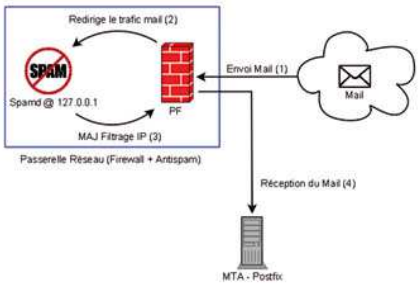
Spamd est un programme antispam fourni dans le système de base OpenBSD. Il réglemente le trafic vers le MTA (Mail Transfer Agent) en interagissant avec PF (Packet Filter). Nous allons étudier ses mécanismes d'éradication, son intégration à PF et donner quelques retours sur son utilisation.

1. Mécanismes

1.1 Généralités

Spamd est un daemon qui va s'intercaler entre le monde extérieur et votre MTA en simulant un faux MTA. La passerelle reçoit le mail (1), le trafic est redirigé vers Spamd (2). Spamd le traite et met éventuellement à jour les règles de filtrage de PF (3). Si le mail est licite, alors le vrai MTA reçoit le message.



Plusieurs points sont à noter :

- Spamd est totalement indépendant du MTA réel (On pourrait, par exemple, le coupler avec Exchange).
- Il agit de manière complémentaire à tout autre filtre anti-spam plus « haut niveau » du style SpamAssassin. Ce genre de combinaison est très intéressant car, malgré la montée en puissance au niveau calcul des machines, SpamAssassin est extrêmement gourmand en ressources et un examen systématique du trafic SMTP peut effondrer votre architecture mail. Idem pour l'analyse antivirale (avec Clamav par exemple).
- Il est très efficace du point de vue utilisation des ressources. Son intégration à PF tire parti des performances de ce dernier.
- Ajouter un premier niveau de filtrage mail sur le pare-feu est intéressant dans la mesure où les machines sont souvent surdimensionnées niveau puissance pour faire uniquement du filtrage IP (mais où l'on n'ajoute rien d'autre dans un souci de sécurité). Cet ajout est possible sans mettre en cause la sécurité de l'ensemble du fait que Spamd est présent dans la base d'OpenBSD et subit un audit sévère du code du point de vue sécurité.

Le routage du trafic mail vers Spamd ou vers le MTA réel est réglementé par les listes. Ces listes contiennent des IP. Le fait pour un émetteur de mails d'appartenir à telle ou telle liste a différents effets sur la transaction SMTP dont il est l'auteur. Nous allons détailler ces différentes listes.

1.2 Les Listes

- Le black listing : cette liste est gérée par Spamd.

Elle est alimentée via le fichier /etc/mail/spamd.conf et contient des adresses de supposés spammeurs. Lorsqu'une transaction arrive sur le faux MTA Spamd avec comme source une de ces adresses, Spamd déclenche une action de tarpitting (c'est-à-dire qu'il va répondre extrêmement lentement à la machine source pour consumer les ressources de ce supposé spammeur) et répond avec un code 550 ou 450. En gros, un supposé spammeur sur black list n'est jamais autorisé à parler au vrai MTA. La table PF associée se nomme ~~<spamd>~~.

- Le white listing : cette liste contient les hôtes autorisés à parler directement au vrai MTA. Cette liste est alimentée par Spamd. Elle est le résultat du processus de décision de Grey Listing (point suivant) et des entrées taguées white dans le fichier ~~/etc/mail/spamd.conf~~ (voir la section « configuration »). La table PF associée se nomme ~~<spamd-white>~~.
- Le grey listing : Spamd n'a pas décidé s'il s'agit d'une connexion licite ou non. Le processus de décision se déroule ainsi : à la première connexion, Spamd renvoie une erreur 451 (serveur temporairement inaccessible, réessayez plus tard). Un timer est déclenché et un triplet temps passé au démarrage de Spamd est utilisé. Le premier élément du triplet fixe le temps (en minutes) au-delà duquel une tentative de reconnexion fait passer l'émetteur de liste grise à liste blanche (autorisée à parler directement au vrai MTA). Le second est une limite de temps (en heures) pour l'existence de l'entrée dans la base Spamd. Si aucune reconnexion n'a été tentée au-delà de cette période, alors l'entrée est supprimée de la base de données Spamd. Le dernier paramètre du triplet donne la durée de validité (en heure) d'une entrée passée de grey en white list. Si l'enregistrement concerné n'est pas utilisé pendant cette période de temps, alors il est supprimé de la white list.
- Spamtrap : cette liste est composée d'adresses mails. Cette méthode définit une (ou plusieurs) adresse(s) mail(s) comme adresse(s) à Spam. Typiquement, on va utiliser des adresses qui ne sont plus usitées ou présentes sur des listes de spammeurs. Quand un hôte est en Grey List et essaie d'envoyer un message à une adresse de Spamtrap, cet hôte est blacklisté pendant 24h. Pas mal de gens s'amusent avec Spamtrap [1] ...

