

Guide d'installation de yacs sur Linux Debian

Version du 14 Janvier 2010

Table des matières

1	Introduction.....	5
1.1	Les caractéristiques techniques de l'architecture cible.....	5
1.2	De quoi avez-vous besoin ?.....	6
1.3	Commande de serveur virtuel privé (RPS) chez OVH.....	6
1.4	Installation de base de Linux Debian.....	7
1.5	Installation d'appliance VMware Linux Debian.....	8
2	Fiches de configuration.....	10
2.1	Paramètres du système.....	10
2.2	Paramètres de site virtuel.....	11
3	Préparation du système.....	12
3.1	Connexion et mise à jour du logiciel.....	12
3.2	Sécuriser l'accès SSH.....	13
3.3	Régler l'heure.....	15
3.4	Installer Apache.....	16
3.5	Optimiser la configuration d'Apache.....	17
3.6	Préparer un premier serveur virtuel.....	18
3.7	Ajouter l'accès HTTPS.....	21
3.8	Installer PHP.....	23
3.9	Ajouter la localisation géographique.....	25
3.10	Optimiser la configuration de PHP.....	26
3.11	Installer MySQL.....	28
3.12	Installer phpMyAdmin.....	31
3.13	Installer proftpd.....	32
3.14	Sécuriser les échanges FTP.....	33
3.15	Installer Postfix.....	34
3.16	Préparer les sauvegardes automatiques.....	36
3.17	Installation de xdebug.....	39
3.18	Quelques vérifications finales.....	39
4	Installation d'un site virtuel.....	42
4.1	Gestion du nom de domaine.....	42
4.2	Ajout d'un utilisateur.....	42
4.3	Installer le site virtuel.....	42
4.4	Créer un site virtuel sécurisé.....	44
4.5	Créer une base de données.....	46
4.6	Télécharger yacs.....	47
4.7	Installer yacs.....	48
4.8	Migrer un site yacs.....	49
4.9	Installer les traitement d'arrière-plan.....	50
4.10	Gérer les logs de yacs.....	51
4.11	Traiter les logs d'Apache.....	51
4.12	Installer piwik.....	52
4.13	Sauvegarder le serveur automatiquement.....	54
4.14	Forcer les accès sécurisés.....	58
4.15	Configurer xdebug.....	58
4.16	Quelques vérifications finales.....	60
4.17	Suppression d'un serveur virtuel.....	60

5 Référence.....	62
5.1 /etc/apt/sources.list.....	62
5.2 /etc/ssh/ssh_config.....	62
5.3 /etc/apache2/apache2.conf.....	63
5.4 /etc/apache2/conf.d/security.....	64
5.5 /etc/apache2/ports.conf.....	64
5.6 /etc/php5/apache2/php.ini.....	65
5.7 /etc/mysql/my.cnf.....	70
5.8 /etc/apache2/sites-available/www.domaine.fr.....	71
5.9 /etc/apache2/sites-available/www.domaine.fr-ssl.....	72

Que trouverez-vous dans ce guide ?

Ce guide sera utile aux personnes devant installer un serveur pour une ou plusieurs communautés yacs. Il est conçu pour optimiser les activités de l'ingénieur système ou du webmestre qui met en place une nouvelle machine, tant dans l'ordonnancement des tâches que dans leur contenu.

Comme vous le découvrirez dans ce guide, il n'est pas besoin d'être un expert Linux pour installer une machine performante. Suivez les instructions fournies et suivez les exemples pour construire, par vous-même, le serveur web dont vous avez besoin.

Une approche progressive et structurée

Ce guide est construit de façon linéaire, avec plusieurs parties correspondant aux grandes étapes de la construction d'un système web.

Section	Description
Introduction	<ul style="list-style-type: none">- les caractéristiques techniques de l'architecture cible- les pré-requis- achat ou location de serveur
Fiches de configuration	<ul style="list-style-type: none">- saisie des paramètres du système- saisie des paramètres de serveur virtuel
Préparation du système	<ul style="list-style-type: none">- mise à jour de Linux Debian- installation et sécurisation de tous les composants logiciels- tests progressifs de validation
Ajout d'un serveur virtuel	<ul style="list-style-type: none">- création d'un compte d'accès FTP- création d'un serveur virtuel Apache- création et configuration d'une instance de yacs- création de la base de données associée- gestion des traitements d'arrière-plan- sécurisation des accès au serveur
Référence	<ul style="list-style-type: none">- contenu des fichiers de configuration essentiels

Conventions

Il faut se rappeler que le texte mis en gras dans les listings est générique, et doit généralement être remplacé par une valeur spécifique à votre installation. Par exemple :

```
mkdir www.domaine.fr
```

Si vous installez un serveur pour le domaine `www.monserveur.fr`, vous taperez en réalité :

```
mkdir www.monserveur.fr
```

A propos de l'auteur

Bernard Paques est le créateur initial du système yacs, l'architecte du logiciel, et son principal développeur. Il est aussi webmestre de plusieurs serveurs, tous construits avec yacs, sur plate-forme Linux. En véritable praticien des plate-forme web dynamiques, il a formalisé son expérience dans ce guide, ensemble de bonnes pratiques développées sur le terrain à travers de multiples projets.

Pour en savoir plus sur l'auteur : <http://www.linkedin.com/in/webnetworker>

Remerciements

Ce document n'aurait pas vu le jour sans la communauté yacsienne. Merci aux muses, aux relecteurs, et à tous ceux qui ont pris un peu de leur temps pour améliorer ce guide.

Pour rejoindre la communauté : <http://www.yacs.fr>

Copyright et licence d'utilisation

Copyright © 2009 Bernard Paques

Ce guide est distribué selon les termes de la licence Creative Commons by-nc-sa 3.0. En conséquence, vous êtes autorisé à copier, modifier, et distribuer publiquement ce document, aux conditions suivantes :

- Attribution – Vous devez reconnaître le travail de l'auteur, en préservant les marques de copyright placées dans le document.
- Non-commercial – Vous ne pouvez pas valoriser ce travail à travers une activité commerciale sans l'autorisation de l'auteur.
- Partage similaire – Si vous modifiez ou réutilisez ce travail, vous pouvez distribuer le document obtenu, mais seulement sous la même licence ou sous une licence similaire.

1 Introduction

1.1 Les caractéristiques techniques de l'architecture cible

A l'issue des opérations décrites dans ce guide, vous obtiendrez un système web performant, évolutif, et sécurisé, tirant parti des meilleurs logiciels du moment :

- L'architecture s'appuie sur des briques de logiciel libre stables et récentes : Apache, PHP, MySQL, phpMyAdmin, Proftpd, Postfix. Ce guide fournit des explications pour installer et configurer chacun de ces éléments.
- L'installation est conçue pour héberger plusieurs serveurs web virtuels sur la même machine. Chaque serveur web est propulsé par une instance yacs distincte, et dispose d'une base de données indépendante.
- La configuration du système est optimisée pour tirer parti des fonctions avancées de yacs, comme par exemple l'URL rewriting, le cache par expiration, ou encore le module Ming.
- Chaque serveur web virtuel dispose d'un double accès, par `http:` et par `https:`. La sécurisation ne demande pas de rajouter d'adresse IP virtuelle, nous utiliserons le module GNUTLS, qui s'avère plus puissant que le module SSL traditionnel.
- Un compte FTP/TLS est créé pour chaque serveur virtuel, qui sera utilisé par l'administrateur du serveur pour gérer les fichiers à distance. Cet accès est sécurisé par chiffrement TLS, et limité au seul serveur virtuel (chroot sur l'espace virtuel).
- Nous avons choisi la version MPM-ITK d'Apache pour éviter les problèmes de permission sur les fichiers manipulés séparément par Apache et FTP, et pour améliorer le cloisonnement entre serveurs virtuels. Cette solution s'avère plus performante que suPHP ou suexec.
- Les traitements d'arrière-plan sont délégués à cron, ce qui allège d'autant le travail de yacs, et améliore les performances vues des internautes.
- Les fichiers de log sont gérés automatiquement, pour éviter l'encombrement progressif des espaces disques.
- Un script de duplication est fourni pour faciliter les sauvegardes des serveurs virtuels. Les fichiers sont soit envoyés à un serveur tiers (rsync), soit dupliqués localement pour récupération ultérieure.
- Le système est propulsé par Linux Debian testing (actuellement: Squeeze), et la procédure décrite ici est capable de migrer les machines installées initialement en Linux Debian Lenny ou Etch.
- L'installation est sécurisée par conception, pour réduire les risques d'attaque par des tiers.

Ce guide n'est pas approprié pour la mise en place d'un environnement à utilisateurs multiples, qui requière une approche différente de la sécurisation des accès à travers le réseau.

1.2 De quoi avez-vous besoin ?

Un serveur web propulsé par yacs est, avant tout, un ordinateur connecté à Internet, ou au réseau privé d'une entreprise. Pour construire ce guide nous avons considéré trois cas différents :

- commande de serveur virtuel privé (RPS) chez OVH – après paiement de votre réservation, l'hébergeur vous mettra à disposition du matériel (carte processeur, mémoire, interface réseau) dédié à votre seul usage, l'espace disque étant distribué sur des baies de disques partagées.
- installation sur un PC de Linux Debian à partir du CD-ROM de base – la procédure marche très bien sur un vieil ordinateur recyclé, à condition qu'il soit correctement relié au réseau pour achever l'installation.
- mise en oeuvre d'une appliance VMWare de Linux Debian – pour les professionnels des centres de données qui souhaitent tirer le maximum des techniques de virtualisation.

A chaque fois, nous sommes parti d'un système d'exploitation « brut de décoffrage », mais comme vous allez le voir en parcourant cette documentation, Linux Debian est finalement assez simple à mettre en oeuvre. Surtout, le système de gestion des paquets, et la solidité du noyau, en font un candidat idéal pour des serveurs professionnels.

Pour ce guide d'installation dédié aux services web propulsé par yacs nous n'avons pas tenu compte des services de nommage (DNS) ni des services de messagerie électronique. Dans le cas d'un serveur virtuel, ces services seront probablement fournis par l'hébergeur sur sa propre infrastructure. Pour les autres cas, vous ferez appel à des ingénieurs système pour vous aider. Il est parfaitement possible de compléter l'installation décrite ici pour obtenir un serveur « tout-en-un », avec DNS, SMTP, POP3, etc. En fait, Debian fournit beaucoup de liberté, mais tout dépend de vos compétences et de votre expérience.

Avant de passer au processus d'installation général, prenons quelques instants pour présenter les différents cas d'obtention d'une machine Linux Debian.

1.3 Commande de serveur virtuel privé (RPS) chez OVH

Je suis fidèle à OVH depuis plusieurs années, et le fait qu'il soit devenu l'un des hébergeurs leader en Europe, tout en maintenant une qualité de service et une flexibilité de l'offre tout à fait remarquables y est pour beaucoup.

Le site d'OVH : <http://www.ovh.com/>

Ce guide s'appuie sur l'installation concrète d'un serveur virtuel privé de type RPS (pour « Real Private Server »), mais il ne devrait pas y avoir beaucoup de différences pour d'autres types de serveurs privés. Et si vous êtes hébergé chez un autre fournisseur, ce guide devrait rester pertinent pour l'essentiel.

Pendant l'installation de serveur RPS chez OVH, choisir l'option Distribution de base, puis

sélectionner Système d'exploitation Linux Debian. Ce guide installe Linux Debian testing grâce aux mises à jour par paquets. Le choix de la langue est affaire de goût, mais par expérience j'ai choisi l'anglais.

Dans le courrier électronique envoyé par l'hébergeur vous trouverez le nom de la machine, son adresse réseau, ainsi que le mot de passe du compte `root`.

Note : par défaut OVH fournit l'espace disque extensible sous `/home`, comme beaucoup d'hébergeurs. Dans ce guide d'installation nous placerons à cet endroit le contenu des serveurs virtuels, ainsi que les bases de données et les sauvegarde.

1.4 Installation de base de Linux Debian

Pour cette installation, nous avons téléchargé l'image minimum (NetInstall), qui a été gravée ensuite sur CD-ROM, et insérée dans le PC au démarrage.

Adresse de l'image téléchargeable : <http://www.debian.org/CD/netinst/>

La documentation qui suit a été préparée à partir de cette installation minimale, qui effectue une bonne partie des opérations à travers le réseau Internet, avec les options de `tasksel` décochées.

Par défaut la machine utilise le protocole DHCP pour obtenir une adresse IP de connexion au réseau.

Si vous avez besoin de positionner une adresse statique, vous pouvez le faire comme suit :

```
nano /etc/network/interfaces
```

Vous indiquerez dans le fichier l'adresse statique, le masque de sous-réseau, ainsi que les adresses du routeur, et des serveurs DNS, en vous inspirant du modèle suivant :

```
auto eth0
iface eth0 inet static
address 192.168.3.90
gateway 192.168.3.1
netmask 255.255.255.0
```

Enregistrer les modifications et quitter l'éditeur.

Pour indiquer les serveurs DNS à utiliser, c'est dans le fichier `/etc/resolv.conf` que ça se passe :

```
nano /etc/resolv.conf
```

Plusieurs serveurs peuvent être listés, comme dans l'exemple suivant :

```
nameserver 206.12.23.13
nameserver 206.12.24.13
```

Enregistrer les modifications et quitter l'éditeur. Relancer les interfaces réseau pour prendre en compte les modifications :

```
/etc/init.d/networking restart
```

En fin de parcours, vous pourrez avoir à compléter l'installation comme suit, pour permettre l'accès distant à la machine:

```
aptitude install ssh
```

1.5 Installation d'appliance VMware Linux Debian

Ce guide s'applique aussi aux installations de serveurs à partir des fichiers préparés pour l'hyperviseur VMware. Voici quelques exemples de sources gratuites de fichiers Linux Debian pour VMware.

Linux Debian 5.0 Lenny Server	http://www.vmware.com/appliances/directory/1000
Linux Debian 4.0 Etch	http://www.visoracle.com/download/debian/

Après le démarrage initial de la machine virtuelle, vous pourrez avoir à changer les préférences pour tenir compte des spécificités du clavier français.

```
dpkg-reconfigure console-data
```

La première chose à faire est, bien entendu, de changer le mot de passe du compte `root`.

```
passwd
```

Si la commande `ifconfig` ne montre aucune interface réseau, vous pouvez tenter les commandes suivantes :

```
/etc/init.d/networking stop  
rmmod pcnet32  
rmmod vmxnet  
depmod -a  
modprobe vmxnet  
/etc/init.d/networking start
```

Ensuite, vous pourrez avoir à configurer le réseau, et ajouter l'accès SSH, comme expliqué au chapitre précédent.

