BARRE Kévin	BTS SIO	Charles de Foucauld Brest

Mise en place d'un serveur Mail de A à Z (Rédaction pas terminée)

Voici le premier tuto complet et intéressant à mon goût que vous pourrez trouver sur mon site web. Nous allons aujourd'hui voir la procédure d'installation d'un serveur de courriel dans sa globalité. Ce tutoriel est destiné à des personnes ayant déjà une bonne expérience en administration des systèmes Linux. Il est important de savoir ce que vous faite et ne pas se contenter de faire des copier, coller… Surtout si vous souhaitez utiliser ce serveur de mail dans la vie de tous les jours.

A la fin de ce tuto, vous serez capable d'envoyer et de recevoir des mails à partir de la webmail Rainloop et même à partir de n'importe quelle client mail (Thunderbird, Outlook, Claws Mail, Evolution…etc)

J'effectue l'installation du serveur sur un VPS avec une version de Debian 8 Jessie. Mais ce tuto est également compatible avec Debian 7 Wheezy et surement avec Debian 9. Mais je n'en suis pas certain.

Pour ce faire, vous aurez besoin d'un **VPS sous Debian 7 ou 8**. Vous pouvez en louer à des prix vraiment pas chers sur des sites comme ovh.com ou même chez pulseheberg.com. Je vous conseille tout de même les services d'OVH qui sont aujourd'hui la référence en matière d'hébergement.

Il vous faudra également un **nom de domaine** avec la possibilité d'édité la **zone DNS** afin d'y enregistrer les champs en tout genre que nous aurons besoin par la suite de ce tutoriel. Encore une fois, je vous conseille ovh.com.

Une fois édité votre zone DNS devra ressembler à cela :

NOM d'HOTE	CLASSE	TYPE D'ENREGISTREMENT	VALEUR
@	IN	A	ipv4 du serveur
hostname	IN	A	ipv4 du serveur
mail	IN	A	ipv4 du serveur
postfixadmin	IN	CNAME	hostname
rainloop	IN	CNAME	hostname
@	IN	MX	10 mail.votredomaine.fr.

Les enregistrements suivants ne sont pas forcément obligatoires.

NOM d'HOTE	CLASSE	TYPE D'ENREGISTREMENT	VALEUR
smtp	IN	CNAME	hostname
imap	IN	CNAME	hostname

```
Les enregistrements suivants sont facultatifs également mais permettent de
        ne pas se retrouver dans des SPAM d'un bote Gmail par exemple. Pour
        plus d'info sur ces champs RDV plus bas.

@                       IN                  TXT                     "v=spf1 a mx
        ip4:IPv4 DU SERVEUR ~all"

mail._domainkey      IN                  TXT                     "k=rsa; p=CLE
        PUBLIQUE DKIM"

_dmarc                  IN                  TXT                     "v=DMARC1;
        p=reject;  rua=mailto:postmaster@votredomaine.fr;  ruf=mailto:admin@
        votredomaine.fr; fo=0; adkim=s; aspf=s; pct=100; rf=afrf; sp=reject"

_domainkey              IN                  TXT                     "o=-;
        r=postmaster@votredomaine.fr "
```

Quelques explications importantes côté logiciel :

- **Postfix** est un serveur libre de messagerie électronique sous Linux. Il est destiné à la transmission des courriels électroniques : c'est donc un MTA « Mail Transfer Agent ».

- **Postfixadmin** est une interface web simplifiant la configuration de Postfix. Il permet de gérer simplement vos domaines, vos adresses virtuelles ainsi que vos alias.

- **Dovecot** est un serveur POP et IMAP sous Unix mettant en avant la sécurité. Il permet entre autres de gérer les formats de boîte de messagerie mbox et Maildir, méthodes d'identifications offertes sont CRAM-MD5, DIGEST-MD5, APOP, NTLM de Microsoft, GSSAPI (Kerberos v5), LDAP, Base de données RPA, LOGIN, à l'aide d'un compte anonyme, OTP et SKEY.

- **SpamAssassin** est un logiciel libre développé par la Apache Software Foundation : auteur du très célèbre serveur Web Apache HTTP Server. Le but de ce logiciel est de filtrer le trafic des courriels pour éradiquer les courriels reconnus comme SPAM ou pourriels en français.

- **ClamAV**, est un logiciel antivirus pour systèmes UNIX. Il est généralement utilisé avec les serveurs de courriels pour filtrer les courriels. ClamAV est l'un des rares antivirus disponible sous GNU/Linux et MacOS.

Définition de termes important :

- **SPF Sender Policy Framework** est une technique qui consiste à authentifier l'émetteur d'un mail. Le principe est simple à comprendre (« MAIL FROM : » de l'enveloppe du message SMTP

et non le champ « From : » de l'entête) une requête DNS de type TXT est effectuée sur le domaine en question pour connaître la liste des serveurs de messagerie autorisés à émettre des e-mails et pour la comparer avec l'adresse IP du serveur émetteur du message.

- **DKIM Domain Keys Identified Mail** regroupe deux technologies : **DomainKeys** de Yahoo et **Identified Internet Mail** de Cisco. DKIM est également une méthode d'authentification comme **SPF**. **DKIM,** lui, permet d'ajouter une signature à l'en-tête du mail de chaque message envoyer. La vérification de la signature se fait via une clef cryptée située dans un enregistrement DNS. DKIM permet donc de vérifier si un message à été intercepté ou modifié lors de son transport entre les différents serveurs SMTP destinataire.

- **DMARC Domain-based Message Authentication, Reporting and Conformance** est une technologie travaillant avec **SPF** et **DMARC** et les unissant pour les rendres plus intelligent en quelque sorte. Il va donc avoir deux fonctions :

    - Il va indiquer au Web Mail et au FAI ce qu'ils doivent faire d'un message ou l'authentification à échouer. (Message en SAPM, le transmettre quand même, le supprimer).
    - DKIM permet également à l'expéditeur d'être averti si l'authentification échoue.

- **TLS Transport Layer Security** est un protocole de sécurisation des échanges de données sur Internet. TLS fonctionne suivant un mode client-serveur. Il permet de satisfaire aux objectifs de sécurité suivants :

    - L'authentification du serveur ;
    - La confidentialité des données échangées (ou session chiffrée) ;
    - L'intégrité des données échangées ;
    - L'authentification du client (généralement c'est le serveur qui gère cet aspect) ;

- **Enregistrement Mail eXchanger** (champs MX) est un type d'enregistrement du Domain Name System qui associe un nom de domaine à un serveur de messagerie électronique associé à son numéro de préférence. Ces enregistrements permettent de déterminer vers quel serveur un courrier électronique doit être acheminé lorsque le protocole SMTP est utilisé.

1°) On va maintenant commencer par l'installation de LEMP Linux, Nginx, MySQL, PHP

Pour Debian 8: Si lors de l'installation de LEMP "apt" ne parvient pas à trouver le paquet en question:

- Verifiez le fichier sources.list

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

- Ce fichier doit contenir les lignes suivantes.

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

- Une fois la modification terminée faites un "**CTRL + X**" pour quiter et un "**o**" ou "**y**" pour confirmer l'enregistrement des modifications.

Installation de Ngnix :

L'installation du serveur web est très simple, quelques commandes suffisent.

- On commence par mettre à **jour** la liste des fichiers disponibles dans les dépôts APT présents dans le fichier de configuration /etc/apt/sources.list

```
sudo apt-get update
```

- On installe le paquet correspondant tout en validant lors de l'installation en appuyant sur « Entrée »

```
apt-get install nginx
```

- Une fois l'installation terminé vous pouvez vérifier si tout est fonctionnel en vous rendant dans votre navigateur préféré et en vous connectant à votre serveur http://votre-serveur-ou-votre-ip. Si tout est opérationnel ceci apparait.

