Configurer sendmail avec les macros m4

23 mai 2001

Table des matières

1 Configurer le mél en local uniquement 2

2 Configurer le mél pour la maison 3

3 Configurer sendmail pour un réseau local 4

4 Réécrire les adresses et le nom de domaine pour les courriers sortants 5

5 Réécrire les logins en Prenom.Nom 7

6 Accepter et regiriger les méls adressés à Prenom.Nom 8

7 S'assurer que les méls locaux restent en local 9

8 Aller encore plus loin 11

9 La lutte anti-spam 14

10 Configurer des hôtes virtuels 15

Nous avons vu il y a quelque temps (`http://www.toolinux.com/linutile/reseau/mail/sendmail/sendmail-kit/`) comment configurer `sendmail` à l'aide du kit jussieu (`http://www.prism.uvsq.fr/~pda/kit-jussieu/`). Voici comment configurer `sendmail` en utilisant les macros `m4`. Le principe est simple : on définit un certain nombre d'options dans un fichier, et on le passe au pré-processeur `m4` qui affiche sur la sortie standard le contenu du fichier `sendmail.cf` qu'on cherche à générer.

Pour commencer, on cherche à configurer le mél en local uniquement, puis en utilisant le serveur de méls du fournisseur d'accès comme relai pour les méls extérieur, enfin on fera en sorte que les méls de notre réseau local soient correctement acheminés vers l'extérieur. Noter qu'à chaque nouvelle section viennent s'ajouter de nouvelles variables qui seront détaillées au fur et à mesure. La lecture des premières sections permet donc de comprendre la signification de certains paramètres utilisés par la suite.

La documentation de référence est le fichier `README`, contenu dans le répertoire de configuration de `sendmail` (`/usr/lib/sendmail-cf/README` sous RedHat, par exemple).

1 Configurer le mél en local uniquement

Pour configurer le mél en local, il n'y a en fait que deux variables à positionner :

OSTYPE('linux') ceci précise quel système d'exploitation est utilisé. En fait, c'est surtout le nom du fichier contenant les informationss sur le système d'exploitation qui est précisé (répertoires des fichiers de configuration, chemins et arguments des mailers, répertoires utilisés par `sendmail` pour stocker les messages...). Cette option est donc obligatoire et devrait apparaître en début de tout fichier de configuration. Ici, ce sera le fichier `linux.mc`, mais avec la Debian par exemple, ce fichier s'appelle `debian.mc`. Il faudra donc mettre `OSTYPE('debian')` ;

MAILER(local) par défaut, le mél est distribué en local. Par souci de clareté, il est sage de le spécifier. Ceci inclut à la fois l'agent de distribution local, utilisé pour envoyer des méls dans la boîte aux lettres des utilisateurs, et le programme utilisé pour envoyer les messages aux programmes locaux.

Comme la plupart des systèmes ont besoin d'avoir au moins la distribution de méls en local, ces deux variables seront positionnées dans la plupart des configurations.

Voici le fichier `local.mc` utilisé pour écrire le fichier `sendmail.cf` :

```
OSTYPE(linux)dnl
MAILER(local)dnl
```

On peut maintenant générer le fichier `sendmail.cf`. Pour cela, il faut chercher le fichier `cf.m4` :

```
# locate cf.m4
/usr/share/sendmail/sendmail.cf/m4/cf.m4
#
```

puis compiler notre fichier à l'aide du pré-processeur `m4` :

```
# m4 /usr/share/sendmail/sendmail.cf/m4/cf.m4 local.mc > sendmail.cf
#
```

Il ne reste alors plus qu'à copier ce fichier dans le répertoire `/etc` et de relancer sendmail pour tester :

```
# /etc/init.d/sendmail restart
Stopping mail transport agent: sendmail.
Starting mail transport agent: sendmail.
#
```

