MESSAGERIE

Présentation de Postfix

1- L'infrastructure Postfix

1.1 Présentation :

Cette présentation ne reprend pas les concepts des protocoles de messagerie.(voir le cours et le TD pour cela).

Vous pouvez consulter la traduction de la documentation postfix sur :

http://postfix.traduc.org/

Le processus principal est le programme « master » qui est lancé par le script de démarrage. Il est résident et se charge d'appeler les différents modules en fonction des traitements et des besoins :

- Réception des messages,
- Analyse des messages,
- Gestion des files d'attente,
- Routage des courriers,
- Distribution des courriers,
- …

Il existe 2 fichiers principaux pour la configuration de Postfix qui sont « *main.cf* » et « *master.cf* ». L'ensemble des processus composant postfix sont définis dans master.cf. Le fichier main.cf permet de configurer l'ensemble de l'infrastructure à travers les variables de configuration.

Main.cf :

Le fichier de configuration typiquement Linux. Toute la configuration générale du serveur de messagerie. Les entrées sont du type VarX= Valeur ou VarY=$VarX ou multi-valeurs var= val1,val2,val3. Certaines variables peuvent aussi faire référence au contenu d'un fichier. La variable permet alors de définir le type et le nom de fichier de définition associé. Il faut ensuite (re)construire le fichier au format indiqué par la commande « postmap ».

Exemple :
« Transport_maps = hash:/etc/postfix/Nom_Fic » permet de définir un fichier contenant les valeurs des transports dans un format « hash ». La commande Postmap /etc/postfix/Nom_Fic va construire le fichier au format hash.

Les variables de ce fichier sont utilisées par les différents modules (processus) composants postfix. Pour différentes instances de programmes de même nature (smtpd généralement) on peut faire varier la configuration en redéfinissant la valeur au niveau de la définition du processus par surcharge des valeurs (dans master.cf).
Les exercices de la suite du cours vous permettront de mieux saisir cette particularité de la configuration.

Chaque ligne de ce fichier est composée de colonnes permettant de préciser le comportement du processus (instance(s) de MTA, client(s) smtp, agent(s) de distribution…).

Exemple:

```
clientsmtp      unix -      -       n       -       16      smtp

127.0.0.1:2626 inet  n   -  n      -       16      smtpd
      -o content_filter=spamassassin
      -o smtp_send_xforward_command=yes
```

Dans main.cf: content_filter=clientsmtp:127.0.0.1:2626

Les différents « processus de traitement » (fils du processus master):

Master est le processus père de tous les autres. Il supervise l'ensemble du système.

- pickup : lit les messages dans la file d'attente locale (file maildrop)
- cleanup : vérifie la cohérence des messages locaux et réseaux
- trivial-rewrite : Traitement des messages (vérifie, modifie, ajoute les entêtes manquantes, convertie les adresses, choisit le bon transport…)
- bouce deferred : Créer un rapport de non-délivrance (notification)
- qmgr est le gestionnaire des files d'attentes

Les agents d'entrés et de distributions principaux :

Entrée :

- smtpd : un serveur smtp (réception des messages du réseau)
- postdrop : processus qui réceptionne les messages locaux (comptes systèmes…)

Distribution :

- smtp, lmtp : se comporte comme un client de messagerie smtp
- - pipe : processus qui attend des infos en entrée pour le faire passer à un script, driver…
- - virtual : se charge des messages pour les boites dîtes « virtual » (compte non système)

Les files d'attente :