Si vous êtes tombé sur une machine virtuelle particulièrement spartiate, vous pourriez être tenté d'installer les outils VMware (en anglais : « VMware Tools »). Cette opération est dangereuse, et fortement déconseillée aux débutants. Si ces avertissements ne vous font pas peur, alors procéder comme suit :

```
aptitude install build-essential psmisc
aptitude install linux-headers-`uname -r`
```

Basculer vers le superviseur VMware, et demander l'installation des VMware Tools. Ceci va créer un disque virtuel visible depuis Linux. Revenir à la console et taper :

```
mount /dev/cdrom /mnt/
tar -C /tmp -zxvf /mnt/VMwareTools-5.5.3-346885.tar.gz
umount /mnt
cd /tmp/vmware-tools-distrib
./vmware-install.pl
```

Pour la configuration, vous pouvez à tout moment entrer la commande :

```
/usr/bin/vmware-config-tools.pl
```

Pour lancer la boîte à outils, taper :

```
vmware-toolbox &
```

2 Fiches de configuration

Imprimez les tableaux suivants pour mémoriser, pendant l'installation, les paramètres dont vous pourrez avoir besoin par la suite pour les besoins de gestion.

2.1 Paramètres du système

Paramètre	Valeur	Notes
Nom de machine		Fourni par la commande <code>hostname</code>
Adresse de réseau		Regarder l'attribut <code>inet addr</code> de la commande <code>ifconfig</code> Exemple : 123.123.123.123
Masque de réseau		Regarder l'attribut <code>Mask</code> de la commande <code>ifconfig</code>
Adresse de routeur		Regarder la colonne Gateway de la commande <code>route</code>
Adresses des serveurs DNS		Les lignes <code>nameserver</code> de la commande <code>cat /etc/resolv.conf</code>
Mot de passe <code>root</code>		Modifiable par la commande <code>passwd</code>
Mot de passe <code>remote</code>		Voir chapitre 3.2 Modifiable par la commande <code>passwd remote</code>
Mot de passe <code>snapshot</code>		Voir chapitre 3.16 Modifiable par la commande <code>passwd snapshot</code>
Mot de passe <code>root</code> pour MySQL		Voir chapitre 3.11
Adresse de courrier électronique		Voir chapitre 3.6 Par défaut : <code>webmaster@localhost</code>
Serveur d'envoi de messages		Le serveur SMTP indiqué à Postfix

2.2 Paramètres de site virtuel

Paramètre	Valeur	Notes
Nom de serveur		Exemple : <code>www.domaine.fr</code>
Adresse de contact		Voir chapitre 4.3 Exemple : <code>contact@domaine.fr</code>
Compte associé		Exemple : <code>domaine</code>
Mot de passe pour FTP pour ce compte		Modifiable par la commande <code>passwd domaine</code>
Mot de passe pour MySQL pour ce compte		Voir chapitre 4.5
Nom de la base de données		Dérivé du nom de serveur, en remplaçant les points par des soulignements. Exemple : <code>www_domaine_fr</code>
Serveur d'envoi de messages		Si différent du serveur configuré par défaut

3 Préparation du système

A ce stade, vous disposez d'un ordinateur doté de Linux Debian et nous allons, dans cette section, ajouter tous les composants logiciels pour en faire un véritable serveur web performant et sécurisé.

3.1 Connexion et mise à jour du logiciel

Pour se connecter, il faut un logiciel qui gère l'écran sur votre poste de travail, et qui mette en oeuvre le protocole de communication SSH avec le serveur.

PuTTY fait ça très bien pour les machines sous Windows. Ce logiciel est disponible en téléchargement à peu près partout. Si vous pratiquez l'anglais, vous pouvez utiliser la page créée par son auteur :

<http://www.chiark.greenend.org.uk/~sgtatham/putty/>

Sur les Mac, comme sur les machines sous Linux, vous activerez une fenêtre de Terminal, et taperez la commande de connexion `ssh` comme suit, en remplaçant `nom_du_serveur` par le nom réel ou, à défaut, par l'adresse IP de la machine cible :

```
ssh root@nom_du_serveur
```

Après que vous ayez indiqué un nom et un mot de passe pour vous faire reconnaître par le serveur, vous êtes en position d'accéder, à travers la session `ssh`, à tous les fichiers, et à toutes les commandes de la machine à installer.

Le système d'exploitation Linux Debian se caractérise par l'emploi de paquets téléchargés à la demande. La première chose à faire est de nous assurer de l'origine des paquets comme suit :

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

L'idée est de mettre en oeuvre la version `testing` de Debian plutôt que la `stable` (c'est-à-dire, au moment de l'écriture de ce guide, de `Squeeze` plutôt que `Lenny`), et de connecter la machine à un référentiel situé en France. Remplacer le contenu du fichier par les lignes suivantes :

```
deb http://ftp.fr.debian.org/debian/ testing main contrib
#deb-src http://ftp.fr.debian.org/debian/ testing main contrib

deb http://security.debian.org/ testing/updates main contrib
#deb-src http://security.debian.org/ testing/updates main contrib
```

Les lignes commentées par le caractère `#` sont utiles seulement sur les machines de développement, lorsqu'il faut recompiler quelque chose. Nous n'en aurons pas besoin ici, puisque nous nous limitons aux machines de production ou de pré-production. Toutefois, les développeurs pourront utiliser le

même fichier pour construire une machine adaptée à leurs besoins.

Enregistrer les modifications et quitter l'éditeur.

Ensuite, nous mettons à jour le référentiel des paquets Debian, ainsi que l'ensemble des paquets installés :

```
aptitude update
aptitude install aptitude
aptitude dist-upgrade
```

Note : la séquence proposée pourra sembler un peu complexe aux spécialistes, mais elle est proposée ici par ce qu'elle fonctionne à la fois pour une simple mise à jour de Linux Debian Testing, ou pour une migration à partir d'une version stable (Lenny ou Etch).

Suivant le cas, ces opérations peuvent être quasiment instantanées, ou bien requérir quelques minutes d'échanges intenses sur le réseau et d'écriture sur le disque dur. Refaire la commande `update` suivie de `dist-upgrade` plusieurs fois si nécessaire, jusqu'à ce que la dernière commande indique qu'il n'y a aucun paquet à rafraîchir.

Profitons-en également pour installer `openssl`, qui servira plusieurs fois pendant l'installation :

```
aptitude install openssl
```

3.2 Sécuriser l'accès SSH

Cette sécurité empêche les accès directs à la machine, ainsi que les transferts de fichiers par SFTP, sauf pour un ou deux comptes réservés à cet usage. Les autres usagers du serveur se contenteront d'un accès par les formulaires de `yacs`, ou par transfert de fichier FTP, comme expliqué plus loin dans ce document.

La procédure mise en oeuvre ici pourra sembler simpliste aux experts de SSH et du `chroot`. En tout cas, elle a le mérite d'être efficace et facile à mettre en oeuvre. Le but est :

- de créer un compte d'utilisateur `remote` pour se connecter en SSH,
- d'interdire le login sur le compte `root` pour éviter les attaques directes sur ce compte,
- d'interdire les accès SSH, sauf pour `remote`.

Peut-être aurez-vous créé le compte `remote` pendant l'installation de Linux Debian, si vous êtes parti du CD-ROM. Dans ce cas, vous pouvez placer le compte dans le groupe `www-data` comme suit :

```
usermod -aG www-data remote
```

Sinon, nous allons le faire maintenant, en prenant le soin de choisir un mot de passe solide :

```
adduser -ingroup www-data remote
```

Passons ensuite au fichier de configuration SSH :

```
nano /etc/ssh/sshd_config
```

Ajouter une ligne pour être sûr des comptes autorisés à utiliser SSH :

```
AllowUsers remote
```

Bloquer l'authentification directe du compte `root` :

```
PermitRootLogin no
```

En profiter pour ajouter un réglage paranoïaque :

```
X11Forwarding no
```

Enregistrer les modifications et quitter l'éditeur.

Relancer le démon SSH pour prendre la modification en compte :

```
/etc/init.d/ssh restart
```

Après la modification, il faudra se connecter en deux temps pour être `root` en SSH. S'authentifier tout d'abord comme `remote`, puis taper la commande `su` et indiquer le mot de passe de `root` pour avoir tous les pouvoirs.

Une autre possibilité, presque équivalente, est de se connecter en SSH avec le compte `remote`, puis d'utiliser la commande suivante, en répétant le mot de passe de `remote`. L'avantage est qu'il n'est même plus nécessaire de connaître le mot de passe de `root` dans ce cas :

```
sudo -s
```

C'est fastidieux, certes, mais la sécurité est à ce prix. En réalité, en dehors des phases d'installation et de configuration, les besoins d'accès en tant que `root` sont relativement rares. Par contre, les risques d'attaque par des tiers sont permanents.

Pour être sûr que `remote` puisse lire les logs du système, nous l'ajoutons au groupe `adm` :

```
usermod -aG adm remote
```

Nous donnons aussi à `remote` le pouvoir d'intervenir directement sur tous les fichiers des serveurs virtuels :

```
mkdir /var/www
chown root:www-data /var/www
chmod 775 /var/www
```

Et puis, pour faciliter l'administration au quotidien, plaçons un raccourci pour le compte `remote` vers tous les sites virtuels :

```
cd /home/remote
ln -s /var/www www
```

Pour simplifier l'usage des commandes réservées à `root`, nous allons installer la commande `sudo`, et indiquer au système que le compte `remote` est habilité à passer toutes les commandes. :

```
aptitude install sudo
visudo
```

Si un éditeur « décent » est lancé, il vous suffira d'insérer la ligne suivante, puis d'enregistrer les modifications et de quitter :

```
remote ALL=(ALL) ALL
```

Si vous vous retrouvez sous l'antique éditeur `vi`, c'est un peu plus compliqué, mais guère plus : se déplacer vers la fin du fichier, puis taper la lettre `i` pour passer en mode insertion, puis la séquence `remote ALL=(ALL) ALL` et enfin la touche `<Esc>` pour repasser en mode commande. Sauver et quitter le programme en tapant les lettres `:wq` suivies de la touche `<Entrée>`.

Dans tous les cas, prenez le temps de vérifier le travail en tapant :

```
tail /etc/sudoers
```

3.3 Régler l'heure

Pour ajuster l'heure du serveur, le mieux est d'installer un paquet capable de se synchroniser avec des serveurs de référence :

```
aptitude install ntpdate
```

Synchroniser le serveur en choisissant une référence proche de lui, par exemple en France :

```
ntpdate fr.pool.ntp.org
```

Pour éviter toute dérive de l'horloge, vous pouvez planifier de lancer la commande de temps en temps :

```
nano /etc/cron.hourly/ntpdate
```

Taper le contenu suivant :

```
#!/bin/sh  
ntpdate fr.pool.ntp.org
```

Enregistrer les modifications et quitter l'éditeur.

Ne pas oublier de rendre le fichier exécutable avec la commande :

```
chmod a+x /etc/cron.hourly/ntpdate
```

3.4 Installer Apache

La commande suivante charge les paquets requis pour Apache :

```
aptitude install apache2-mpm-itk
```

Note : Nous choisissons une version d'Apache qui permet de distinguer les différents propriétaires des serveurs virtuels, ce qui est plus simple et plus efficace que l'installation du suexec ou de suPHP.

Plusieurs modules d'extension Apache sont intéressants pour une exécution optimisée de yacs. Nous verrons, lors de l'installation des serveurs virtuels, comment les exploiter efficacement par l'assistant de construction du fichier `.htaccess`. Pour l'instant, contentons-nous de les intégrer au système de base.

Taper la commande suivante pour installer les modules en charge de la ré-écriture des liens, de la compression et du cache par expiration :

```
a2enmod rewrite deflate expires
```

3.5 Optimiser la configuration d'Apache

La configuration par défaut fonctionne très bien, et les modifications que nous allons effectuer ont seulement pour but d'obtenir encore mieux d'Apache, soit en termes de sécurité, soit en termes de performances.

Modifier le fichier de configuration principal :

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

Nous allons tenter de tirer le meilleur parti de la machine en raisonnant de façon très simple sur les quantités de mémoire. La seule information dont vous avez besoin à ce stade est la taille de la mémoire physique du serveur.

A raison de 15 Mo de mémoire par processus observés sur un serveur yacs en production, nous limiterons, très grossièrement, leur nombre à la moitié de la mémoire du serveur, divisée par 15. Soit, pour une machine avec 2 Go de mémoire vive, dont 1 Go affectés à Apache :

Nombre maximal de processus = $1000 / 15 = 66 \sim 70 = \text{MaxClients}$

Au-delà de la limite que nous nous fixons, les tentatives de connexion seront placées en file d'attente, sans consommer trop de ressources de la machine. Les internautes qui seront servis le seront correctement. Et les autres ne le seront pas du tout. En plafonnant la quantité de requêtes acceptables, nous évitons l'écroulement des temps de réponse en cas de surcharge du serveur. Cette précaution est importante, et vous ferez attention de donner une valeur réaliste au paramètre `MaxClients`, car c'est une assurance importante contre toutes sortes de problèmes.

Chercher dans le fichier la section `prefork MPM` et changer les valeurs de `MaxClients` et de `ServerLimit` par le résultat de notre calcul. Soit, dans l'exemple d'une machine avec 2 Go de mémoire :

```
MaxClients 70
ServerLimit 70
```

Nous allons aussi améliorer la stabilité du système dans le temps, et demander à Apache de purger chaque processus après qu'il ait traité un certain nombre de requêtes :

```
MaxRequestsPerChild 50000
```

Par défaut, Apache attend 15 secondes avant de casser les connexions TCP. En pratique, on peut diviser cette valeur par 2, ce qui accroît d'un même facteur la quantité de connexions disponibles à un instant donné.

```
KeepAliveTimeout 7
```

Enregistrer les modifications et quitter l'éditeur.