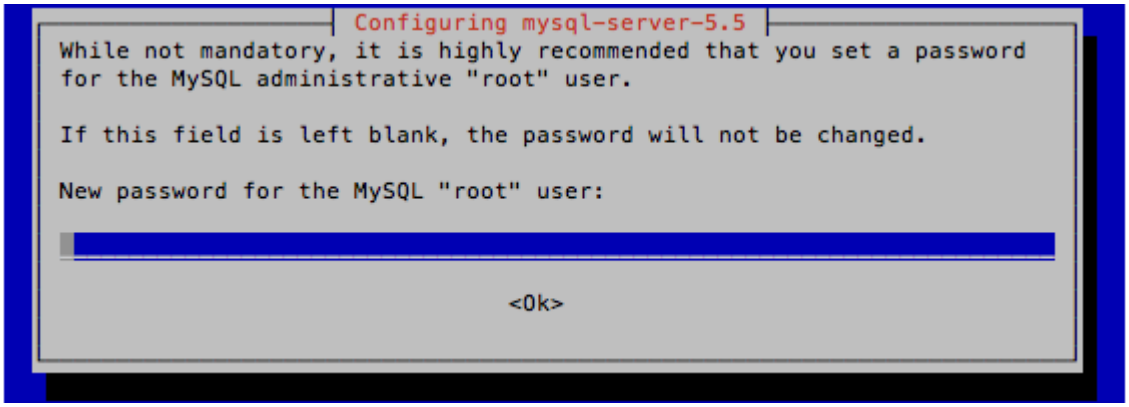
Installation de mysql :

- Maintenant que le serveur web est installé, vous devez installer mysql qui est une base de données qui va nous servir ici à stocker les informations des domaines.

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

- Lors de l'installation de mysql-server, le script d'installation va vous demandez de renseigner un mot de passe qui sera celui du compte root de mysql.

- Je vous conseille d'utiliser un mot de passe compliqué composé de chiffres, de lettres, et de caractères spéciaux.



- L'installation est maintenant terminée. Mais ce n'est pas fini, il faut maintenant modifier certains paramètres de sécurité.

```
sudo mysql_secure_installation
```

Il vous sera demandé d'entrer le mot de passe que vous avez défini pour le compte root MySQL. Ensuite, il vous sera demandé si vous voulez changer ce mot de passe. Si vous êtes satisfait de votre mot de passe actuel, entrez **N** pour **non** :

Pour le reste des questions que le script demande, vous devez appuyer sur **Y** ou **O**, suivi de la touche ENTRÉE à chaque demandes. Cela supprimera certains utilisateurs anonymes et la base de données de test, désactivera les connexions root distantes et chargera ces nouvelles règles afin que MySQL respecte immédiatement les modifications que vous avez apporté.

Votre base de données est maintenant sécurisée est prête à l'emploi.

Installation de php :

Nous avons maintenant Nginx installé pour servir nos pages et MySQL installé pour stocker et gérer nos données. Cependant, nous n'avons toujours rien qui puisse générer du contenu dynamique. C'est là que PHP intervient.

Puisque Nginx ne contient pas de traitement PHP natif comme d'autres serveurs web, nous devons installer **fpm** qui signifie "**gestionnaire de processus fastCGI**". Nous dirons à Nginx de transmettre les requêtes PHP à ce logiciel pour traitement. Nous allons également installer un paquet d'assistance supplémentaire qui permettra à PHP de communiquer avec notre backend de base de données MySQL. L'installation tirera dans les fichiers de base PHP nécessaires pour que cela fonctionne.

- Ensuite, installez les modules php5-fpm et php5-mysql:

```
sudo apt-get install php5-fpm php5-mysql
```

- Les composants php maintenant installés, nous allons modifier la ligne **cgi.fix_pathinfo** du fichier situé **/etc/php5/fpm/php.ini**

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

C'est un paramètre extrêmement peu sûr car il indique à PHP de tenter d'exécuter le fichier le plus proche qu'il puisse trouver si le fichier PHP demandé est introuvable. Cela permettrait essentiellement aux utilisateurs de créer des requêtes PHP d'une manière à leur permettre d'exécuter des scripts qu'ils ne devraient pas être autorisés à exécuter.

- Nous allons changer ces deux conditions en décommentant la ligne et en la mettant à "0" comme ceci :

```
cgi.fix_pathinfo=0
```

Configuration de Nginx pour utiliser PHP :

Maintenant, nous avons tous les composants requis installés. Le seul changement de configuration que nous devons effectuer est de dire à Nginx d'utiliser notre PHP pour le contenu dynamique.

Nous faisons cela au niveau du bloc de serveur (les blocs de serveur sont similaires aux hôtes virtuels d'Apache). Ouvrez le fichier de configuration du bloc de serveur Nginx par défaut en tapant :

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

Par défaut ce fichier ressemble à cela :

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    root /var/www/html;
    index index.html index.htm index.nginx-debian.html;

    server_name _;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

Nous devons apporter quelques modifications à ce fichier pour notre site.

Pour commencer, nous devons ajouter **index.php** comme première valeur de notre directive index afin que les fichiers nommés **index.php** soient servis (si disponible) lorsqu'un répertoire est demandé.

- Nous pouvons modifier la directive **server_name** pour pointer vers le nom de domaine ou l'adresse IP publique de notre serveur.

- Pour le traitement PHP réel, nous avons juste besoin de décommenter un segment du fichier qui gère les demandes PHP. Ce sera le bloc d'emplacement **~ \ .php $** location, l'extrait de **fastcgi-php.conf** inclus, et le socket associé à php-fpm.

- Nous allons également décommenter le bloc de localisation traitant des fichiers **.htaccess**. Nginx ne traite pas ces fichiers. Si l'un de ces fichiers trouve son chemin dans la racine du document, il ne doit pas être accessible aux visiteurs.

Votre fichier **/etc/nginx/sites-available/default** doit ressembler à cela après son édition**.**

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    root /var/www/html;
    index index.php index.html index.htm index.nginx-debian.html;

    server_name your_server_ip;

    location / {
        try_files $uri $uri/ =404;
    }

    location ~ \.php$ {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/var/run/php5-fpm.sock;
    }

    location ~ /\.ht {
        deny all;
    }
}
```

Afin de vérifier si votre fichier de configuration est Ok effectuez la commande suivante :

```
nginx -t
```

Si des erreurs sont signalées, revenez en arrière et revérifiez votre fichier avant de continuer.

- Il faut maintenant redémarrer nginx pour que la nouvelle configuration soit prise en compte :

```
systemctl reload nginx
ou
```

```
service nginx reload
```

Création du fichier PHP pour tester la configuration

Votre pile LEMP devrait maintenant être complètement configurée. Nous pouvons le tester pour valider que Nginx peut correctement transférer les fichiers. **php** à notre a PHP.

- Nous pouvons le faire en créant un fichier PHP de test dans le répertoire racine web **/var/www/html**. Créer donc un nouveau fichier appelé **info.php**.

```
nano /var/www/html/info.php
```

- Tapez ou collez les lignes suivantes dans le nouveau fichier. C'est un code PHP valide qui retournera des informations sur notre serveur :

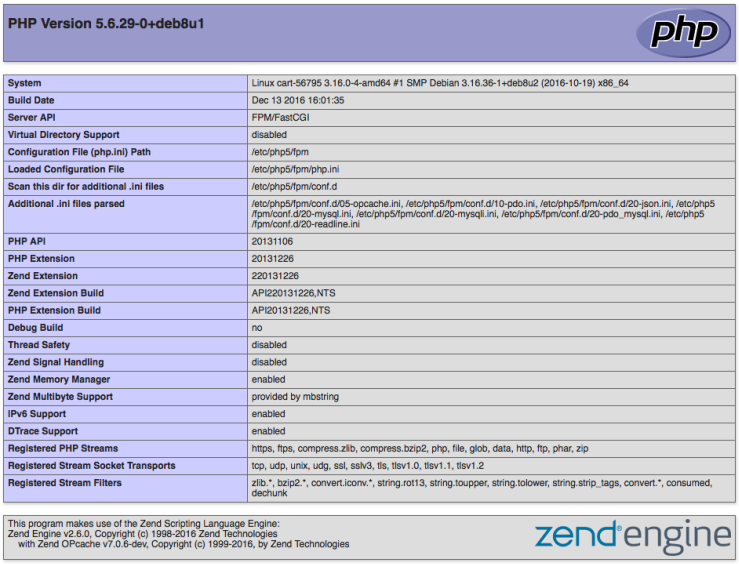
```
<?php
  phpinfo();
?>
```

- Lorsque vous avez terminé, enregistrez et fermez le fichier **CTRL + X** .

- Vous pouvez maintenant visiter cette page dans votre navigateur Web en accédant au nom de domaine de votre serveur ou à l'adresse IP publique suivie de /info.php.