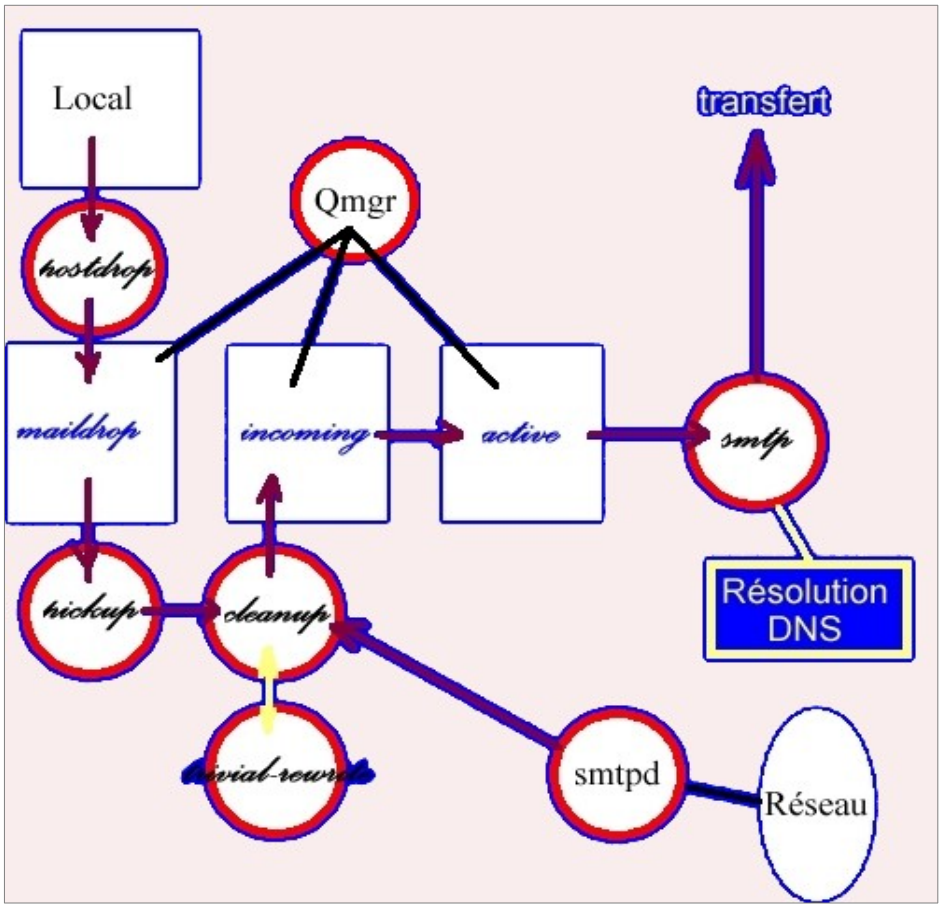
- maildrop : les messages locaux tombent dans cette file d'attente (provenance postdrop)
- incoming : les messages conformes (après analyse par cleanup et trivial-rewrite) transitent dans cette file. Ils sont passés à la file active.
- active : file des messages prêt à être traité par un transport (agent de distribution).
- deffered : file des messages ne pouvant être distribué
- corrupt : file des messages corrompus
- hold : file des messages conservés indéfiniment.

Exemple de la liste des processus (utilisation pstree)

```
├─master(2139)─┬─cleanup(6663)
│              ├─local(6665)
│              ├─pickup(6302)
│              ├─qmgr(4455)
│              ├─trivial-rewrite(6664)
│              └─smtpd(6735)
```

Bien entendu, l'affichage peut varié en fonction du moment où nous tapons la commande. En effet, certains processus sont exécutés à intervalles réguliers ou à l'arrivée d'événements particuliers.

Synoptique simplifié (cas local et réseau):



La notion de domaine :

Domaine Canonique : domaine local, domaine du serveur
Domaine relayé : les domaines a relayer (on ne stocke pas dans une boite, on ne sert que d'intermédiaire (MX de faible valeur pour un domaine ami). Les messages sont en file d'attente.
Domaine virtuel : domaine hébergé mais pas canonique. On distingue les domaines d'alias virtuel (re-routage vers comptes unix) et les boites aux lettres virtuelle (boites ne correspondant pas à un compte système : les messages sont stockés dans des fichiers ou répertoires) ou utilisent un transport particulier pour router les messages (lmtp, pipe...).

Les alias :
Les alias permettent de rediriger une adresse vers une autre, un domaine par un autre. Les alias sont configurés à travers une table des alias. La variable « virtual_alias_maps » est utilisée pour indiquer le format et le nom du fichier de stockage des alias.

Exemple :
Virtual_alias_maps = hash:/etc/postfix/virtual.alias

Dans le fichier alias on a:

Adresse1@doamine adresse@domaine...

Les fichiers « listes » après une modification se reconstruisent via la commande « postmap Nom_Fichier ». La commande produit un ficher au format indiqué et utilisé par le système.
La commande « postmap /etc/postfix/virtual.alias » va produire un fichier .db au format indiqué

Routage par les transports :

Il existe une table de transport pour diriger les messages en fonction de critères définis. La variable transport_maps permet d'indiquer le format et le fichier liste qui permet ce routage.
« Transport_maps = hash:/etc/postfix/transport »

Dans le fichier transport on a :
adresse@domaine Nom_du_transport

Nom_du_transport correspond à une entrée du fichier master.cf (un pipe, un script...)
Attention! Avant de faire un transport, le fichier des alias est consulté.