2 Configurer le mél pour la maison

```
OSTYPE(`linux')dnl
FEATURE(nocanonify)dnl
FEATURE(always_add_domain)dnl
FEATURE(local_procmail)dnl
GENERICS_DOMAIN(machine.domaine machine localhost)
define(`SMART_HOST', `smtp8:[mail.fai.fr]')dnl
MAILER(local)
MAILER(smtp)
```

FEATURE(nocanonify) D'habitude, `sendmail` tente de compléter toute les adresses composées du nom d'hôte seul (de la forme **machine**) en leur nom pleinement qualifié (de la forme **machine.domaine**). Pour cela, il consultant un serveur de noms et tente de résoudre les noms d'hôtes (de la forme **machine**) en leur nom pleinement qualifié (de la forme **machine.domaine**). Dans notre cas, on ne veut pas de cette option car on n'a pas de DNS à disposition. Ceci suppose donc que l'on fournisse toujours à `sendmail` des adresses canoniques ;

FEATURE(always_add_domain) Si cette option est définie, le nom de domaine est ajouté même pour les méls délivrés en local ;

FEATURE(local_procmail) Cette option indique à sendmail d'utiliser `procmail` pour délivrer les couriers à la machine ; `procmail` permet de filtrer les méls de chaque utilisateur : triage, suppression automatique selon les champs *From :*, *To :* de chaque mél...(voir `http://www.toolinux.com/linutile/reseau/mail/procmail/index.htm`) ;

GENERICS_DOMAIN(tarsier.athome.chezmoi tarsier localhost) Liste des nom de domaines pour lesquels on doit pouvoir accepter les méls ;

SMART_HOST Définit quel hôte peut être utilisé pour relayer nos courriers vers l'extérieur. Si on utilise un "Smart Host", on n'a pas besoin de DNS. Cette solution est souvent utilisée dans le cas d'une connexion via modem ou si on est derrière un firewall. Ici, on utilise SMTP8 pour envoyer des courriers à **mail.fai.fr** contenant des caractères codés sur 8 bits ;

MAILER(smtp) Le moyen le plus utilisé pour transporter du mél sur Internet est SMTP. En incluant cet agent de transport, quatre mailers sont configurés : smtp (le SMTP basique), esmtp (Extended SMTP, qui permet entre autres d'utiliser les accents), smtp8 (SMTP sur 8 bits) et relay (prédestiné pour relayer des messages entre des hôtes) ;

3 Configurer sendmail pour un réseau local

Pour une configuration plus poussée, nous allons utiliser le fichier `tcpproto.mc` fournit avec `sendmail` pour faire notre fichier `host.mc`. Voici le contenu de ce fichier :

```
divert(-1)
#
# Copyright (c) 1998 Sendmail, Inc.  All rights reserved.
```

```
# Copyright (c) 1983 Eric P. Allman.  All rights reserved.
# Copyright (c) 1988, 1993
#       The Regents of the University of California.  All rights reserved.
#
# By using this file, you agree to the terms and conditions set
# forth in the LICENSE file which can be found at the top level of
# the sendmail distribution.
#
#

divert(0)dnl
VERSIONID(`@(#)tcpproto.mc      8.10 (Berkeley) 5/19/1998')
OSTYPE(unknown)
FEATURE(nouucp)
MAILER(local)
MAILER(smtp)
```

Détaillons les nouvelles options :

VERSIONID Cette macro optionelle est utile pour enregistrer le numéro de la version du fichier source de configuration `.mc`. Ce numéro est affiché entre autres lors d'une connexion via `telnet` :

En général, les numéros de versions utilisés sont au format RCS ou SCCS ;

FEATURE(nouucp) `sendmail` peut être configuré pour être dégourdi avec les adresses UUCP (un vieux protocole plus trop utilisé), ou il peut ne rien faire de spécial. Si on ne se préocupe pas des adresses UUCP, ou si vous n'avez jamais entendu parler de ce protocole, il faut laisser *nouucp*.

