1',
  PRIMARY KEY  (`domain`)
) ENGINE=MyISAM COMMENT='Postfix Admin - Domaines Virtuels';
```

```
> CREATE TABLE `mailbox` (
  `email` varchar(255) NOT NULL default '',
  `password` varchar(255) NOT NULL default '',
  `quota` int(10) NOT NULL default '0',
  `actif` tinyint(1) NOT NULL default '1',
  `imap` tinyint(1) NOT NULL default '1',
  `pop3` tinyint(1) NOT NULL default '1',
  PRIMARY KEY  (`email`)
) ENGINE=MyISAM COMMENT='Postfix Admin - Boites Emails Virtuelles';
```

```
> CREATE TABLE `alias` (
  `source` varchar(255) NOT NULL default '',
  `destination` text NOT NULL,
  `actif` tinyint(1) NOT NULL default '1',
  PRIMARY KEY  (`source`)
) ENGINE=MyISAM COMMENT='Postfix Admin - Alias Virtuels';
```

On rajoute l'utilisateur mysql pour postfix (pensez à remplacer MONMOTDEPASSE par le mot de passe désiré).

```
> GRANT SELECT ON `postfix`.* TO 'postfix'@'%' IDENTIFIED BY 'MONMOTDEPASSE';
```

On applique les paramètres :

```
> FLUSH PRIVILEGES;
```

Et on termine notre session mysql.

```
> exit;
```

VI-D - Fichiers de configuration

On va éditer/créer nos fichiers de configuration. N'oubliez pas de mettre vos informations à jour, sous peine d'erreurs.

```
# nano /etc/postfix/main.cf
smtp_banner = $myhostname ESMTP (Debian / GNU)
biff = no
disable_vrfy_command = yes
smtpd_helo_required = yes

append_dot_mydomain = no

mydestination = nom_de_votre_serveur, localhost, localhost.localdomain

mydomain = nom_de_votre_serveur

myhostname = nom_de_votre_serveur

relayhost =

mynetworks = 127.0.0.0/8 ip_de_votre_serveur
inet_interfaces = all

smtpd_sender_restrictions =
        permit_mynetworks,
        reject_unknown_sender_domain,
        warn_if_reject reject_unverified_sender

smtpd_recipient_restrictions =
        permit_mynetworks,
        reject_unauth_destination,
        reject_unknown_recipient_domain,
        reject_non_fqdn_recipient

smtpd_client_restrictions =
        reject_unknown_client,
        permit_mynetworks

virtual_alias_maps = mysql:/etc/postfix/mysql-virtual_aliases.cf,
mysql:/etc/postfix/mysql-virtual_aliases_mailbox.cf
virtual_mailbox_domains = mysql:/etc/postfix/mysql-virtual_domains.cf
virtual_mailbox_maps = mysql:/etc/postfix/mysql-virtual_mailboxes.cf
virtual_mailbox_base = /var/spool/vmail/
virtual_uid_maps = static:5000
virtual_gid_maps = static:5000
```

nano /etc/postfix/main.cf
```
virtual_create_maildirsize = yes
virtual_mailbox_extended = yes
virtual_mailbox_limit_maps = mysql:/etc/postfix/mysql-virtual_mailbox_limit_maps.cf
virtual_mailbox_limit_override = yes
virtual_maildir_limit_message = "Desole, la boite email de l'utilisateur est pleine. Veuillez re-essayer plus tard !"
virtual_overquota_bounce = yes
```

nano /etc/postfix/mysql-virtual_mailbox_limit_maps.cf
```
hosts = 127.0.0.1
user = postfix
password = VOTREMOTDEPASSE
dbname = postfix
select_field = quota
table = mailbox
where_field = email
```

nano /etc/postfix/mysql-virtual_aliases_mailbox.cf
```
hosts = 127.0.0.1
user = postfix
password = VOTREMOTDEPASSE
dbname = postfix
select_field = email
table = mailbox
where_field = email
additional_conditions = AND actif='1'
```

nano /etc/postfix/mysql-virtual_aliases.cf
```
hosts = 127.0.0.1
user = postfix
password = VOTREMOTDEPASSE
dbname = postfix
select_field = destination
table = alias
where_field = source
additional_conditions = AND actif='1'
```

nano /etc/postfix/mysql-virtual_mailboxes.cf
```
hosts = 127.0.0.1
user = postfix
password = VOTREMOTDEPASSE
dbname = postfix
select_field = CONCAT(SUBSTRING_INDEX(email,'@',-1),'/',SUBSTRING_INDEX(email,'@',1),'/')
table = mailbox
where_field = email
additional_conditions = AND actif='1'
```

nano /etc/postfix/mysql-virtual_domains.cf
```
hosts = 127.0.0.1
user = postfix
password = VOTREMOTDEPASSE
dbname = postfix
select_field = 'virtual'
table = domain
where_field = domain
additional_conditions = AND actif='1'
```

On modifie les droits de ces fichiers de configuration :