Dans tous les cas, on passe d'abord par les alias (virtual_alias_maps) donc si on veut qu'une adresse particulière d'un domaine virtuelle par exemple soit redirigée vers une adresse locale on peut le faire en plaçant une entrée dans cette table. On peut également créer un compte fourre tout en plaçant le nom de domaine virtuel sur boite unique (ex: @etab.ac-amiens.fr service). Par contre, si on place cette entrée dans le fichier des alias plus rien n'arrive au domaine virtuel car les alias sont prioritaires sur les domaines virtuels (voir la suite de la présentation).
Réécrire des émetteurs et destinataires d'un mail :

Cette fonction utilise les canonical maps. Celles-ci peuvent être appliqués en émission, en réception ou les deux.

Voyons comment opérer au moyen de 3 exemples simples.

1- Changement d'émetteur :

Admettons que vous avez un utilisateur root@monserveur.lan, nous allons réécrire l'adresse avec une canonical map afin de la faire apparaître en tant que root-serveur@example.org

Ajoutez la ligne suivante dans votre fichier /etc/postfix/main.cf

sender_canonical_maps = hash :/etc/postfix/sender_canonical

Insérez la ligne suivante dans le fichier /etc/postfix/sender_canonical.

root@monserveur.lan root-serveur@example.org

Enfin tapez :

postmap /etc/postfix/sender_canonical

service postfix reload

2- Changement de destinataire

Vous souhaitez que les messages adressés à hostmaster@dns1.example.org soient adressés à dns-master@example.org. Nous allons créer une canonical map pour cela.

Ajoutez la ligne suivante dans /etc/postfix/main.cf

recipient_canonical_maps = hash :/etc/postfix/recipient_canonical

Insérez la ligne suivante dans le fichier /etc/postfix/recipient_canonical :

hostmaster@dns1.example.org dns-master@example.org

Enfin tapez :

postmap /etc/postfix/recipient_canonical

service postfix reload

Changer un nom de domaine

3-Réécrire de toutes les adresses d'un nom de domaine donné et pour cela nous allons créer une canonical map générique (envoi + réception) afin de remplacer domain.lan par example.org.

Ajoutez la ligne suivante dans /etc/postfix/main.cf

canonical_maps = hash :/etc/postfix/canonical

Insérez la ligne suivante dans le fichier /etc/postfix/canonical :

@domain.lan @example.org

Enfin tapez

postmap /etc/postfix/canonical

service postfix reload

Installation du serveur SMTP : Postfix

Par les paquets binaires ou sources.

Sources :

Assurez-vous d'avoir une machine parfaitement configurée: c'est important pour le fonctionnement de postfix ! (réseau, résolution DNS, Domaine, nom de la machine, fichier hosts…),
Vérifier la version correspondant à votre distribution,
Vérifier et Installer les dépendances,
Télécharger et installer cette version à partir des sources (téléchargement des sources par par les dépôts et installation par make, make install sans le .configure),
Le service va fonctionner via UID et GID postfix/postdrop, créer cet utilisateur et ce groupe (adduser, groupadd, addgroup).
Vérifier que postfix appartient au groupe postdrop,
Installer si ce n'est pas le cas le MDA procmail (dans le cadre du TP, c'est lui qui va stocker le message dans le fichier mail de l'utilisateur du système dans /var/spool/mail/user).

Configuration du serveur SMTP : Postfix

Le fichier principal est « main.cf » dans le répertoire de configuration. La syntaxe est du type paramètre=val (comme pour de nombreux services linux classiquement). Pour donner à val la valeur d'un autre paramètre, il suffit de placer un « $ » devant le non du paramètre dans la zone valeur (comme pour les variables linux).

