Découvrir Debian pour mettre en place un serveur LAMP complet

Version 1.0

Thomas Bassetto, Julien Molina

15 September 2009

Table des matières

1 Introduction

1.1 Contexte

De nos jours on peut trouver des serveurs dédiés à bas prix. C'est une bonne solution pour ceux qui veulent héberger leur site Web pour un prix abordable tout en conservant la possibilité de configurer entièrement le système d'exploitation. Dans ce tutoriel, nous allons nous placer dans l'optique où plusieurs étudiants paient ensemble un serveur dédié, qu'ils se partagent pour héberger plusieurs sites Web. Par conséquent plusieurs concessions ont été faites, il n'est pas facile de trouver le juste milieu entre performance, sécurité et facilité d'utilisation...

En suivant ce tutoriel, vous obtiendrez un serveur sécurisé et bien configuré pour héberger vos sites Web et éviter un minimum d'attaques malveillantes.

Warning : Gardez à l'esprit que devenir administrateur système est beaucoup plus long que la lecture de ce tutoriel. Si à la fin vous aurez acquis une abse solide, ça ne remplacera pas la pratique sur le long terme. N'oubliez pas qu'un serveur mal configuré ou non entretenu peu rapidement être détourné en relais à spam ou se faire tout simplement hacker.

1.2 Finalité

- Le système d'exploitation utilisé est une **Debian Lenny 5.0**.
- Nous allons tout installer avec les paquets fournis par **Debian Lenny 5.0** pour faciliter les mises à jour.
- Le shell par défaut, **Bash**, sera conservé mais personnalisé.
- Nous allons apprendre à nous servir de **screen**.
- **Apache** et **PHP** seront configurés pour être le moins "bavards" possible.
- **Apache** et **PHP** seront par défaut peu permissifs. Les besoins particuliers des sites Web seront réglés au cas par cas dans le *VirtualHost*.
- Nous allons configurer **Postfix** pour avoir une gestion minimale des emails (envoi par le système et **PHP**).
- Le SGBD (Système de Gestion de Base de Données) installé sera MySQL.
- Nous allons créer **un compte système par site Web**, pour faciliter la gestion des droits.
- Aucun serveur FTP ne sera installé ! À la place nous allons utiliser **SFTP**, un protocole plus sûr même si moins répandu.
- Nous allons mettre en place un pare-feu, ainsi que les outils **Fail2ban**, **rkhunter** et **Monit**.
- Nous allons mettre en place un système de sauvegardes.

2 Prérequis

2.1 Compétences

Quand on écrit ce genre de tutoriel il est dur, d'un part, de faire abstraction des acquis que l'on a déjà concernant GNU/Linux et, d'autre part, de connaître les compétences des futurs lecteurs. Nous avons essayer de trouver le juste milieu, tout en donnant des liens complémentaires pour ceux qui souhaitent plus d'informations.

Si vous débutez **vraiment**, nous vous conseillons vivement de lire le tutoriel disponible sur le Site du zéro intitulé *Reprenez le contrôle avec Linux !* [1]. Avant de continuer il est aussi recommandé de savoir se servir de la ligne de commandes, modifier les fichiers de configuration (décommenter les lignes, etc.) et utiliser un éditeur de texte.

2.2 Serveur dédié chez un prestataire comme Dédibox, OVH, Gandi, etc.

Nous n'allons pas donner de conseils sur un prestataire en particulier, c'est à vous de choisir.

Dans la majorité des cas, l'installation du système d'exploitation est faite par le prestataire, nous n'allons donc pas en parler.

2.3 Le cas des "Panel"

Il existe des logiciels que l'on appelle "Panel" qui servent à configurer votre serveur via une interface graphique. C'est une solution qui vaut ce qu'elle vaut, mais un bon administrateur GNU/Linux doit de toute façon comprendre comment le serveur fonctionne sous le capot !

3 Debian, accès root et bases

3.1 Passer en mode super-administrateur (root)

Comme vous devez déjà le savoir ce mode vous permet d'avoir tout les droits sur votre système. Contrairement à Ubuntu (pour ceux qui y sont habitués) où il suffit de rajouter **sudo** devant la commande à exécuter, Debian requiert que l'on se connecte vraiment avec l'utilisateur *root*. Pour faire cela rien de plus simple : utiliser la commande **su -** puis rentrer le mot de passe *root* défini durant l'installation ou fourni par votre prestataire.

Ne restez connecté en *root* que le temps de faire vos modifications. Plus longtemps vous utilisez ce compte, plus vous avez de chance de faire une bêtise irréparable ! Pour se déconnecter il suffit de taper la commande **exit**.

1. http ://www.siteduzero.com/tutoriel-3-12827-reprenez-le-controle-avec-linux.html

Gardez à l'esprit que mis à part les utilisateurs systèmes (on ne se préoccupe pas d'eux, ils sont juste utile pour le fonctionnement de certains programmes), il n'existe pour l'instant que deux utilisateurs sur votre système : le super-administrateur *root* et votre utilisateur par défaut que l'on nommera *john* dans la suite de ce tutoriel.

3.2 Gestionnaire de paquets : apt-get

Sous Debian, le principal logiciel pour gérer les (paquets) logiciels installés se nomme **apt-get**. Il faut l'utiliser connecté en *root*.

Ci-dessous une liste des options les plus importantes :

apt-get update Mise à jour de la liste des paquets avant l'installation d'un paquet ou d'une mise à jour du système.

apt-get upgrade Mise à jour tous les paquets du système en installant leurs dépendances.

apt-get install bar Installe *bar* avec ses dépendances.

apt-cache showpkg foo bar ... Affiche l'information sur les paquets *foo bar* ....

apt-get remove bar Supprime le paquet *bar* mais garde ses fichiers de configuration.

apt-get remove –purge bar Supprime le paquet *bar* et tous ses fichiers de configuration.

apt-cache search bar Affiche les paquets dont le nom contient *bar*.

L'option **-s** couplée avec **install** ou **update** permet de simuler l'action (par exemple : **apt-get install -s bar**). C'est par exemple utile pour vérifier les dépendances d'un paquet.

3.3 Répertoires importants

Il peut être utile de savoir où sont stockés les fichiers importants :

`/home/` Contient les répertoires personnels des utilisateurs.

`/root/` Répertoire personnel du super-utilisateur *root*.

`/etc/` Contient les fichiers de configurations des logiciels, répartis entre les sous-répertoires.

`/var/log/` Contient les fichiers journaux (logs) des logiciels, répartis entre les sous-répertoires.

3.4 Quelques commandes utiles

pwd Cette commande permet de connaître le chemin complet du répertoire où l'on se trouve.

ls Cette commande permet d'obtenir beaucoup d'informations sur les fichiers présents dans un répertoire. **ls -al** permet d'afficher des informations les fichiers et répertoires cachés et les affiche en colonnes avec plus d'informations comme les droits, le propriétaire, etc.

mkdir Cette commande permet de créer un répertoire, sa syntaxe est la suivante : **mkdir [option] répertoire-à-créer**. L'option **-p** permet de ne pas afficher d'erreur si le répertoire existe déjà.

touch Cette commande permet de changer la date de modification d'un fichier, ou de le créer s'il n'existe pas. Sa syntaxe est la suivante : **touch fichier-à-créer**.

cat et less La commande **cat** permet de lire des fichiers. **less** a l'avantage d'afficher le fichier page par page.

cp et mv La commande **cp** permet de copier des fichiers, sa syntaxe est la suivante : **cp [option] fichier-origine fichier-destination** ou **cp [option] fichier répertoire**. **mv** déplace les fichiers. On s'en sert aussi pour renommer les fichiers.

rm et rmdir La commande **rm** permet de supprimer des fichiers. L'option **-R** permet de le faire de façon récursive. **rmdir** permet de supprimer des répertoires, si ils sont vides ! La commande **rm -rf nom_du_repertoire/** permet de forcer la suppression du répertoire et de tout ce qu'il contient. Cette commande n'affiche aucun message même quand les fichiers sont inexistants, attention aux fausses manipulations avec cette commande, les résultats pourraient être catastrophiques.

find La commande **find** permet de retrouver des fichiers ou répertoires, sa syntaxe est la suivante : **find [options]**. Les options les plus utiles sont **-name nom-de-l-élément-à-trouver** et **-type type-de-l-élément-à-trouver**.