```
http://domaine-ou-ip-serveur/info.php
```

Vous devriez voir une page web qui a été générée par PHP avec des informations sur votre serveur.



PHP Version 5.6.29-0+deb8u1

System	Linux cart-56795 3.16.0-4-amd64 #1 SMP Debian 3.16.36-1+deb8u2 (2016-10-19) x86_64
Build Date	Dec 13 2016 16:01:35
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/fpm
Loaded Configuration File	/etc/php5/fpm/php.ini
Scan this dir for additional .ini files	/etc/php5/fpm/conf.d
Additional .ini files parsed	/etc/php5/fpm/conf.d/05-opcache.ini, /etc/php5/fpm/conf.d/10-pdo.ini, /etc/php5/fpm/conf.d/20-json.ini, /etc/php5/fpm/conf.d/20-mysql.ini, /etc/php5/fpm/conf.d/20-mysqli.ini, /etc/php5/fpm/conf.d/20-pdo_mysql.ini, /etc/php5/fpm/conf.d/20-readline.ini
PHP API	20131106
PHP Extension	20131226
Zend Extension	220131226
Zend Extension Build	API220131226,NTS
PHP Extension Build	API20131226,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	enabled
Registered PHP Streams	https, ftps, compress.zlib, compress.bzip2, php, file, glob, data, http, ftp, phar, zip
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, sslv3, tls, tlsv1.0, tlsv1.1, tlsv1.2
Registered Stream Filters	zlib.*, bzip2.*, convert.iconv.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk

This program makes use of the Zend Scripting Language Engine:
Zend Engine v2.6.0, Copyright (c) 1998-2016 Zend Technologies
with Zend OPcache v7.0.6-dev, Copyright (c) 1999-2016, by Zend Technologies

Si vous voyez une page qui ressemble à ceci, vous avez réussi le traitement PHP avec Nginx.

Après avoir vérifié que Nginx restitue correctement la page, il est préférable de supprimer le fichier que vous avez créé car il peut effectivement donner des indications sur votre configuration à des utilisateurs non autorisés.
Une fois la pile LEMP correctement installée, nous allons pouvoir passer aux choses sérieuses.

2°) Configuration du nom d'hôte, du FQDN de la machine.

Le nom d'hôte est le nom de la machine. Celui-ci permet d'identifier un équipement de manière unique au sein d'un réseau. Il doit être configuré afin d'être le même que celui que vous allez renseigner du côté de votre **ZONE DNS**. Ce **HostName** peut contenir des chiffres et des lettres mais en aucun cas des espaces ou des points.

- Pour changer votre nom d'hôte, c'est simple : il suffit de modifier le fichier **/etc/hostname.**

```
nano /etc/hostname
```

Modifiez ce fichier et entrez le nom de machine que vous voulez. Une fois fait on pense à quitter **CTRL + X** et enregistrer **O** ou **Y** et **Entrée**.

- Ou alors, une autre méthode qui marche bien aussi :

```
echo "hostname" > /etc/hostname
```

- Remplacez hostname par le nom de votre machine.

Le FQDN (Fully Qualified Domain Name) lui aussi doit être également configurer (sur le serveur mais également dans la **ZONE DNS**, il s'agit du nom complet de la machine suivit de votre nom de domaine il va permettre l'accessibilité de la machine sur internet.

- Pour modifier le FQDN, il faut éditer le fichier **/etc/hosts.**

```
nano /etc/hosts
```

- Remplacez le contenu du fichier par ceci :

```
127.0.0.1 localhost.localdomain localhost
IP DU SERVEUR hostname.votredomaine.fr hostname
```

- Une fois fait, on pense à quitter **CTRL + X** et enregistrer **O** ou **Y** et **Entrée**.
- Pour bien prendre en compte les modifications, pensez à redémarrer le serveur via la commande.

```
reboot
```

- Puis, une fois le serveur rebooté faites la commande :

```
hostname -f
ou
hostname --fqdn
```

- Le Shell doit vous retourner :

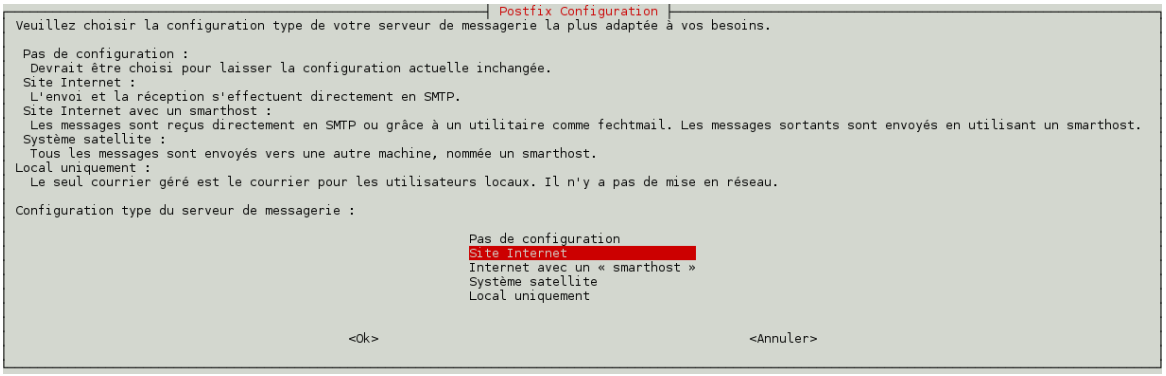
```
votre-hostname.votredemaine.fr
```

3°) Installation de Postfix

- On installe maintenant le **MTA** présenté précédemment ainsi que sa dépendance MySql. Car oui, les domaines utilisateurs seront stockés dans une base de données.

```
apt-get install postfix postfix-mysql
```

- Lors du processus d'installation de Postfix, vous devez choisir le type du serveur de messagerie : choisissez **Site Internet** pour utiliser SMTP.



- Par la suite, on vous demande de saisir le nom de votre domaine : entrez alors le **FQDN** de votre serveur.

4°) Création de la base de données

- Les lignes commençant par # sont simplement des commentaires. Ne pas les prendre en compte lors de vos copier-coller.

```
# Connexion au serveur MySQL en tant que root
mysql -u root -p
# Entrez votre mot de passe MySql que l'on ç configurer lors de l'installation.

# Création de la base de données "postfix"
mysql> CREATE database postfix;

# Création de l'utilisateur "postfix" et ajout des permissions
mysql> CREATE USER 'postfix'@'localhost' IDENTIFIED BY 'MOT DE PASSE';
mysql> GRANT USAGE ON *.* TO 'postfix'@'localhost';
mysql> GRANT ALL PRIVILEGES ON postfix.* TO 'postfix'@'localhost';

# On quitte la console MySQL
mysql> exit
```

- La création de l'utilisateur peut également être faite par le billet d'interface web comme phpmyadmin ou même adminer.

5°) Installation et configuration de PostfixAdmin

```
# On se déplace dans le dossier /var/www
cd /var/www

# On télécharge l'archive contenant PostfixAdmin

wget http://downloads.sourceforge.net/project/postfixadmin/postfixadmin/postfixadmin-2.92/postfixadmin-2.92.tar.gz

# On extrait le contenue de l'archive

tar -xzf postfixadmin-2.92.tar.gz

# On renomme le dossier postfixadmin-2.92
mv postfixadmin-2.92 postfixadmin

# On supprime l'archive téléchargée
rm -rf postfixadmin-2.92.tar.gz

# On change le propriétaire du dossier postfixadmin
chown -R www-data:www-data postfixadmin

# Et enfin on installe le paquet php5-imap
apt-get install php5-imap
```

- La configuration de Postfixadmin est à retrouver ici **/var/www/postfixadmin/config.inc.php**

```
nano /var/www/postfixadmin/config.inc.php
```

Recherchez et modifiez les paramètres suivants :

```
$CONF['configured'] = true;
$CONF['default_language'] = 'fr';
$CONF['database_type'] = 'mysqli';
$CONF['database_host'] = 'localhost';
$CONF['database_user'] = 'postfix';
$CONF['database_password'] = 'MOT DE PASSE UTILISATEUR postfix';
$CONF['database_name'] = 'postfix';
$CONF['admin_email'] = 'admin@domaine.fr;
$CONF['domain_path'] = 'YES';
$CONF['domain_in_mailbox'] = 'NO';
$CONF['fetchmail'] = 'NO';
```

- Enregistrez et quittez l'éditeur.
- Ensuite nous allons créer un virtual host sur Nginx afin d'y accéder via une URL de type http://postfixadmin.votredomaine.fr
- Pour ce faire, on va éditer le fichier **/etc/nginx/sites-enabled/postfixadmin.conf**