4 Réécrire les adresses et le nom de domaine pour les courriers sortants

On peut rajouter des options pour être plus "sioux". Voici des améliorations classiques que l'on peut apporter à notre configuration :

- le nom d'hôte de la machine expéditeur est réécrit en nom de domaine dans l'en-tête et dans l'enveloppe pour tous les méls qui sortent. Par exemple, un mél envoyé de **machine.domaine** par l'utilisateur **utilisateur** aura pour champ From **utilisateur@domaine** au lieu de **utilisateur@machine.domaine**. Cette option est classique, et permet de rendre indépendante d'un nom de machine du réseau local, si bien que si celle-ci vient à disparaître ou à être renomée, l'adresse mél reste valide ;

– le nom de l'utilisateur envoyant un mél sera réécrit sous la forme *Prénom.Nom*. Par exemple, l'utilisateur Jean Bon ayant pour login **jbon** enverra des méls avec un champ From **Jean.Bon@domaine** au lieu de **jbon@domaine** ;

– les méls adressés à **Prénom.Nom@domaine** doivent donc être acceptés et redirigés vers les boîtes aux lettres de l'utilisateur correspondant ;

Pour faire cela, il est nécessaire d'apporter quelque modifications à notre fichier de configuration `host.mc`. Pour cela, on rajoute simplement une ligne `DOMAIN` contenant le nom d'un fichier (par exemple le nom du domaine), et on édite un nouveau fichier `domaine.m4`. Voici donc le contenu du fichier `host.mc` :

```
VERSIONID(`@(#)tcpproto.mc      8.10 (Berkeley) 5/19/1998')
OSTYPE(linux)
DOMAIN(domaine)
FEATURE(nouucp)
MAILER(local)
MAILER(smtp)
```

Et voici le contenu du fichier `domaine.m4` :

```
MASQUERADE_AS(domaine)
FEATURE(masquerade_envelope)
FEATURE(genericstable)
```

Ce fichier doit se trouver dans le répertoire prévu à cet effet. Sous Debian, c'est le répertoire `/usr/share/sendmail/sendmail.cf/domain`, comme l'indique le message d'erreur se produisant si le fichier n'est pas au bon endroit :

```
$ m4 /usr/share/sendmail/sendmail.cf/m4/cf.m4 host.mc > sendmail.cf
$
```

Pour trouver le répertoire sous d'autre distributions, chercher le fichier `generic.m4` (avec la commande `locate`, par exemple), qui devrait se trouver dans ce même répertoire.

Détaillons les nouvelles options :

DOMAIN(domaine) Indique le nom du fichier contenant des options spécifiques au domaine. Ce peut être interessant si on souhaite configurer le service de courrier électronique pour un grand nombre de machines dans notre réseau ;

MASQUERADE_AS(domaine) Réécrit le nom du domaine dans le champ From en *domaine*, au lieu de *machine.domaine*.

Ce nom de domaine est donc le nom de domaine à montrer après le @ dans le champ From : des méls sortants : une personne recevant un mél verra From : login@domaine. Ce domaine devrait être le nom de domaine pleinement qualifié (**machine.domaine**) si ce domaine est effectivement enrégistré par un DNS, ou le nom de domaine du fournisseur d'accès. Dans le cas d'un intranet avec un nom de domaine "bidon" (car enrégistré par aucun DNS de l'Internet), il faut cacher ce nom car des utilisateurs vérifient les noms de domaines des méls qu'ils reçoivent, sous peine de les mettre directement à la poubelle.

Cette option est employée avec la caractéristique `genericstable` ;

FEATURE(masquerade_envelope) Si on veut que l'enveloppe des méls (ainsi que les en-têtes) laissent croire qu'ils viennent d'un nom de domaine particulier, il faut préciser cette option ;

FEATURE(genericstable) Permet d'utiliser la caractéristique *genericstable*, qui sert à réécrire les adresses méls des utilisateurs expédiant des couriers. La mise en oeuvre de cette caractéristique est détaillée dans la section suivante. Noter que les noms et chemins de la base de donnée peuvent être spécifiés comme suit : `FEATURE(genericstable, 'hash -o /etc/mabd')` pour une base de données se trouvant dans le fichier `/etc/mabd.db` ;

5 Réécrire les logins en Prenom.Nom

Pour réécrire les logins des utilisateurs en leur Prenom.Nom, il faut une base de données faisant la correspondance. Cette base de données peut être générée automatiquement à partir du fichier `/etc/passwd`, voir...