chmod et chown La commande **chmod** permet de modifier les droits du fichier ou répertoire. Les différents droits sont *r* pour *read* équivaut à 4 (le droit de lecture), *w* pour *write* équivaut à 2 (le droit d'écriture), *x* pour *execute* équivaut à 1 (le droit d'exécution). La commande peut s'utiliser de deux façons. La première **chmod nnn élément-à-modifier** définit les droits pour chaque groupe, le premier *n* correspond à la somme des valeurs des droits pour l'utilisateur, le second *n* est pour le groupe et le troisième *n* est pour les autres. La seconde **chmod groupe+droit élément-à-modifier** permet de modifier les droits pour un groupe (*u* pour *user*, *g* pour *group* et *o* pour *other*), on peut mettre autant de droits que l'on veut en ajoutant des **+droit**, on peut les enlever en mettant des **-droit**. Exemple **chmod u+r-x fichier**. La commande **chown** permet de changer le propriétaire et le groupe du fichier ou répertoire, sa syntaxe est **chown propriétaire :groupe élément-à-modifier**.

df et du La commande **df** permet de connaître la taux d'utilisation des disques durs, sa syntaxe est la suivante : **du [option]**. L'option **-h** permet de rendre le résultat facilement lisible. La commande **du** permet de connaître l'espace pris sur le disque dur par le répertoire courant, sa syntaxe est la suivante : **du [option]**. L'option **-hs** permet de rendre le résultat plus lisible et de faire un résumé.

Note : De manière générale, si vous souhaitez avoir des informations sur une commande utilisez **man**. Cette commande s'utilise de la manière suivante : **man nom_commande_inconnue**.

Voir aussi :

Les commandes fondamentales de Linux Présentation un peu plus avancée de commandes Linux utiles.

Aide-mémoire des commandes Linux Aide-mémoire assez complet.

Les astuces les plus intéressantes Sélection des commandes les plus pratiques.

4 nano, l'éditeur de texte

Sous **Linux** il existe de nombreux éditeurs de texte en ligne de commande. **nano** est simple, installé par défaut mais beaucoup moins complet que des éditeurs comme **emacs** ou **vi**. Malgré tout il suffira pour faire les quelques modifications utiles dans ce tutoriel.

Pour éditer un fichier il suffit d'utiliser **nano nom_du_fichier**. On rentre ensuite en mode édition du fichier. Les commandes sont affichées en bas de l'écran. Le caractère `^` représente la touche `Ctrl` du clavier.

Les raccourcis les plus importants sont :

`Ctrl + G` Afficher l'aide.

`Ctrl + K` Couper la ligne de texte (et la mettre dans le presse-papier).

`Ctrl + U` Coller la ligne de texte que vous venez de couper.

`Ctrl + C` Afficher à quel endroit du fichier votre curseur est positionné (numéro de ligne...).

`Ctrl + W` Rechercher dans le fichier.

`Ctrl + O` Enregistrer le fichier.

`Ctrl + X` Quitter nano.

Un très bon tutoriel d'initiation à **nano** est disponible sur le Site du zéro [2]

Note : **nano** est configurable, il suffit de modifier le fichier `.nanorc`. Pour cela connectez-vous en **root** puis copiez le fichier `/etc/nanorc` dans votre répertoire home : **cp /etc/nanorc /home/votre_pseudo/.nanorc**. Puis changez le propriétaire : **chown votre_pseudo /home/votre_pseudo/.nanorc**.

Les lignes que nous vous conseillons de décommenter sont :

```
set autoindent        # Autoindentation
set backup    # Créé automatiquement une sauvegarde nom_fichier~
set nonewlines        # Pas de nouvelles lignes à la fin du fichier
set nowrap    # Ne coupe pas le texte (wrap)
set rebinddelete # Utile quand on pilote le serveur via un Mac
set smooth    # Defilement doux
```

2. http ://www.siteduzero.com/tuto-3-24614-1-nano-l-editeur-de-texte-du-debutant.html

Et surtout la coloration automatique :

```
include "/usr/share/nano/nanorc.nanorc"       # Du fichier .nanorc
include "/usr/share/nano/html.nanorc"         # Des fichiers HTML
include "/usr/share/nano/sh.nanorc"           # Des fichiers bash
```

Voir aussi :

Nano, l'éditeur de texte du débutant Documentation avec QCM par le Site du Zéro

Vim : l'éditeur de texte du programmeur Documentation sur un autre éditeur de texte en ligne de commande plus puissant mais plus long à maîtriser

5 bash, le shell

Par défaut le shell installé est **bash**. Il est possible de le personnaliser en éditant le fichier `.bashrc` qui se trouve dans le répertoire personnel de l'utilisateur (que ce soit **john** ou **root**).

Les deux choses qui nous intéressent le plus ici sont les *alias* et la mise en valeur du prompt quand on est connecté en **root**.

Les *alias* permettent de créer de nouvelles commandes utilisables dans le terminal, des sortes de raccourcis. Quelques alias pratiques sont (à rajouter à la fin du fichier `/root/.bashrc`) :

```
alias ls='ls -al' # Affiche le contenu de dossier en colonne et avec les fichiers cachés
alias rm='rm -i' # Force rm à demander des confirmations pour les étourdis
alias df='df -h' # Donne l'occupation des disques dans un format lisible
alias du='du -hs' # Idem mais par répertoires
alias golog='cd /var/log' # Aller directement dans les répertoires des fichiers journaux (logs)
```

Pour personnaliser le prompt, rajoutez à la fin du même fichier `export PS1='[\t]\[\e[41;1;37m\]\u@\h :\w\[\e[0m\]\$ '`. Vous aurez ainsi un prompt blanc sur fond rouge avec la date au début. Pour prendre en compte ces modifications il faut se reconnecter ou utiliser la commande **source /root/.bashrc**.

6 Gestion des utilisateurs

La gestion des utilisateurs occupe un chapitre à elle toute seule car il y a beaucoup de connaissances théoriques à intégrer avant d'aller plus loin. Comme nous l'avons déjà évoqué, vous avez pour l'instant deux utilisateurs pouvant se connecter à votre système : **root** et **john**.

6.1 Fichiers et répertoires à connaître

`/etc/passwd` Ce fichier contient la liste des utilisateurs avec leurs caractéristiques, leur mot de passe chiffré et leur shell de connexion.

`/etc/group` Ce fichier contient la liste des groupes disponibles sur le serveur.

`/etc/shells` Ce fichier contient la liste des shells de connexion autorisés pour les utilisateurs.

`/etc/skel/` Contient la liste des fichiers qui seront placés dans le répertoire personnel d'un utilisateur lors de sa création.

`/home/` Contient les dossiers personnels des utilisateurs, comme `/home/john/` par exemple.

6.2 Création d'un groupe et d'un utilisateur

Nous avons fait le choix de créer un utilisateur Linux par site Web hébergé sur le serveur. Une autre solution aurait été de créer un répertoire `/var/www/nom_site/` pour chaque site Web. Chaque méthode à ses avantages et ses inconvénients. En faisant comme nous, vous aurez moins de soucis pour gérer les droits des fichiers et vous pourrez facilement abandonner le protocole FTP pour passer à une méthode plus sûre de transfert de fichier nommé SFTP (FTP over SSH).

Même si on reviendra plus tard sur SSH, il faut s'assurer que vous savez vous connecter à votre serveur. Normalement votre prestataire devrait vous l'avoir dit. Dans une console, tapez la commande **ssh john@ip_serveur**. Et voilà !

Pour créer (respectivement supprimer) un utilisateur sous Debian il existe la commande **adduser** (respectivement **deluser**). Ces commandes sont spécifiques à Debian et ses dérivés, les commandes classiques sont **useradd** et **userdel** mais elles proposent moins d'options.

La commande pour définir ou changer le mot de passe d'un utilisateur est **passwd john** et celle pour créer un groupe d'utilisateur est **addgroup**.