```
nano /etc/nginx/sites-enabled/postfixadmin.conf
```

- Pensez bien à changer la ligne server_name par vos propres paramètres.

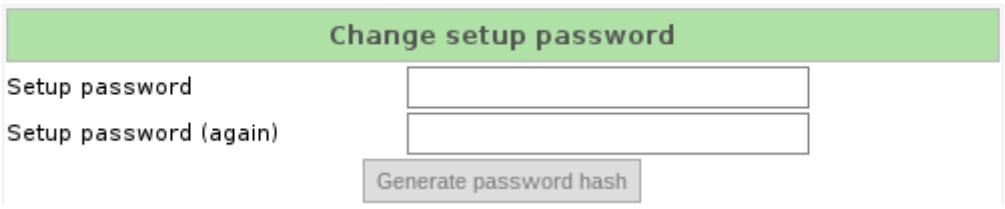
```
server {
   listen 80;
   server_name     postfixadmin.votredomaine.fr;
   root            /var/www/postfixadmin;
   index           index.php;
   charset         utf-8;

   location / {
      try_files $uri $uri/ index.php;
   }

   location ~* \.php$ {
        include       /etc/nginx/fastcgi_params;
        fastcgi_pass  unix:/var/run/php5-fpm.sock;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
   }
}
```

- Afin de prendre en compte les modifications, redémarrer Nginx.

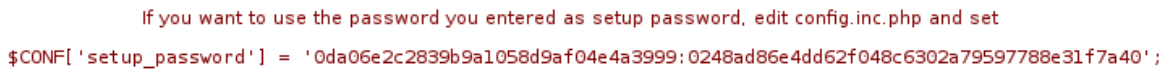
```
service nginx restart
```

- Vous pouvez maintenant accéder à l'installation de PostfixAdmin via d'adresse : http://postfixadmin.votredomaine.fr/setup.php
- Le script d'installation doit normalement créer les tables de Postfix prochainement utilisé. Si vous avez des messages d'erreur, vérifiez que les étapes précédentes ont bien été respectées. Si vous avez encore des problèmes, essayer de googleliser le message d'erreur.

- Tout en bas de la page, vous devez maintenant créer un mot de passe pour le setup d'installation. Préférez un mot de passe complexe.



- Cliquez ensuite sur **Generate pasword hash.** Copiez et garde ce hash.



- Rendez-vous ensuite dans le fichier **/var/www/postfixadmin/config.inc.php** avec le hash en presse-papier.

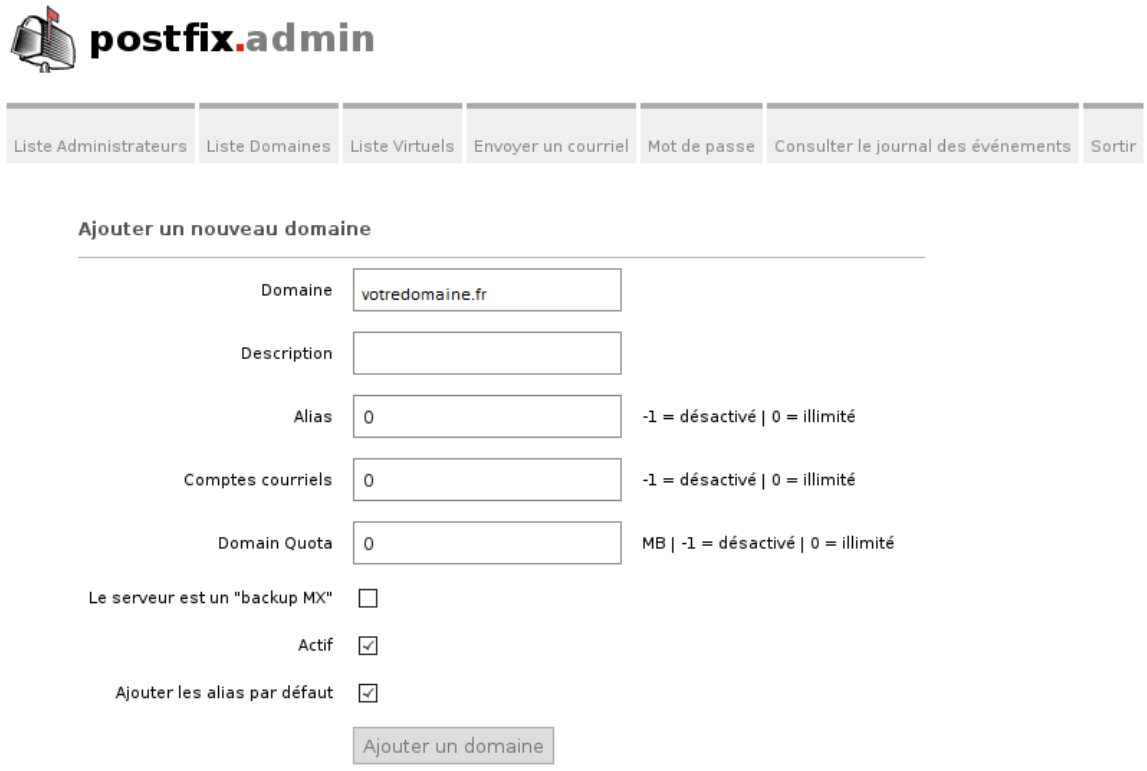
```
nano /var/www/postfixadmin/config.inc.php
```

- Trouvez et modifier la ligne **$CONF['setup_password'] = 'HASH';** par la valeur de votre hash généré précédemment. Enregistrez et quitter.

- Retournez sur votre navigateur web et maintenant créer un compte Administrateur admin@votredomaine.fr avec comme Setup password : le mot de passe créé juste avant (pas le HASH). Et comme mot de passe de ce compte utiliser un autre code complexe également mais différent du setup password, pour plus de sécurité.

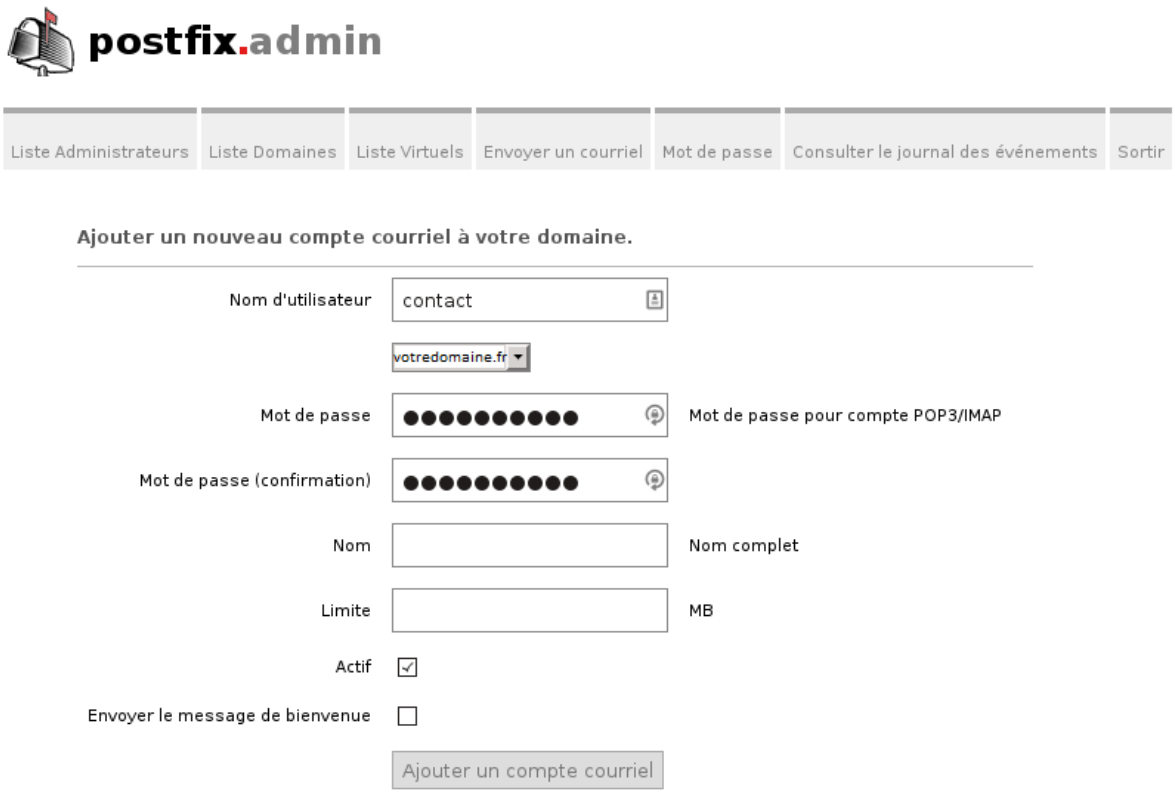
- Le compte administrateur configuré, vous pouvez vous identifier via l'adresse http://postfixadmin.votredomaine.fr/login.php

- Une fois connecté, vous allez créer un domaine virtuel : **votredomaine.fr**. Pensez bien à activer le domaine en cochant la case **Actif**. Vous pouvez également limiter le nombre d'utilisateur qui va pouvoir être créé sur ce domaine et géré le nombre d'alias…



- Maintenant, créez vos utilisateurs que vous voulez. Ils auront donc comme adresse mail : **nomutilisateur@votredomaine.fr**.

Créer l'utilisateur contact@votredomaine.fr et admin@votredomaine.fr



Pour la partie configuration de PostfixAdmin c'est terminé. Mais on est loin de la fin du tuto.

6°) Configuration de Postfix

La configuration de Postfix se trouve dans **/etc/postfix/main.cf.** Nous allons modifier ce fichier afin que Postfix prenne en charge les connexions SMTP et également l'envoie de mails via les utilisateurs créer précédemment sur PostfixAdmin.

- Pensez à faire une sauvegarde du fichier de conf original de Postfix.