Ex :
Mon_domaine = dom.com
Mon_origine = $Mon_domaine

Les principaux paramètres du fichier main.cf
Les chemins d'accès sont renseignés automatiquement après l'installation :
queue_directory : spécifie le répertoire des files d'attente
command_directory : spécifie le répertoire des commandes administratives
daemon_directory : spécifie le répertoire des démons de postfix
mail_spool_directory : spécifie le répertoire des boites mail
mail_owner : spécifie le compte utilisé par le démon et les queues
myhostname : le non du serveur pleinement qualifié
mydomain : le domaine du serveur
myorigin : utilisé pour renseigner la partie droite d'une adresse mail non renseignée ($myhostname ou $mydomain)
mydestination : permet d'indiquer pour qui on accepte les mails localement ($myorigin, localhost, une liste dans une base…)
alias_maps : indique l'ensemble des bases d'alias d'adresses (correspondances d'adresse)
alias_database : idem pour ce qui se rapporte au monde unix
mailbox_command : permet d'indiquer l'agent de délivrance à utiliser (voir le cours)
relayhost :Pour une délivrance des messages vers un autre MTA

La conf minimale :

En plus des variables pré-renseignées (les chemins de configurations) il faut paramétrer un minimums de variables:

mynetworks = le réseau autorisé à se connecter ex :172.30.177.0/24
 # Le nom du domaine
mydomain = le domaine reconnu ex : in.ac-amiens.fr
 # Le nom de la machine pleinement qualifié
myhostname = ex : linux.in.ac-amiens.fr
 # Pour des adresses non pleinement qualifiées ajouter myorigin à droite du @
myorigin = $mydomain
 # Pour identifier les domaines locaux (à delivrer sur cette machine)
mydestination = $myhostname, localhost, $mydomain
 # identifier le MDA pour délivrer un message local
mailbox_command = /usr/bin/procmail

L'agent de délivrance des messages sera /usr/bin/procmail car en effet, dans un premier temps, nous utilisons les comptes systèmes (linux) dans le cadre du TP.

Procmail est un MDA. Le fonctionnement de procmail est très simple : il est appelé avec un mail en entrée standard, et traite le mail en fonction des filtres contenus dans le fichier *.procmailrc* (si il existe) du répertoire home de l'utilisateur. Si ce fichier n'existe pas il le dépose simplement dans la boite locale de l'utilisateur /var/mail/compte. Vous pouvez également utiliser le MDA « virtual » pour gérer des comptes sans avoir à créer un compte sur le système (faire un « man virtual » pour obtenir plus d'infos).
Voir également l'excellent article sur :
http://www.linux-france.org/article/mail/procmail/procmail.html

alias_maps = hash:/chemin/aliases permet d'indiquer le fichier des alias. Il faut construire cette BDD (ici un simple format hash) via la commande postmap (même si le fichier est vide au début de l'installation) et vérifier la création du fichier .db sinon le service ne fonctionne pas correctement.

Pour une config mydestination=myhostname,localhost et mydomain, les comptes valides seront les adresses du type nom@NomMachine, nom@Localhost, nom@domaine . Les logs du démon sont stockés dans /var/log/maillog (Ils sont écrit par le démon syslog).
Le test de la configuration se fait via la commande « postfix check » pour lancer arrêter le service utiliser « start » et « stop » à la place de « check ». Pour avoir la configuration de toutes les variables même celles par défaut la commande est « postconf » et pour avoir les valeurs par défauts il faut utiliser l'option -d.

Travail à réaliser :

Installer votre serveur SMTP et placer les directives minimales dans le fichier main.cf, créer 2 comptes systèmes, démarrer et connectez vous et sur votre service SMTP et tester le fonctionnement en envoyant un message d'un compte vers l'autre (comme vous l'avez fait en

TD). Vérifiez au niveau système via la commande mail (en étant connecté via le compte destinataire du message) ou directement en affichant le contenu du fichier mails (/var/spool/mail/Compte_User) et dans les traces du fichier de log.

root@debian:~/f# telnet 172.30.177.86 25
Trying 172.30.177.86...
Connected to 172.30.177.86.
Escape character is '^]'.
220 smtp.beauvelo.fr ESMTP Postfix (Debian/GNU) TP ISRI
...