```
# chgrp postfix /etc/postfix/mysql-virtual_*.cf
# chmod u=rw,g=r,o= /etc/postfix/mysql-virtual_*.cf
```

Un redémarage de Postfix, et un test de la configuration :

```
# /etc/init.d/postfix restart
# postfix check
```

VI-E - Configuration du courrier

```
# apt-get install courier-base courier-authdaemon courier-authmysql courier-imap courier-pop
```

A la question posée, on répond par oui.

On va maintenant signaler à postfix que l'on utilise mysql pour l'identification :

```
# nano /etc/courier/authdaemonrc
/etc/courier/authdaemonrc
```

Et on modifie la ligne qui commence par authmodulelist.

```
authmodulelist="authmysql"
```

Je remplace le contenu du fichier /etc/courier/authmysqlrc par ce qui suit :

```
# >/etc/courier/authmysqlrc
# nano /etc/courier/authmysqlrc
```

```
MYSQL_SERVER            localhost
MYSQL_USERNAME          postfix
MYSQL_PASSWORD          MOTDEPASSE
MYSQL_PORT              0
MYSQL_OPT               0
MYSQL_DATABASE          postfix
MYSQL_USER_TABLE        mailbox
MYSQL_CRYPT_PWFIELD     password
MYSQL_UID_FIELD         5000
MYSQL_GID_FIELD         5000
MYSQL_LOGIN_FIELD       email
MYSQL_HOME_FIELD        "/var/spool/vmail/"
MYSQL_MAILDIR_FIELD     CONCAT(SUBSTRING_INDEX(email,'@',-1),'/',SUBSTRING_INDEX(email,'@',1),'/')
MYSQL_QUOTA_FIELD       quota
```

On pense à redémarrer les différents démons que l'on a paramétré :

```
# /etc/init.d/courier-authdaemon restart
# /etc/init.d/courier-imap restart
# /etc/init.d/courier-pop restart
```

L'accès POP ou IMAP échouera si la boîte mail n'existe pas ! En conséquence, je vous conseille d'envoyer un email de bienvenue lors de la création de vos comptes. Cela vous permettra de créer le répertoire !

VI-F - Test du serveur

On commence par installer les outils :

```
# apt-get install mailutils
```

On test que notre domaine est bien configuré dans bind pour accepter notre serveur mail :

```
# host -t MX nom_domaine
```

> *Il doit répondre nom_domaine mail is handled by 10 mail.nom_domaine.tld. Dans le cas contraire, il est nécessaire d'aller configurer BIND !*

On va ajouter notre compte dans notre base de données. N'oubliez pas de remplacer les valeurs par les vôtres.

```
# mysql -u root -p
==> On entre le mot de passe
> use postfix;
> INSERT INTO domain (domain, actif) VALUES ('nom_domaine.com', '1');
> INSERT INTO mailbox (email,password,quota,actif,imap,pop3) VALUES
('user@nom_domaine.com',ENCRYPT('MONMOTDEPASSEDUCOMPTEMAIL'),0,1,1,1);
> exit;
```

Et maintenant, on va envoyer un mail à notre propre utilisateur, et à une adresse externe (j'ai testé avec Gmail sans que ce soit reconnu comme spam) :

```
mail -s sujet
To : user@nom_domaine.com
Cc : user@domaine_externe.com
texte du mail
.
```

- mail -s ==> pour envoyer le sujet. Remplacez le mot sujet par un sujet
- To : l'adresse du compte créé ci-dessus
- Cc : je mets une autre adresse pour tester la réception vers une adresse externe
- Il faut terminer par une ligne contenant un . pour envoyer le mail

Si vous vérifiez votre logiciel mail, vous devriez obtenir le mail envoyé en Cc. Pour tester le mail envoyé à votre utilisateur, il vous suffit de rentrer les bons paramètres dans votre logiciel de mail préféré (Thunderbird).