La première étape est d'écrire un fichier qu'on peut appeller `genericstable`, à placer dans le répertoire `/etc` ou `/etc/mail`. Ce fichier existe sans doutes déjà :

```
# locate genericstable
/etc/mail/genericstable
/etc/mail/genericstable.db
#
```

On voit ici que le fichier existe bien déjà. On constate aussi que la base de données associée, `genericstable.db`, se trouve dans ce même répertoire.

Ce fichier `genericstable` peut ressembler à ceci :

```
jbon: Jean.Bon
jdeuf: John.Deuf@domaine1.com
```

Il faut maintenant générer la base de données à l'aide de `sendmail` à partir du répertoire où se trouve le fichier `genericstable` (`/etc` ou `/etc/mail`, en général) :

```
# sendmail -bi -oA/etc/genericstable
#
```

Une autre solution qui revient au même (à vérifier...) consiste à écrire un fichier ayant un format légèrement différent (sans les " :") et d'appeller `makemap`.

Voici le fichier `genericstable` :

```
jbon  Jean.Bon
jdeuf John.Deuf@domaine1.com
```

Et voici comment générer la base de données, toujours à partir du répertoire où se trouve le fichier `genericstable` :

```
# makemap hash /etc/mail/genericstable < /etc/mail/genericstable
#
```

6 Accepter et regiriger les méls adressés à Prenom.Nom

C'est la réciproque de l'opération précédente.

Pour faire en sorte que les méls adressés à **Prénom.Nom@domaine** soient redirigés vers la boîte aux lettre du bon utilisateur, il faut éditer le fichier `/etc/aliases` :

```
Jean.Bon@domaine: jbon
John.Deuf@domaine: jdeuf

# Tous les mels adresses a root sont redistribues a sysadm
```

```
root: sysadm

# Tous les mels adresses a sysadm sont redistribues a admin et moi

sysadm: admin moi

# Une liste de discussion

admin: jbon, Harry.Cover@labas.com

# Pour moi, on execute le script /home/moi/monfiltre

moi: |/home/moi/monfiltre
```

Encore une fois, ce fichier existe sans doute déjà, avec des alias prédéfinits (comme **postmaster** qui est plus que conseillé par la RFC...). Il faut donc sans doutes éditer le fichier existant.

Si un alias commence par "|", c'est un script qui sera lancé.

Lancer la commande `newaliases` (équivalente à `sendmail -bi`) pour mettre à jour la base de données (`/etc/aliases.db`).

Et voilà, ce n'est pas plus compliqué que ça ! Il ne reste plus qu'à recompiler notre fichier `host.mc` et à relancer `sendmail` comme précédement :

```
# m4 /usr/share/sendmail/sendmail.cf/m4/cf.m4 host.mc > sendmail.cf
# /etc/init.d/sendmail restart
Stopping mail transport agent: sendmail.
Starting mail transport agent: sendmail.
#
```

7 S'assurer que les méls locaux restent en local

Le problème avec la configuration précédente, c'est que si un utilisateur **user1** veut envoyer un mél à un autre utilisateur en local **user2**, l'adresse du de l'expéditeur sera réécrite en nom de domaine officielement enrégistré par un DNS (celui du fournisseur d'accès dans les exemples qu'on a vu), par exemple **user1@fai.fr**. Si l'utilisateur veut répondre à ce mél, il va répondreà **user1@fai.fr**, ce qui implique que le mél partira vers le serveur de méls du fournisseur d'accès pour revenir dans notre réseau local. Cela marchera, mais présente deux inconvégnients : le premier est que le mél va faire du trajet pour rien, prenant de la bande passante inutilement et mettant du temps à être acheminé ; le second est que si la connexion avec le fournisseur d'accès est interompue (par soucis d'économie, par exemple), le courrier ne sera acheminé que lorsque la connexion sera rétablie.