```
cp /etc/postfix/main.cf /etc/postfix/main.cf.old
```

- Editez le fichier **/etc/postfix/main.cf**

```
nano /etc/postfix/main.cf
```

Les lignes commençant par « **#** » sont des commentaires. Elles ne sont pas utiles au programme, mais servent simplement à vous permettre de mieux comprendre ce que l'on fait. Les lignes commençant par « **#** » peuvent également être des options de configuration que nous ne souhaitons pas prendre en compte.

```
# Règles sur les adresses de destination
# permit_sasl_authenticated : Accepter la connexion lorsque le client est authentifié
# reject_non_fqdn_recipient : Refuser les adresses de destinations invalides (non FQDN)
smtpd_recipient_restrictions =
     permit_mynetworks,
     permit_sasl_authenticated,
     reject_non_fqdn_recipient,
     reject_unauth_destination,
     reject_unknown_recipient_domain,
     reject_rbl_client zen.spamhaus.org

# Règles sur l'échange HELO qui survient avant la connexion
# reject_invalid_helo_hostname : Refuser les échanges HELO invalides
# reject_non_fqdn_helo_hostname : Refuser les noms d'hôte invalides (non FQDN)
# reject_unknown_helo_hostname : Refuser les noms d'hôte qui n'ont pas de champ DNS A ou MX
dans leurs DNS.
smtpd_helo_restrictions =
     permit_mynetworks,
     permit_sasl_authenticated,
     reject_invalid_helo_hostname,

     reject_non_fqdn_helo_hostname,
     # reject_unknown_helo_hostname

# Règles de connexion des clients
# permit_sasl_authenticated : Accepter la connexion lorsque le client est authentifié
# reject_plaintext_session : Refuser les connexions non sécurisées
# reject_unauth_pipelining : Refuser les défauts lors de la connexion
smtpd_client_restrictions =
     permit_mynetworks,
     permit_inet_interfaces,
     permit_sasl_authenticated,
     # reject_plaintext_session,
     # reject_unauth_pipelining

# Règles sur les expéditeurs
# reject_non_fqdn_sender : Refuser les expéditeurs invalides (non FQDN)
# reject_unknown_sender_domain : Refuser les expéditeurs qui n'ont pas de champ DNS A ou MX
dans leurs DNS.
# reject_sender_login_mismatch : Refuser les expéditeurs locaux non authentifiés
smtpd_sender_restrictions =
     reject_non_fqdn_sender,

     reject_unknown_sender_domain,
     reject_sender_login_mismatch

#Paramètres de chiffrement via TLS

# Smtp ( OUTGOING / Client )
smtp_tls_loglevel            = 1
smtp_tls_security_level      = may
smtp_tls_CAfile              = /etc/ssl/certs/ca.cert.pem
smtp_tls_protocols           = !SSLv2, !SSLv3
smtp_tls_mandatory_protocols = !SSLv2, !SSLv3
smtp_tls_mandatory_ciphers   = high
```

```
smtp_tls_exclude_ciphers     = aNULL, eNULL, EXPORT, DES, 3DES, RC2, RC4, MD5, PSK, SRP, DSS,
AECDH, ADH
smtp_tls_note_starttls_offer = yes

# ------------------------------------------------------------------------------------------

# Smtpd ( INCOMING / Server )
smtpd_tls_loglevel            = 1
smtpd_tls_auth_only           = yes
smtpd_tls_security_level      = may
smtpd_tls_received_header     = yes
smtpd_tls_protocols           = !SSLv2, !SSLv3
smtpd_tls_mandatory_protocols = !SSLv2, !SSLv3
smtpd_tls_mandatory_ciphers   = medium

# Infos (voir : postconf -d)
# Medium cipherlist = aNULL:-aNULL:ALL:!EXPORT:!LOW:+RC4:@STRENGTH
# High cipherlist   = aNULL:-aNULL:ALL:!EXPORT:!LOW:!MEDIUM:+RC4:@STRENGTH

# smtpd_tls_exclude_ciphers   = NE PAS modifier cette directive pour des raisons de
compatibilité
#                               avec les autres serveurs de mail afin d'éviter une erreur du
type
#                               "no shared cipher" ou "no cipher overlap" puis un fallback en
#                               plain/text...
# smtpd_tls_cipherlist        = Ne pas modifier non plus !

smtpd_tls_CAfile              = $smtp_tls_CAfile
smtpd_tls_cert_file           = /etc/ssl/certs/mailserver.crt
smtpd_tls_key_file            = /etc/ssl/private/mailserver.key
smtpd_tls_dh1024_param_file   = $config_directory/dh2048.pem
smtpd_tls_dh512_param_file    = $config_directory/dh512.pem

tls_preempt_cipherlist = yes
tls_random_source      = dev:/dev/urandom

smtp_tls_session_cache_database  = btree:${data_directory}/smtp_scache
smtpd_tls_session_cache_database = btree:${data_directory}/smtpd_scache
lmtp_tls_session_cache_database  = btree:${data_directory}/lmtp_scache



# Paramètres de connexion SASL
# C'est ici que l'on déclare Dovecot comme une passerelle pour authentifier les utilisateurs.
# Postfix peut s'appuyer sur Dovecot pour identifier les connexions SMTP.
smtpd_sasl_auth_enable          = yes
smtpd_sasl_type                 = dovecot
smtpd_sasl_path                 = private/auth
smtpd_sasl_security_options     = noanonymous
smtpd_sasl_tls_security_options = $smtpd_sasl_security_options
smtpd_sasl_local_domain         = $mydomain
smtpd_sasl_authenticated_header = yes

broken_sasl_auth_clients = yes
```

- Enregistrez le fichier de configuration et quitter le. Nous allons y revenir dans quelques instants après la création des certificats.
- Maintenant il faut créer le certificat SSL dans le dossier **/etc/ssl.** Ce certificat sera auto signée. Nous verrons donc par la suite, la création d'un certificat valide et signé via Let's Encrypt.