Installation du serveur pop/imap de l'université de Washington:

Les sources sont accessibles via le site http://www.washington.edu/imap/
(imap-2007f pour la version disponible)
ou par le paquet source de votre distribution (uw-imap pour ubuntu 16.04 ou 14.04)
Recherche via :

https://packages.ubuntu.com/fr/

https://packages.ubuntu.com/search?
suite=xenial&section=all&arch=any&lang=fr&keywords=uw-imap&searchon=sourcenames

https://packages.ubuntu.com/fr/source/xenial/uw-imap pour ubuntu

add-apt-repository "deb http://archive.canonical.com/ubuntu $(lsb_release -sc) partner"
apt-get update

wget https://www.mirrorservice.org/sites/ftp.cac.washington.edu/imap/imap-2007f.tar.gz
apt-get build-dep uw-imap
vi Makefile → modifier les variables
make slx
* Pour une autre version Linux, il faut certainement lister les dépendances du paquet uw-imap et les installer manuellement avant de compiler le programme.

root@PV-Linux:/usr/src/imap-2007f# make slx

make[1]: entrant dans le répertoire « /usr/src/imap-2007f »

++

+ Building in NON-COMPLIANCE with RFC 3501 security requirements:

+ Non-compliant:

++ TLS/SSL encryption is NOT supported

++ Unencrypted plaintext passwords are permitted

+

+ In order to rectify this problem, you MUST build with:

++ SSLTYPE=nopwd

+ You must also have OpenSSL or equivalent installed.

++

Do you want to continue this build anyway? Type y or n please:

Y
…

`cat ../c-client/CCTYPE` -I../c-client `cat ../c-client/CFLAGS` -c -o tquota.o tquota.c

`cat ../c-client/CCTYPE` -I../c-client `cat ../c-client/CFLAGS` -o tmail tmail.o tquota.o ../c-client/c-client.a `cat ../c-client/LDFLAGS`

make[2]: quittant le répertoire « /usr/src/imap-2007f/tmail »

make[1]: quittant le répertoire « /usr/src/imap-2007f »

root@PV-Linux:/usr/src/imap-2007f#

Les binaires ipopd et imapd sont bien compilés :

```
root@PV-Linux:/usr/src/imap-2007f# ls -l ipopd

total 6028

-rwxr-xr-x 1 root dip 2942668 déc.  19 09:57 ipop2d

lrwxrwxrwx 1 root dip      38 déc.  19 09:57 ipop2d.c ->
/usr/src/imap-2007f/src/ipopd/ipop2d.c

-rw-r--r-- 1 root dip  100872 déc.  19 09:57 ipop2d.o

-rwxr-xr-x 1 root dip 2963420 déc.  19 09:57 ipop3d

lrwxrwxrwx 1 root dip      38 déc.  19 09:57 ipop3d.c ->
/usr/src/imap-2007f/src/ipopd/ipop3d.c

-rw-r--r-- 1 root dip  159104 déc.  19 09:57 ipop3d.o

lrwxrwxrwx 1 root dip      38 déc.  19 09:57 Makefile ->
/usr/src/imap-2007f/src/ipopd/Makefile

root@PV-Linux:/usr/src/imap-2007f# ls -l imapd

total 3628

-rwxr-xr-x 1 root dip 3136709 déc.  19 09:57 imapd

lrwxrwxrwx 1 root dip      37 déc.  19 09:57 imapd.c ->
/usr/src/imap-2007f/src/imapd/imapd.c

-rw-r--r-- 1 root dip  576872 déc.  19 09:57 imapd.o

lrwxrwxrwx 1 root dip      38 déc.  19 09:57 Makefile ->
/usr/src/imap-2007f/src/imapd/Makefile

root@PV-Linux:/usr/src/imap-2007f#


root@PV-Linux:/usr/src/imap-2007f# cp ./ipopd/ipop3d /usr/bin

root@PV-Linux:/usr/src/imap-2007f# cp ./imapd/imapd /usr/bin
```

Ce paquetage renferme un serveur pop(s) et un serveur imap(s) (uniquement la version imap sous certaines distributions). La configuration par défaut compile l’application avec la couche SSL. Nous simplifions l'installation, en supprimant la couche SSL, utilisation en ipv4 et sans

chiffrement des mots de passe. il est donc nécessaire de modifier les sources (fichier Makefile) option SSLTYPE, IP, et PASSWDTYPE. Installer les dépendances. Ces services fonctionnent au dessous du super démon inetd (inetd, ou xinetd ou openbsd-inetd en fonction des distributions. Faire la compilation en fonction de votre système (linux : make slx), copier le démon dans le répertoire /usr/sbin (imapd et ipopd), vérifier le bit x, ajouter les lignes nécessaires dans /etc/inetd.conf ainsi que dans le fichier /etc/services si besoin. (PASSWDTYPE=std, SSLTYPE=none, IP=4)
N'oublier pas de relancer le service inetd (ou la variante de votre distribution).