On va donc commencer par ajouter un groupe **sitesweb**, dans lequel seront tous nos utilisateurs/sites Web :

```
# addgroup sitesweb
Adding group 'sitesweb' (GID 1000) ...
Done.
```

6.3 Le problème du shell

À moins de vouloir faire un serveur FTP pour distribuer des fichiers (comme *ftp.debian.org* par exemple), vous n'aurez pas besoin d'un logiciel dédié à ce service. Nous allons donc éviter les solutions telles que **proftpd** ou **vsftpd** pour passer par **SSH**, qui est déjà installé sur la machine. Cette solution nous permettra en plus d'utiliser le protocole SFTP au lieu de FTP, qui est plus sûr (les transferts sont chiffrés).

Chaque utilisateur que l'on va créer pour les sites Web aura un accès SSH et donc l'accès SFTP. Mais l'accès SSH veut donc dire qu'il a accès à la console et à toutes les commandes disponibles sur le serveur. C'est le cas de notre utilisateur *john* par exemple. Cependant ce n'est pas forcément ce qu'on désire pour nos autres utilisateurs. Pour régler ces ennuis, nous allons installer un *shell* alternatif ne permettant d'utiliser que SFTP et nous allons aussi "chrooter" leur répertoire personnel pour qu'ils ne puissent pas se "balader" au delà.

6.4 SFTP et chroot

Il existe deux principaux shells pour n'autoriser que SFTP : *rssh* et *scponly*. Nous allons utiliser ce dernier car le paquet fourni un script prêt à l'emploi qui nous facilite le travail.

Installons *scponly* : **apt-get install scponly**. Lors de l'installation, la ligne `/usr/sbin/scponlyc` est directement rajoutée à la fin du fichier `/etc/shells` qui représente la liste des *shells* utilisables.

Ce paquet a rajouté à notre système un script tout fait pour rajouter des utilisateurs Linux ne pouvant utiliser que SFTP et disposant d'un environnement chrooté. Pour l'obtenir aller dans le bon répertoire : **cd /usr/share/doc/scponly/setup_chroot/** et décompresser le : **gunzip setup_chroot.sh.gz**. Donnons les droits d'exécution au fichier : **chmod +x setup_chroot.sh** et lançons le pour créer notre premier utilisateur : **./setup_chroot.sh**.

```
./setup_chroot.sh

Next we need to set the home directory for this scponly user.
please note that the user's home directory MUST NOT be writeable
by the scponly user. this is important so that the scponly user
cannot subvert the .ssh configuration parameters.

for this reason, a writeable subdirectory will be created that
the scponly user can write into.
```

```
Username to install [scponly]monsiteweb-fr
```

Dans un premier temps le script vous demande le nom de l'utilisateur a créer (*monsiteweb-fr* dans notre cas)

```
home directory you wish to set for this user [/home/monsiteweb-fr]
```

Ici on vous demande le chemin du répertoire personnel de l'utilisateur, il est conseillé de laisser la valeur par défaut.

```
name of the writeable subdirectory [incoming]www
```

A cette étape il faut entrer le nom d'un répertoire qui sera créé dans le répertoire personnel de l'utilisateur et pour lequel celui-ci aura les droits en écritures (*www* dans notre cas). Il n'a pas accès en écriture à la racine de son répertoire personnel sinon il pourrait modifier les fichier du répertoire *.ssh/* et s'accorder plus de droits que nous le lui avons donné !

```
creating  /home/monsiteweb-fr/www directory for uploading files

Your platform (Linux) does not have a platform specific setup script.
This install script will attempt a best guess.
If you perform customizations, please consider sending me your changes.
Look to the templates in build_extras/arch.
 - joe at sublimation dot org
```

Pour finir on vous demande d'entrer un mot de passe pour le nouvel utilisateur.

```
please set the password for monsiteweb-fr:
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
if you experience a warning with winscp regarding groups, please install
the provided hacked out fake groups program into your chroot, like so:
cp groups /home/exemple-com/bin/groups
```

Il faut maintenant activer le bit SUID sur `/usr/sbin/scponlyc` : **chmod +s /usr/sbin/scponlyc**. Cette commande n'est à faire que lors de la création du premier utilisateur ! Pour les autres, il ne sera plus utile de la faire.

Sous **Debian Etch 4.0** et **Debian Lenny 5.0** (aucune idée pour les autres distributions), il y a un bug à corriger : il faut créer un fichier `/dev/null` dans le répertoire personnel :

```
# cd ~monsiteweb-fr
# mkdir dev
# mknod -m 666 dev/null c 1 3
```

> **Warning :** Pour finir, il faut rajouter notre nouvel utilisateur au groupe *sitesweb* : **usermod -aG sitesweb monsiteweb-fr**. Ça y est vous pouvez vous connecter en SFTP avec l'utilisateur *monsiteweb-fr*.

7 SSH

Attention, si vous avez sautez le chapitre sur les utilisateurs, vous n'avez sûrement pas créé l'utilisateur *monsiteweb-fr* comme demandé. C'est le moment de le faire.

Comme on l'a déjà vu, la commande pour se connecter à notre serveur, la commande à utiliser est : **ssh john@adresse_ip_du_serveur**.

Afin de sécuriser un minimum l'accès SSH au serveur, éditons le fichier `/etc/ssh/sshd_config` : **nano /etc/ssh/sshd_config**.

Modifiez (ou ajoutez cas échéant) les lignes suivantes :

```
PermitRootLogin no        # Ne pas permettre de connexion directement avec root.
Protocol 2                # Utilisation du protocole v2 uniquement.
AllowGroups john sitesweb # L'utilisateur doit appartenir à un de ces groupes pour pouvoir se con
```

Le groupe *sitesweb* a été créé dans le chapitre précédent (les utilisateurs). N'oubliez pas non plus de rajouter le groupe de votre utilisateur "admin" (pour nous **john**) sinon vous ne pourrez plus jamais vous connecter à votre serveur :o)

Nous pourrions aussi changer le port de connexion par défaut pour éviter quelques attaques par bruteforce sur le port 22, mais c'est un leurre un peu faible. Un scan **nmap** de votre serveur permettra de trouver facilement le nouveau port pour **SSH**. Nous n'allons pas parler du port-knoking non plus car son utilisation quotidienne devient vite pénible.

Pour prendre en compte les changements il faut relancer le serveur SSH connecté en **root** : **/etc/init.d/ssh restart**.

Note : Nous verrons en dans le chapitre "Firewall" comment bannir les adresses IP essayant de se connecter en **SSH** par bruteforce.

8 screen

8.1 Généralités

Quand vous êtes connecté depuis votre ordinateur sur votre serveur vous n'avez en général qu'une seule fenêtre avec un seul prompt. Cette situation peut vite devenir gênante si vous êtes en train d'éditer un fichier et que vous redémarrez un service en même temps pour voir si vos modifications fonctionnent. Le cheminement dans ce cas là est le suivant : * vous ouvrez le fichier à modifier dans un éditeur de texte, * vous faites vos modifications, * vous fermez l'éditeur, * vous relancez le serveur, * vous vous apercevez que ça n'a pas marché, * vous ouvrez à nouveau le fichier pour refaire des modifications, * etc.

screen permet d'ouvrir des fenêtres parallèles au sein de votre terminal pour par exemple en avoir une ouverte sur le fichier à éditer et une autre pour redémarrer le programme. Vous pouvez également vous déconnecter de **SSH** mais laisser vos *screens* ouverts, ce qui vous permettra de vous y *rattacher* lors de votre prochaine connexion.

screen Permet de créer un *screen* de type *bash*. Pour créer un screen d'un type différent il suffit de lancer **screen commande**, ce qui créé un *screen* spécialement dédié à cette commande.

screen -t "nom" Permet de créer un *screen* avec le titre "nom".

screen -S NOM_DE_SESSION Permet de créer un *screen* avec un nom de session plus explicite permettant ainsi de le rattacher plus facilement.

screen -r PID ou NOM_DE_SESSION Permet de se rattacher à un *screen* détaché au préalable grâce à son PID ou son NOM_DE_SESSION s'il s'agit d'un *screen* créé avec cette option.

screen -ls Permet d'obtenir la liste des *screens* avec leurs PID.

8.2 Liste des commandes sous screen

L'ensemble des commandes sous screen commencent toujours par `Ctrl+a`. Voici la liste des commandes les plus utiles dans les screens.

`Ctrl+a c` Permet de créer un nouveau *screen*.

`Ctrl+a A` Permet de modifier le titre d'un *screen*.