Ensuite nous passons aux directives de sécurité :

```
nano /etc/apache2/conf.d/security
```

Changer la valeur des paramètres suivants, pour minimiser l'exposition du service web aux pirates éventuels :

```
ServerTokens Prod
ServerSignature Off
TraceEnable Off
```

Enregistrer les modifications et quitter l'éditeur.

Nous voulons aussi que PHP ait la priorité sur HTML pour les pages d'index :

```
nano /etc/apache2/mods-available/dir.conf
```

Modifier la ligne comme suit pour que `index.php` figure juste après le mot-clé `DirectoryIndex` :

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

Enregistrer les modifications et quitter l'éditeur.

3.6 Préparer un premier serveur virtuel

Nous prévoyons de mettre en service plusieurs serveurs virtuels partageant la même adresse réseau, et c'est ce qu'il faut indiquer à Apache.

```
nano /etc/apache2/ports.conf
```

Indiquer que les adresses réseau vont être partagées :

```
NameVirtualHost *:80
Listen 80

<IfModule mod_gnutls.c>
    NameVirtualHost *:443
    Listen 443
</IfModule>
```

Attention, il faut changer le nom du module par défaut, de `mod_ssl.c`, en `mod_gnutls.c`.

La deuxième partie du fichier anticipe sur la sécurisation des échanges de données, que nous aborderons un peu plus tard.

Enregistrer les modifications et quitter l'éditeur.

Visiter le répertoire `/var/www` pour vérifier l'existence du répertoire `default` :

```
cd /var/www
ls
```

Si le répertoire `apache2-default` existe, le renommer en `default` comme suit :

```
mv apache2-default default
```

Si, en revanche, aucun répertoire n'existe à cet endroit, en créer un pour le serveur virtuel par défaut, et déplacer la page d'index de bienvenue :

```
mkdir default
mv index.html default
```

Pour bénéficier du cloisonnement entre les serveurs virtuels, nous allons assigner un propriétaire pour chacun d'eux, et aligner les permissions sur les fichiers et les comptes d'exécution d'Apache sur cette assignation.

Nous choisirons pour chaque serveur virtuel un compte créé spécifiquement pour cet usage, donc différent de `www-data`, le compte utilisé par défaut par Apache, et de `root`. Pour le serveur virtuel par défaut, et afin de faciliter les interactions par FTP, nous prendrons le compte `remote`.

Pour le groupe, le mieux est d'utiliser `www-data` pour permettre à ce compte d'intervenir sur tous les sous-répertoires. Ceci donne donc la combinaison suivante :

```
chown -R remote:www-data default
```

Vérifions à présent la configuration du serveur virtuel par défaut :

```
nano /etc/apache2/sites-available/default
```

Le contenu du fichier de configuration pour ce serveur virtuel sera défini comme suit :

```
<VirtualHost *:80>
    AssignUserID remote www-data
    ServerAdmin webmaster@localhost

    DocumentRoot /var/www/default
    <Directory />
```

```

        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /var/www/default>
        Options FollowSymLinks
        AllowOverride FileInfo Indexes Options
        Order allow,deny
        allow from all
    </Directory>

    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
    <Directory "/usr/lib/cgi-bin">
        AllowOverride None
        Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
        Order allow,deny
        Allow from all
    </Directory>

    ErrorLog /var/log/apache2/error.log
    LogLevel warn
    CustomLog /var/log/apache2/access.log combined

</VirtualHost>

```

Note : La directive `AssignUserID` est spécifique à la version Apache MPM ITK, c'est celle qui permet de définir le compte et le groupe associés à l'exécution des scripts pour ce serveur virtuel.

Ce fichier de configuration est particulièrement dépouillé, pour des raisons de sécurité, et vous devez commentez ou supprimer les lignes qu'il pourrait y avoir en plus. On s'en tient au strict nécessaire, c'est plus sûr.

Enregistrer les modifications et quitter l'éditeur.

Redémarrer Apache pour prendre en compte toutes les modifications :

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

Dans un navigateur, faire pointer sur l'adresse IP du serveur et vérifier que Apache retourne bien un petit message en HTML : « It works! ».

En cas de problème, regarder le contenu de la log d'erreur comme suit :

```
tail /var/log/apache2/error.log
```

Pour analyser les logs produites par Apache, nous installerons le paquet `awstats` :

```
aptitude install awstats
```

Nous allons rajouter un fichier de configuration Apache partagé par tous les serveurs virtuels:

```
nano /etc/apache2/conf.d/awstats
```

La seule directive nécessaire sert à indiquer l'emplacement des icônes utilisées dans les rapports construits par le programme :

```
Alias /awstats-icon/ /usr/share/awstats/icon/
```

Enregistrer les modifications et quitter l'éditeur.

Supprimons le lancement générique de l'outil, puisque nous l'activerons séparément pour chaque serveur virtuel par la suite :

```
rm /etc/cron.d/awstats
```

L'analyse des logs par `awstats` sera finalisée lors de l'installation de chaque serveur virtuel, nous verrons cela un peu plus loin dans ce guide.

3.7 Ajouter l'accès HTTPS

Vous pouvez passer directement au chapitre suivant si vous n'avez pas besoin de chiffrer les communications vers le serveur web.

La méthode d'accès HTTPS est absolument nécessaire pour les sites de commerce électronique, ou lorsque certaines informations doivent être chiffrées pendant leur transmission. Ceci s'applique au contenu du serveur, bien sûr, mais aussi à la transmission des mots de passe pendant l'authentification yacs par exemple.

Cette sécurisation ajoute un peu de complexité à l'installation, pour deux raisons :

- il faut installer un certificat numérique authentifiant le serveur pendant l'établissement des connexions sécurisées,
- et il faut gérer des serveurs virtuels distingués par leur nom, mais partageant la même adresse de réseau (IP).

Pour le certificat numérique, nous utiliserons ici le certificat fourni par défaut lors de l'installation de `openssl`. Il est auto-signé, et produit donc une alerte de sécurité dans les navigateurs lors de la première connexion. Mais pour démarrer une installation, et pour permettre de sécuriser son administration, c'est largement suffisant.

Pour le support des serveurs virtuels, nous devons utiliser une option avancée du protocole TLS, et pour ceci nous remplacerons le module SSL habituel dans Apache par le module GNUTLS, comme suit :

```
aptitude install libapache2-mod-gnutls
a2dismod ssl
a2enmod gnutls
```

Créer un serveur virtuel pour l'accès HTTPS comme suit :

```
nano /etc/apache2/sites-available/default-ssl
```

Attention, cette configuration diffère des modèles habituels basés sur `mod_ssl`. Ici, nous nous basons sur les fonctionnalités de `mod_gnutls`, qui sont plus puissantes :

```
<IfModule mod_gnutls.c>
<VirtualHost *:443>
    AssignUserID remote www-data
    ServerAdmin webmaster@localhost

    DocumentRoot /var/www/default
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /var/www/default>
        Options FollowSymLinks
        AllowOverride FileInfo Indexes Options
        Order allow,deny
        allow from all
    </Directory>

    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
    <Directory "/usr/lib/cgi-bin">
        AllowOverride None
        Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
        Order allow,deny
        Allow from all
    </Directory>

    ErrorLog /var/log/apache2/error.log
    LogLevel warn
```

```
CustomLog /var/log/apache2/access.log combined

GnuTLSEnable on
GnuTLSCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem
GnuTLSKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
GnuTLSPriorities NORMAL

</VirtualHost>
</IfModule>
```

Il est très important de supprimer toutes les lignes faisant référence au module `mod_ssl`. Le mieux est de s'en tenir strictement au contenu présenté ici, ni plus ni moins.

Enregistrer les modifications et quitter l'éditeur.

Activer le nouveau serveur virtuel comme suit :

```
a2ensite default-ssl
/etc/init.d/apache2 reload
```

Pour tester le fonctionnement, indiquer l'adresse sécurisée du serveur virtuel dans la fenêtre d'un navigateur web :

```
https://www.domaine.fr/
```

Le navigateur doit se plaindre du fait que le certificat est auto-signé. Accepter l'exception de sécurité pour valider l'accès. Normalement, cette opération ne doit être faite qu'une seule fois. Ensuite le navigateur affiche le message : « It works! ».

En cas de problème, regarder les messages d'erreur dans `/var/log/apache2/error.log`.

En environnement public, il sera bon d'acheter un certificat signé par une autorité de certification bien connue, et d'installer le certificat correspondant dans le répertoire `/etc/ssl/certs` avec les autres certificats connus sur cette machine. Le fichier de configuration `default-ssl` sera modifié pour pointer sur ce certificat en lieu et place de celui que nous avons créé nous-mêmes, et il faudra redémarrer Apache pour faire prendre en compte le changement. Les internautes pourront alors naviguer sur le service sécurisé sans recevoir de message d'avertissement du navigateur.

3.8 Installer PHP

Passons ensuite à l'installation de PHP :

```
aptitude install libapache2-mod-php5 php5-cli
```

Le moteur d'exécution des programmes PHP sera utilisé à la fois comme module Apache (pour la création dynamique de page web), et avec l'interface en ligne de commande (pour les traitements d'arrière-plan).

Comme nous l'avons fait pour le serveur Apache, nous allons compléter l'environnement d'exécution PHP avec plusieurs modules utiles à yacs. La gestion des paquets de Debian va faire tout le travail de téléchargement et d'installation, une fois de plus :

```
aptitude install php5-gd php5-curl php5-ming php5-xcache php5-xsl
```

Cette simple commande ajoute :

- le module gd pour créer les vignettes des images téléversées,
- le module curl pour se connecter à des machines distantes (intégration RSS),
- le module ming pour permettre à yacs de construire quelques objets Flash dynamiques,
- la fonction xcache pour accélérer l'exécution des scripts PHP,
- le module xsl pour la transformation de données XML

L'installation du module xcache est particulièrement efficace pour améliorer les temps de réponse d'au moins un facteur 2. Il s'agit donc d'une solution extrêmement simple, et gratuite, qui vous permettra de tirer le meilleur parti des ressources CPU et mémoire disponibles.

Pour tirer le meilleur parti de xcache, vous pouvez avoir à lui indiquer que la machine dispose de plusieurs processeurs, ou encore que vous souhaiteriez mettre en cache aussi quelques variables.

Les paramètres `xcache.count` et `xcache.var_count` doivent mentionner le nombre de processeurs de la machine. Pour le trouver, tapez la commande suivante :

```
cat /proc/cpuinfo | grep -c processor
```

Pour modifier les paramètres d'exécution de xcache :

```
nano /etc/php5/conf.d/xcache.ini
```

Pour information, voici quelques-uns des paramètres utilisés pour la mise en place de www.yacs.fr :

```
xcache.size = 16M
xcache.count = 2
xcache.var_size = 16M
xcache.var_count = 2
xcache.cacher = Off
xcache.stat = Off
```

```
xcache.optimizer = On
```

Enregistrer les modifications et quitter l'éditeur.

Pour vérifier l'installation vous pouvez taper la commande :

```
php -v
```

Si vous avez besoin de faire authentifier les usagers par un annuaire d'entreprise, vous pouvez ajouter le module LDAP comme suit :

```
aptitude install php5-ldap
```

Faire prendre en compte les nouvelles extensions de PHP par Apache :

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

3.9 Ajouter la localisation géographique

Plus exactement, il s'agit d'exploiter le fait que les adresses de réseau sont affectées à des zones territoriales clairement identifiées. L'adresse utilisée par un surfeur donnera donc des indications précieuses sur son lieu de connexion au réseau. Cette fonction sera particulièrement utile aux sites de commerce électronique, et elle est très facile à mettre en oeuvre sous Debian :

```
aptitude install libapache2-mod-geoip
```

Activer le module dans Apache :

```
nano /etc/apache2/mods-available/geoip.conf
```

Enlever le commentaire pour indiquer l'emplacement du fichier des données, pour obtenir :

```
<IfModule mod_geoip.c>  
    GeoIPEnable On  
    GeoIPDBFile /usr/share/GeoIP/GeoIP.dat  
</IfModule>
```

Enregistrer les modifications et quitter l'éditeur.

Faire prendre en compte la nouvelle configuration dans Apache :

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

On pourra alors créer un petit script PHP dans le répertoire du serveur virtuel :

```
nano /var/www/default/test.php
```

Y placer le contenu suivant, puis sauvegarder et enregistrer le fichier :

```
<?php echo $_SERVER['GEOIP_COUNTRY_NAME']; ?>
```

Dans un navigateur, faire pointer sur l'adresse IP du serveur avec le chemin `/test.php` et vérifier qu'un nom de pays, ou de continent, est bien affiché à l'écran. La mention 'N/A' est normale sur un réseau à adressage privé.

Note : la base de données utilisée par GeoIP sera mise à jour en même temps que les autres paquets installés sur le serveur, en lançant périodiquement les commandes suivantes :

```
aptitude update  
aptitude dist-upgrade
```

3.10 Optimiser la configuration de PHP

Les changements apportés au fonctionnement de PHP sont importants, soit parce qu'ils sécurisent le fonctionnement du serveur, soit parce qu'ils augmentent les ressources fournies à `yacs`. vous devez impérativement vous y tenir, sous peine de piratage ou, à tout le moins, de limitations fonctionnelles importantes.

Changeons le fichier de configuration de PHP :

```
nano /etc/php5/apache2/php.ini
```

Le paramètre qui suit empêche l'exécution de commande à trop bas niveau. A part la fonction `shell_exec`, qui est utilisée pour certains appels système, nous considérons que les autres seront mieux dans un script shell ou une crontab, si besoin. Et le chargement de bibliothèques, c'est dans le `php.ini` que ça se passe, et pas ailleurs :

```
disable_functions = system, passthru, exec, popen, proc_open, dl
```

Nous allons aussi empêcher l'exécution de script distant, pour éviter les attaques par redirection :

```
allow_url_fopen = Off
```

Le paramètre qui suit limite l'exposition de PHP, il rend l'installation conforme aux standards à long terme du web :

```
expose_php = Off
```

Nous allons aussi autoriser les gros téléchargements, pour dépasser la limite par défaut qui est seulement de 2M. Pour déplacer cette limite à 32M, changer deux paramètres comme suit :

```
post_max_size = 64M
upload_max_filesize = 32M
```

Le paramètre qui suit est important pour l'exécution correcte de yacs. La valeur indiquée ici est celle proposée par défaut lors de l'installation de PHP 5.

```
memory_limit = 128M
```

Nous allons aussi capturer les erreurs d'exécution PHP, histoire de garder une trace des problèmes rencontrés par les visiteurs :

```
log_error = On
error_log = /var/log/php.error.log
```

Yacs n'utilise pas le tableau `$_ENV`, ni les anciennes variables `$HTTP_*`, donc nous pouvons épargner à PHP quelques cycles CPU pour éviter de générer tout cela :

```
variables_order = GPCS
register_long_arrays = Off
```

Yacs ne force pas les appels par référence, conformément aux recommandations de la communauté PHP :

```
allow_call_time_pass_reference = Off
```

Les paramètres suivants sont positionnés correctement avec PHP 5. Ils sont indiqués pour mémoire, au cas où vous auriez besoin de reprendre une installation ancienne...

```
register_globals = off
```

Enregistrer les modifications et quitter l'éditeur.