Exemple de configuration :

Pour inetd:

```
pop3    stream    tcp    nowait    root    /usr/sbin/ipop3d    ipop3d
```

Pour xinetd :

```
service pop3
{
        socket_type        = stream
        wait               = no
        user               = root
        server             = /usr/bin/ipop3d
        log_on_success  += HOST DURATION
        log_on_failure    += HOST
        instances          = 150
        cps                = 70 30
        disable            = no
}
```

Relancer le service xinetd et tester la connexion :

root@PV-Linux:/etc/xinetd.d# telnet 10.1.16.50 110

Trying 10.1.16.50...

Connected to 10.1.16.50.

Escape character is '^]'.

+OK POP3 PV-Linux 2007f.104 server ready

QUIT

+OK Sayonara

Connection closed by foreign host.

Test de la configuration (voir TD):

Connectez vous sur le serveur pop et imap avec l'un des deux comptes et vérifiez la connexion. La connexion se fait en mode commande via un telnet sur le port 110 (143 pour imap).

Nous avons donc maintenant une architecture de messagerie opérationnelle avec un MTA serveur (SMTP) gérant le domaine de l'entreprise et les MAA serveurs (IMAP, POP) pour permettre aux usagers de retirer leurs messages. Il faut maintenant consolider notre infrastructure (anti-virus, anti-spam...).

Un peu plus loin avec Postfix...

Postfix permet de nombreux contrôles paramétrables : sur le contexte de la communication (réseaux, ip, résolution DNS…), au niveau des commandes : Enveloppe : HELO, MAIL FROM…, au niveau des variables d'en-têtes, du corps du message...

voir http://postfix.traduc.org et http://www.postfix.org/uce.html

C'est ce que nous allons voir maintenant...

Mise en place des contrôles d'accès au serveur de mail : Un début d'anti-spam !

Exemples de paramètres de contrôles :
au niveau de la connexion SMTP (directive *smtpd_client_restrictions)*
au niveau de l'adresse donnée dans la phase HELO/EHLO (directive *smtpd_helo_restrictions)*
au niveau de l'adresse donnée dans la phase MAIL FROM: (directive *smtpd_sender_restrictions)*
au niveau du RCPT TP: (directives smtpd_recipient_restrictions= reject_unauth_destinations)

Exemple:

Par défaut Postfix accepte tous les mails de n'importe quel site émetteur. Pour constituer une liste noire , on peut créer un fichier */etc/postfix/access* dont la syntaxe est la suivante (c'est un moyen rapide pour créer une règle d'anti-spam) :

Site1.com REJECT
Site2.com REJECT
Site3.dom..com OK

Comme expliqué précédemment, ce fichier sera indexé via la commande « postmap » dbm:*/etc/postfix/access* ou hash:*/etc/postfix/access* selon les types d'indexation supportés par votre compilation» (nous utiliserons le format hash).

Ce fichier de liste-noire peut être utilisé dans les variables "*smtpd_sender_restrictions"* et "*smtp_client_restrictions"* pour restreindre les adresses émettrices dont on accepte les mails

smtpd_sender_restrictions = check_sender_access hash:*/etc/postfix/access*

Interdire les fichiers dangereux :
Il est également possible de sécuriser la boite aux lettres de vos utilisateurs en testant les pièces jointes transitant par le serveur.

Ex : refuser les fichiers portant une extension « dangereuse » exe com bat…

Ligne à rajouter dans /etc/postfix/main.cf :

mime_header_checks = regexp:/etc/postfix/mime_header_checks

Cela permet d'indiquer de tester les formats mimes via un fichier de règles construites via des expressions régulières.

Le contenu du fichier avec la règle :

```
/filename="?.*\.(bat|com|pif|vb|exe|lnk|scr|reg|chm|wsh|js|inf|shs|job|ini|
shb|scp|scf|wsc|sct|dll|msc|msi|ppt|pps|mdb|rar|bmp|wav|mpg|mpeg|wma|wmv)/
REJECT
/^Content-(Disposition|Type):.*(file)?name="?.*\.(bat|com|pif|vb|exe|lnk|
scr|reg|chm|wsh|js|inf| shs|job|ini|shb|scp|scf|wsc|sct|dll|msc|msi|ppt|
pps|mdb|rar|bmp|wav|wma|wmv)/ REJECT Extension de fichier joint refusée.
Desole!
```

Tester via votre client graphique.