`Ctrl+a k` Permet de détruire le *screen* courant (demande une confirmation).

`Ctrl+a w` Permet d'obtenir une liste des *screens*.

`Ctrl+a 0-9` Permet d'aller directement au *screen* numéro 0-9.

`Ctrl+a n` Permet d'aller au *screen* suivant.

`Ctrl+a p` Permet d'aller au *screen* précédent.

`Ctrl+a Ctrl+a` Permet de jongler entre les deux derniers *screens* utilisés.

Ctrl+a ? Permet d'afficher l'ensemble des commandes *screen*.

Ctrl+a d Permet de se détacher du *screen* courant.

Ctrl+a Ctrl+ Permet de fermer *screen* (demande une confirmation).

Ctrl+a [Permet de démarrer le mode copie.

Ctrl+a] Permet de coller ce qui a été précédemment copié.

8.3 Mode copie

Une fois que vous avez démarré le mode copie grâce à la commande `Ctrl+a [`, il faut définir des délimiteurs entre lesquels le contenu sera copié. Pour cela vous pouvez vous déplacer grâce aux touches `h`, `j`, `k` et `l` (représentant les touches directionnelles) mais également grâce à `0` ou `^` pour aller en début de ligne ou `$` pour aller en fin de ligne. Une fois que vous êtes au point de départ de la copie appuyer sur `Entrer`, déplacez-vous jusqu'au point de fin de copie puis appuyer sur `Entrer`. Un message signale que la copie a été enregistrée. Pour coller il faut utiliser la commande `Ctrl+a ]`, elle peut être utilisée dans un autre *screen*.

8.4 Fichier de configuration `.screenrc`

Voici un exemple de fichier de configuration `.screenrc` qui doit être placé dans le répertoire personnel de l'utilisateur courant. Ce fichier permet de configurer **screen** mais également d'afficher deux lignes en bas du terminal sous **screen**. La première permet de d'afficher les différents *screens* et affiche en rouge le *screen* courant, la deuxième affiche le date et la distribution.

```
deflogin on
vbell off
defscrollback 1024
startup_message off

# pour l'affichage des differents onglets (avant derniere ligne)
caption always "%{+u wk}%?%-w%?%{rk}\%n %t/%{wk}%?%+w%?"
# la ligne tout en bas
hardstatus alwayslastline "%{+b kg}%H%{ky} > le %d/%m/%Y %c %{kg}%=%42`"
```

9 Postfix, le serveur d'email

Postfix est le serveur SMTP de messagerie électronique libre le plus répandu. Il se charge de la livraison des e-mails et a été conçu de façon modulaire autour de différents programmes dévolus chacun à une tâche précise.

Nous n'allons pas traiter ici de la configuration d'un vrai serveur email, par nom de domaine. Nous allons juste mettre en place le minimum pour pouvoir envoyer des emails via PHP et pour les alertes des logiciels de surveillance système, c'est tout.

9.1 Installation

L'installation est assez simple : **apt-get install postfix mailx**.

Si l'installation ne vous propose pas directement de répondre aux questions de configuration des paquets, utilisez la commande : **dpkg-reconfigure postfix**.

Voici les réponses-types à apporter dans la langue de Shakespeare. En ce qui concerne le nom du serveur, remplacez bien sûr *monserveur* par votre hostname (de préférence celui contenu dans le fichier `/etc/hostname`) :

```
General type of configuration? <-- Internet Site
Where should mail for root go <-- rien (laisser blanc)
Mail name? <-- monserveur
Other destinations to accept mail for? (blank for none) <-- monserveur, localhost
```

```
Force synchronous updates on mail queue? <-- No
Local networks? <-- 127.0.0.0/8
Use procmail for local delivery? <-- Yes
Mailbox size limit <-- 0
Local address extension character? <-- +
Internet protocols to use? <-- all
```

9.2 Configuration minimale

Le fichier de configuration principal de Postfix est `/etc/postfix/main.cf`.

Description de certaines directives :

myhostname = monserveur Vérifiez que cette option contient bien le nom de domaine (FQDN) de votre serveur.

myorigin = /etc/mailname Cette option définit le nom utilisé par le serveur pour s'identifier. En précisant un nom de fichier (par défaut `/etc/mailname`), le contenu de celui-ci sera lu et assigné à l'option. Dans notre cas, `/etc/mailname` contient *monserveur*, n'hésitez pas à vérifier l'exactitude du contenu de ce fichier.

smtpd_banner = $myhostname ESMTP $mail_name (Debian/GNU) Bannière affichée lors de la connexion SMTP sur le port 25. Les variables commençant par $ seront remplacées par leurs valeurs définies précédemment. Soyez sobre.

mydestination = monserveur, localhost.localdomain, localhost Liste des domaines pour lesquels le serveur doit accepter le courrier.

relayhost = Pour effectuer les livraisons de courrier via un relais (ici vide).

mynetworks = 127.0.0.0/8 Réseaux locaux autorisés.

mailbox_size_limit = 0 Limite de taille pour les boîtes aux lettres, en octets (0 = illimité).

message_size_limit = 51200000 Limite de la taille maximum d'un message, en octets (ici 50 Mo).

À ces directives je vous conseille d'ajouter les suivantes pour empêcher votre serveur de devenir un relais de SPAM :

```
# Autorise les connexions depuis le réseau sûr seulement.
smtpd_client_restrictions = permit_mynetworks, reject

# Ne pas accepter de courrier des domaines qui n'existent pas.
smtpd_sender_restrictions = reject_unknown_sender_domain

# Liste blanche: les clients locaux peuvent indiquer n'importe quelle destination, pas les autres
smtpd_recipient_restrictions = permit_mynetworks, reject_unauth_destination

# Bloquer les clients qui parlent trop tôt
smtpd_data_restrictions = reject_unauth_pipelining
```

Après toute modification de ce fichier, redémarrez Postfix ou rechargez plus simplement la configuration grâce à **postfix reload**, ou encore **/etc/init.d/postfix reload**.

Pour tester l'envoi d'email vous pouvez utiliser le code suivant dans le terminal : **echo "Salut, je suis un email." | mail -s "Hello world" john@gmail.com**.

9.3 Rediriger les emails envoyer à root

Il est possible que certains logiciels envoient les emails au compte **root** de votre machine. Il seront stockés dans le fichier `/var/mail/root` ce qui n'est pas très pratique pour les lire... Le mieux à faire est de rediriger les emails envoyés à **root** vers votre adresse email. Pour cela éditez le fichier `/etc/aliases` : **nano /etc/aliases** et modifiez la ligne commençant par *root :* pour y mettre votre adresse email. Exemple :

```
root: john@gmail.com
```

Warning : Pour prendre en compte cette modification il vous faut ensuite utiliser la commande **newaliases**. Dans la suite du tutoriel, on configurera les logiciels de monitoring pour qu'ils envoient leurs emails à *root@monserveur*.

Voir aussi :

Postfix, SMTP, SASL (SSL), TLS, POP3 et IMAP Pour aller plus loin. C'est d'ailleurs le tutoriel qui a servi de base à la rédaction de cette partie.

10 Firewall

10.1 Netfilter et iptables

Depuis le noyau Linux 2.6, on dispose par défaut de la commande **iptables** et de Netfilter pour mettre en place les meilleures règles de firewalls pour Linux. Il existe de nombreux scripts de configuration, en voici un à adapter à votre configuration. À tout instant, vous pouvez utiliser la commande **iptables -L -v** pour lister les règles en place.

Celles-ci portent sur 3 chaînes : INPUT (en entrée), FORWARD (dans le cas d'un routage réseau) et OUPUT (en sortie). Les actions à entreprendre sont ACCEPT (accepter le paquet), DROP (le jeter), QUEUE et RETURN.

