28 juin 2008

Serveur SMTP : Routage des mails avec Postfix

Catégorie : Administration système Tags : GLMF

~~Retrouvez cet article dans : Linux Magazine 86~~

Dans un article précédent, nous avons abordé quelques éléments de configuration avancée de Postfix permettant de sécuriser la messagerie à l'intérieur d'un réseau. Toujours au travers de la résolution de problèmes de configuration, nous abordons ici les questions de routage du courrier.

Haute disponibilité avec les champs MX

Nous commencerons ici avec un petit rappel sur le fonctionnement du routage Internet. Pour expédier un message à utilisateur@domaine.com, le serveur de messagerie interroge le DNS pour trouver les échangeurs de messagerie correspondant à domaine.com. Exemple de requête MX :

[1] Cyrus-Imap est un logiciel serveur IMAP4 et gestionnaire de boîtes aux lettres. Il est un des rares à implémenter complètement et correctement la norme IMAP4. Il fournit également un serveur POP3 et un serveur de news.

```
$ host -t MX free.fr
free.fr mail is handled by 10 mx.free.fr.
free.fr mail is handled by 30 mrelay1-2.free.fr.
free.fr mail is handled by 30 mrelay2-1.free.fr.
free.fr mail is handled by 30 mrelay2-2.free.fr.
free.fr mail is handled by 40 mx1-1.free.fr.
```

Comme on le voit ici, un domaine dispose en général de plusieurs échangeurs de messagerie avec des poids respectifs (ici de 10 à 40). Le serveur va tenter d'expédier le message au serveur

disposant du poids le plus faible et en cas d'échec, essaie les autres dans l'ordre croissant des poids. Lorsque plusieurs serveurs ont le même, Postfix répartit la charge. Si la livraison échoue, le message est remis en file d'attente en attendant la tentative de livraison suivante. En revanche, si aucun enregistrement MX n'est trouvé dans le DNS, les échangeurs effectuent une recherche sur les enregistrements de type A (correspondant à des adresses du type utilisateur@machine).
Dans Postfix, ce comportement se traduit par la configuration par défaut suivante dans le fichier de configuration ~~main.cf~~ :

[2] La page des exemples de configuration standard est disponible ici : http://postfix.traduc.org/index.php/STANDARD_CONFIGURATION_README.html

```
default_transport = smtp
```

Ce paramètre s'applique à tous les messages dont Postfix n'est pas la destination finale. Dans le cas où notre serveur n'est pas un enregistré comme MX sur Internet, il y a de fortes chances pour que nous ne puissions pas livrer directement le courrier car les serveurs distants nous rejetteront. La solution consiste alors à passer par le serveur SMTP de notre fournisseur d'accès :

```
relayhost = smtp.mon.fai
```

Tel qu'est écrite cette ligne, si le serveur ~~smtp.mon.fai~~ ne répond pas, le courrier sera mis dans la file d'attente des messages en souffrance, puis renvoyé à l'expéditeur. On ne bénéficie donc pas en apparence de la haute disponibilité fournie par le mécanisme des champs MX. Or Postfix ne cherche pas en réalité à joindre ~~smtp.mon.fai~~, mais effectue la recherche standard comme indiquée ci-dessus, c'est-à-dire qu'il cherche d'abord des enregistrements MX correspondant au sous-domaine ~~smtp.mon.fai~~ et à défaut, se rabat sur les enregistrements de type A. Si on veut réellement utiliser une machine et demander à Postfix de n'effectuer que la recherche des enregistrements de type A, il faut utiliser les crochets ~~[]~~. La syntaxe correcte est donc :

```
relayhost = un.(sous).domaine
# ou
relayhost = [une.machine]
# ou encore
relayhost = [10.1.2.3]
```

Si on utilise Postfix sur une passerelle censée renvoyer le courrier interne à un autre serveur, on peut utiliser les champs MX pour créer un système de meilleure disponibilité. L'astuce consiste à créer un sous-domaine comme suit :

```
interne.mon.domaine.  MX  10  serveur1
interne.mon.domaine.  MX  10  serveur2
interne.mon.domaine.  MX  10  serveur3
```