Exercices :

- Un envoi massif de spam de l'adresse contact-info@pub.fr arrive dans les boites de vos utilisateurs, vous réalisez un filtre qui permet de bloquer cette adresse. Vous testez votre filtre.

- Rechercher tous les paramètres postfix du type « reject_* ». Tester ces paramètres dans la configuration de votre serveur.

Configuration du serveur SMTP en serveur de relayage (dit null client)

Un relais ne sert pas de comptes locaux il transmet uniquement le message vers un autre serveur SMTP après toutefois les avoir stockés ou même fait des traitements particuliers si nécessaire (spam, anti-virus…).

relayhost = $mydomain
relayhost = [gateway.example.com]
relayhost = [ip_add]

Pour tester cette configuration positionner l'adresse du serveur smtp de la FAC et utiliser une connexion telnet vers votre MTA SMTP local pour envoyer un message sur votre compte de la FAC. Votre serveur local doit relayer vers le MTA de l'université. Vérifier l'envoi du message et contrôler dans le source du message le passage par les différents MTA.

Pour forwarder uniquement un domaine particulier :

Il faut identifier un transport particulier pour le domaine à « forwarder » (ex : ac-amiens.fr forwarde vers smtp.ac-amiens.fr). La ligne « **ac-amiens.fr smtp:[smtp.ac-amiens.fr]** » est a placer dans le fichier transport.
Il ne faut pas oublier de placer la ligne dans le fichier main.cf et de re-construire la BDD via un postmap.

Identification du fichier transport : « transport_maps = hash:/etc/postfix/transport »

Contenu du fichier transport :
ac-amiens.fr smtp:[smtp.ac-amiens.fr]

Construction de la BDD :
postmap transport

Exercice :

Vous êtes le MX de secours d'une filiale qui possède son propre domaine et infrastructure de messagerie (beauvtt.fr) . Que faut il faire pour palier à un problème de connexion internet de cette filiale ? Elle se propose également de devenir secours pour votre domaine... que faut il faire ?

Mutualiser plusieurs noms de domaine sur un même domaine:

On peut en avoir besoin pour mutualiser un domaine interne et un domaine externe par exemple

Définir le ou les domaines virtuels partagés:
virtual_alias_domains = alias.ac-amiens.fr (c'est un exemple)
Définir la table des alias pour les différents domaines
virtual_alias_maps = hash:/etc/postfix2/virtual_alias

Dans le fichier virtual_alias, on définit les équivalences :
info@domaineexterne.fr util@domaineinterne.fr

Ici, un message pour « info@domaineexterne.fr » tombe dans la boite de « util@domaineinterne.fr »

Pour héberger plusieurs domaines de messagerie dans l'arborescence du serveur :

Dans le cadre du TP, nous avons configurer le serveur pour utiliser les boites systèmes des utilisateurs. Avec cette option, nous allons utiliser la notion de domaine vituel ce qui va permettre de stocker les messages dans un fichier classique même si l'utilisateur n'a pas de compte système.
on utilise pour cela le parametre virtual_mailbox_domains:
"virtual_mailbox_domains = dom1.fr, dom2.fr"
ou via un fichier si on utilise de nombreux domaines virtuels
« virtual_mailbox_domains = /etc/postfix/dom_virtuels »

On utilise ensuite
virtual_mailbox_base pour identifier la racine du repertoire de stockage
Il reste a identifier la boite (le fichier) pour chacun des comptes via les directives :

virtual_mailbox_domains = etab.fr
virtual_mailbox_base = /var/spool/etab
virtual_mailbox_maps = hash:/etc/postfix/mailbox_etab
virtual_uid_maps = static:113
virtual_gid_maps = static:120
(user 113 postfix et group 120 postfix dans ce cas)

root@debian:/etc/postfix# cat mailbox_etab
prof@etab.fr prof
Créer le fichier .db

Après envoi d'un message:
root@debian:/etc/postfix# ls -l /var/spool/etab/
total 4
-rw------- 1 postfix postfix 466 janv. 9 18:03 prof
root@debian:/etc/postfix# cat /var/spool/etab/prof