Arguments utilisés :
– i : interface d'entrée (input)
– o : interface de sortie (output)
– t : table (par défaut filter contenant les chaînes INPUT, FORWARD, OUTPUT)
– j : règle à appliquer (Jump)
– A : ajoute la règle à la fin de la chaîne (Append)
– I : insère la règle au début de la chaîne (Insert)
– R : remplace une règle dans la chaîne (Replace)
– D : efface une règle (Delete)
– F : efface toutes les règles (Flush)
– X : efface la chaîne
– P : règle par défaut (Policy)
– lo : localhost (ou 127.0.0.1, machine locale)
Nous allons créer un script qui sera lancé à chaque démarrage pour mettre en place des règles de base. D'abord créons et remplissions notre fichier : **nano /etc/init.d/firewall**.

```
#!/bin/sh

# Vider les tables actuelles
iptables -t filter -F

# Vider les règles personnelles
iptables -t filter -X

# Interdire toute connexion entrante et sortante
iptables -t filter -P INPUT DROP
iptables -t filter -P FORWARD DROP
iptables -t filter -P OUTPUT DROP

# ---

# Ne pas casser les connexions etablies
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

# Autoriser loopback (localhost)
iptables -t filter -A INPUT -i lo -j ACCEPT
iptables -t filter -A OUTPUT -o lo -j ACCEPT
```

```
# ICMP (Ping)
iptables -t filter -A INPUT -p icmp -j ACCEPT
iptables -t filter -A OUTPUT -p icmp -j ACCEPT

# ---

# SSH In : vérifiez bien votre port...
iptables -t filter -A INPUT -p tcp --dport 22 -j ACCEPT

# SSH Out : vérifiez bien votre port...
iptables -t filter -A OUTPUT -p tcp --dport 22 -j ACCEPT

# DNS In/Out
iptables -t filter -A OUTPUT -p tcp --dport 53 -j ACCEPT
iptables -t filter -A OUTPUT -p udp --dport 53 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 53 -j ACCEPT
iptables -t filter -A INPUT -p udp --dport 53 -j ACCEPT

# NTP Out : pour la mise à jour automatique de l'heure
iptables -t filter -A OUTPUT -p udp --dport 123 -j ACCEPT

# Whos is
iptables -t filter -A OUTPUT -p tcp --dport 43 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 80 -j ACCEPT

# HTTP + HTTPS Out
iptables -t filter -A OUTPUT -p tcp --dport 80 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 443 -j ACCEPT

# HTTP + HTTPS In
iptables -t filter -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 443 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 8443 -j ACCEPT

# Mail SMTP : on ne tient pas compte de POP(S) et IMAP(S) dans notre tuto
iptables -t filter -A INPUT -p tcp --dport 25 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 25 -j ACCEPT

# Monit : on va le voir plus tard :)
iptables -t filter -A INPUT -p tcp --dport 1337 -j ACCEPT
```

Lorsque vous avez défini toutes les règles, rendez ce fichier exécutable : **chmod +x /etc/init.d/firewall**.

Vous pourrez le tester en l'exécutant directement en ligne de commande : **/etc/init.d/firewall**. Assurez-vous d'avoir toujours le contrôle de votre machine (reconnectez-vous en SSH, vérifiez la disponibilité des services web, ftp, mail...). En cas d'erreur, redémarrez le serveur, les règles seront oubliées et vous permettront de reprendre la main. En revanche, si les tests s'avèrent concluants, ajoutez le script au démarrage pour que celui-ci protège le serveur dès le boot. Afin de l'ajouter aux scripts appelés au démarrage : **update-rc.d firewall defaults**.

Pour le retirer, vous pouvez utiliser la commande suivante : **update-rc.d -f firewall remove**.

Redémarrez, ou exécutez **/etc/init.d/firewall** pour activer le filtrage.

N'oubliez pas de tester vos règles. Un mauvais choix peut entraîner une indisponibilité de votre serveur ou une perte de contrôle sur celui-ci avec le blocage de votre connexion SSH.

..todo : : update-rc.d : warning : /etc/init.d/firewall missing LSB information update-rc.d : see <http ://wiki.debian.org/LSBInitScripts>

10.2 Fail2ban

Fail2ban est un script surveillant les accès réseau grâce aux logs des serveurs. Lorsqu'il détecte des erreurs d'authentification répétées, il prend des contre-mesures en bannissant l'adresse IP grâce à **iptables**. Cela permet d'éviter nombre d'attaques bruteforce et/ou par dictionnaire.

Pour l'installer il suffit de taper la commande **apt-get install fail2ban**. Ensuite on va le configurer en éditant **/etc/fail2ban/fail2ban.conf**.

Vérifiez la présence et l'activation de la ligne *logtarget = /var/log/fail2ban.log* qui correspond au fichier de log de **fail2ban**.

Les services à contrôler sont stockés dans le fichier `jail.conf`. Il est recommandé d'en effectuer une copie nommée `jail.local` qui sera automatiquement utilisée à la place du fichier d'exemple : **cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local** puis **nano /etc/fail2ban/jail.local**. Ce fichier est divisé en sections, que l'on peut identifier grâce aux crochets : *[DEFAULT]*, *[ssh]*, etc.

Quelques paramètres utiles :

ignoreip = 127.0.0.1 Liste des adresses IP de confiance à ignorer par fail2ban

findtime = 200000 Durée sur laquelle Fail2ban analyse le fichier de log. Fail2ban n'analyse pas le fichier de log entièrement mais seulement sur cette durée, si cette option n'est pas défini elle est égale au bantime.

bantime = 86400 Temps de ban en secondes. Ici une journée, on ne rigole pas.

maxretry = 3 Nombre d'essais autorisés pour une connexion avant d'être banni

destemail root@monserveur Adresse e-mail destinataire des notifications

Chaque section possède ses propres paramètres qui prennent le pas sur les globaux s'ils sont mentionnés :

enabled Monitoring activé (true) ou non (false)

port Port concerné

logpath Fichier de log à analyser pour détecter des anomalies

filter Filtre utilisé pour l'analyser du log

Les filtres, par défaut, sont stockés dans */etc/fail2ban/filter.d*. Ils contiennent en général une instruction `failregex` suivie d'une expression régulière matchant la détection d'une authentification erronée. Par exemple pour le service SSH :

```
failregex = (?:(?:Authentication failure|Failed [-/\w+]+) for(?: [iI](?:llegal|nvalid) user)?|[Ii
```

Nous vous conseillons d'activer **Fail2ban** pour **SSH**, mais aussi pour *postfix*, *apache* et *apache-noscript*. Cette dernière section vérifie les erreurs 404 sur les fichiers *.php* et *.asp*. Quand un site est bien codé, il ne devrait pas ressortir de telles erreurs, sauf si quelqu'un cherche une faille. Attention quand même si un jour pour changer toutes les URLs de votre site. Il y aura ainsi beaucoup d'erreur 404 pour les internautes provenant d'un moteur de recherche ou de leur marque-pages et ils seront bannis !

Vérifiez aussi la configuration de la section *[postfix]* car le fichier par défaut `/var/log/postfix.log` n'existe peut-être pas sur votre machine. Remplacez-le par `/var/log/mail.log` si c'est le cas.

Note : Nous vous déconseillons de remplir et d'activer le champ *destemail* si vous comptez utiliser *logwatch* comme indiqué plus tard dans le tutoriel. En effet *logwatch* vous enverra **déjà** un compte-rendu des adresses IPs bannies, et fera doublon avec les emails de **Fail2ban**.

Note : Si vous souhaitez quand même être averti par email, profitez-en pour ne décommentez que la troisième ligne *action* qui permet d'inclure dans l'email la provenance de l'IP et les lignes de log incriminés. N'oubliez pas d'installer l'utilitaire *whois* s'il n'est pas par défaut sur votre système : **apt-get install whois**.

Après modification de la configuration, n'oubliez pas de redémarrer **Fail2ban** : **/etc/init.d/fail2ban restart**.

10.3 Protection SYN/ACK

Pour se protéger des attaques SYN/ACK exécutez la commande : **echo 1 > /proc/sys/net/ipv4/tcp_syncookies**

11 Monitoring

Il est important de surveiller l'état de votre serveur et d'être prévenu si une anomalie survient, le plus souvent par e-mail.

11.1 Monit

Monit est une application permettant de surveiller l'état des services (notamment web, ftp, mail, mysql, ssh) par une interface web, et de notifier l'administrateur si nécessaire (trop grande charge cpu, redémarrage, indisponibilité...).

L'installation de Monit se réalise en quelques secondes grâce au paquet éponyme : **apt-get install monit**.

La configuration de Monit se fait en deux temps. Tout d'abord, autoriser le démarrage du service en éditant `/etc/default/monit` : **nano /etc/default/monit**.