Pour remédier à ce problème, il y a plusieurs solutions :

– En répondant à un mél, changer l'adresse du destinataire à la main. Cela peut s'avérer fastidieux si on envoie souvent des méls en local (pour un réseau local d'une entreprise, par exemple) ;
– Ecrire un script scrutant à intervalles réguliers la file d'attente des méls en partance (et donc bloqués en cas de non connexion avec le fournisseur d'accès) et réécrivant les champs si le méls est destiné à un utilisateur local. Mais en cas de connexion, le mél transite quand même par le fournisseur d'accès, ce qui est dommage ;
– Utiliser un principe similaire aux `genericstable` pour réécrire les adresses de l'enveloppe. C'est cette dernière méthode qui est présentée par la suite ;

Pour cela, il faut rajouter les lignes suivantes à la fin du fichier de configuration à passer à `m4` :

```
Kinterneuser hash -o /etc/mail/interneuser
LOCAL_RULE_0
R$+ < @ $+ . > $*        $: < $(interneuser $1 @ $2 $: $) > $1 < @ $2 . > $3
R$+ < @ $+ $~. > $*      $: < $(interneuser $1 @ $2 $3 $: $) > $1 < @ $2 $3 >$4
R< $+ > $+ < @ $+ > $*   $@ $>97 $1 $4
R<> $+ < @ $+ > $*       $: $1 < @ $2 > $3
```

Il faut ensuite éditer le fichier qui va servir à générer la base de données servant à la réécriture des adresses. Le fichier s'appelle `interneuser` :

```
loginlabas@fai.fr loginici
```

Ce fichier est à placer dans le répertoire `/etc/mail`, par exemple. Pour compiler la base de données, on fait :

```
# makemap hash /etc/mail/interneuser < /etc/mail/interneuser
#
```

De même, pour s'assurer que tout les mél de notre domaine est délivré en local, et que tout le reste passe par le ***SMART HOST***, on peut rajouter ceci :

```
LOCAL_NET_CONFIG
R$* < @ $* .$m. > $*     $#smtp $@ $2.$m. $: $1 < @ $2.$m. > $3
```

Il s'agit d'une règle de réécriture comme `sendmail` les aimes. Se reporter au guide d'administration réseau sous Linux [1] pour plus de détails.

8 Aller encore plus loin

Cette section présente des options qui peuvent s'avérer utiles.

define('SMTP_MAILER_FLAGS', 'e9') Les courriers envoyés par smtp sont coûteux à envoyer ('e' pour *expensive*). L'option **9** force la conversion 8 vers 7 bits sur le contenu du texte ;

FEATURE(allmasquerade) Si cette option est définie, les adresses des destinataires sembleront aussi provenir du nom de domaine précisé par `genericstable`, au lieu du nom d'hôte local.

Cela peut outrepasser les alias locaux. Par exemple, si on envoie un mél à **aliaslocal**, `sendmail` va trouver que cet alias est associé à **aliaslocal@domaine**. Le mél sera alors envoyé à **login@autredomaine** avec un champ To : **aliaslocal@domaine** au lieu de **login@autredomaine**. Si cette adresse n'existe pas, une réponse à **aliaslocal@domaine** échouera. Il ne faut donc utiliser cette option QUE si on peut garantir que TOUTE les adresses **aliaslocal@domaine** fonctionneront. En cas de doute, il est plus sage de ne pas utiliser cette option.

Cw D'habitude, `sendmail` est configuré pour accepter les méls pour notre nom de domaine. Cependant, dans certains cas, on ne veut pas que sendmail le fasse, en particulier si le mél pour le domaine est renvoyé par un champ MX du DNS vers un autre hôte. En cas de doutes, il est plus sage de laisser cette option de coté.