Comme nous utilisons PHP aussi en ligne de commande, nous allons créer un lien symbolique vers la configuration que nous avons préparé pour le module Apache, comme suit :

```
cd /etc/php5/cli
```

```
ln -fs ../apache2/php.ini php.ini
```

Avec cette disposition, tout changement des paramètres PHP côté Apache sera aussi effectif pour les traitements d'arrière-plan.

Redémarrer Apache pour prendre en compte toutes les modifications au niveau de PHP :

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

On pourra alors créer un petit script PHP dans le répertoire du serveur virtuel comme suit :

```
echo "<?php phpinfo(); ?>" >/var/www/default/test.php
```

Dans un navigateur, faire pointer sur l'adresse IP du serveur avec le chemin `/test.php` et vérifier qu'on obtient effectivement des informations complètes sur le moteur d'exécution PHP, y compris la présence des modules curl, gd et ming.

3.11 Installer MySQL

Installer MySQL avec les paquets Debian :

```
aptitude install mysql-server mysql-client php5-mysql
```

Changer le mot de passe `root` pour MySQL, en choisissant une chaîne compliquée, et différente de celle utilisée pour le compte système `root`. Si cela n'est pas demandé pendant l'installation de MySQL, vous pouvez le faire par vous-même comme dans l'exemple ci-dessous, en remplaçant `new-password` par sa vraie valeur :

```
mysql -p
mysql> use mysql;
mysql> update user set password=PASSWORD('new-password') where user='root';
mysql> flush privileges;
mysql> exit;
```

Si vous avez besoin de déplacer les bases de données de leur emplacement par défaut à, disons, `/home`, vous pouvez procéder comme suit :

```
/etc/init.d/mysql stop
mkdir /home/mysql
chown mysql:mysql /home/mysql
usermod --home /home/mysql mysql
cp -Rp /var/lib/mysql/* /home/mysql/
rm -R /var/lib/mysql
```

```
ln -s /home/mysql /var/lib/mysql
```

Le fichier de configuration de MySQL est dans `/etc/mysql` et, pour l'éditer, vous pouvez utiliser la commande suivante :

```
nano /etc/mysql/my.cnf
```

Tous les paramètres qui suivent doivent être placés après le mot-clé `[mysqld]`.

Par défaut, l'indexation du texte réalisée par MySQL se limite aux mots de quatre lettres ou plus. Pour les usagers, il est plus confortable de changer ce réglage pour prendre en compte aussi les mots de trois lettres, en ajoutant la ligne suivante :

```
ft_min_word_len = 3
```

Les paramètres suivants sont ceux qui ont été utilisés pour un site yacs assez important, avec une seule base de données :

```
key_buffer                = 64M
max_allowed_packet        = 16M
read_buffer                = 1M
sort_buffer                = 4M
table_cache               = 256
thread_cache_size         = 32
thread_stack               = 192K
max_connections           = 128
#
# * Query Cache Configuration
#
query_cache_limit         = 2M
query_cache_size          = 32M
query_cache_type          = 1
```

Nous activons aussi la détection des requêtes longues en décommentant les lignes suivantes :

```
log_slow_queries = /var/log/mysql/mysql-slow.log
long_query_time = 2
```

Enregistrer les modifications et quitter l'éditeur.

Redémarrer MySQL pour être sûr de prendre la nouvelle configuration en compte :

```
/etc/init.d/mysql restart
```

Si le redémarrage du serveur s'avère problématique, ce peut-être dû à une erreur dans le fichier de configuration. Pas de panique ! regarder plutôt dans les endroits suivants si un message d'erreur ne fournit pas une piste.

```
tail /var/log/syslog
tail /var/log/mysql.log
tail /var/log/mysql.err
```

Pour faciliter les opérations sur la base de données faite par `root` nous allons enregistrer le mot de passe MySQL comme suit :

```
nano /root/.my.cnf
```

Taper le contenu suivant, en indiquant le mot de passe, en clair, du compte `root` de MySQL :

```
[client]
user=root
password="mot_de_passe"
```

Enregistrer les modifications et quitter l'éditeur.

Pour être sûr de limiter l'accès à ce fichier, nous changeons les permissions comme suit :

```
chmod 600 /root/.my.cnf
```

Tester l'accès au serveur MySQL en tapant simplement :

```
mysql
```

Si vous obtenez le prompt de MySQL à l'écran sans avoir à taper le mot de passe c'est que la configuration est correcte. Pour sortir de MySQL, et revenir au prompt système, tapez :

```
exit;
```

Nous copions ce fichier dans le répertoire de `remote` pour que ce compte bénéficie aussi de l'accès direct à la commande `mysql` :

```
cp /root/.my.cnf /home/remote
```

L'installation de MySQL est à présent terminée.

Pour avoir un rapide check-up de votre serveur à tout moment vous pouvez utiliser l'un des outils fournis avec MySQL 5.0 :

```
mysqlreport
```

Pour lire le rapport fourni par `mysqlreport` vous pourrez vous référer à la documentation officielle, en anglais, à l'adresse suivante :

```
http://hackmysql.com/mysqlreportguide
```

Note : si vous perdez le mot de passe MySQL du compte `root`, il existe une procédure pour s'en sortir, comme indiqué à l'adresse suivante :

```
http://dev.mysql.com/doc/refman/5.1/en/resetting-permissions.html
```

3.12 Installer phpMyAdmin

Installer l'outil grâce aux paquets Debian :

```
aptitude install phpmyadmin
```

Si l'outil d'installation demande avec quelle version d'Apache il doit s'intégrer, sélectionner `apache2`. Sinon, réaliser l'opération manuellement comme suit :

```
cd /etc/apache2/conf.d
ln -fs /etc/phpmyadmin/apache.conf phpmyadmin.conf
/etc/init.d/apache2 reload
```

Comme nous avons spécifié un groupe d'exécution particulier pour ce serveur virtuel, et puisque nous l'utiliserons aussi pour les suivants, nous devons nous assurer des permissions sur les fichiers de configuration de phpMyAdmin :

```
cd /etc/phpmyadmin
chmod o+r config-db.php
```

Pour tester la configuration, faire pointer un navigateur sur l'adresse IP du serveur vers le chemin `/phpmyadmin` et visualiser l'écran d'authentification de phpMyAdmin. En cas d'erreur, regarder dans `/var/log/apache2` les différents fichiers de logs.

L'installation est mutualisée entre tous les serveurs virtuels, chaque webmestre pourra donc ajouter `/phpmyadmin` à l'adresse de son serveur virtuel pour gérer en direct sa base de données, faire ses sauvegardes, etc. Bien sûr, chacun ne verra que « sa » base à travers l'outil, en fonction du nom et du mot de passe fournis dans le formulaire d'entrée dans phpMyAdmin.

Si vous avez installé le chiffrement des communications web, vous pourrez aussi vérifier la méthode d'accès `https` : pour être sûr de disposer d'une méthode sécurisée de manipulation des données à distance. Quel bonheur de gérer la base de données par Internet, sans craindre que les échanges soient espionnés par un tiers !

3.13 Installer proftpd

Note : l'accès par FTP n'est pas nécessaire si vous êtes seul à effectuer des mises à jour de fichier. Vous pouvez transférer des fichiers à partir d'un compte SSH tel que `remote` en cas de besoin.

Pour installer un serveur FTP, télécharger le paquet Debian `proftpd`, comme suit :

```
aptitude install proftpd
```

Sélectionner l'option 'standalone' lorsque le programme d'installation demande de choisir entre 'inetd' ou 'standalone'.

Reprendre la configuration du serveur :

```
nano /etc/proftpd/proftpd.conf
```

Vérifier les paramètres suivants :

```
# résoudre les problèmes d'adressage éventuels
UseIPv6 off

# empêcher la recherche d'identité
IdentLookups off

# empêcher la recherche de noms (ligne à insérer)
UseReverseDNS off

# nom du serveur affiché aux usagers
ServerName "Serveur FTP"

# chrooter les utilisateurs
DefaultRoot ~

# limiter le nombre d'usagers simultanés (ligne à insérer)
Maxclients 30

# limiter les connexions (ligne à insérer)
MaxClientsPerHost 50
```

Vérifier que la ligne suivante n'est pas commentée par un # intempestif, ce qui empêcherait la sécurisation des accès :

```
Include /etc/proftpd/tls.conf
```

Enregistrer les modifications et quitter l'éditeur.

3.14 Sécuriser les échanges FTP

Passer au chapitre suivant si vous n'avez pas besoin de chiffrer les mots de passe des comptes FTP, ni les fichiers échangés avec le serveur.

L'accès FTP est sécurisé par TLS, pour protéger les mots de passe et les fichiers échangés avec le serveur. Ce mode de transmission est très efficace, et accepté par la plupart des logiciels FTP dignes d'intérêt, tel que Filezilla.

La première étape est de créer un certificat et une clé privée pour le serveur, en utilisant le nom réel de la machine, ou son adresse réseau, à la place de `nom_de_serveur` ci-dessous :

```
openssl req -new -x509 -days 3650 -nodes \  
-out /etc/ssl/certs/nom_de_serveur.pem \  
-keyout /etc/ssl/private/nom_de_serveur.pem
```

Attention : la commande de génération des clés ci-dessus doit être tapée en une seule ligne, sans les signes \ qui, ici, représentent une continuité avec la ligne suivante.

Pendant la création du certificat, indiquez au moins les deux lettres du pays où se trouve le serveur (FR pour la France) et le nom réel de la machine (par exemple : `ftp.domaine.fr`), ou son adresse réseau (par exemple : `123.45.67.89`), comme Common Name.

Note : pour favoriser la ré-utilisation des certificats requis pour TLS d'un serveur virtuel à l'autre, les certificats sont regroupés dans `/etc/ssl/certs` et les clés privées dans `/etc/ssl/private`. Les fichiers sont nommés d'après le champ Common Name. Par exemple, le certificat SSL de `ftp.domaine.fr` sera placé dans `/etc/ssl/certs/ftp.domaine.fr.pem`.

Modifier le fichier de configuration TLS :

```
nano /etc/proftpd/tls.conf
```

Positionner le module TLS comme suit, en remplaçant `nom_de_serveur` par sa valeur réelle :

```
<IfModule mod_tls.c>  
    TLSEngine on  
    TLSLog /var/log/proftpd/tls.log  
    TLSProtocol SSLv23  
    TLSRSACertificateFile /etc/ssl/certs/nom_de_serveur.pem
```

```
TLRSACertificateKeyFile /etc/ssl/private/nom_de_serveur.pem
TLSOptions NoCertRequest
TLSVerifyClient off
TLSRequired off
TLSRenegotiate required off

</IfModule>
```

Enregistrer les modifications et quitter l'éditeur.

Relancer le serveur pour prendre en compte les modifications :

```
/etc/init.d/proftpd restart
```

Avec ces paramètres la machine acceptera aussi bien les sessions en clair que les sessions chiffrées. C'est donc à la personne qui se connecte de sélectionner le mode de communication adapté.

Utiliser un client FTP/TLS tel que Filezilla pour valider la configuration. Attention de choisir l'option d'accès FTPES « FTP over explicit TLS/SSL » pour bénéficier des communications chiffrées. Présenter le nom `remote` et le mot de passe associé, puis vérifier que vous pouvez accéder au répertoire distant, et téléverser un fichier.

En cas de problème, regardez les enregistrements dans le répertoire `/var/log/proftpd`.

Si la communication ne passe pas, essayez de reprendre les échanges en utilisant le protocole FTP ordinaire. Ceci sera nécessaire par exemple si un pare-feu, placé entre l'ordinateur distant et le serveur, interdit l'établissement du flux de données lorsque la transmission des commandes est chiffrée.

3.15 Installer Postfix

Dans l'absolu, `yacs` n'a pas besoin de Postfix pour envoyer et recevoir des courriers électroniques. En effet, vous pouvez indiquer dans le panneau de configuration système le nom du serveur de courrier à utiliser, et `yacs` s'adressera alors directement à cette machine, à travers le réseau.

Toutefois, il est fortement conseillé d'installer Postfix, pour permettre à l'ensemble du système d'envoyer des messages. Ce composant sert par exemple à recevoir dans votre boîte aux lettres le résultat des traitements d'arrière-plan, ou les messages d'alertes que vous pourriez configurer pour surveiller le fonctionnement de la machine.

Si votre serveur s'adresse à un serveur SMTP connu, dont vous avez l'adresse ainsi qu'un compte d'authentification valide, alors vous pouvez configurer Postfix pour envoyer les messages électroniques :

```
aptitude install postfix libsasl2-modules
```

Il est possible que l'installateur propose d'enlever `exim`, qui est le service de messagerie par défaut. Répondez par l'affirmative à toutes les questions.

Pendant l'installation, choisir le mode 'Satellite system'. Indiquer le nom du système relais si vous le connaissez.