```
cd /etc/ssl/

openssl genrsa -out ca.key.pem 4096

openssl req -x509 -new -nodes -days 1460 -sha256 -key ca.key.pem -out ca.cert.pem
```

```
Country Name (2 letter code) [AU]: FR

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

Locality Name (eg, city) []: Votre-Ville

Organization Name (eg, company) [Internet Widgits Pty Ltd]: Nom-De-Votre-Site ou De-Votre-Compagnie

Organizational Unit Name (eg, section) []: Nom-Autorité-Certification

Common Name (e.g. server FQDN or YOUR name) []: Root CA
```

```
openssl genrsa -out mailserver.key 4096
openssl req -new -sha256 -key mailserver.key -out mailserver.csr
```

```
Country Name (2 letter code) [AU]: FR

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

Locality Name (eg, city) []: Votre-Ville

Organization Name (eg, company) [Internet Widgits Pty Ltd]: Nom-De-Votre-Site ou De-Votre-Compagnie

Organizational Unit Name (eg, section) []: server
Common Name (e.g. server FQDN or YOUR name) []: mail.votre-domaine.fr
```

```
openssl x509 -req -days 1460 -sha256 -in mailserver.csr -CA ca.cert.pem -CAkey ca.key.pem -
CAcreateserial -out mailserver.crt

#On change les droits d'accès aux fichier clé généré précédemment.
chmod 444 ca.cert.pem
chmod 444 mailserver.crt
chmod 400 ca.key.pem
chmod 400 mailserver.key

#On déplacer les clés dans leur dossier approprié
mv ca.key.pem private/
mv ca.cert.pem certs/
mv mailserver.key private/
mv mailserver.crt certs/
```

- Maintenant nous allons générer les paramètres de chiffrement Diffie-Hellman :

```
openssl dhparam -out /etc/postfix/dh2048.pem 2048
openssl dhparam -out /etc/postfix/dh512.pem 512
```

Nous en avons pas encore terminé avec la configuration de Postfix.

Editez le fichier **/etc/postfix/main.cf**

```
nano /etc/postfix/main.cf
```

- Nous allons donc définir les paramètres de connexion via SASL (Simple Authentication and Security Layer) qui est un protocole d'authentication et de sécurisation.

```
# Paramètres de connexion SASL
# C'est ici que l'on déclare Dovecot comme une passerelle pour
authentifier les utilisateurs.
# Postfix peut s'appuyer sur Dovecot pour identifier les connexions SMTP.
smtpd_sasl_auth_enable          = yes
smtpd_sasl_type                 = dovecot
smtpd_sasl_path                 = private/auth
smtpd_sasl_security_options     = noanonymous
smtpd_sasl_tls_security_options = $smtpd_sasl_security_options
smtpd_sasl_local_domain         = $mydomain
smtpd_sasl_authenticated_header = yes

broken_sasl_auth_clients = yes
```

Les mails seront gérés par un user **vmail** qui aura **UID/GID** de **5000**, et un **HOME** pointant sur le répertoire **/var/mail**. C'est donc ce dossier qui contiendra les mails. On indique donc ces paramètres à la suite du fichier de configuration de Postfix et après nous allons créer cet utilisateur.

```
# Gestion stockage des mail
virtual_uid_maps        = static:5000
virtual_gid_maps        = static:5000
```

```
virtual_minimum_uid     = 5000
virtual_mailbox_base    = /var/mail
```

- Nous devons maintenant indiquer à Postfix comment il doit procéder pour récupérer les informations de domaine, adresse virtuel. Postfix va donc se connecter à base de données : MySql créer précédemment lors de l'installation de Postfiaxadmin. Il faut donc lui préciser les identifiants de cette base. Nous reviendrons sur la création de ces fichiers après avoir terminé la configuration principale de Postfix.

```
virtual_mailbox_domains = mysql:/etc/postfix/mysql-virtual-mailbox-
domains.cf
virtual_mailbox_maps    = mysql:/etc/postfix/mysql-virtual-mailbox-
maps.cf
virtual_alias_maps      = mysql:/etc/postfix/mysql-virtual-alias-maps.cf

smtpd_sender_login_maps  = mysql:/etc/postfix/mysql-sender-login-maps.cf
```

- Le paramètre suivant : **virtual_transport** est extrêmement important. Il permet donc à Postfix de savoir ce qu'il doit faire des mails reçus. Il doit donc les envoyer à Dovecot afin que les utilisateurs authentifiés puissent récupérer leur courrier.

```
virtual_transport = lmtp:unix:private/dovecot-lmtp
```

- Il ne reste plus qu'à configurer les paramètres généraux de Posftix en modifiant bien sûr les paramètres **myhostname** et **myorigin** par votre **FQDN**. Vous pouvez également modifier la valeur de (**Debian/GNU**) de la smtpd_banner. Si vous ne voulez pas communiquer la version de votre OS/Distribution au client mail. La modification de ce champ peut permettre de rendre plus complexe la recherche de faille dans votre système par un pirate. Vous pouvez donc remplacer cette valeur par le nom de votre serveur ou même ce que vous voulez.

```
# Parameters généraux de Posftix
smtpd_banner         = $myhostname ESMTP $mail_name (Debian/GNU)
biff                 = no
append_dot_mydomain  = no
readme_directory     = no
delay_warning_time   = 4h
mailbox_command      = procmail -a "$EXTENSION"
recipient_delimiter  = +
disable_vrfy_command = yes
message_size_limit   = 502400000
mailbox_size_limit   = 1024000000

inet_interfaces = all
inet_protocols = ipv4
```

```
myhostname     = hostname.domain.tld
myorigin       = hostname.domain.tld
mydestination = localhost localhost.$mydomain
mynetworks     = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128
relayhost      =

alias_maps      = hash:/etc/aliases
alias_database = hash:/etc/aliases
```

Nous en avons enfin terminé pour la configuration principale de Postfix. Vous pouvez vous rendre sur le site officiel pour plus d'informations sur cette partie : ici

Le fichier de configuration complet est disponible également **ici**

7°) Postfix et MySQL

- Il est maintenant temps de créer les 4 fichiers qui vont permettre à Postfix de récupérer les informations sur notre base de données. Pensez bien à modifier le champs **MOT DE PASSE**, par le mot de passe que vous avez indiqué lors de la création de la base de données **postfix**.

Récupération des informations sur les domaines :

```
nano /etc/postfix/mysql-virtual-mailbox-domains.cf
```

```
hosts = 127.0.0.1
user = postfix
password = MOT DE PASSE
dbname = postfix

query = SELECT domain FROM domain WHERE domain='%s' and backupmx = 0 and
active = 1
```

Récupération des adresses mail du domaine :

```
nano /etc/postfix/mysql-virtual-mailbox-maps.cf
```

```
hosts = 127.0.0.1
user = postfix
password = MOT DE PASSE
dbname = postfix

query = SELECT maildir FROM mailbox WHERE username='%s' AND active = 1
```

Récuparation des informations d'alias:

```
nano /etc/postfix/mysql-virtual-alias-maps.cf
```

```
hosts = 127.0.0.1
user = postfix
password = MOT DE PASSE
dbname = postfix

query = SELECT goto FROM alias WHERE address='%s' AND active = 1
```

Récupération des informations de correspondances entre le login SASL et les adresses d'expéditeur (MAIL FROM) :

```
nano /etc/postfix/mysql-sender-login-maps.cf
```

```
hosts = 127.0.0.1
user = postfix
password = MOT DE PASSE
dbname = postfix

query = SELECT username FROM mailbox WHERE username='%s' AND active = 1
```

- Afin de vous connectez de manière sécurisé en SMTPS, il faut donc activer le port 587. Tout se passe dans le fichier: **/etc/postfix/master.cf** en décommentant et en modifiant les lignes suivantes :

```
nano /etc/postfix/master.cf
```

```
submission inet n       -       -       -       -       smtpd
  -o syslog_name=postfix/submission
  -o smtpd_tls_dh1024_param_file=${config_directory}/dh2048.pem
  -o smtpd_tls_security_level=encrypt
  -o smtpd_sasl_auth_enable=yes
  -o smtpd_client_restrictions=permit_sasl_authenticated,reject
```

Cherchez également la ligne suivante et décommentez la si ce n'est pas déjà le cas.

```
smtp  inet  n       -       -       -       -       smtpd
```

Le fichier de configuration complet est disponible également **ici**

8°) Installation de Dovecot

- On installe donc les paquets suivants : dovecot-core dovecot-imapd dovecot-lmtpd dovecot-mysql.

dovecot-core étant le paquet principal. Les trois autres sont les client/serveur que va utiliser Dovecot pour le bon fonctionnement de notre système de messagerie électronique.

```
apt-get install dovecot-core dovecot-imapd dovecot-lmtpd dovecot-mysql
```

9°) Configuration de Dovecot

- Nous allons commencer par modifier le fichier **/etc/dovecot/dovecot.conf** afin de configurer les protocoles qui seront utilisés par Dovecot.

- **IMAP:** Pour la récupération des mails par les clients (Rainloop, Mozilla Thunderbird ...)
- **LMTP:** Pour le transfert de mail entre Postfix et Dovecot.