FEATURE(use_cw_file) En plus du nom de domaine local, on peut ajouter un certain nombre d'autre noms de domaine à reconnaître pour les méls arrivant. Si d'autre hôtes ont un champ MX du DNS pointant vers le nom de domaine local, c'est ici qu'il faut les lister.

Cette liste est sauvegardée dans le fichier `/etc/mail/sendmail.cw` de façon à pouvoir être éventuellement changé plus tard.

FEATURE(use_ct_file) `sendmail` autorise un groupe spécial d'utilisateurs à modifier l'adresse de leur enveloppe "From" en utilisant l'option *-f* sans générer de message d'erreur. Un programme tel que `Majordomo` peut avoir besoin de cela. Noter que les utilisateurs **root**, **daemon** et **uucp** sont inclus d'office, si bien que on n'a pas besoin de les spécifier.

Cette liste est sauvegardée dans le fichier `/etc/sendmail.ct` ou `/etc/mail/trusted-users` de façon à pouvoir être éventuellement changée plus tard.

FEATURE(redirect) Cette option permet de faire un alias avec la nouvelle adresse d'un utilisateur. `sendmail` retourne alors un mél d'erreur à l'envoyeur lui indiquant la nouvelle adresse du destinataire. Pour mettre en oeuvre cette solution, il faut ajouter une ligne telle que celle ci dans le fichier `/etc/alias` :

```
plusla@domaine     Ilestla@tatoune.com.REDIRECT
```

Un message d'erreur sera donc envoyé à l'expéditeur lui indiquant la nouvelle adresse ;

FEATURE(stickyhost) Si cette option est définie, les méls adressés à **login@domaine** sont marqués comme "sticky". Cela signifie que les adresses locales ne sont pas en conflit avec UDB et ne passent pas par la règle de réécriture 5.

Ceci est utilisé si on veut une configuration où **login** n'est pas forcémment le même que **login@domaine**.

Lorsque cette fonctionalité est spécifiée, `LOCAL_RELAY` gère toute les adresses locales n'ayant pas de nom de domaine, et `MAIL_HUB` gère toute les adresses locales comportant un nom de machine.

En cas de doutes, il est plus sage de laisser cette option de coté.

FEATURE(nodns) Cette option doit être définie si on est connecté directement à l'Internet et que l'on a accès à un serveur de noms.

FEATURE(bestmx_is_local) Cette option va permettre à `sendmail` d'accepter les méls de tous les hôtes qui listent la machine locale comme le meilleur échangeur de méls possible par le champ MX d'un DNS. Cela génère du trafique DNS supplémentaire, mais devrait être bon pour des hôtes ayant peu de trafique. Cela revient en fait à accepter le courrier comme si il était adressé localement pour toute machine dont on est le serveur MX ;

FEATURE(mailertable) Cette option va permettre à `sendmail` de lire les règles de routage dans le fichier `/etc/mail/mailertable`. Ceci est nécessaire pour des mailers inhabituels comme `ifmail` ou tous les programmes de fax.

Pour plus d'information, voir le fichier `op.txt.gz`.

FEATURE(smrsh) Cette option va permettre à `sendmail` d'utiliser un shell restrictif (`smrsh`) au lieu de `/bin/sh` pour envoyer des méls aux programmes. Ceci accroît la possibilité de contôler ce qui transite via le mél. Ainsi, seuls les programmes se trouvant dans un répertoire spécial peuvent être lancés. Si cette option est définie, il est conseillé de lire attentivement la page de manuel de smrsh (8) pour plus d'informations.

define('confMAILER_NAME', 'Postmaster') Cette option sert à changer le nom utilisé pour les messages sortants générés en interne. Ce ne serait une mauvaise idée de mettre **Postmaster**.

define('confCON_EXPENSIVE', 'True') Indique au système de conserver le courrier en partance provenant des mailers ayant l'indicateur *e* monté, jusqu'à ce que la file soit réactivée ;

define('confME_TOO', 'True') Normalement, `sendmail` n'envoie pas les messages de l'envoyeur d'une liste de diffusion. Cette option est utilisée pour que l'envoyeur reçoive aussi les messages.