Il reste maintenant à indiquer les paramètres de connexion au serveur SMTP. Chez OVH, remplacer `smtp.exemple.com` par `ns0.ovh.net` :

```
postconf -e 'relayhost = smtp.exemple.com'
postconf -e 'smtp_sasl_auth_enable = yes'
postconf -e 'smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd'
postconf -e 'smtp_sasl_security_options = noanonymous'
```

Le nom et le mot de passe à utiliser pour la connexion sont placés dans un fichier séparé.

```
echo "smtp.exemple.com nom:mot_de_passe" > /etc/postfix/sasl_passwd
```

Note : chez certains hébergeur, vous devrez indiquer comme nom l'intégralité de l'adresse, en remplaçant le caractère @ par %. Par exemple, pour le compte `yacs@yacs.fr` chez OVH :

```
echo "ns0.ovh.net yacs%yacs.fr:mot_de_passe" > /etc/postfix/sasl_passwd
```

Personne d'autre que `root` ne devrait avoir accès à ce fichier :

```
chown root:root /etc/postfix/sasl_passwd
chmod 600 /etc/postfix/sasl_passwd
```

Il faut aussi convertir le fichier des mots de passe dans un format directement utilisable par postfix :

```
postmap /etc/postfix/sasl_passwd
```

Relançons postfix pour prendre en compte cette nouvelle configuration :

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

Il nous reste seulement à vérifier l'envoi de message en ligne de commande, en remplaçant `nom@domaine` par une adresse valide dans l'exemple suivant :

```
echo "Bonjour le monde" | mail nom@domaine
```

Si cela ne marche pas, regarder les logs `/var/log/mail.err` ainsi que `/var/log/mail.info`.

Après avoir configuré correctement l'envoi de courrier électronique, vous pourrez indiquer où envoyer les messages envoyés par les traitements d'arrière-plan :

```
nano /etc/crontab
```

Une solution évidente est de ne pas transmettre du tout les messages générés par les traitements d'arrière-plan. Dans ce cas, utilisez la directive :

```
MAILTO=""
```

Ceci permettra au moins d'éviter de générer des fichiers dans le répertoire `/var/mail`, ce qui crée un risque d'encombrement progressif de l'espace disque.

Ou alors, si vous préférez lire le résultat de chaque traitement d'arrière-plan, indiquez l'adresse de courrier électronique de destination :

```
MAILTO=cron@supervision.fr
```

Cette adresse recevra automatiquement le résultat des commandes lancées par cron. Dans la suite de ce guide d'installation, nous indiquerons des adresses différentes pour les traitements spécifiques à chaque serveur virtuel, afin de dispatcher les messages vers les différents points de contact si besoin.

3.16 Préparer les sauvegardes automatiques

La sauvegarde d'un serveur yacs concerne à la fois les fichiers et le contenu de la base de données. Nous allons mettre en place une stratégie de duplication simple, comme suit :

- Un compte `snapshot` est créé pour héberger, dans son répertoire, l'ensemble des copies des serveurs.
- Les fichiers d'un serveur virtuel, et le contenu de la base de données associée, sont archivés à intervalle réguliers dans un sous-répertoire de `/home/snapshot`.

La sauvegarde elle-même pourra prendre deux formes au choix :

- synchrone - Le contenu de `/home/snapshot` est synchronisé périodiquement vers une machine distincte par le protocole `rsync`.
- asynchrone - Ou alors, il est créé une archive unique pour chaque serveur virtuel, et cette archive est récupérée périodiquement par FTP.

L'une ou l'autre des deux options sera mise en place pendant l'installation d'un serveur virtuel, comme nous le verrons ultérieurement. Pour l'instant, créons simplement un compte pour la centralisation des données de sauvegarde, en choisissant un mot de passe bien sécurisé :

```
adduser snapshot
```

Ce compte sera utilisé dans les scripts pour archiver le contenu des bases de données dans le répertoire `/home/snapshot`. Cette opération est dangereuse, parce qu'elle risque d'exposer le mot de passe du compte `root` de MySQL dans les scripts mis en oeuvre. Pour éviter cela, nous allons centraliser l'accès à ce mot de passe dans un seul fichier, comme suit :

```
cp /root/.my.cnf /home/snapshot
chown snapshot /home/snapshot/.my.cnf
```

Pour tester le système, il faut invoquer le client MySQL dans une session du compte `snapshot` :

```
su - snapshot
mysql
```

Le prompt `mysql>` doit apparaître à l'écran, prouvant l'acceptation du login. Nous pouvons stopper l'exécution du client MySQL, puis la session de `snapshot`.

```
exit;
exit
```

Cette sécurisation permettra d'invoquer `mysqldump` dans les scripts de sauvegarde, sans avoir à indiquer le mot de passe correspondant. Ceci ne fonctionnera que pour les scripts invoqués par le compte `snapshot`, et c'est ce compte qui devra être appelé dans la crontab.

Vérifions également que le paquet de synchronisation des fichiers est installé :

```
aptitude install rsync
```

La suite de ce chapitre explique comment téléverser les fichiers sur un serveur de sauvegarde externe, par le protocole `rsync` sécurisé sur SSH. Si vous ne prévoyez pas de mettre en oeuvre ce genre de choses, vous pouvez passer directement au chapitre suivant.

Sinon, nous allons, en plusieurs étapes, installer un tunnel de synchronisation entre votre machine et une autre. Le serveur de sauvegarde recevra les fichiers dupliqués et vous aurez besoin, pour la configuration, de pouvoir y accéder en SSH.

La première étape est de préparer une clé de sécurisation des échanges SSH pour remplacer la demande interactive du mot de passe lors de chaque établissement de session. Pour cela, nous allons créer une clé privée et une clé publique dans le répertoire du compte `snapshot`.

```
su - snapshot
mkdir -p .ssh
ssh-keygen -t rsa
exit
```

Lorsque le programme demande un nom de fichier ou un mot de passe, appuyer simplement sur la touche <Entrée>.

La clé publique `id_rsa.pub` est prête à être poussée vers un serveur de sauvegarde. Dans les lignes qui suivent, on a nommé `remote` le compte utilisé sur le `serveur_de_sauvegarde`, mais vous devez utiliser les vraies valeurs.

```
scp .ssh/id_rsa.pub remote@serveur_de_sauvegarde:/home/remote/
```

Ensuite, on ouvre une session SSH sur le serveur de sauvegarde pour installer la clé publique de connexion :

```
ssh remote@serveur_de_sauvegarde
cd /home/remote
mkdir -p .ssh
cd .ssh
cat ../id_rsa.pub >>authorized_keys
rm ../id_rsa.pub
chmod 600 authorized_keys
```

Pour vérifier que tout fonctionne, il faut casser la connexion SSH en cours vers le serveur de sauvegarde, pour en rouvrir une autre en utilisant la clé publique. Tapez successivement :

```
exit
ssh -i /home/snapshot/.ssh/id_rsa remote@serveur_de_sauvegarde
```

Normalement, on ne doit pas recevoir le prompt de demande de mot de passe, mais plutôt un superbe prompt de bienvenue.

Ce message est un test positif, puisqu'il indique que l'appariement des clés est correct, que le script de validation a bien été activé, et qu'il a fait son travail de blocage comme prévu.

Si un prompt apparaît pour demander un mot de passe, c'est qu'il y a un problème quelque part. Il faut se connecter par SSH au serveur de sauvegarde et regarder dans `/var/log/auth.log`. On peut aussi vérifier, toujours sur le serveur de sauvegarde, que la clé n'a pas été considérée comme trop vulnérable :

```
ssh-vulnkey /home/remote/.ssh/authorized_keys
```

La mise en place du script de sauvegarde sera faite avec la mise en place du premier serveur virtuel.

3.17 Installation de xdebug

Attention, cette opération concerne seulement les machines de développement ou d'intégration. Il ne faut pas installer ce composant sur un serveur de production.

```
aptitude install php5-xdebug
```

Lors de l'installation un fichier `/etc/php5/conf.d/xdebug.ini` est créé, avec une seule ligne pour charger la librairie additionnelle :

```
zend_extension=/usr/lib/php5/20060613+lfs/xdebug.so
```

Nous pourrions modifier ce fichier pour activer le profiling, ou le debugging, mais la configuration s'appliquerait alors à tous les serveurs virtuels. Nous verrons, dans la prochaine section de ce guide, comment activer xdebug sur un seul serveur virtuel à la fois.

3.18 Quelques vérifications finales

Pour mieux protéger votre serveur des espions éventuels il convient de supprimer le petit script de validation de la configuration PHP :

```
rm /var/www/default/test.php
```

Regardons du côté de l'espace disque comment les choses se présentent :

```
df -h
```

Pour connaître la taille occupée par chaque serveur virtuel sur le disque, vous pouvez taper:

```
cd /var/www
du -s *
```

Et pour savoir la taille totale d'un répertoire, c'est encore plus simple :

```
du -s /home/mysql
```

Pour surveiller les processus mémoire, on peut utiliser `top`, en appuyant sur la touche `q` pour en sortir :

```
top
```

La même chose existe pour les échanges réseau, à condition d'installer un paquet supplémentaire :

```
aptitude install iftop
iftop
```

Encore plus fort : on peut également suivre le fonctionnement d'un serveur Apache en quasi-temps réel, avec l'outil `apachetop` :

```
aptitude install apachetop
apachetop -f /var/log/apache2/access.log
```

Nous ajoutons dans le répertoire du compte `remote` une commande qui permettra de suivre le fonctionnement du serveur virtuel en temps réel :

```
cd /home/remote
nano apachetop
```

Il suffit de faire pointer la commande `apachetop` vers le bon fichier de log comme suit :

```
#!/bin/sh
sudo apachetop -f /var/log/apache2/access.log
```

Enregistrer les modifications et quitter l'éditeur, puis indiquer au système que ce fichier est une commande exécutable :

```
chmod a+x apachetop
```

Pour gagner un peu de place disque on peut supprimer les copies des paquets installés :

```
aptitude clean
```

Pour vérifier que la liste des ports ouverts aux communications extérieures est conforme à notre attente, et limitée au strict minimum, taper :

```
netstat --inet --listening
```

Si la machine écoute sur les ports portmapper, et que vous n'avez pas besoin de partager de disque en réseau, alors tenter de supprimer NFS:

```
aptitude remove nfs-common portmap
```

A un niveau encore plus bas, taper la commande suivante pour repérer d'éventuelles erreurs de transmission réseau qui auraient pu se produire pendant l'installation:

```
ifconfig
```

Si la commande n'indique aucune erreur, c'est que la machine est saine et que nous pouvons passer à l'étape suivante, c'est-à-dire l'installation d'un premier serveur virtuel yacs.

4 Installation d'un site virtuel

Dans cette section nous allons ajouter un serveur virtuel à l'environnement préparé précédemment. La même séquence est, en fait, utilisée pour chaque nouveau serveur virtuel.

Pour chaque nouveau domaine installé sur la machine :

- un compte utilisateur sera créé pour l'accès FTP à distance (et aussi pour exécuter les scripts qui constituent yacs), nous le désignerons par le raccourci `domaine` dans cette section
- un serveur virtuel sera créé, et géré indépendamment des autres (fichiers de configuration, répertoire, base de données), nous le désignerons par `www.domaine.fr` dans les exemples

4.1 Gestion du nom de domaine

Le DNS n'est pas traité dans le cadre du présent guide. Pour la gestion de noms publics, vous utiliserez vraisemblablement la console fournie par votre fournisseur de services (OVH, ...).

Le principe est simple : vous devez associer chaque site virtuel à l'adresse réseau de la machine que nous sommes en train d'installer.

4.2 Ajout d'un utilisateur

Ce compte sera celui utilisé pour exécuter les scripts de yacs, et aussi pour accéder aux fichiers à distance. Remplacer `domaine` par sa vraie valeur.

```
adduser domaine
```

Le mot de passe est celui qui doit être utilisé pour se connecter par FTP/TLS.

4.3 Installer le site virtuel

Créer le répertoire du site virtuel, en reprenant le nom complet du site. On ajoute le groupe `www-data` pour permettre aux outils associés à Apache d'y intervenir si besoin :

```
cd /home/domaine
mkdir www
chown domaine:www-data www
cd /var/www
ln -s /home/domaine/www www.domaine.fr
cd www.domaine.fr
```

Construire un script PHP minimum comme suit :

```
echo '<?php phpinfo(); ?>' >/var/www/www.domaine.fr/test.php  
chown domaine:www-data /var/www/www.domaine.fr/test.php
```

Indiquer le nouveau site à Apache :

```
nano /etc/apache2/sites-available/www.domaine.fr
```

S'inspirer du texte suivant pour configurer le site virtuel, en remplaçant `www.domaine.fr` et les autres paramètres par leurs vraies valeurs :

```
<VirtualHost *:80>  
    AssignUserID domaine www-data  
    ServerName www.domaine.fr  
    ServerAlias domaine.fr  
    ServerAdmin contact@domaine.fr  
    DocumentRoot /var/www/www.domaine.fr  
    <Directory />  
        Options FollowSymlinks  
        AllowOverride None  
    </Directory>  
    <Directory /var/www/www.domaine.fr>  
        Options FollowSymlinks  
        AllowOverride FileInfo Indexes Options  
        Order allow,deny  
        allow from all  
    </Directory>  
    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/  
    <Directory "/usr/lib/cgi-bin">  
        AllowOverride None  
        Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch  
        Order allow,deny  
        Allow from all  
    </Directory>  
    ErrorLog /var/log/apache2/error-www.domaine.fr.log  
    LogLevel warn  
    CustomLog /var/log/apache2/access-www.domaine.fr.log combined  
</VirtualHost>
```

Enregistrer les modifications et quitter l'éditeur.

Activer le site comme suit :

```
a2ensite www.domaine.fr
/etc/init.d/apache2 reload
```

Dans un navigateur, faire pointer sur l'adresse ip du serveur vers le script `/test.php` et vérifier que PHP est déclenché, sans provoquer d'erreur interne.

Note : il est normal d'avoir un message d'erreur si l'on pointe juste à la racine du serveur (sans indiquer `/test.php`), puisque nous n'avons pas encore créé d'index à ce stade. Veillez donc à bien ajouter le nom du script à exécuter.

Nous ajoutons dans le répertoire du compte `remote` une commande qui permettra de suivre le fonctionnement du serveur virtuel en temps réel :

```
cd /home/remote
nano apachetop-www.domaine.fr
```

Il suffit de faire pointer la commande `apachetop` vers le bon fichier de log comme suit :

```
#!/bin/sh
sudo apachetop -f /var/log/apache2/access-www.domaine.fr.log
```

Enregistrer les modifications et quitter l'éditeur, puis indiquer au système que ce fichier est une commande exécutable :

```
chmod a+x apachetop-www.domaine.fr
```

4.4 Créer un site virtuel sécurisé

Vous n'avez pas besoin de ce chapitre si vous ne souhaitez pas sécuriser les échanges avec le serveur web.