From pascal@beauvelo.fr Thu Jan 9 18:03:25 2014
Return-Path: <pascal@beauvelo.fr>
X-Original-To: prof@etab.fr
Delivered-To: prof@etab.fr
Received: from pascal (testcal.in.ac-amiens.fr [172.30.177.86])
        by smtp.beauvelo.fr (Postfix) with SMTP id 0F015C5D95
        for <prof@etab.fr>; Thu, 9 Jan 2014 18:02:58 +0100 (CET)
subject: essai
Message-Id: <20140109170309.0F015C5D95@smtp.beauvelo.fr>
Date: Thu, 9 Jan 2014 18:02:58 +0100 (CET)
From: pascal@beauvelo.fr

test

root@debian:/etc/postfix#

Il faut créer le répertoire /var/spool/etab avec les bons droits (uid gid du compte postfix)

<u>Petite précisions :</u>

Dans tous les cas on passe d'abord par les alias (virtual_alias_maps) donc si on veut qu'une adresse particulière du domaine virtuelle soit redirigée vers une adresse locale on peut le faire en plaçant une entrée dans la table des alias.
On peut egalement créer un compte fourre tout en plaçant le nom de domaine sur boite unique
ex: @etab.ac-amiens.fr   service
Mais attention, si on place cette entrée dans le fichier des alias plus rien n’arrive au domaine virtuel car les alias sont prioritaires sur les domaines virtuels.

Pour envoyer un message vers un script pour un traitement particulier:

Nous allons ici configurer notre serveur pour qu'il route un message vers un script classique Linux. Nous illustrons cette option par un exemple à travers une utilisation via Nagios. Nagios est application de supervision qui est capable d'envoyer une alerte par mail. Nous allons récupérer ce message d'alerte pour alimenter un fichier xml qui va servir d'entrée à un flux rss et qui va être affiché via un serveur web.
Un mail pour rss@in.ac-amiens.fr va être associé au transport « traitement » qui renvoie sur un pipe qui va servir un script en entrée. Le message est placé en entrée de la commande fluxrss.sh.

Dans master.cf:

```
traitement  unix  -     n     n     -     -     pipe  user=nobody argv=/usr/local/bin/fluxrss.sh
```

Exemple de fichier fluxrss.sh:

```
# on place le message reçu en entrée dans le fichier « /var/spool/mail/message_nagios »
tee > /var/spool/mail/message_nagios
# ici on va découper le message pour placer les infos dans des variables
Host=`cat /var/spool/mail/message_nagios |grep Host| sed 's/Host: //'`
Address=`cat /var/spool/mail/message_nagios |grep Address| sed 's/Address: //'`
State=`cat /var/spool/mail/message_nagios |grep State| sed 's/State: //'`
Service=`cat /var/spool/mail/message_nagios |grep Service| sed 's/Service: //'`
Nom=`cat /var/spool/mail/message_nagios |grep Nom| sed 's/Nom: //'`
Date=`cat /var/spool/mail/message_nagios |grep Time| sed 's/Date\/Time: //'`
#formatage du fichier rss. On le place dans le repertoire du serveur apache
sed -i '/\/channel/d' /usr/local/httpd/htdocs/fluxrss.xml
sed -i '/\/rss/d' /usr/local/httpd/htdocs/fluxrss.xml
echo " <item>" >> /usr/local/httpd/htdocs/fluxrss.xml
echo "  <title>$Address-$Service</title>" >> /usr/local/httpd/htdocs/fluxrss.xml
echo "    <link>http://nagios.agriates.ac-amiens.fr/nagios/cgi-bin/status.cgi?
host=$Nom#$Date</link>" >> /usr/local/httpd/htdocs/fluxrss.xml
echo "    <guid isPermaLink="False">$Service-$Date</guid>" >>
/usr/local/httpd/htdocs/fluxrss.xml
echo "      <description>" >> /usr/local/httpd/htdocs/fluxrss.xml
…
```

Exercice :

1- Sur le même principe pour générer une page html consultable via votre navigateur à travers votre serveur Web (Vous avez reçu un messade de xxx le contenu est xxx).
2- rechercher, télécharger et installer le script « répondeur » (écrit en Perl) qui permet d'envoyer un mail de réponse à un courrier reçu sur un compte particulier (revoir la notion de transport). Dans ce cas le traitement va permettre de récupérer l'émetteur du message, de construire un message de réponse et de l'envoyer par une commande du type « mail ».