Modifier l'option *startup* à *1*.

```
startup=1
```

Ensuite, éditer le fichier de configuration `/etc/monit/monitrc` contenant la description de tous les services à surveiller.

Voici un exemple complet. Celui-ci est relativement explicite et à adapter selon votre configuration.

set mailserver Indique le(s) serveur(s) de mail à utiliser pour l'envoi des notifications

set alert Indique les adresses destinataires

set httpd port Spécifie le port de connexion web. Vous pourrez ensuite vous connecter grâce à votre navigateur sur l'IP et le port correspondant (ex : http ://monserveur :1337)

allow monlogin :monpassword Spécifie le couple login/password pour l'accès web (à renseigner)

check device Va permettre de surveiller l'espace disque restant : il faut ici indiquer le bon chemin vers /dev/XXX correspondant à la partition à monitorer (ex : /dev/sda, /dev/md1... selon votre configuration)

```
# Config
set daemon 120
set logfile syslog facility log_daemon
set mailserver localhost
set mail-format {
    from: monit@$HOST
    subject: $HOST - Monit : $EVENT $SERVICE
    }
set alert root@monserveur
set httpd port 1337 and
   allow login:password

# Apache 2
check process httpd with pidfile /var/run/apache2.pid
group apache
start program = "/etc/init.d/apache2 start"
stop program = "/etc/init.d/apache2 stop"
if failed host 127.0.0.1 port 80
protocol http then restart
if 5 restarts within 5 cycles then timeout
if cpu is greater than 85% for 2 cycles then alert
if cpu > 90% for 5 cycles then restart
if children > 250 then restart

# MySQL
check process mysqld with pidfile /var/run/mysqld/mysqld.pid
group database
start program = "/etc/init.d/mysql start"
```

```
stop program = "/etc/init.d/mysql stop"
if failed host 127.0.0.1 port 3306 then restart
if 5 restarts within 5 cycles then timeout

# SSH
check process sshd with pidfile /var/run/sshd.pid
group ssh
start program "/etc/init.d/ssh start"
stop program "/etc/init.d/ssh stop"
if failed host 127.0.0.1 port 22 protocol ssh then restart
if 5 restarts within 5 cycles then timeout

# Postfix
check process postfix with pidfile /var/spool/postfix/pid/master.pid
group mail
start program = "/etc/init.d/postfix start"
stop  program = "/etc/init.d/postfix stop"
if failed port 25 protocol smtp then restart
if 5 restarts within 5 cycles then timeout

# Disk
check device sda1 with path /dev/sda1
if space usage > 85% then alert
group system
```

Vous êtes libre d'ajouter tous les services à monitorer sur votre machine (fail2ban...). La syntaxe est abordable et les exemples nombreux. Pour vérifier cette syntaxe, utilisez la commande : **/etc/init.d/monit syntax**.

Si aucun message d'erreur n'est indiqué, vous pourrez ensuite démarrer monit : **/etc/init.d/monit start**.

Vérifiez une nouvelle fois la bonne interprétation de la configuration grâce à **monit -v**.

11.2 Logwatch

Logwatch est par un démon pouvant analyser et résumer les logs générés par les autres services durant la journée pour en détecter d'éventuelles anomalies ou en tirer des statistiques. Il permet d'envoyer un e-mail récapitulatif quotidien à l'administrateur. Son installation est elle aussi très simple grâce à APT et au paquet éponyme : **apt-get install logwatch**.

La configuration par défaut suffit amplement, il suffit de modifier le destinataire dans le fichier `/usr/share/logwatch/default.conf/logwatch.conf`.

Modifiez l'option *MailTo* :

```
MailTo = root@monserveur
```

11.3 Rkhunter

rkhunter est un programme Unix qui permet de détecter les rootkits, portes dérobées et exploits. Pour cela, il compare le hash MD5 des fichiers importants avec les hash connus, qui sont accessibles à partir d'une base de données en ligne. Ainsi, il peut détecter les répertoires généralement utilisés par les rootkit, les permissions anormales, les fichiers cachés, les chaînes suspectes dans le kernel et peut effectuer des tests spécifiques à Linux et FreeBSD.

Vous pouvez l'installer grâce à : **apt-get install rkhunter**. Il a besoin de la commande **strings** qui est disponible en installant **binutils** : **apt-get install binutils**.

Il procédera à des détections journalières anti-rootkits et enverra des notifications par e-mail si nécessaire. Il est conseillé de l'installer très tôt car il calcule l'empreinte MD5 des programmes installés afin de détecter d'éventuels changements. Editez `/etc/default/rkhunter` pour indiquer l'adresse de notification et l'exécution journalière.

```
REPORT_EMAIL="root@monserveur"
CRON_DAILY_RUN="yes"
```

En cas de fausses détections positives sur des répertoires ou fichiers existants et sains, éditez `/etc/rkhunter.conf` pour les ajouter à la liste des éléments autorisés.

Pour lancer un test à n'importe quel moment il suffit de taper `rkhunter -c`. Décommentez aussi les liens suivante dans `/etc/rkhunter.conf` :

```
ALLOWHIDDENDIR=/dev/.udev
ALLOWHIDDENDIR=/dev/.static
```

Il vous reste à mettre à jour le logiciel : `rkhunter --update`.

Enfin, si vous faites des modifications (mises à jour ou installations) dans les dossiers contrôlés par **rkhunter** il faut exécuter la commande suivante : `rkhunter --propupd`. Elle permet de mettre à jour les tables de **rkhunter** afin qu'il ne vous envoie pas de mail alors que vous avez fait consciemment des modifications.

12 Apache

12.1 Installation

Pour installer Apache il suffit de faire : **apt-get install apache2**.

Les modules sont stockés dans */etc/apache2/mods-available/*. Si il en manque vous pouvez voir la liste des modules disponibles avec la commande **dpkg -l libapache2-mod-*** (un ii devant le nom signifie que le paquet est déjà installé).

Tous les fichiers de configuration Apache sont stockés dans */etc/apache2/* :

apache2.conf Configuration générale (ancien httpd.conf).

mods-available/ Modules disponibles.

mods-enabled/ Modules activés.

sites-available/ Sites disponibles.

sites-enabled/ Sites activés.

Afin d'activer les modules, utilisez **a2enmod** (Apache 2 Enable Module) suivi du nom du module. Par exemple, activons *mod_rewrite* pour permettre la réécriture des URLs : **a2enmod rewrite**. Afin d'activer un site dont la configuration est stockée dans *sites-available*, utilisez **a2ensite** (Apache 2 Enable Site). Pour désactiver un site, utilisez **a2dissite**. Pour désactiver un module, utilisez **a2dismod**.

12.2 Configuration & sécurité

Le fichier */etc/apache2/ports.conf* fait la liste des ports écoutés par apache, vous pouvez vérifier que vous avez bien les lignes suivantes :

```
Listen 80
<IfModule mod_ssl.c>
        Listen 443
</IfModule>
```

Passons enfin à la modification de */etc/apache2/apache2.conf*. Modifiez ou ajoutez les lignes suivantes :

```
ServerTokens Prod               # Mettre Prod pour eviter au serveur d'etre trop bavard quand a l
ServerSignature Off             # Idem
TimeOut 60                      # La valeur par defaut, 300, est bien trop grande
```

On modifie également la fin du fichier en ajoutant juste avant les lignes

```
# Include the virtual host configurations:
Include /etc/apache2/sites-enabled/
```

Le code suivant :

```
LogLevel info
ErrorLog /var/log/apache2/error.log
CustomLog /var/log/apache2/access.log combined

<Directory />
    Options None
    AllowOverride None
    Order Deny,Allow
    Deny from All
</Directory>
```

Le code compris dans *<Directory />* est général. Il interdit l'accès au site à quiconque, désactive toutes les options et empêche la prise en compte des fichiers *.htaccess* On réglera ces options au cas par cas pour chaque site.

L'hébergement virtuel par nom (une seule IP pour tous) est habituellement plus simple, car il vous suffit de configurer votre serveur DNS pour que chaque domaine pointe sur l'adresse IP dont vous disposez, et de configurer votre serveur Apache afin qu'il reconnaisse ces domaines. À noter que l'hébergement virtuel par nom ne peut pas être utilisé avec des serveurs sécurisés SSL à cause de la nature même du protocole SSL...