La table transport de la passerelle contiendra alors :

```
mon.domaine   smtp:interne.mon.domaine
```

On peut bien entendu jouer sur les poids pour privilégier certains serveurs. Cette astuce est bien entendu valable dans toutes les directives appelant des transporteurs (voir plus bas) telles ~~content_filter, mailbox_transport~~, ~~local_transport~~, etc... ainsi que dans les tables utilisées avec ~~transport_maps~~.

Le système de transport de Postfix

Dans la formule default_transport = smtp, contrairement aux apparences, l'argument smtp ne désigne pas le protocole SMTP, mais le « transporteur » smtp livré avec Postfix. On peut mentionner ici n'importe quelle entrée du fichier de configuration master.cf de Postfix correspondant à un transporteur. Par exemple, on peut interdire la livraison locale en insérant local_transport = error dans main.cf (error est un démon qui rejette systématiquement le courrier).

Postfix propose dans master.cf plusieurs transporteurs : smtp, lmtp, local, error, virtual, relay.

Définition des transporteurs

Une entrée de master.cf est définie comme suit :

```
mysmtp  unix  -  -  -  -  -  smtp
   -o <éventuelle(s) option(s)>
```

Le premier argument nous donne le nom du service qui nous servira dans main.cf ou les tables de transport tandis que le dernier avant les options désigne le processus qui sera réellement lancé. Dans l'exemple ci-dessus, l'entrée mysmtp ne présente aucune différence avec smtp si ce n'est les options. En créant un transporteur dans master.cf, on peut le paramétrer en deux points :

- dans le fichier master.cf pour les paramètres passés au processus (smtp ici). Elles sont insérées chacune derrière un -o. Par exemple, on peut redéfinir smtp_helo_name : -o smtp_helo_name=autre.nom. On peut également agir sur les autres paramètres de la ligne tel le nombre maximum de processus lancés [6] ;
- dans main.cf pour les paramètres passés au gestionnaire des files d'attente (qmgr) : chaque transporteur dispose de ses propres paramètres concernant par exemple le nombre maximum de destinataires. Elles sont définies par transporteur_nom_d_option et ont comme valeur par défaut default_nom_d_option. Par exemple, pour ne pas dépasser 30 destinataires à la fois dans notre transporteur mysmtp, on utilise : mysmtp_destination_recipient_limit = 30 [4].

Petit exemple issu de la documentation de Postfix : supposons que notre serveur passerelle livre un grand volume de courrier à un serveur interne chargé et générant des messages type « 421 Server busy errors ». Le gestionnaire des files d'attente de Postfix risque de mettre régulièrement en liste noire ce serveur voire finir par retourner des messages aux expéditeurs. La solution proposée consiste à limiter le nombre de processus livrant le courrier vers cette destination et rendre Postfix plus tolérant quant au nombre d'erreurs.
Ainsi dans master.cf, on crée une entrée limitant par exemple à 10 le nombre de clients :

```
mysmtp  unix  -  -  -  -  10  smtp
```

puis dans main.cf, on augmente le nombre de tentatives :

```
initial_destination_concurrency = 200
mysmtp_destination_concurrency_limit=200
```

Ceci ne change rien pour les autres transporteurs qui restent limités par default_destination_concurrency_limit si ce n'est que le gestionnaire des files d'attente en lancera plus initialement. Postfix utilise lui-même ce mécanisme : la valeur par défaut du transporteur des

domaines relayés (relay_transport) n'est pas smtp, mais relay. Vous observerez que ces deux entrées de master.cf ne diffèrent pas initialement.