```
pop: mail.nom_domaine.com
smtp: votre FAI
login: user@nom_domaine.com
pass: *****
```

VI-G - Conclusion

Voilà, votre MTA (Mail Transfert Agent) est normalement configuré. Si vous ne recevez rien, vérifiez bien tous les paramètres. Il est important d'avoir autorisé la connexion en local lors de l'installation de mysql, par exemple. En cas de doute, ou de souci, allez voir les logs :

```
# nano /var/log/mail.log
```

Je tiens tout particulièrement à remercier Fred036 pour son aide, et la mise à disposition de son tutorial, qui m'a bien aidé, et dont je me suis largement inspiré pour ce tutorial sur Postfix.

VII - Script

VII-A - Installation d'un nouvel hébergement

Ce script permet de créer les paramètres nécessaires pour créer un nouveau site. Il configure :

- Apache (virtualhost)
- Bind
- Vsftpd
- Postfix (création du compte mail webmaster)
- Copie des fichiers de base dans le répertoire home

Ce script a besoin, pour être exécuté, de 3 paramètres et d'un répertoire :

- nom de l'utilisateur shell créé
- mot de passe de cet utilisateur
- le nom de domaine correspondant

Avant d'utiliser le script pour la première fois, il faut commencer par créer l'arborescence de base qui devra exister dans le nouvel hébergement créé. Dans le même répertoire que votre script, créer l'arborescence suivante :

```
fic_originaux/
fic_originaux/logs/
fic_originaux/public_html/
fic_originaux/public_html/index.html
```

Vous pouvez évidemment mettre ce que vous voulez dans le dossier fic_originaux/public_html/. Pour ma part, je mets un fichier index.html contenant un message de bienvenue, et un fichier info.php qui contient la commande phpinfo(). Cela permet de vérifier que le php est activé et fonctionne correctement.

> *Ce script est livré en l'état, sans aucune assurance. Il peut être adapté, mais en aucun cas redistribué, sous quelque forme que ce soit. Il n'est certainement pas optimisé, mais il fonctionne avec le reste du tutoriel !*

Pour exécuter le script, il faut lui donner les droits d'exécution (X), et le lancer sous cette forme :

```
./new_site.sh user mdp domaine
```

new_site.sh

```
#
#------------------------------------------------------------------------------------------------------------------------
#                                                Script de création d'hébergement
#
# Auteur:         Lange Olivier
# Date:         2006/2007
# Description:
#                 Script permettant d'ajouter un nouvel hébergement dans le serveur. Il crée tous
les fichiers
#                 nécessaires, et effectue les configurations de base.
# Paramètre: 1° Nom de l'utilisateur Shell
#                 2° Mot de passe de l'utilisateur
#                 3° Nom de domaine
#------------------------------------------------------------------------------------------------------------------------


#!/bin/sh

# Récupère le nom d'utilisateur créé
new_user=$1
```

new_site.sh

```
# Récupère le nom de domaine
nom_domaine=$3

# Récupère le mot de passe souhaité
password=$2

# Définit le mot de passe root de mysql
mdp_mysql_root = XXXXXXXXXXXX

# Récupère la date de création pour générer le fichier Bind
date_creation=`date +%Y%m%d`01

# Récupère l'IP du serveur
mon_ip=`grep $HOSTNAME /etc/hosts |cut -f1`

# Définit le répertoire de base de l'utilisateur
rep=/home/$new_user

# Test si le répertoire existe, et donc l'utilisateur
if [ -d $rep ]; then
        echo "L'utilisateur est déja géré par ce serveur"
else
        if [ -e /etc/apache2/sites-enabled/$nom_domaine ]; then
                echo "Le nom de domaine est déja géré par ce serveur"
        else