La sécurisation s'appuie sur un certificat que l'on pourra acheter à un prestataire spécialisé ou, pour des besoins de tests, que l'on pourra fabriquer soi-même, en utilisant le nom virtuel de la machine à la place de `www.domaine.fr` ci-dessous :

```
openssl req -new -x509 -days 3650 -nodes \
  -out /etc/ssl/certs/www.domaine.fr.pem \
  -keyout /etc/ssl/private/www.domaine.fr.pem
```

Attention : la commande de génération des clés ci-dessus doit être tapée en une seule ligne, sans les signes `\` qui, ici, représentent une continuité avec la ligne suivante.

S'il y a besoin de sécuriser les accès au serveur web par TLS, on pourra ajouter un deuxième fichier

de configuration dérivé de default-ssl :

```
cd /etc/apache2/sites-available
cp default-ssl www.domaine.fr-ssl
nano www.domaine.fr-ssl
```

Voici le contenu type du fichier de configuration d'un serveur virtuel sécurisé :

```
<IfModule mod_gnutls.c>
<VirtualHost *:443>
    AssignUserID domaine www-data
    ServerName www.domaine.fr
    ServerAlias domaine.fr
    ServerAdmin contact@domaine.fr
    DocumentRoot /var/www/www.domaine.fr
    <Directory />
        Options FollowSymlinks
        AllowOverride None
    </Directory>
    <Directory /var/www/www.domaine.fr>
        Options FollowSymlinks
        AllowOverride FileInfo Indexes Options
        Order allow,deny
        allow from all
    </Directory>
    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
    <Directory "/usr/lib/cgi-bin">
        AllowOverride None
        Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
        Order allow,deny
        Allow from all
    </Directory>
    ErrorLog /var/log/apache2/error-www.domaine.fr.log
    LogLevel warn
    CustomLog /var/log/apache2/access-www.domaine.fr.log combined

    GnuTLSEnable on
    GnuTLSCertificateFile /etc/ssl/certs/www.domaine.fr.pem
    GnuTLSKeyFile /etc/ssl/private/www.domaine.fr.pem
    GnuTLSPriorities NORMAL
```

```
</VirtualHost>
</IfModule>
```

Grâce à votre œil exercé vous aurez remarqué que nous utilisons les mêmes fichiers de log que pour l'accès non sécurisé. Ainsi, l'analyse des erreurs, ou les statistiques sur les pages consultées, seront globales et indépendantes du type d'accès.

Enregistrer les modifications et quitter l'éditeur.

Activer le site comme suit :

```
a2ensite www.domaine.fr-ssl
/etc/init.d/apache2 reload
```

Vérifions l'accès sécurisé en faisant pointer le navigateur vers l'adresse suivante :

```
https://www.domaine.fr/test.php
```

Le navigateur peut se plaindre du fait que le certificat est auto-signé. Accepter l'exception de sécurité pour valider l'accès. Normalement, cette opération ne doit être faite qu'une seule fois. Ensuite le navigateur affiche le message : « It works! ».

En cas de problème, regarder les messages d'erreur comme suit :

```
tail /var/log/apache2/error-www.domaine.fr.log
```

En environnement public, il sera bon d'acheter un certificat signé par une autorité de certification bien connue, et d'installer le certificat correspondant dans le répertoire `/etc/ssl/certs` avec les autres. Le fichier de configuration du serveur virtuel sera modifié pour pointer sur ce certificat en lieu et place de celui que nous avons créé nous-mêmes, et il faudra redémarrer Apache pour faire prendre en compte le changement. Les internautes pourront alors naviguer sur le service sécurisé sans recevoir de message d'avertissement du navigateur.

4.5 Créer une base de données

Yacs est capable de créer les tables par lui-même pendant l'installation. A ce stade, nous avons seulement besoin d'ajouter une base de données, et un compte utilisateur pour y accéder.

Dans l'exemple suivant, remplacez `www_domaine_fr` et `password` par les véritables valeurs. Veuillez noter que les points du nom de domaine sont remplacés par des caractères soulignés, c'est une contrainte qui vient de MySQL. Il est préférable de choisir une paire unique identifiant - mot de passe pour chaque base de données, pour des raisons de sécurité.

```
mysql
mysql> create database www_domaine_fr;
mysql> grant all on www_domaine_fr.* to 'domaine'@'localhost' identified by \
```

```
'password';  
mysql> exit;
```

Les paramètres à saisir dans le panneau de configuration de yacs seront donc les suivants :

Nom de la base de données	La valeur réelle de www_domaine_fr
Nom d'utilisateur	La valeur réelle de domaine
Mot de passe	La valeur saisie pour password

4.6 Télécharger yacs

Télécharger la version la plus récente de yacs, et extraire les fichiers de l'archive comme suit. Pour cela, se placer directement dans le répertoire du serveur virtuel, qui est aussi le répertoire d'installation de yacs :

```
cd /var/www/www.domaine.fr
```

Télécharger l'archive directement depuis le serveur de référence, en tapant la commande suivante sur une seule ligne :

```
wget http://www.yacs.fr/yacs.tgz
```

Si vous préférez bénéficier des évolutions les plus récentes, vous téléchargerez la version en cours de test, comme suit :

```
wget http://www.yacs.fr/yacs.testing.tgz
```

Extraire les fichiers de l'archive comme ici, pour la version stable de yacs :

```
tar xvf yacs.tgz
```

Supprimer l'archive devenue inutile :

```
rm yacs.tgz
```

Donner les droits d'accès à chacun des fichiers. Encore une fois, nous permettons au groupe `www-data` d'intervenir à distance si besoin.

```
cd /home/domaine  
chown -R domaine:www-data www
```

Faire pointer le navigateur vers la racine du serveur virtuel pour lancer l'assistant d'installation.

4.7 Installer yacs

Faire pointer un navigateur vers `http://www.domaine.fr/` pour déclencher l'assistant d'installation, et passer à travers les différentes phases de la configuration :

- tests de l'environnement PHP
- test de l'accès à la base de données, et enregistrement du fichier de configuration système
- recherche des extensions de yacs, et enregistrement du fichier de configuration des extensions
- création des tables de la base de données
- enregistrement du fichier de configuration du rendu visuel

En fin de parcours, vérifier sur le site principal s'il faut installer un patch de sécurité, ou s'il faut suivre des directives de mise à jour particulières :

```
http://www.yacs.fr/
```

Naviguer vers le Panneau de Contrôle principal, puis cliquer sur l'onglet de configuration pour optimiser certains paramètres.

Lancer l'assistant de création du fichier `.htaccess` qui va d'abord évaluer les possibilités réelles du serveur Apache, puis préparer un fichier adapté à la situation. Le fichier est écrit à la racine du serveur virtuel, c'est-à-dire :

```
/var/www/www.domaine.fr/.htaccess
```

Dans le panneau de configuration système, changer le paramètre Génération des liens en choisissant la troisième option, celle qui permet la ré-écriture des liens grâce aux directives placées dans le fichier `.htaccess`.

Activer aussi la compression gzip pour réduire la taille des pages transmises, dont le fameux fichier `/included/browser/library.js` qui consolide la plupart des bibliothèques javascript utilisées par yacs.

Modifier aussi les paramètres relatifs à l'envoi de courrier électronique en changeant la limite des envois de 50 à 1000 par heure. Si vous n'avez pas configuré le système de base pour l'envoi de courrier par postfix, vous pourrez indiquer l'adresse du serveur SMTP et les paramètres d'authentification POP3.

Pensez également à demander et configurer les clés Google qui permettront d'intégrer les cartes de Google Maps, et d'analyser le fonctionnement de votre serveur. Les liens à utiliser pour faire vos demandes sont affichées dans le panneau de configuration des services web de yacs.

4.8 Migrer un site yacs

Si vous avez besoin de migrer un serveur existant, procédez à l'installation normale comme expliqué précédemment, puis transférez ensuite :

- le contenu de la base de données,
- les images attachées aux pages,
- les fichiers téléversés sur le site.

Pour la base de données, le plus simple est d'exporter l'ancienne base, par exemple avec le script fourni avec yacs, ou, pour les trop grosses bases, avec phpMyAdmin. Si vous utilisez phpMyAdmin, veillez à décocher les options « Complete inserts » et « Extended inserts », et choisissez la compression gzip.

L'importation sur le nouveau site pourra se faire en téléversant le fichier directement depuis le formulaire de restauration de yacs. Pour les bases très importantes, qui dépasserait les limites de transfert d'Apache, il est plus simple de téléverser le fichier par FTP puis de l'importer en ligne de commande SSH. Par exemple, si vous avez envoyé le fichier `sauvegarde.sql.gz` dans le répertoire racine du compte `domaine` :

```
cd /home/domaine
gunzip sauvegarde.sql.gz
mysql -D www_domaine_fr <sauvegarde.sql
```

La restauration peut durer de longues minutes, ne vous inquiétez pas de ne pas revenir au prompt tout de suite.

Pour les images et les fichiers, le mieux est de récupérer l'ensemble de l'arborescence yacs par FTP. Dans un deuxième temps, connectez-vous vers le nouveau site par FTP, et téléverser les sous-répertoires de `/files` et de `/images`.

Vous pouvez aussi copier de la même façon le contenu du répertoire `/parameters`, à l'exception de `control.include.php` qui contient les paramètres d'accès à la base de données. Ce fichier ne devrait jamais être recopié à l'identique, sous peine de perte d'accès à la base de données. Lorsque j'ai le temps, je préfère re-créeer tous les panneaux de configuration un par un, plutôt que de recopier les anciens.

Et si vous devez rediriger toutes les pages d'un ancien site vers le nouveau, je vous conseille de remplacer le fichier `.htaccess` placé à la racine de votre ancien site, avec le contenu suivant :

```
<IfModule mod_rewrite.c>
    RewriteEngine on
    RewriteRule ^(.*)$ http://www.domaine.fr/$1 [R=301,L]
</IfModule>
```

Bien sûr, cette redirection sera installée seulement après toutes les vérifications d'usage du nouveau serveur, y compris les notifications par courrier électronique.

4.9 Installer les traitement d'arrière-plan

Lors de l'installation du premier serveur virtuel, préparer l'environnement d'exécution comme suit :

```
cd /etc/cron.d
cp /var/www/www.domaine.fr/tools/yacs_crontab www_domaine_fr
nano /etc/cron.d/www_domaine_fr
```

Note : attention de ne pas mettre de point dans les noms des fichiers placés dans le répertoire `/etc/cron.d`, ils ne seraient pas pris en compte dans les traitements d'arrière-plan. C'est pour cela que nous utilisons `www_domaine_fr`, avec des caractères de soulignement, exactement comme nous l'avons fait précédemment pour les bases de MySQL.

Si vous souhaitez envoyer le résultat des traitements d'arrière-plan à une adresse particulière, insérer en début de fichier une ligne comme suit :

```
MAILTO=cron@domaine.fr
```

Juste en-dessous, il faut modifier le fichier pour indiquer l'emplacement exact du script `cron.php` et le compte utilisateur correspondant.

```
09,19,29,39,49,59 * * * * domaine php /var/www/www.domaine.fr/cron.php
```

Enregistrer les modifications et quitter l'éditeur.

Vérifier, par acquis de conscience, que la commande placée dans le fichier cron est correcte, en l'exécutant à la main, sans oublier de remplacer `www.domaine.fr` par sa vraie valeur :

```
php /var/www/www.domaine.fr/cron.php
```

Dans le navigateur, ouvrir le panneau de configuration système pour signaler à yacs que les traitements en tâche de fond sont gérés par cron. Attendre dix minutes, et faire afficher les informations système d'arrière-plan pour vérifier que l'heure des derniers traitements est incrémentée régulièrement.

4.10 Gérer les logs de yacs

Yacs enregistre dans un fichier les événements liés à la vie du serveur. Ce fichier doit être purgé de temps à autre sous peine d'encombrement de l'espace disque.

Pour cela, on pourra ajouter un fichier de configuration à `logrotate`, l'outil Debian qui gère ce genre de fonction, en remplaçant `www.domaine.fr` par sa valeur :

```
nano /etc/logrotate.d/www.domaine.fr
```

Les directives suivantes sont explicites, se rapporter à la documentation de `logrotate` pour plus d'informations.

```
/var/www/www.domaine.fr/temporary/*.txt {  
    rotate 12  
    monthly  
    compress  
    delaycompress  
    missingok  
    notifempty  
    create 660 domaine www-data  
}
```

Enregistrer les directives et quitter l'éditeur. Le fichier sera pris en compte automatiquement tous les mois.

4.11 Traiter les logs d'Apache

La première chose à faire est de préparer un fichier de configuration pour `awstats` :

```
cd /etc/awstats  
cp awstats.conf awstats.www.domaine.fr.conf  
nano awstats.www.domaine.fr.conf
```

Nous indiquons dans le fichier l'emplacement de la log et son format, en changeant les paramètres suivants :

```
LogFile="/var/log/apache2/access-www.domaine.fr.log"  
LogType=W  
LogFormat = 1  
LogSeparator=" "  
SiteDomain="www.domaine.fr"  
DNSLookup=1
```

Attention de bien écrire `apache2` dans la valeur du paramètre `LogFile`, par défaut c'est `apache` tout seul et ça ne marche pas.

Enregistrer les modifications et quitter l'éditeur.

Pour vérifier la configuration, et pour initialiser la base gérée par `awstats` pour ce domaine, nous lançons en ligne de commande :

```
/usr/lib/cgi-bin/awstats.pl -config=www.domaine.fr -update
```

Faire pointer le navigateur vers le serveur virtuel, avec l'adresse `/cgi-bin/awstats.pl` pour vérifier l'accès aux données statistiques.

L'étape suivante est de lancer l'analyse des logs chaque jour, vers 3 heures du matin. Pour cela, nous allons reprendre le fichier `crontab` pour ce domaine :

```
nano /etc/cron.d/www_domaine_fr
```

Rajouter deux lignes comme suit :

```
0 3 * * * root nice /usr/lib/cgi-bin/awstats.pl \  
-config=www.domaine.fr -update
```

Note : Le mot-clé `nice` est inséré pour réduire les priorités d'exécution des commandes d'analyse, pour éviter d'affecter les surfeurs pendant leur visite.

Enregistrer les modifications et quitter l'éditeur.