```
nano /etc/dovecot/dovecot.conf
```

```
!include_try /usr/share/dovecot/protocols.d/*.protocol
protocols = imap lmtp
listen = *

# Assurez-vous que cette ligne est bien décommentée :
!include conf.d/*.conf
```

Le fichier de configuration complet est disponible également **ici**

- Maintenant on indique à Dovecot comment il doit gérer le stockage de mail comme nous l'avons fait précédemment pour Postfix.

```
nano /etc/dovecot/conf.d/10-mail.conf
```

```
mail_location = maildir:/var/mail/vhosts/%d/%n/mail
maildir_stat_dirs=yes

namespace inbox {
    inbox = yes
}

mail_uid = 5000
mail_gid = 5000

first_valid_uid = 5000
last_valid_uid = 5000

mail_privileged_group = vmail
```

Le fichier de configuration complet est disponible également **ici**

- Vous l'aurez donc compris: les mails seront stockés **/var/mail.** Ce dossier étant déjà existant, il faut créer un répertoire pour les mails de votre domaine. Pensez bien à remplacer par votre domaine, celui que vous avez créer dans Postfixadmin.

```
mkdir -p /var/mail/vhosts/votre-domaine.fr
```

- On va maintenant créer l'utilisateur **vmail** et le groupe **vmail** dont nous avons parlé tout à l'heure.

```
groupadd -g 5000 vmail
useradd -g vmail -u 5000 vmail -d /var/mail
chown -R vmail:vmail /var/mail
```

- Modifiez maintenant le fichier **10-auth.conf**

```
nano /etc/dovecot/conf.d/10-auth.conf
```

```
disable_plaintext_auth = yes
auth_mechanisms = plain login
#!include auth-system.conf.ext # Commenter cette ligne
!include auth-sql.conf.ext     # décommenter cette ligne
```

Le fichier de configuration complet est disponible également **ici**

- Editez le fichier **auth-sql.conf.ext** pour y configurer les méthodes de connexion. Ce qui permet à Dovecot de savoir où il doit récupérer les informations des utilisateurs.

```
nano /etc/dovecot/conf.d/auth-sql.conf.ext
```

```
# Le mot de passe est obtenu à partir de la base de données
passdb {
  driver = sql
  args = /etc/dovecot/dovecot-sql.conf.ext
}

# Par contre le nom d'utilisateur est obtenu de manière statique à partir du conteneur local
# %d = domaine.tld
# %n = utilisateur
userdb {
  driver = static
  args = uid=vmail gid=vmail home=/var/mail/vhosts/%d/%n
}
```

- Nous allons donc maintenant configurer les paramètres de connexion au serveur Mysql. N'oubliez pas de changer la valeur de =**MOT DE PASSE**

```
nano /etc/dovecot/dovecot-sql.conf.ext
```

```
# Paramètres principale de connexion
driver = mysql
connect = host=127.0.0.1 dbname=postfix user=postfix password=MOT DE PASSE

# Permet de définir l'algorithme de hachage.
# Pour plus d'information: http://wiki2.dovecot.org/Authentication/PasswordSchemes
# /!\ ATTENTION : ne pas oublier de modifier le paramètre $CONF['encrypt'] de PostfixAdmin
default_pass_scheme = MD5-CRYPT

# Requête de récupération du mot de passe du compte utilisateur
password_query = SELECT password FROM mailbox WHERE username = '%u'
```

- Modifiez les permission d'accès sur le repertoire **/etc/dovecot**

```
chown -R vmail:dovecot /etc/dovecot
chmod -R o-rwx /etc/dovecot
```

- Il nous reste encore quelques fichiers à éditer, dont **10-master.conf**.

```
nano /etc/dovecot/conf.d/10-master.conf
```

```
service imap-login {

  inet_listener imap {
    port = 143
  }

  inet_listener imaps {
    port = 993
    ssl = yes
  }

  service_count = 0

}

service imap {

}

service lmtp {

  # On autorise Postfix à transférer les mails dans le spooler de Dovecot via LMTP
  unix_listener /var/spool/postfix/private/dovecot-lmtp {
      mode = 0600
      user = postfix
      group = postfix
  }

}

service auth {

  # On autorise Postfix à se connecter à Dovecot via LMTP
  unix_listener /var/spool/postfix/private/auth {
      mode = 0666
      user = postfix
      group = postfix
  }

  # On indique à Dovecot les permissions du conteneur local
  unix_listener auth-userdb {
      mode = 0600
      user = vmail
      group = vmail
  }

  user = dovecot

}

service auth-worker {

  user = vmail

}
```

Le fichier de configuration complet est disponible également **ici**

- Pour en finir avec la configuration de Dovecot, éditez le fichier **10-ssl.conf** et modifier les paramètres suivants concernant l'authentification via ssl.

```
nano /etc/dovecot/conf.d/10-ssl.conf
```

```
ssl = required
ssl_cert = </etc/ssl/certs/mailserver.crt
ssl_key = </etc/ssl/private/mailserver.key
ssl_protocols = !SSLv2 !SSLv3
ssl_cipher_list =
ALL:!aNULL:!eNULL:!LOW:!MEDIUM:!EXP:!RC2:!RC4:!DES:!3DES:!MD5:!PSK:!SRP:!DSS:!AECDH
:!ADH:@STRENGTH
ssl_prefer_server_ciphers = yes # Dovecot > 2.2.x
ssl_dh_parameters_length = 2048 # Dovecot > 2.2.x
```

Le fichier de configuration complet est disponible également **ici**

10°) Redémarrage des services et vérification des ports + premier test de connexion

- Il est maintenant temps de redémarrer Postfix et Dovecot pour que les nouvelles configurations soient prises en compte et également pour voir s'il y a des problèmes.

```
service postfix restart
service dovecot restart
```

Si un problème vous est remonté, c'est que vous avez surement fait une petite erreur dans les fichiers de configuration édité précédemment.

- Pour en savoir plus sur le problème, vous pouvez affiché le statu de postfix ou de dovecot via la commande:

```
service postfix status
service dovecot status
```

Postfix intègre directement une commande permettant de vérifier sa configuration.

- Exécutez la commande.

```
postfix check
```

A partir des informations données par la commande précédente, vous pourez probablement voir sur quel fichier de configuration il y a un problème et surement même voir la ligne qui pose problème. Vous pouvez même copier/coller le message d'erreur obtenu par la commande précédente afin de vous renseigner sur Google.com. Vous trouverez votre bonheur.

Une autre façon, un peu moins drôle est de reprendre les fichiers de configuration un par un pour les vérifier.

On va maintenant voir les ports en écoute sur la machine. Les ports **25** (SMTP), **587** (SMTPS) et **993** (IMAPS) doivent bien être en écoute sur 0.0.0.0

- Vérifiez la présence des lignes suivantes.

```
netstat -ptna
```

```
tcp 0 0 0.0.0.0:25 0.0.0.0: LISTEN 4995/master
tcp 0 0 0.0.0.0:587 0.0.0.0: LISTEN 4995/master
tcp 0 0 0.0.0.0:993 0.0.0.0:* LISTEN 5030/dovecot
```

- Si les ports sont bien en écoute on peut faire un petit essai afin de simuler une connexion SMTP via TELNET.

```
telnet localhost 25
ehlo localhost
```

Normalement après avoir fait le "**ehlo**" vous devriez avoir la réponse suivante :

```
250-hostname.domain.tld
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-STARTTLS
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN
```

Si vous voyez **250-STARTTLS**, c'est une bonne nouvelle. Cela veut dire que le serveur supporte bien l'authentification par **STARTTLS**.

11°) Premier test de fonctionnement

Avant de passer aux étapes suivantes du tutorial, il est important de vérifier si tout ce que nous avons mis en place jusqu'à présent fonctionnent bien.

- Pour se faire, nous allons tenter de nous connecter à notre serveur mail via un client qui est Mozilla Thunderbird. Il est téléchargeable ici. Vous pouvez également installer Claws Mail qui lui est un client bien plus léger.

Les informations de connexion suivante sont à adapter sellon votre configuration bien sur.

Partie compte utilisateur

Votre nom **# Vous pouvez y mettre ce que vous voullez**	BARRE Kévin
Addresse Mail **# Il s'agis de l'E-mail d'un des compte que vous avez créé sur PostfixAdmin**	contact@votre-domaine.fr
Mot de passe **# Il s'agis du mot de passe du compte que vous avez créé sur PostfixAdmin**	votre-mot-de-passe

Partie IMAP SMTP

	Protocole	**Serveur hostname**	**Port**	**SSL / TLS**	**Authentification**	**Nom Utilisateur**
Courrier entrant	IMAP	hostname.votredomaine.fr **#Le FQDN de votre serveur**	993	SSL/TLS	Normal password	contact@votre-domaine.fr
Courrier sortant	SMPT	hostname.votredomaine.fr **#Le FQDN de votre serveur**	587	STRARTLS	Normal password	contact@votre-domaine.fr

Test de la connexion via IMAP

Une fois les informations de connexion entrée, vous devez simplement réouvrir votre client mail pour qu'il se connecte au serveur.

Si vous n'avez pas eu de message d'erreur a l'ouverture c'est que vous êtres probablement connecter. Pour le vérifier nous allons jeter un œil dans le fichier log.