Commençons par modifier le fichier de configuration par défaut, celui qui est exécuté quand on accède au serveur via son adresse IP : **nano /etc/apache2/sites-available/default** :

```
NameVirtualHost  *:80
<VirtualHost *:80>
  DocumentRoot /var/www/apache2-default
  <Directory /var/www/apache2-default>
    Order allow,deny
    Allow from all
    php_admin_value open_basedir "/var/www/apache2-default/"
    php_admin_value error_log "/var/log/apache2/php.log"
  </Directory>
</VirtualHost>
```

Warning : Maintenant il est temps de créer la configuration pour notre premier site *www.monsiteweb.fr*. Rappelez-vous, c'est le nom de l'utilisateur qu'on a créé au début du tutoriel. Créons des sous-dossiers et attribuons leur les bon droits : **cd /home/monsiteweb-fr/ ; mkdir logs tmp sessions ; chown -R www-data :www-data logs tmp sessions**.

Créons maintenant le fichier de configuration de notre site : **nano /etc/apache2/sites-available/monsiteweb-fr** et mettons-y :

```
<VirtualHost *:80>
    ServerAdmin webmaster@monsiteweb.fr
    ServerName www.monsiteweb.fr
    ServerAlias monsiteweb.fr *.monsiteweb.fr

    DocumentRoot /home/monsiteweb-fr/www/
    <Directory /home/monsiteweb-fr/www/>
        Order allow,deny
        Allow from all
        php_admin_value open_basedir "/home/monsiteweb-fr/www/"
        php_admin_value error_log "/home/monsiteweb-fr/logs/error.php.monsiteweb-fr.log"
  php_admin_value upload_tmp_dir "/home/monsiteweb-fr/tmp/"
  php_admin_value session.save_path "/home/monsiteweb-fr/sessions/"
    </Directory>

    ErrorLog /home/monsiteweb-fr/logs/error.monsiteweb-fr.log
```

```
    CustomLog /home/monsiteweb-fr/logs/access.monsiteweb-fr.log combined
</VirtualHost>
```

Si on veut rajouter la prise en compte des *.htaccess* (même si ça implique une baisse des performances) il faut rajouter *AllowOverride All*. Pour activer des *Options*, comme la création de la liste de fichiers d'un dossier vide par exemple il faut mettre un + devant : *Options +Indexes*. On remarquera que les logs sont bien créé dans le dossier *logs/* du site web.

N'oubliez pas d'activer votre site Web : **a2ensite monsiteweb-fr**. Après tout changement de configuration, n'oubliez pas de recharger la configuration : **/etc/init.d/apache2 force-reload**. Mais il y a mieux encore ! Quand on veut redémarrer Apache je vous conseille : **apache2ctl graceful**. Cette commande vérifie d'abord la syntaxe du fichier de conf Apache, puis redémarre Apache après que toute les connexions ouvertes aient été fermées.

13 PHP

13.1 Installation

Pour installer PHP, la commande est : `apt-get install libapache2-mod-php5 php5 php5-common php5-mysql`.

Vous remarquerez qu'on installe les modules PHP un par un (ici *php5-mysql* seulement). Pour installer l'extension GD (exemple) `apt-get install php5-gd`.

13.2 Configuration & sécurité

La configuration de PHP est stockée dans le fichier */etc/php5/apache2/php.ini*

Je vous encourage à modifier (ajouter le cas échéant) les lignes suivantes :

```
allow_url_fopen = Off    # Rarement utile et source de problemes
magic_quotes_gpc = Off   # Source de probleme (ajout de slashs aux variables GPC)
memory_limit = 8M        # Si vous avez besoin de plus pour un site soit il est mal code soit ce s
post_max_size = 2M       # Avez-vous vraiment de si gros formulaire ? Si oui on reglera au cas par
short_open_tag=Off       # Pas de scripts qui commence par <? au lieu de <?php
register_globals = Off   # Les variables EGPC ne seront pas globales par defaut et c'est tant mieu
enable_dl = Off          # Inutile sauf cas particulier et piratage...
expose_php = Off         # Serveur moins bavard sur sa configuration
display_errors = Off     # On n'affiche pas les erreurs sur un serveur en production
log_errors = On          # Par contre il faut quand meme les enregistrer quelque part !
error_log = /var/log/apache2/php.log # Fichier de log des erreurs
```

Et si vous souhaitez faire de l'UTF-8 (très conseillé) :

```
mbstring.language=UTF-8
mbstring.internal_encoding=UTF-8
mbstring.http_input=UTF-8
mbstring.http_output=UTF-8
mbstring.detect_order=auto
```

Pour plus de sécurité, il est possible de désactiver des fonctions système. Ne le faites que si vous êtes sûr de leur non-emploi.

```
disable_functions = symlink,shell_exec,exec,proc_close,proc_open,popen,system,dl,passthru,escapesh
```

N'oubliez pas de redémarrer Apache !

14 MySQL

14.1 Installation

Pour installer MySQL la commande est : **apt-get install mysql-server mysql-client**.

14.2 Configuration & sécurité

Le fichier de configuration est `/etc/mysql/my.cnf`.

Les lignes à modifier à l'intérieur sont :

```
language = /usr/share/mysql/french # Choisir la langue par defaut pour les messages du serveur
#log_bin = /var/log/mysql/mysql-bin.log # Desactiver le log binaire
#expire_logs_days = 10 # Idem
#max_binlog_size = 100M # Idem
log_slow_queries = /var/log/mysql/mysql-slow.log # Mettre en log les requetes lentes
long_query_time = 2 # Duree (en secondes) a partir de laquelle une requete est consideree comme l

[client]
default-character-set = utf8 # Jeu de caracteres par defaut pour le client, si vous voulez de l'U

[mysqld]

default-character-set = utf8 # Jeu de caracteres par defaut pour le serveur, si vous voulez de l'
default-collation = utf8_general_ci # Collation du jeu de caracteres, si vous voulez de l'UTF-8
```

Et enfin, redémarrer le serveur MySQL : **/etc/init.d/mysql reload**.

Il est facile de mettre en place quelques règles simples pour sécuriser le serveur, grâce au script **mysql_secure_installation**. Et en répondant à quelques questions (le script demande le mot de passe root, il est vide la première fois, appuyez juste sur kbd :*Entrée*) :

Set root password ? Y Permet de modifier le mot de passe root (de mysql)

Remove anonymous users ? Y Retire les accès anonymes

Disallow root login remotely ? Y Retire l'accès root distant (recommandé)

Remove test database and access to it ? Y Retire la base test et ses accès

Reload privilege tables now ? Y Recharge les privilèges suite aux modifications

14.3 Utilisation

Vous pourrez vous connecter en ligne de commande grâce au client : **mysql -ulogin -p**. Pour les réfractaires, on verra plus tard comment utiliser un client graphique comme phpMyAdmin.

Commandes **shell** utiles :

mysqldump -ulogin -pmotdepasse nom_base –opt > fichier.sql Exporte la base nom_base vers fichier.sql

mysqldump -ulogin -pmotdepasse –all-databases –opt > fichier.sql Exporte toutes les bases vers fichier.sql

mysql -ulogin -pmotdepasse < fichier.sql Importe les instructions du fichier SQL

mysqladmin Toute une collection d'outils pour administrer le serveur

Commandes **MySQL** pour créer un utilisateur :

create database monsiteweb-fr ; Dans un premier temps il faut créer la base de données, nous choisissons le même nom que l'utilisateur

grant all privileges on monsiteweb-fr.* to 'monsiteweb-fr'@'localhost' identified by "motdepasse" ; Ensuite nous donnons les droits à l'utilisateur monsiteweb-fr sur la base monsiteweb-fr qui pour se connecter grâce à son motdepasse

flush privileges ; Enfin il faut recharger les droits de mysql

Référez-vous à la documentation (**man mysqldump**) pour plus de renseignements.

Voir aussi :

Ajouter un utilisateur : http ://dev.mysql.com/doc/refman/5.0/fr/adding-users.html

15 phpMyAdmin

15.1 Installation

Pour installer phpMyAdmin la commande est : **apt-get install phpmyadmin**.