                # Génère les fichiers de base du nouveau site
                /usr/sbin/useradd -p $password $new_user
                cp -R fic_originaux $rep
                chown -R www-data:www-data $rep
                chmod -R 755 $rep

#
#-----------------------------------------------------------------------------------------------------------------------
#                                                 Configuration d'apache
#-----------------------------------------------------------------------------------------------------------------------

                # Crée le fichier de configuration du répertoire virtuel
                echo "
<VirtualHost *>

        ServerAdmin postmaster@$nom_domaine
        ServerName www.$nom_domaine
        ServerAlias $nom_domaine *.$nom_domaine

        DocumentRoot /home/$new_user/public_html/

        <Directory /home/$new_user/public_html/>
                Options -Indexes FollowSymLinks MultiViews
                AllowOverride All
        </Directory>

        ErrorLog /home/$new_user/logs/error.log
        LogLevel warn
        CustomLog /home/$new_user/logs/access.log combined

        ServerSignature Off

</VirtualHost>


" >> /etc/apache2/sites-available/$nom_domaine
                ln -s /etc/apache2/sites-available/$nom_domaine
/etc/apache2/sites-enabled/$nom_domaine

#
#-----------------------------------------------------------------------------------------------------------------------
#                                                 Configuration de BIND
#-----------------------------------------------------------------------------------------------------------------------

                # Crée le fichier de déclaration de zone BIND
                echo "
\$ttl 86400
$nom_domaine.   IN      SOA     $HOSTNAME. webmaster.$nom_domaine. (
                        $date_creation
                        21600
                        3600
```

new_site.sh

```
                        604800
                        86400 )
                IN      NS      $HOSTNAME.
                IN      NS      ns.kimsufi.com.
                IN      MX      10 mail.$nom_domaine.
                IN      A       $mon_ip
www             IN      A       $mon_ip
mail            IN      A       $mon_ip
smtp            IN      A       $mon_ip
pop             IN      A       $mon_ip
pop3            IN      A       $mon_ip
imap            IN      A       $mon_ip
sql             IN      A       $mon_ip
mysql           IN      A       $mon_ip
" > /etc/bind/db.$nom_domaine

                # Crée la zone BIND
                echo "zone \"$nom_domaine\" {
        type master;
        file \"/etc/bind/db.$nom_domaine\";
};
" >> /etc/bind/zones.conf


#
#-------------------------------------------------------------------------------------------------------------------------------
#                                                 Configuration du ftp principal
#-------------------------------------------------------------------------------------------------------------------------------

                # On ajoute l'utilisateur et son mdp
                echo "$new_user
$password" > /etc/vsftpd/login.txt

                # On met à jour la base berkeley
                db3_load -T -t hash -f /etc/vsftpd/login.txt /etc/vsftpd/login.db

                # On ajoute les paramètres pour notre nouveau site
                echo "anon_world_readable_only=NO
local_root=$rep
write_enable=YES
anon_upload_enable=YES
anon_mkdir_write_enable=YES
anon_other_write_enable=YES" > /etc/vsftpd/vsftpd_user_conf/$new_user

#
#-------------------------------------------------------------------------------------------------------------------------------
#                                               Installation du compte mail par défaut
#-------------------------------------------------------------------------------------------------------------------------------

mysql -u root -e "INSERT INTO domain (domain, actif) VALUES ('$nom_domaine', '1');"
-p$mdp_mysql_root postfix
mysql -u root -e "INSERT INTO mailbox (email,password,quota,actif,imap,pop3) VALUES
('webmaster@$nom_domaine',ENCRYPT('$password'),0,1,1,1);" -p$mdp_mysql_root postfix


#
#-------------------------------------------------------------------------------------------------------------------------------
#                                           Redémarre les services
#-------------------------------------------------------------------------------------------------------------------------------
                /etc/init.d/bind9 restart
                /etc/init.d/apache2 restart
                /etc/init.d/vsftpd restart
                /etc/init.d/postfix restart

                echo "L'utilisateur $new_user a bien été enregistré sur le serveur"
                echo "Le serveur a été redémarré"
        fi
fi
```