En résumé, à l'issue de cette étape, nous obtenons pour chaque serveur virtuel une adresse bien utile pour exploiter les logs collectées par Apache :

```
http://www.domaine.fr/cgi-bin/awstats.pl
```

4.12 Installer piwik

Piwik est un outil d'analyse des navigations sur un site, qui fournit des informations très précieuses aux webmestres, à la façon de Google Urchin.

Après avoir obtenu l'archive la plus récente de Piwik depuis le site web <http://piwik.org/> vous extrairez tous les fichiers sur votre machine. Ensuite copiez le répertoire `piwik` vers le répertoire d'installation de `yacs`, à travers FTP. Ce nouveau répertoire sera listé, côté serveur, juste entre `parameters` et `scripts`.

Basculez ensuite vers le navigateur web, et faites le pointer sur le répertoire piwik du serveur, comme suit :

```
http://www.domaine.fr/piwik
```

L'écran de bienvenue de l'installateur piwik est affiché, laissons-nous guider. Sur le panneau de configuration de la base de données, indiquer exactement les mêmes paramètres de connexion que ceux utilisés par yacs. Garder le préfixe `piwik_` pour distinguer les tables de celles utilisées par yacs. Choisir le connecteur `mysqli` pour gagner en rapidité.

Ne pas tenir compte du message d'avertissement relatif à la connexion UTF-8, le cas échéant, et continuer l'installation.

Piwik demande ensuite de créer un compte pour accéder et pour gérer les informations de navigation. On pourra saisir les mêmes informations (nom et mot de passe) que pour l'utilisateur du site yacs, ou d'autres si besoin. Indiquer une adresse de courrier électronique valide.

Sur le panneau suivant, vous indiquerez un nom court pour votre site, ainsi que son adresse web.

Le tag Javascript indiqué par Piwik doit être ajouté par yacs à toutes les pages créées dynamiquement. Pour cela, coller le texte indiqué par l'installateur de Piwik dans le panneau de configuration du rendu visuel, dans le champ « Bas de page ». Enregistrer la modification, puis faire afficher la source d'une page yacs pour vérifier que le code Javascript de Piwik y figure bien.

Pour traiter les logs une fois par jour plutôt qu'au fil des navigations sur votre site, vous devriez modifier le fichier cron comme suit :

```
nano /etc/cron.d/www_domaine_fr
```

Ajouter la ligne suivante, en remplaçant `www.domaine.fr` par le nom de votre serveur :

```
5 0 * * * root /var/www/www.domaine.fr/piwik/misc/cron/archive.sh >/dev/null
```

S'assurer que le script est bien exécutable, sinon il ne se passera pas grand-chose :

```
chmod a+x /var/www/www.domaine.fr/piwik/misc/cron/archive.sh
```

Faire prendre en compte, par Piwik, le fait que le traitement des logs est effectué en arrière-plan :

```
nano /var/www/www.domaine.fr/piwik/config/config.ini.php
```

Insérez les lignes suivantes, puis enregistrer les modifications.

```
[General]  
time_before_today_archive_considered_outdated = 3600  
enable_browser_archiving_triggering = false
```

Et voilà ! Revenir dans quelques jours pour admirer les superbes rapports fournis par l'outil.

4.13 Sauvegarder le serveur automatiquement

Puisque nous avons accès à tous les recoins de la machine, profitons-en pour automatiser la consolidation des données, et leur transmission éventuelle à une machine tierce.

Créons un script, dans le répertoire du compte `snapshot`, pour regrouper toutes les commandes nécessaires :

```
nano /home/snapshot/.snapshot
```

Note : en plaçant le caractère `.` au début du nom du script, on le rendra invisible lors des accès par FTP pour récupérer les sauvegardes.

Ce script n'aura pas la même chose à faire selon que vous disposiez d'un serveur de sauvegarde synchrone ou non. Deux méthodes peuvent être mises en oeuvre :

- asynchrone – le script prépare des archives sur le serveur, et vous les récupérez quand vous le pouvez par FTP/TLS, qui est sécurisé.
- synchrone – le script envoie directement les fichiers au serveur de sauvegarde par le protocole `rsync` sur une liaison sécurisée SSH

La méthode asynchrone est la plus souple, mais elle requière une vraie discipline. En cas de panne du disque dur du serveur, vous perdez tout, y compris les archives de sauvegarde. Il est donc très important d'aller récupérer ces archives à intervalles très régulier. Sous Unix, vous pourrez automatiser cette opération grâce aux commandes `wget` ou `curl`, et vous êtes invités à regarder les pages d'aide de votre système. Pour l'instant, intéressons-nous au contenu du script qui va créer les archives de sécurité :

```
#!/bin/sh

# sauver le contenu de chaque base dans un fichier
SNAPSHOT='/home/snapshot/sql'
mkdir -p $SNAPSHOT
cd $SNAPSHOT
DUMP='mysqldump -u root --opt'
$DUMP www_domaine_fr | gzip - >www_domaine_fr.sql.gz

# archiver chaque serveur virtuel
SNAPSHOT='/home/snapshot/www'
mkdir -p $SNAPSHOT
cd /var/www
tar czhf $SNAPSHOT/www.domaine.fr.tgz www.domaine.fr
```

Si une machine de sauvegarde est disponible en permanence pour recevoir des flux de sauvegarde par le protocole `rsync` alors vous pourrez utiliser plutôt le script suivant. La variable `serveur_de_sauvegarde` est le nom ou l'adresse du serveur qui recevra les données, tandis que `nom_de_machine` désigne le nom ou l'adresse de la machine sur laquelle on travaille :

```
#!/bin/sh

# sauver le contenu de chaque base dans un fichier
DUMP='mysqldump -u root --opt'
SNAPSHOT='/home/snapshot/sql'
mkdir -p $SNAPSHOT
$DUMP www_domaine_fr >$SNAPSHOT/www_domaine_fr.sql

# dupliquer les fichiers vers une machine tierce
CIBLE='remote@serveur_de_sauvegarde:/home/remote/nom_de_machine'
rsync -CaLvz --delete -e "ssh -i /home/snapshot/.ssh/id_rsa" \
    /var/www/ $CIBLE/www
rsync -CaLvz --delete -e "ssh -i /home/snapshot/.ssh/id_rsa" \
    $SNAPSHOT/ $CIBLE/sql
```

Enregistrer les modifications et quitter l'éditeur.

Le script sera marqué comme étant exécutable comme suit :

```
chmod a+x /home/snapshot/.snapshot
```

Si l'on a choisi d'envoyer les données à un serveur de sauvegarde, il faudra s'y connecter pour préparer les répertoires d'accueil des fichiers :

```
cd /home/remote
mkdir nom_de_machine
mkdir nom_de_machine/sql
mkdir nom_de_machine/www
chown -R remote:www-data nom_de_machine
```

Que l'on ait choisi de dupliquer localement, ou d'exporter les données à travers le réseau, on pourra alors tenter une toute première opération de synchronisation manuelle. Il faut passer temporairement par une session de `snapshot` pour accéder à la base de données :

```
su - snapshot
./snapshot
exit
```

Si vous avez mis en oeuvre la méthode de sauvegarde synchrone alors la quantité d'informations échangées est indiquée pendant l'exécution du script. On pourra lancer la commande une deuxième fois pour constater l'efficacité du protocole `rsync` sur les mises à jour incrémentales.

Il nous reste à intégrer la commande aux traitements d'arrière-plan :

```
nano /etc/cron.d/snapshot
```

Pour synchroniser chaque jour à 5 heures du matin, indiquer le contenu suivant :

```
0 5 * * * snapshot nice /home/snapshot/.snapshot
```

Le mot-clé `nice` est inséré pour réduire la priorité d'exécution de la commande, et donc pour éviter d'affecter les traitements ordinaires de la machine. Après tout, les sauvegardes doivent rester aussi invisibles que possible des internautes.

Enregistrer les modifications et quitter l'éditeur.

Vérifier le contenu du répertoire `/home/snapshot` le lendemain, pour s'assurer que le contenu de la base de données a été placé dans un fichier compressé.

Si la commande `tar` a été activée, on pourra automatiser la récupération des archives en se connectant par FTPS sur le compte `snapshot`, par exemple avec la commande `curl` comme suit :

```
curl --ftp-ssl -k -u snapshot:mot_de_passe \  
ftp://www.domaine.fr/sql/www.domaine.fr.sql.gz -O\  
ftp://www.domaine.fr/www/www.domaine.fr.tgz -O
```

Si c'est `rsync` qui a été mis en oeuvre, on vérifiera sur le serveur de sauvegarde que le transfert s'est effectué comme prévu.

Il est possible de construire sur le serveur de sauvegarde une sorte de miroir du système sauvegardé, soit pour effectuer des tests, soit pour restaurer une machine en état de fonctionnement s'il arrivait un désastre. Si vous n'êtes pas intéressé par la construction d'un serveur de secours, vous pouvez économiser du temps de lecture et aller directement au chapitre suivant.

Toujours là ? C'est donc que vous avez hâte d'apprendre à configurer un site miroir. L'installation s'effectue en deux temps. Tout d'abord, vous devrez installer un site virtuel « normal », comme nous l'avons déjà expliqué quelques chapitres plus tôt (création d'un compte, d'une base de données, d'une instance de yacs). Ensuite, vous préparerez un script pour alimenter ce site virtuel à partir des données de sauvegarde, comme suit :

```
nano /home/remote/reload-www.domaine.fr
```

Le contenu de ce script copie à la fois les fichiers et les données, mais pas le logiciel ni les paramètres du site virtuel :

```
#!/bin/sh

# emplacement de la sauvegarde
BACKUP=/home/remote/nom_de_machine
RHOST=www.domaine.fr
RDATABASE=www_domaine_fr

# noms locaux
LUSER=domaine
LHOST=www.domaine.fr
LDATABASE=www_domaine_fr

# copie des fichiers
rsync -CaLvz --delete $BACKUP/www/$RHOST/files /var/www/$LHOST
rsync -CaLvz --delete $BACKUP/www/$RHOST/images /var/www/$LHOST
rsync -CaLvz --delete $BACKUP/www/$RHOST/skins /var/www/$LHOST
sudo chown -R $LUSER:www-data /var/www/$LHOST/

# rechargement de la base
sudo mysql -D $LDATABASE <$BACKUP/sql/$RDATABASE.sql
```

Enregistrer les modifications et quitter l'éditeur.

Le script sera marqué comme étant exécutable comme suit :

```
chmod a+x /home/remote/reload-www.domaine.fr
```

Il y a plusieurs façons d'utiliser ce script :

- Sur une machine d'intégration ou d'assurance-qualité, on rechargera le site virtuel avant d'entamer une campagne de modifications ou de tests.
- Sur une machine de secours, on rechargera le site virtuel en cas de sinistre sur le serveur principal. Puis on redirigera le nom de domaine `www.domaine.fr` vers l'adresse réseau du serveur de secours, en gardant à l'esprit que la propagation DNS peut s'étaler sur plusieurs heures.
- Sur une machine de démonstration, on rechargera le site automatiquement, à intervalles réguliers, pour purger le système des modifications qui auraient pu être faites par des tiers.

Pour une restauration à la demande, le compte `remote` se connecte en SSH et tape la commande `./reload-www.domaine.fr` pour ré-initialiser le serveur virtuel cible.

Pour une restauration automatique, on modifiera le fichier crontab du site virtuel cible :

```
nano /etc/cron.d/www_domaine_fr
```

Insérer la ligne suivante pour recharger le site virtuel chaque jour à 5 heures du matin :

```
0 5 * * * root nice /home/remote/reload-www.domaine.fr
```

4.14 Forcer les accès sécurisés

Dans le cas d'un serveur à protéger entièrement, vous devez vous assurer que toutes les requêtes sont chiffrées par TLS. Pour cela, activez le panneau de configuration système, et choisissez l'option de redirection des requêtes non sécurisées.

A partir de cet instant, yacs redirigera poliment toutes les demandes reçues par HTTP vers leurs homologues HTTPS. Cette opération sera indolore pour la plupart des internautes, et vous n'avez pas besoin de mentionner `https:` lorsque vous partagez l'adresse du serveur ou même d'une page particulière.

Un autre avantage du chiffrement systématique des transmissions est que vous pouvez mettre en place les cookies d'authentification permanente sans risque qu'ils soient interceptés par des personnes mal intentionnées. C'est sur le panneau de configuration des utilisateurs, à l'onglet Authentification. L'identification « longue durée », c'est un réel confort supplémentaire pour les membres d'une communauté en ligne, parce qu'elle évite de saisir le mot de passe à chaque visite du site.

4.15 Configurer xdebug

Dans certains cas nous pouvons être amené à faire des tests particuliers pour une instance virtuelle, indépendamment des autres. Par exemple, nous pourrions décider de profiler les scripts du serveur virtuel `www.domaine.fr` dans des fichiers au format `cachegrind` dans le répertoire `/var/www/www.domaine.fr/xdebug`.

Pour cela, il faut modifier les paramètres utilisés par PHP, mais seulement pour un seul serveur virtuel. La solution la plus simple est d'ajouter quelques directives `php_flag` dans le fichier de configuration correspondant, comme suit :

```
nano /etc/apache2/sites-available/www.domaine.fr
```

Insérons quelques lignes pour indiquer ses paramètres de fonctionnement à xdebug, pour obtenir un fichier qui ressemble à ceci :

```
<VirtualHost *:80>
    ServerName www.domaine.fr
    ServerAlias domaine.fr
    ServerAdmin contact@domaine.fr
```

```

DocumentRoot /var/www/www.domaine.fr
<Directory />
    Options None
    AllowOverride None
</Directory>
<Directory /var/www/www.domaine.fr>
    Options FollowSymlinks
    AllowOverride FileInfo Indexes Options
    Order allow,deny
    allow from all
    php_value xdebug.profiler_enable 1
    php_value \
        xdebug.profiler_output_dir /var/www/www.domaine.fr/xdebug
    php_value xdebug.profiler_append On
    php_value xdebug.profiler_output_name cachegrind.out.%s
</Directory>
ErrorLog /var/log/apache2/error-www.domaine.fr.log
LogLevel warn
CustomLog /var/log/apache2/access-www.domaine.fr.log combined
</VirtualHost>

```

Enregistrer les modifications et quitter l'éditeur.