Desserte locale

Comme Sendmail et les autres logiciels de routage de courrier, le métier initial de Postfix n'est pas de gérer des boîtes aux lettres, mais de router le courrier.
Toutefois, il peut gérer le stockage du courrier selon les méthodes UNIX traditionnelles « mailbox » ou « maildir ». Par défaut, la première méthode est utilisée et les messages sont stockés dans les fichiers /var/spool/mail/user si l'utilisateur dispose d'un compte Unix local ; ils sont rejetés sinon. L'exemple proposé ici consiste à livrer le courrier local au système Cyrus-Imap [1] via le protocole lmtp.

[3] L'utilisation du mot-clef query dans les fichiers SQL n'est possible qu'à partir de la version 2.2 de Postfix. Pour les versions précédentes, elle est découpée en plusieurs champs (voir http://postfix.traduc.org/index.php/MYSQL_README.html).

Avant de commencer, étudions les caractéristiques de la livraison locale par défaut dans Postfix. Lorsqu'un message entrant correspond à $mydestination, Postfix le considère comme local. Ci-dessous les principales étapes de la livraison locale :

[4] La liste des paramètres du gestionnaire des files d'attente pouvant être personnalisés par transporteur peut être obtenue en consultant la page de manuel qmgr(8). Ces paramètres sont ceux qui commencent par transport_, mot à remplacer par le nom du transporteur créé.

- Dès la réception de la commande RCPT TO, Postfix vérifie que le message correspond à un utilisateur enregistré en consultant les tables passées au paramètre local_recipient_maps et le rejette sinon. Dans les versions 1.x de Postfix, ce paramètre était laissé vide. Le message n'était donc pas contrôlé à son arrivée, générant ainsi des accusés de non-remise pour lorsque le destinataire était inconnu. A partir de Postfix 2, local_recipient_maps pointe par défaut sur la base des comptes UNIX et celle des alias (/etc/aliases).
- Le gestionnaire des files d'attente attribue le message au transporteur déclaré dans local_transport : par défaut, c'est l'entrée local de master.cf qui appelle le démon du même nom.
- Le démon local gère ensuite les réécritures d'adresses si le destinataire se trouve dans /etc/aliases ou si l'utilisateur existe et dispose d'un fichier .forward.
- Si le destinataire est trouvé dans une table mailbox_transport_maps ou si le paramètre mailbox_transport est renseigné, local délègue la livraison.
- Sinon, le message est livré dans le fichier utilisateur du répertoire /var/spool/mail.

Pour livrer le courrier à Cyrus, nous devons donc choisir si nous souhaitons utiliser les alias ou non. Dans le dernier cas, il suffit de remplacer local dans local_transport :

```
local_transport =
   lmtp:unix:/var/cyrus/soc
```

où /var/cyrus/soc est la socket où Cyrus attend les messages, alors que pour bénéficier du mécanisme des alias, nous devons passer par local puis utiliser mailbox_transport :

```
mailbox_transport =
```

```
lmtp:unix:/var/cyrus/soc
```

Il nous reste à gérer le problème du rejet des destinataires ne disposant pas de comptes UNIX. Pour ce faire, deux solutions :

- Omettre la vérification en vidant le paramètre ~~local_recipient_maps~~. Cette solution présente l'inconvénient de devoir générer les accusés de non-remise.
- Renseigner ~~local_recipient_maps~~ avec une table renvoyant quelque chose si l'utilisateur est valide. On peut utiliser ici une table LDAP [voir encadré] mais, si votre serveur est très chargé, il est préférable d'interroger régulièrement l'annuaire avec un script cron qui crée un fichier local car les consultations LDAP ralentissent le traitement du courrier et nécessitent un serveur LDAP très disponible. Comme Postfix ne tient pas compte de la valeur renvoyée, vous pouvez utiliser une table construite à d'autres fins (contrôle d'accès, etc.).

[5] Sur Debian, seul le support des expressions régulières POSIX est installé par défaut avec Postfix. Pour ajouter le support des expressions régulières Perl : apt-get install postfix-pcre . Vous ne pensiez tout de même pas que j'allais oublier de citer ma distribution préférée ;-)
Simple non ?