15.2 Configuration & sécurité

Dans un premier temps il faut modifier le fichier de configuration qui est `/etc/phpmyadmin/config.inc.php`.

Il est conseillé de changer le type d'authentification de cookie en http :

```
$cfg['Servers'][$i]['auth_type'] = 'http'; // Authentication method (config, http or cookie based)
```

Dans un second temps il faut modifier le fichier `/etc/apache2/sites-available/default` et ajouter les lignes suivantes à la fin du VirtualHost :

```
Alias /phpmyadmin /usr/share/phpmyadmin

<Directory /usr/share/phpmyadmin>
  Options Indexes FollowSymLinks
  DirectoryIndex index.php
  Order allow,deny
  Allow from all
  <Files setup.php>
      Order allow,deny
      Deny from all
  </Files>
</Directory>
```

16 Sauvegardes

17 Conclusion

Voilà, si vous avez suivi notre tutoriel jusqu'à la fin vous avez un serveur LAMP efficace. Bien sûr, il y a toujours des points à améliorer, ce que nous ne manquerons pas de faire dans notre prochaine version.

Petit bonus, si vous souhaitez pouvoir rajouter un nouveau site Web facilement, nous vous conseillons d'utiliser le script prévu spécialement à cet effet :

```
#!/bin/bash
#############################################################
##              Script d'ajout d'un site web               ##
##           Par Thomas Bassetto et Julien Molina          ##
#############################################################
execute ( ) {
  eval "$@"
  if [ $? -gt 0 ]; then
    echo "La commande $@ a échoué."
    exit 1
```

```
  fi;
}
exitOnEmpty ( ) {
  if [ -z "$@" ]; then
    echo "Vous devez rentrer une valeur !"
    exit 1
  fi;
}

clear
echo -n "Nom de domaine (sans le www) : "
read NDD
exitOnEmpty "$NDD"

echo -n "Nom de l'utilisateur SFTP : "
read NAME
exitOnEmpty "$NAME"

echo -n "Adresse email : "
read EMAIL
exitOnEmpty "$EMAIL"

echo -n "Mot de passe MySQL (celui système sera demandé plus tard) : "
read MDPMYSQL
exitOnEmpty "$MDPMYSQL"

echo "############################################################"
echo "##                  Demarrage de scponly                  ##"
echo "############################################################"
execute "cd /usr/share/doc/scponly/setup_chroot/; ./setup_chroot.sh /home/$NAME $NAME www;cd -"
echo "############################################################"
echo "##               Fin de l'execution de scponly            ##"
echo "############################################################"
echo ""
echo "Execution de scponly                                  [OK]"

execute "mkdir -p /home/$NAME/dev"
execute "mknod -m 666 /home/$NAME/dev/null c 1 3"
echo "Creation du dev/null                                  [OK]"

execute "usermod -aG sitesweb $NAME"
echo "Ajout de l'utilisateur au groupe sitesweb             [OK]"

execute "mkdir /home/$NAME/logs /home/$NAME/tmp /home/$NAME/sessions"
execute "chown -R www-data:www-data /home/$NAME/logs /home/$NAME/tmp /home/$NAME/sessions"
echo "Creation des dossiers logs tmp et sessions            [OK]"

echo '<VirtualHost *:80>

        ServerAdmin '$EMAIL'
        ServerName www.'$NDD'
        ServerAlias '$NDD' *.'$NDD'

        DocumentRoot /home/'$NAME'/www/

        <Directory /home/'$NAME'/www/>
                Order allow,deny
                Allow from all
                php_admin_value open_basedir /home/'$NAME'/www/
                php_admin_value error_log /home/'$NAME'/logs/error.php.'$NAME'.log
                php_admin_value upload_tmp_dir "/home/'$NAME'/tmp/"
                php_admin_value session.save_path "/home/'$NAME'/sessions/"
```

```
        </Directory>
        ErrorLog /home/'$NAME'/logs/error.'$NAME'.log
        CustomLog /home/'$NAME'/logs/access.'$NAME'.log combined
</VirtualHost>' > /etc/apache2/sites-available/$NDD
echo "Creation du vhost                                          [OK]"

execute "a2ensite $NDD > /dev/null 2>&1"
echo "Activation du site                                         [OK]"

execute "apache2ctl graceful > /dev/null 2>&1"
echo "Redemarrage d'Apache                                       [OK]"

echo "Entrez votre mot de passe root pour MySQL :"
execute "mysql -u root -p --exec=\"CREATE DATABASE $NAME;GRANT ALL PRIVILEGES ON $NAME.* TO '$NAM
echo "Creation de l'utilisateur MySQL                            [OK]"
```

Note : Pour l'utiliser, enregistrer le dans un fichier `creeSite.sh` par exemple, rajouter les droits d'éxecution, `chmod +x creeSite.sh`, et exécuter le en étant **root**.

En superbonus, un script pour désactiver l'accès à MySQL, l'accès SFTP et l'accès HTTP à un utilisateur/site :

```
#!/bin/bash
############################################################
##           Script de désactivation d'un site web        ##
##           Par Thomas Bassetto et Julien Molina         ##
############################################################
execute ( ) {
  eval "$@"
  if [ $? -gt 0 ]; then
    echo "La commande $@ a échoué."
    exit 1
  fi;
}
exitOnEmpty ( ) {
  if [ -z "$@" ]; then
    echo "Vous devez rentrer une valeur !"
    exit 1
  fi;
}

clear
echo -n "Nom de domaine (sans le www) : "
read NDD
exitOnEmpty "$NDD"

echo -n "Nom de l'utilisateur SFTP : "
read NAME
exitOnEmpty "$NAME"

execute "usermod -L $NAME > /dev/null 2>&1"
echo "Utilisateur $NAME bloque                                   [OK]"

execute "a2dissite $NDD > /dev/null 2>&1"
echo "Desactivation du site                                      [OK]"

execute "apache2ctl graceful > /dev/null 2>&1"
echo "Redemarrage d'Apache                                       [OK]"

echo "Entrez votre mot de passe root pour MySQL :"
execute "mysql -u root -p --exec=\"REVOKE ALL PRIVILEGES,GRANT OPTION FROM '$NAME'@'localhost';FL
echo "Desactivation de l'utilisateur MySQL                       [OK]"
```

Et un dernier pour supprimer toutes traces d'un site/compte :

```bash
#!/bin/bash
############################################################
##           Script de suppression d'un site web          ##
##           Par Thomas Bassetto et Julien Molina         ##
############################################################
execute ( ) {
  eval "$@"
  if [ $? -gt 0 ]; then
    echo "La commande $@ a échoué."
    exit 1
  fi;
}
exitOnEmpty ( ) {
  if [ -z "$@" ]; then
    echo "Vous devez rentrer une valeur !"
    exit 1
  fi;
}

clear
echo -n "Nom de domaine (sans le www) : "
read NDD
exitOnEmpty "$NDD"

echo -n "Nom de l'utilisateur SFTP : "
read NAME
exitOnEmpty "$NAME"

execute "deluser --remove-home $NAME > /dev/null 2>&1"
echo "Fichiers de $NAME supprime                                    [OK]"

#execute "delgroup $NAME > /dev/null 2>&1"
#echo "Groupe $NAME supprime                                         [OK]"

execute "rm -f /etc/apache2/sites-enabled/$NDD > /dev/null 2>&1;rm -f /etc/apache2/sites-available
echo "Suppression du site dans Apache                               [OK]"

execute "apache2ctl graceful > /dev/null 2>&1"
echo "Redemarrage d'Apache                                          [OK]"

echo "Entrez votre mot de passe root pour MySQL :"
execute "mysql -u root -p --exec=\"DROP DATABASE $NAME;DROP USER $NAME;FLUSH PRIVILEGES;\""
echo "Suppression de l'utilisateur MySQL et de sa base              [OK]"
```

18 Rermerciements

– Sébastien Bonnegent, pour la relecture complète et le soutien lors de la première rédaction.
– Xavier Belanger, pour la relecture complète.
– Pleins d'autres personnes de LinuxFR.org : farvardin, geb, Tanguy Ortolo et ultimat.

19 Notes de versions

19.1 Version 1.0 (1 août 2009)

– Première version publique “stable”.

20 Index et tables

– *Index*
– *Page de recherche*