define('confCOPY_ERRORS_TO', 'Postmaster') Adresse à laquelle il faut envoyer une copie des messages d'erreur ;

define('confDEF_CHAR_SET', 'ISO-8859-1') Table de caractères par défaut pour les données MIME 8-bits sans étiquetage particulier ;

define('confMIME_FORMAT_ERRORS','True') Indique au système d'envoyer les messages d'erreur (destinataire non trouvé, par exemple) avec une encapsulation MIME ;

define('confTO_QUEUEWARN', '24h') `sendmail` enverra un message à l'envoyeur si il ne peut délivrer un message dans un intervalle de temps raisonnable. Il enverra aussi en message d'erreur pour dire qu'il abandonne les essais d'envoyer un message si il ne peut le livrer dans un intervalle de temps raisonnable. Les intervalles de temps par défaut sont respectivement de 4 heures et 5 jours (4h/5d), mais beaucoup de gens trouvent qu'un avertissement après seulement 4 heures est un peu prématuré.

9 La lutte anti-spam

`sendmail` peut aider à lutter contre le spam, ces méls indésirables qui encombrent nos boîtes aux lettres. Pour cela, on utilise la “liste noire” par défaut du serveur **rbl.maps.vix.com**.

Il suffit juste de rajouter la ligne suivante dans le fichier de configuration à passer à `m4` :

```
FEATURE(rbl)
```

Il existe un système plus flexible, permettant de choisir quels hôtes ou quels utilisateurs on veut rejeter. Pour chaque connexion SMTP, `sendmail` compare les informations contenues dans l'en-tête du message, puis accepte ou rejette le courrier suivant ce qu'il a trouvé. La base d'accès est un ensemble de règles décrivant ce qu'il faut faire pour les messages reçus. Il y a plusieurs actions possibles :

OK Accepte le mél ;

RELAY Accepte ce mél même si il n'est pas destiné pour notre machine. Cela autorise en fait le relai de messages ;

REJECT Rejette le mél avec un message générique ;

*Message d'erreur* Retourne un message d'erreur en utilisant le code d'erreur ### et *Message d'erreur* comme message d'erreur ;

Voici un exemple de fichier `/etc/mail/access` à écrire :

```
mechant@nefaste.com REJECT
spam.com            REJECT
212.62.169.138      REJECT
postmaster@spam.com OK
autre.com           RELAY
```

Dans cet exemple, on rejette tous les méls venant de **mechant@nefaste.com**, de n'importe quelle hôte appartenant au domaine **spam.com** et de l'hôte **212.62.169.138**. En dépit du refus des méls venant de **spam.com**, on accepte quand même les messages de la part de **postmaster@spam.com**. Enfin, on accepte de relayer les méls adressés à n'importe quel hôte du domaine **autre.com**.

Il ne reste plus qu'à rajouter dans le fichier de configuration à passer à `m4` la ligne :

```
FEATURE(access_db)
```

et à regénérer la base de données :

```
makemap hash /etc/mail/access < /etc/mail/access
```

Il est parfois utile de refuser du mél pour des programmes qui ne le liront jamais, par exemple. Il suffit alors de mettre dans le fichier `/etc/mail/access` les lignes :

```
daemon       550 Un daemon ne lit pas le mel...
punis@domine 550 Lui, il est punit, il n'a plus le droit aux mels
```

et de regénérer la base de données.

10 Configurer des hôtes virtuels

Les hôtes virtuels permettent d'héberger sur une même machine plusieurs domaines. Cette machine doit donc être capable d'accepter les courriers à destination des différents hôtes, de les distribuer à leurs destinataires et rediriger les courriers venant de la part d'un domaine vers l'extérieure. Heureusement, tout cela se fait simplement.