Utiliser une table LDAP

L'emploi d'un annuaire LDAP pour effectuer des réécritures d'adresses ne pose pas de difficultés particulières.
Toutefois, Postfix attend de l'annuaire LDAP un résultat correspondant à ce qu'il cherche ; ainsi dans une table de transport, le résultat attendu est du type ~~lmtp:unix:/socket~~ ou ~~smtp:[serveur]~~.
Exemple de gestion des alias :

```
# main.cf
alias_maps =
      proxy:ldap:/etc/postfix/ldap.cf

# /etc/postfix/ldap.cf
# Connexion
server_host      = ldap.mon.domaine
search_base      = dc=mon,dc=domaine
bind             = no
# requête LDAP
query_filter     = (mail=%s)
# attribut LDAP contenant le résultat
result_attribute = mailbox
```

Le système est très simple : on appelle un fichier qui contient les paramètres de connexion, la requête à effectuer et l'attribut à lire. La variable ~~%s~~ sera remplacée par l'adresse examinée. Pour connaître tous les attributs utilisables dans le fichier, consultez la page de manuel ~~ldap_table(5)~~.
L'utilisation du service ~~proxy~~ permet de mutualiser les connexions au serveur LDAP [voir encadré].

Gérer plusieurs systèmes dans un même domaine

Dans cet exemple, nous sommes confrontés à une migration progressive du système interne de

messagerie sans interruption globale de service. Le serveur Postfix que nous configurons est une passerelle qui va devoir gérer l'aiguillage. Nous nous trouvons donc avec des utilisateurs du même domaine mais répartis sur plusieurs systèmes, un routage habituel des messages avec les tables ~~transport_maps~~ ne convient plus.
Toutefois, depuis la version 2, Postfix consulte toujours ces tables avec l'adresse mail avant d'utiliser le domaine de destination. Pour le courrier extérieur entrant, la résolution du problème est donc extrêmement simple : nous allons établir des tables de consultation nous renvoyant vers le bon système de messagerie :

```
# main.cf
transport_maps = /etc/postfix/transport
# /etc/postfix/transport
user1@domaine.fr   smtp:mx-système1
user2@domaine.fr   smtp:mx-système2
...
```

On peut également utiliser une table LDAP, avec la réserve indiquée dans l'encadré, ou encore une table SQL. Nous utilisons toujours des champs MX comme proposé dans le premier paragraphe de cet article pour garantir une certaine disponibilité. Pour le courrier sortant, nous devons indiquer aux deux systèmes de passer par notre passerelle, ce qui ne pose pas de difficultés. Le problème majeur restant est d'expliquer aux deux systèmes qu'ils ne détiennent pas la totalité de la base des comptes. Quelques exemples : cette configuration ne pose pas de problèmes à Postfix (il suffit d'écrire la table de transport) ni à Exchange 2000 qui dispose d'un paramètre permettant de renvoyer à un autre serveur le courrier du domaine n'ayant pas trouvé de destinataire (on peut aussi renseigner l'annuaire avec les paramètres de l'autre serveur). En revanche, pour Exchange 5, cette configuration est plus délicate. L'astuce consiste à changer le domaine qu'il considère être le sien :

- Chaque utilisateur dispose de deux adresses dans l'annuaire : user1@domaine.fr et user1@bidon.fr.

Utiliser une table MySQL

Contrairement à LDAP, les requêtes SQL nous donnent plus de souplesse pour gérer les tables :

```
# main.cf
alias_maps =
      proxy:mysql:/etc/postfix/mysql

# /etc/postfix/mysql
user = nom-de-connexion
password = mot-de-passe
dbname = nom-de-la-BD
query = SELECT 'smtp:[' mailhost ']' \
        FROM users WHERE mail='%s'
```

On obtient ainsi ce qu'attend Postfix (« ~~smtp:[machine]~~ ») à partir d'un enregistrement ne contenant qu'un nom de machine [3].