```
tail -f /var/log/mail.log
```

Et si vous pouvez voir les lignes suivantes à la fin du fichier c'est que la connexion IMAP/TLS est bien OK, et c'est déja pas mal.

```
Fev 12 19:47:41 hostname dovecot: auth-worker(xxx): mysql(127.0.0.1): Connected to database
postfix

Fev 12 19:47:41 hostname dovecot: imap-login: Login: user=<contact@votre-domaine.fr>,
method=PLAIN, rip=ADRESSE IP CLIENT, lip=ADRESSE IP SERVEUR, mpid=xxx, TLS, session=<xxxxxxxx>
```

Envoie de mail via SMTP

Maintenant nous allons envoyer un mail via notre serveur. Pour ce faire envoyer un mail sur une de vos addresse mail : votre-mail@gmail.com l'objet et le contenue du mail sont completement arbitraire pour le teste.

Pour voir si l'envie du mail par SMTP est bien opérationnel nous allons jeter un oeuil dans le fichier log.

```
tail -f /var/log/mail.log
```

Et si vous avez les lignes suivantes, c'est bien jouer😊

```
Fev 12 19:57:01 hostname postfix/submission/smtpd[xxx]: connect from [VOTRE ADRESSE IP]
Fev 12 19:57:01 hostname dovecot: auth-worker(xxx): mysql(127.0.0.1): Connected to database
postfix
Fev 12 19:57:01 hostname postfix/submission/smtpd[xxx]: client=[VOTRE ADRESSE IP],
sasl_method=PLAIN, sasl_username=votre-mail@gmail.com
Fev 12 19:57:01 hostname postfix/smtp[xxx]: to=<contact@votre-domaine.fr>, relay=[ADRESSE IP
RELAI]:25, status=sent (250 OK)
```

Réception de mail via IMAP

Pour ce dernier test vous allez vouy rentre sur votre compte Gmail ou autre et envoyer un mail sur votre addresse contact@votre-domaine.fr

Si vous avez les lignes suivantes dans le fichiers mail.log, c'est que la réception par IMAP et le transfert de l'email par LMTP sont bons aussi.

```
Aug 16 20:04:58 hostname postfix/smtpd[xxx]: Anonymous TLS connection established from
mail.google.com: TLSv1 with cipher ECDHE-RSA-RC4-SHA (128/128 bits)
Aug 16 20:05:00 hostname postfix/qmgr[xxx]: from=<votre-mail@gmail.com>, size=1748, nrcpt=1
(queue active)
Aug 16 20:05:01 hostname dovecot: lmtp(xxx, contact@votre-domaine.fr): saved mail to INBOX
```

```
Aug 16 20:05:01 hostname postfix/lmtp[xxx]: to=<contact@votre-domaine.fr>,
relay=mail.domain.tld[private/dovecot-lmtp], status=sent (250 2.0.0 <contact@votre-domaine.fr>
Saved
```

12°) Installation et configuration d'OpenDKIM

Pour commencer installation d'OPEN DKIM entrer les commandes suivantes.

```
apt-get install opendkim opendkim-tools
```

Editez le fichier de configuration **opendkim.conf** avec le contenu suivant :

```
nano /etc/opendkim.conf
```

```
AutoRestart             Yes
AutoRestartRate         10/1h
UMask                   002
Syslog                  Yes
SyslogSuccess           Yes
LogWhy                  Yes

OversignHeaders         From
AlwaysAddARHeader       Yes
Canonicalization        relaxed/simple

ExternalIgnoreList      refile:/etc/opendkim/TrustedHosts
InternalHosts           refile:/etc/opendkim/TrustedHosts
KeyTable                refile:/etc/opendkim/KeyTable
SigningTable            refile:/etc/opendkim/SigningTable

Mode                    sv
PidFile                 /var/run/opendkim/opendkim.pid
SignatureAlgorithm      rsa-sha256

UserID                  opendkim:opendkim

Socket                  local:/var/spool/postfix/opendkim/opendkim.sock
```

Postfix va communiquer avec OpenDKIM via un socket, on va donc créer un répertoire **/var/spool/postfix/opendkim**

```
mkdir /var/spool/postfix/opendkim
chown opendkim: /var/spool/postfix/opendkim
usermod -aG opendkim postfix
```

On indique maintenant à Postfix comment entré en contact avec OpenDKIM. On édite donc les fichier main.cf de Postfix.

```
nano /etc/postfix/main.cf
```

```
milter_protocol = 6
milter_default_action = accept
smtpd_milters = unix:/opendkim/opendkim.sock
non_smtpd_milters = unix:/opendkim/opendkim.sock
```

Enregistrez les modifications puis nous allons passer à la suite.

Maintenant créer le fichier TrustedHosts dans le repertoire /etc/opendkim

```
nano /etc/opendkim/TrustedHosts
```

```
127.0.0.1
localhost
::1
*.votre-domaine.fr
```

Remplacez bien sur ***.votre-domaine.fr** par la valeur vous correspondant.

Créez le répertoire keys dans /etc/opendkim

```
mkdir -p /etc/opendkim/keys
```

Ce répertoire contiendra la clé publique est privé d'open DKIM. On ce depalce dans ce dernier et nous allons créer un autre répertoire qui aura comme nom, votre nom de domaine.

```
cd /etc/opendkim/keys && mkdir domain.tld && cd domain.tld
```

Plus de formulaires de connexion sur rainloop.!!

Ce problème m'est arrivé après la désinstallation de PHP7.0 en le remplaçant par PHP5

Suppression de **/var/www/rainloop** tout en faisant un back up avant.

- Réinstallation de rainloop

```
:wget http://repository.rainloop.net/v2/webmail/rainloop-latest.zip
mkdir /var/www/rainloop
unzip rainloop-latest.zip -d /var/www/rainloop
rm -rf rainloop-latest.zip
```

Modifiez les permissions pour que le serveur web ai accés au répertoire **/var/www/rainloop**

```
cd /var/www/rainloop
find . -type d -exec chmod 755 {} \;
find . -type f -exec chmod 644 {} \;
chown -R www-data:www-data .
```

- Connexon http://rainloop.votredomaine.fr

Vous aurez peut être un message vous disant que la dépendance CURL pour PHP n'est pas disponible dans votre version de PHP.

- Pour connaitre votre version de PHP

```
php -v
```

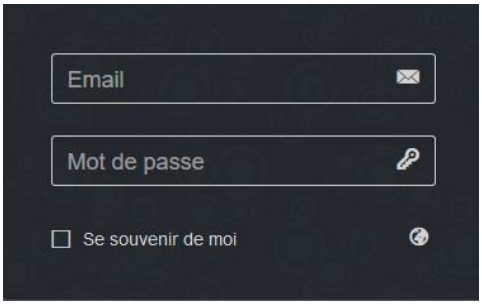
```
PHP 5.6.33-0+deb8u1 (cli) (built: Jan  5 2018 15:46:26)
Copyright (c) 1997-2016 The PHP Group
Zend Engine v2.6.0, Copyright (c) 1998-2016 Zend Technologies
    with Zend OPcache v7.0.6-dev, Copyright (c) 1999-2016, by Zend Technologies
```

Je vois que je suis en PHP5, j'installe donc CURL pour celui ci.

```
apt-get install php5-curl
```

Puis un **service nginx reload ou restart** pour prendre en compte les modifications.

Et normalement tout rentre dans l'ordre.



https://mondedie.fr/d/5750-Tuto-Installer-un-serveur-de-mail-avec-Postfix-Dovecot-et-Rainloop

https://www.digitalocean.com/community/tutorials/how-to-install-linux-nginx-mysql-php-lemp-stack-on-debian-8

https://www.supinfo.com/articles/single/3558-installer-un-certificat-ssl-sur-nginx-avec-let-s-encrypt