Il faut d'abord dire à `sendmail` quels sont les hôtes dont il a la charge, c'est à dire pour lesquels il doit accepter les courriers. Cela se fait en mettant la ligne suivante dans le fichier à passer à `m4` :

```
FEATURE(use_cw_file)
```

Le fichier contenant la liste des domaines est `/etc/mail/local-host-names` pour les distributions utilisant comme répertoire de fichiers de configuration `/etc/mail/` ou `/etc/sendmail.cw` pour les autres. On peut préciser le nom de ce fichier comme suit :

```
define(`confCW_FILE',`/etc/virtualnames')
```

Ce fichier doit simplement contenir le nom des domaines pour lesquels on accepte les méls :

```
domaine1.com
domaine2.com
domaine3.com
```

Ensuite, il faut s'assurer que les enregistrements MX de ces domaines pointent vers la machine qui les hébergent.

Il ne reste plus alors qu'à renvoyer ces courriers vers leurs destinataires. Pour cela, il faut rajouter la ligne suivante dans le fichier à passer à `m4` :

```
FEATURE(virtusertable)
```

pour activer la possibilité de transmettre ces courriers.

Le fichier `/etc/mail/virtusertable` doit contenir les correspondances pour chaque utilisateur :

```
Jean.Bon@domaine1.com     jbon
John.Deuf@domaine2.com    jdeuf@domaine3.com
@domaine4.com             mail@domaine5.com
@domaine6.com             $1@domaine7.com
```

Compiler (`makemap hash /etc/mail/virtusertable < /etc/mail/virtusertable`.

Dans cet exemple, on accepte les méls à destination des domaines **domaine1.com**, **domaine2.com**, **domaine4.com**, et **domaine6.com**. Les courriers destinés à **Jean.Bon@domaine1.com** seront distribués à l'utilisateur local **jbon**, ceux destinés à **John.Deuf@domaine2.com** seront redistribués à l'adresse extérieure **jdeuf@domaine3.com**. Tous les courriers destinés à un utilisateur du domaine **domaine4.com** seront envoyés à **mail@domaine5.com**, et ceux destinés à un utilisateur du domaine **domaine6.com** seront envoyés au domaine **domaine7.com** avec le même login (**jbon@domaine6.com** sera redirigé vers **jbon@domaine7.com**, par exemple).

Références

[1] **Guide de l'administration réseau sous Linux**, Olaf KIRCH et Terry DAWSON, seconde édition, juin 2000.

Version française (version 1, plus à jour !) :
`ftp://ftp.lip6.fr/pub/linux/french/books/nag.french.eoit-1.0.tar.gz`.
Version originale (en anglais) :
`http://www.linuxdoc.org/LDP/nag2/index.html`.

[2] **Page d'accueil de Sendmail**
`http://www.sendmail.org/`

[3] **Questions fréquemment posées sur Sendmail**
`http://www.sendmail.org/faq/`

[4] **Gérer des hôtes virtuels avec Sendmail**
http://www.sendmail.org/virtual-hosting.html

[5] **Page d'accueil de Sendmail**
http://www.sendmail.org/

[6] **Page d'accueil de Fetchmail**
http://www.tuxedo.org/~esr/fetchmail/index.html

[7] **Lire et envoyer du courrier off-line sur sa machine**, Éric Jacoboni, 1er Juillet 1998.
http://www.linux-france.org/article/mail/sendmail/

[8] **Envoyer du courrier avec sendmail**, Olivier Hoarau, 26 Novembre 2000.
http://www.funix.org/fr/linux/main-linux.php3?ref=sendmail&page=menu

[9] **Lire et envoyer du courrier off-line sur sa machine**, Éric Jacoboni V1.0, 1er Juillet 1998.
http://www.linux-france.org/article/mail/sendmail/

[10] **Administration Système UNIX**, Thierry BESANCON, Pierre DAVID, Joël MARCHAND, 1 avril 1996.
ftp://ftp.uvsq.fr/pub/sysadmin/cookbook/cookbook/2.0/acc.ps.gz