Mutualiser les consultations de table

Dans beaucoup de consultations de tables, on peut mutualiser les connexions en utilisant le démon ~~proxymap~~. Il présente deux intérêts :

- pouvoir consulter une table qui se trouve en dehors de la cage ~~chroot~~,
- mutualiser les connexions.

Dans le cas de requêtes SQL ou LDAP, ce service permet de conserver des connexions ouvertes évitant la renégociation d'une session à chaque création d'un nouveau processus. Pour l'utiliser, il suffit de faire précéder la déclaration de la table du préfixe ~~proxy:~~ :

```
alias_maps =
     proxy:mysql:/etc/postfix/mysql
```

On ne peut en revanche l'utiliser partout. La liste des paramètres de ~~main.cf~~ acceptant ce service peut être obtenue avec la commande ~~postconf -d proxy_read_maps~~.

- Le domaine interne d'Exchange est bidon.fr.
- Les adresses sortantes sont réécrites de user1@bidon.fr en user1@domaine.fr.

Cette configuration fonctionne parfaitement tant qu'Exchange ne se perd pas dans son annuaire. Si vous gérez plusieurs « sites » au sens Exchange du terme, il risque fort de s'y perdre. Pour corriger ce bug, il suffit de récrire les adresses type ~~user1@domaine.fr~~ en ~~user1@bidon.fr~~. Ces modifications d'adresses sont sans conséquences pour les utilisateurs d'Outlook qui ne voient jamais leur adresse mail ni pour les utilisateurs POP/IMAP dont la configuration est locale. Exemple de réécriture avec un peu d'expressions régulières compatibles Perl [5] :

```
# main.cf
sender_canonical_maps = \
   pcre:/etc/postfix/expedition
recipient_canonical_maps = \
   hash:/etc/postfix/reception
# /etc/postfix/expedition
/^([^@]+)@bidon\.fr/ $1@domaine.fr
# /etc/postfix/reception
user2@domaine.fr user2@bidon.fr
...
```

On ne peut utiliser d'expressions régulières pour les messages entrants car cela ne concerne qu'une partie de nos utilisateurs. En revanche, ces réécritures nous évitent le routage par utilisateur car, en sortant de Postfix, les utilisateurs des deux systèmes ne se trouvent plus dans le même domaine :

```
# /etc/postfix/transport
bidon.fr   smtp:mx-exchange5
domaine.fr smtp:mx-exchange2k
```

Duplication de messages

Postfix offre la possibilité d'effectuer à la volée des copies cachées (dites « BCC »). On peut intégralement dupliquer le courrier passant par Postfix en utilisant ~~always_bcc = adresse@domaine~~. Outre l'aspect légal de cet espionnage, sur un serveur chargé, le volume reçu peut être considérable et générer des rejets de la boîte aux lettres ~~adresse@domaine~~. L'expéditeur recevra alors un avis lui montrant que son courrier a été dupliqué. Le problème posé dans l'exemple ci-dessous est de créer une base de sauvegarde de certaines boîtes aux lettres afin de garder une trace des messages effacés. On souhaite ainsi conserver un historique complet de hotlines sans que les utilisateurs de ces boîtes ne se trouvent devant une interface de plusieurs

milliers de messages. L'idée est donc de créer une copie cachée des messages émis ou reçus par cette boîte :

[6] Pour connaître les paramètres de main.cf qui agissent sur un processus particulier, il suffit de consulter la page de manuel correspondant à ce processus. Par exemple, pour lmtp, regardez lmtp(8).

```
 sender_bcc_maps = \
    pcre:/etc/postfix/hotlines
recipient_bcc_maps = $sender_bcc_maps
# /etc/postfix/hotlines
/^(hotline[^@]+)@domaine.fr$/ $1@sauv.fr
```

L'astuce utilisée ici consiste à créer un domaine de messagerie (~~sauv.fr~~) dédié à la sauvegarde. Le Postfix correspondant à ce domaine distribue normalement les messages dans des boîtes portant le même nom mais dans le domaine ~~sauv.fr~~. Enfin, pour que les utilisateurs puissent chercher dans ces boîtes de secours sans pouvoir effacer les messages, on peut retirer le droit « d » dans le cas où on utilise Cyrus-Imap. Reste à configurer un petit webmail et le tour est joué !