N'oublions pas de préparer le répertoire qui accueillera les fichiers créés par xdebug :

```

mkdir /var/www/www.domaine.fr/xdebug
chown domaine:www-data /var/www/www.domaine.fr/xdebug

```

Relancer Apache pour être sûr qu'il n'y a pas d'erreur dans les fichiers de configuration, et pour faire prendre en compte les modifications :

```

/etc/init.d/apache2 restart

```

Pour tester le fonctionnement, faire pointer le navigateur sur un script PHP, et vérifier que des fichiers sont créés dans le répertoire `/var/www/www.domaine.fr/xdebug`. Ces fichiers pourront être exploités avec l'un des outils suivants :

Unix	http://sourceforge.net/projects/kcachegrind/
Windows	http://sourceforge.net/projects/wincachegrind/
OSX	http://www.maccallgrind.com/
Web	http://code.google.com/p/webgrind/

4.16 Quelques vérifications finales

Par sécurité, il vaut mieux supprimer le petit script de test utilisé pour valider les étapes de l'installation :

```
rm /var/www/www.domaine.fr/test.php
```

De la même façon, on pourra supprimer le script d'index créé dans le répertoire du compte `domaine` pendant l'installation de yacs :

```
rm /home/domaine/index.php
```

Le lendemain de l'installation, vérifier le contenu du répertoire `/home/snapshot` et l'éventuelle duplication vers une machine tierce.

Au cas où vous chercheriez vos petits quelque temps après l'installation d'un site virtuel, voici un résumé de la situation :

- Le répertoire `/home/domaine/www` est le répertoire d'installation de yacs. Il contient tous les scripts PHP, les fichiers de paramètres, les fichiers et les images téléversés sur le site.
- Le répertoire `/var/www/www.domaine.fr` est un alias du précédent, conforme aux bonnes pratiques de gestion des sites Apache.
- La base de données pour ce site est faite de tous les fichiers placés dans le répertoire `/home/mysql/www_domaine_fr`. Attention, il est impératif de passer par phpMyAdmin ou par un outil de MySQL pour exporter ou importer des données. N'essayez pas de manipuler les fichiers directement, vous pourriez le regretter.
- Les fichiers de configuration des serveurs virtuels Apache sont dans le répertoire `/etc/apache2/sites-available`
- Les traitements d'arrière-plan sont indiqués dans `/etc/cron.d/www.domaine.fr`

4.17 Suppression d'un serveur virtuel

Bien sûr, il est beaucoup plus facile de casser que de construire. Si un jour vous aviez besoin de supprimer complètement un site virtuel, vous pourriez procéder comme suit :

```
rm /etc/cron.d/www_domaine_fr
rm /etc/awstats/awstats.www.domaine.fr.conf
rm /var/lib/awstats/*.www.domaine.fr.txt
a2dissite www.domaine.fr
a2dissite www.domaine.fr-ssl
/etc/init.d/apache2 reload
rm /etc/apache2/sites-available/www.domaine.fr
```

```
rm /etc/apache2/sites-available/www.domaine.fr-ssl
rm /var/www/www.domaine.fr
deluser --force --remove-home domaine
echo "drop database www_domaine_fr" | mysql
```

5 Référence

Cette section reprend le contenu type des fichiers de configuration essentiels. Elle permet de contrôler une installation, indépendamment du déroulement linéaire de l'installation tel qu'abordé dans les sections précédentes de ce guide. Seules les lignes actives sont listées, les commentaires ayant été retirés pour faciliter l'accès rapide à l'information.

5.1 /etc/apt/sources.list

```
deb http://ftp.fr.debian.org/debian/ testing main
deb http://security.debian.org/ testing/updates main
```

5.2 /etc/ssh/sshd_config

```
Port 22
Protocol 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
UsePrivilegeSeparation yes
KeyRegenerationInterval 3600
ServerKeyBits 768
SyslogFacility AUTH
LogLevel INFO
AllowUsers remote
LoginGraceTime 120
PermitRootLogin no
StrictModes yes
RSAAuthentication yes
PubkeyAuthentication yes
IgnoreRhosts yes
RhostsRSAAuthentication no
HostbasedAuthentication no
PermitEmptyPasswords no
ChallengeResponseAuthentication no
X11Forwarding no
X11DisplayOffset 10
PrintMotd no
PrintLastLog yes
TCPKeepAlive yes
AcceptEnv LANG LC_*
```

```
Subsystem sftp /usr/lib/openssh/sftp-server
UsePAM yes
```

5.3 /etc/apache2/apache2.conf

```
ServerRoot "/etc/apache2"
LockFile /var/lock/apache2/accept.lock
PidFile ${APACHE_PID_FILE}
Timeout 300
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 3
<IfModule mpm_prefork_module>
    StartServers          5
    MinSpareServers       5
    MaxSpareServers       10
    MaxClients            70
    ServerLimit           70
    MaxRequestsPerChild   50000
</IfModule>

User ${APACHE_RUN_USER}
Group ${APACHE_RUN_GROUP}

AccessFileName .htaccess

<Files ~ "^\.ht">
    Order allow,deny
    Deny from all
</Files>

DefaultType text/plain

HostnameLookups Off

ErrorLog /var/log/apache2/error.log
LogLevel warn

Include /etc/apache2/mods-enabled/*.load
Include /etc/apache2/mods-enabled/*.conf
```

```
Include /etc/apache2/httpd.conf
```

```
Include /etc/apache2/ports.conf
```

```
LogFormat "%v:%p %h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" vhost_combined
```

```
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
```

```
LogFormat "%h %l %u %t \"%r\" %>s %b" common
```

```
LogFormat "%{Referer}i -> %U" referer
```

```
LogFormat "%{User-agent}i" agent
```

```
CustomLog /var/log/apache2/other_vhosts_access.log vhost_combined
```

```
Include /etc/apache2/conf.d/
```

```
Include /etc/apache2/sites-enabled/
```

5.4 /etc/apache2/conf.d/security

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

```
ServerTokens Prod
```

```
ServerSignature Off
```

```
TraceEnable Off
```

5.5 /etc/apache2/ports.conf

```
NameVirtualHost *:80
```

```
Listen 80
```

```
<IfModule mod_gnutls.c>
```

```
    NameVirtualHost *:443
```

```
    Listen 443
```

```
</IfModule>
```

5.6 /etc/php5/apache2/php.ini

```
[PHP]
engine = On
zend.ze1_compatibility_mode = Off
short_open_tag = On
asp_tags = Off
precision      = 12
y2k_compliance = On
output_buffering = Off
zlib.output_compression = Off
implicit_flush = Off
unserialize_callback_func=
serialize_precision = 100
allow_call_time_pass_reference = On
safe_mode = Off
safe_mode_gid = Off
safe_mode_include_dir =
safe_mode_exec_dir =
safe_mode_allowed_env_vars = PHP_
safe_mode_protected_env_vars = LD_LIBRARY_PATH
disable_functions = shell_exec, passthru, exec, popen, proc_open, dl
disable_classes =
expose_php = Off
max_execution_time = 30
max_input_time = 60
memory_limit = 128M
error_reporting = E_ALL & ~E_NOTICE
display_errors = On
display_startup_errors = Off
log_errors = Off
log_errors_max_len = 1024
ignore_repeated_errors = Off
ignore_repeated_source = Off
report_memleaks = On
track_errors = Off
variables_order = "EGPCS"
register_globals = Off
register_long_arrays = On
register_argc_argv = On
auto_globals_jit = On
```

```
post_max_size = 64M
magic_quotes_gpc = On
magic_quotes_runtime = Off
magic_quotes_sybase = Off
auto_prepend_file =
auto_append_file =
default_mimetype = "text/html"
doc_root =
user_dir =
enable_dl = Off
file_uploads = On
upload_max_filesize = 32M
allow_url_fopen = Off
allow_url_include = Off
default_socket_timeout = 60
```

[Date]

[filter]

[iconv]

[sqlite]

[xmlrpc]

[Pcre]

[Syslog]

```
define_syslog_variables = Off
```

[mail function]

```
SMTP = localhost
```

```
smtp_port = 25
```

[SQL]

```
sql.safe_mode = Off
```

[ODBC]

```
odbc.allow_persistent = On
```

```
odbc.check_persistent = On
```

```
odbc.max_persistent = -1
odbc.max_links = -1
odbc.defaultlrl = 4096
odbc.defaultbinmode = 1
```

[MySQL]

```
mysql.allow_persistent = On
mysql.max_persistent = -1
mysql.max_links = -1
mysql.default_port =
mysql.default_socket =
mysql.default_host =
mysql.default_user =
mysql.default_password =
mysql.connect_timeout = 60
mysql.trace_mode = Off
```

[MySQLi]

```
mysqli.max_links = -1
mysqli.default_port = 3306
mysqli.default_socket =
mysqli.default_host =
mysqli.default_user =
mysqli.default_pw =
mysqli.reconnect = Off
```

[mSQL]

```
msql.allow_persistent = On
msql.max_persistent = -1
msql.max_links = -1
```

[OCI8]

[PostgreSQL]

```
pgsql.allow_persistent = On
pgsql.auto_reset_persistent = Off
pgsql.max_persistent = -1
pgsql.max_links = -1
pgsql.ignore_notice = 0
pgsql.log_notice = 0
```

```
[Sybase]
sybase.allow_persistent = On
sybase.max_persistent = -1
sybase.max_links = -1
sybase.min_error_severity = 10
sybase.min_message_severity = 10
sybase.compatibility_mode = Off
```

```
[Sybase-CT]
sybct.allow_persistent = On
sybct.max_persistent = -1
sybct.max_links = -1
sybct.min_server_severity = 10
sybct.min_client_severity = 10
```

```
[bcmath]
bcmath.scale = 0
```

```
[browscap]
```

```
[Informix]
ifx.default_host =
ifx.default_user =
ifx.default_password =
ifx.allow_persistent = On
ifx.max_persistent = -1
ifx.max_links = -1
ifx.textasvarchar = 0
ifx.byteasvarchar = 0
ifx.charasvarchar = 0
ifx.blobinfile = 0
ifx.nullformat = 0
```

```
[Session]
session.save_handler = files
session.use_cookies = 1
session.name = PHPSESSID
session.auto_start = 0
session.cookie_lifetime = 0
session.cookie_path = /
session.cookie_domain =
```

```
session.cookie_httponly =
session.serialize_handler = php
session.gc_divisor      = 100
session.gc_maxlifetime = 1440
session.bug_compat_42 = 1
session.bug_compat_warn = 1
session.referer_check =
session.entropy_length = 0
session.entropy_file =
session.cache_limiter = nocache
session.cache_expire = 180
session.use_trans_sid = 0
session.hash_function = 0
session.hash_bits_per_character = 4
url_rewriter.tags = "a:href,area:href,frame:src,input:src,form=,fieldset="
```

[MSSQL]

```
mssql.allow_persistent = On
mssql.max_persistent = -1
mssql.max_links = -1
mssql.min_error_severity = 10
mssql.min_message_severity = 10
mssql.compatibility_mode = Off
mssql.secure_connection = Off
```

[Assertion]

[COM]

[mbstring]

[FrontBase]

[gd]

[exif]

[Tidy]

```
tidy.clean_output = Off
```

[soap]

```
soap.wsdl_cache_enabled=1
soap.wsdl_cache_dir="/tmp"
soap.wsdl_cache_ttl=86400
```

5.7 /etc/mysql/my.cnf

```
[client]
port                = 3306
socket              = /var/run/mysqld/mysqld.sock

[mysqld_safe]
socket              = /var/run/mysqld/mysqld.sock
nice                = 0

[mysqld]
user                = mysql
pid-file            = /var/run/mysqld/mysqld.pid
socket              = /var/run/mysqld/mysqld.sock
port                = 3306
basedir             = /usr
datadir             = /home/mysql
tmpdir              = /tmp
language            = /usr/share/mysql/english
skip-external-locking

bind-address        = 127.0.0.1
ft_min_word_len    = 3
key_buffer          = 64M
max_allowed_packet = 16M
read_buffer         = 1M
sort_buffer         = 4M
table_cache        = 256
thread_stack       = 128K
thread_cache_size  = 32
myisam-recover     = BACKUP
query_cache_limit  = 2M
query_cache_size   = 32M
query_cache_type   = 1
expire_logs_days   = 10
max_binlog_size    = 100M
skip-bdb
```

```

[mysqldump]
quick
quote-names
max_allowed_packet = 16M

[mysql]

[isamchk]
key_buffer                = 16M
!includedir /etc/mysql/conf.d/

```

5.8 /etc/apache2/sites-available/www.domaine.fr

```

<VirtualHost *:80>
    AssignUserID domaine www-data
    ServerName www.domaine.fr
    ServerAdmin contact@domaine.fr
    DocumentRoot /var/www/www.domaine.fr
    <Directory />
        Options FollowSymlinks
        AllowOverride None
    </Directory>
    <Directory /var/www/www.domaine.fr>
        Options FollowSymlinks
        AllowOverride FileInfo Indexes Options
        Order allow,deny
        allow from all
    </Directory>
    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
    <Directory "/usr/lib/cgi-bin">
        AllowOverride None
        Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
        Order allow,deny
        Allow from all
    </Directory>
    ErrorLog /var/log/apache2/error-www.domaine.fr.log
    LogLevel warn
    CustomLog /var/log/apache2/access-www.domaine.fr.log combined
</VirtualHost>

```

5.9 /etc/apache2/sites-available/www.domaine.fr-ssl

```
<IfModule mod_gnutls.c>
<VirtualHost *:443>
    AssignUserID domaine www-data
    ServerName www.domaine.fr
    ServerAlias domaine.fr
    ServerAdmin contact@domaine.fr
    DocumentRoot /var/www/www.domaine.fr
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /var/www/www.domaine.fr>
        Options FollowSymLinks
        AllowOverride FileInfo Indexes Options
        Order allow,deny
        allow from all
    </Directory>
    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
    <Directory "/usr/lib/cgi-bin">
        AllowOverride None
        Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
        Order allow,deny
        Allow from all
    </Directory>
    ErrorLog /var/log/apache2/error-www.domaine.fr.log
    LogLevel warn
    CustomLog /var/log/apache2/access-www.domaine.fr.log combined

    GnuTLSEnable on
    GnuTLSCertificateFile /etc/ssl/certs/www.domaine.fr.pem
    GnuTLSKeyFile /etc/ssl/private/www.domaine.fr.pem
    GnuTLSPriorities NORMAL

</VirtualHost>
</IfModule>
```