Conclusion

Cet article ne couvre bien entendu pas l'étendue des possibilités de configuration du routage dans Postfix, mais comme les exemples de la documentation en ligne [2] apportent une solution à la plupart des configurations standards, j'ai choisi quelques problèmes qui pourront peut-être vous donner des idées si vous en rencontrez de similaires. La découverte du fonctionnement du fichier ~~master.cf~~ nous servira également dans le prochain article pour intégrer des processus tels les antivirus.

Liens:

- Documentation de Postfix : http://www.postfix.org/documentation.html
- Traduction de la documentation par l'auteur : http://postfix.traduc.org/
- Cyrus-Imap : http://asg.web.cmu.edu/cyrus/imapd/
- Postfix avec Cyrus-Imap : http://www.linuxorable.net/1-Cyrus-IMAP-58-installer-Cyrus.html

~~Retrouvez cet article dans : Linux Magazine 86~~

Posté par (La rédaction) | Signature : Xavier Guimard | Article paru dans 

Laissez une réponse

Vous devez avoir ouvert une session pour écrire un commentaire.

« Précédent Aller au contenu »

Identifiez-vous
Inscription
S'abonner à UNIX Garden

- # Articles de 1ère page

- Lutter contre le spam avec Postfix
- Yafray, le moteur de rendu photoréaliste libre, une première approche
- Ajouter des logiciels : la gestion des paquets
- Web : récupérez vos marque-pages
- Une nouvelle disposition de clavier français pour Xorg
- Installation et configuration d'E17
- Migration des données
- Ubuntu, un peu d'histoire...
- H.P.Anvin : M. ISOLinux / SYSLinux
- Les marques déposées et les Logiciels libres

- # Il y a actuellement

- 639 articles/billets en ligne.

Recherche

- # Catégories

- Administration réseau
- Administration système
- Agenda-Interview
- Audio-vidéo
- Bureautique
- Comprendre
- Distribution
- Embarqué
- Environnement de bureau
- Graphisme
- Jeux
- Matériel
- News
- Programmation
- Réfléchir
- Sécurité
- Utilitaires
- Web

Archives

- juillet 2008
- juin 2008
- mai 2008
- avril 2008
- mars 2008
- février 2008
- janvier 2008
- décembre 2007
- novembre 2007
- février 2007

GNU/Linux Magazine

- GNU/Linux Magazine 107 - Juillet/Août 2008 - Chez votre marchand de journaux
- Edito : GNU/Linux Magazine 107
- GNU/Linux Magazine HS 37 - Juillet/Août 2008 - Chez votre marchand de journaux
- Edito : GNU/Linux Magazine HS 37
- GNU/Linux Magazine 106 - Juin 2008 - Chez votre marchand de journaux !

GNU/Linux Pratique

- Linux Pratique N°48 -Juillet/Août 2008 - Chez votre marchand de journaux
- Edito : Linux Pratique N°48
- Linux Pratique Essentiel N°2 - Juin/Juillet 2008 - Chez votre marchand de journaux
- Edito : Linux Pratique Essentiel N°2
- Linux Pratique Hors-Série N°15 - Juin / Juillet 2008 - chez votre marchand de

journaux.

- **MISC Magazine**

  - Références de l'article « Détection de malware par analyse système » d'Arnaud Pilon paru dans MISC 38
  - Références de l'article « La sécurité des communications vocales (3) : techniques numériques » d'Éric Filiol paru dans MISC 38
  - Misc 38 : Codes Malicieux, quoi de neuf ? - Juillet/Août 2008 - Chez votre marchand de journaux
  - Edito : Misc 38
  - Misc 37 : Déni de service - Mai/Juin 2008 - Chez votre marchand de journaux