1.3 Architecture & fonctionnement

Spamd se compose de deux daemons :

- Spamd, un faux MTA traditionnellement bindé sur l'interface de loopback. Son rôle est d'examiner les transactions mail (à destination du port SMTP) passant par le pare-feu. Il va gérer les listes blanches, grises et noires.
- Spamlogd, programme qui surveille les requêtes mail (à destination du port SMTP) sur l'interface pflog. C'est lui qui va entretenir la liste blanche propre a Spamd. Les règles d'acceptation du trafic SMTP au niveau PF doivent être loguées. Un exemple :

```
pass in log quick on $ext_if proto tcp from any to 192.168.0.1 port smtp
```

Note sur pflog
L'interface pflog va produire un affichage lorsqu'une règle notée « log » va être appliquée par notre pare-feu. Cet affichage contient le numéro de la règle qui a déclenché l'action de log.

```
# tcpdump -netti pflog0
1194186759.230304 rule 2/(match) block in on nve0: une_ip > une_autre_ip: [|tcp]
```

Spamd va interagir avec PF de deux manières :

- Statique sous la forme de redirections de ports (~~rdr~~).
- Dynamique grâce aux tables. Ces tables sont utilisées par les règles de ~~rdr~~ pour orienter le trafic mail soit vers le daemon Spamd (black et grey list), soit vers le vrai MTA (white list).

En pratique, nous allons déployer le banc de test suivant : le célèbre domaine « example.org » avec comme enregistrement MX (Mail Exchanger) l'IP publique 195.30.25.4 (Point d'entrée du trafic SMTP pour notre réseau). Une machine sur le réseau interne qui va servir de MTA réel (ici Postfix). Son IP privée est la 192.168.0.1.Une machine pare-feu sous OpenBSD avec Spamd bindé sur 127.0.0.1 port 8025 (par défaut).

2. Configuration

2.1 Règles PF

Configurons le pare-feu pour intercaler Spamd entre notre MTA réel et le reste du monde. Voici un extrait de notre ~~/etc/pf.conf~~ (Section ~~NAT/BiNAT/rdr~~) :

```
(1) rdr pass on $ext_if proto tcp from <whitelist> to 195.30.25.4 port smtp -> 192.168.0.1 port smtp
(2) rdr pass on $ext_if proto tcp from <blacklist> to 195.30.25.4 port smtp -> 127.0.0.1 port 8025
(3) rdr pass on $ext_if proto tcp from <spamd> to 195.30.25.4 port smtp -> 127.0.0.1 port 8025
(4) rdr pass on $ext_if proto tcp from <spamd-white> to 195.30.25.4 port smtp -> 192.168.0.1 port smtp
(5) rdr pass on $ext_if proto tcp from !<spamd-white> to 195.30.25.4 port smtp -> 127.0.0.1 port 8025
```

Les règles (1) et (2) sont respectivement une whitelist et une blacklist qui vont court-circuiter complètement le mécanisme Spamd. Elles sont présentes sur le disque sous forme d'un fichier texte contenant des IP les unes à la suite des autres. Pour la whitelist (1), on redirige le trafic directement vers le MTA réel. Pour la blacklist (2), on envoie le trafic vers Spamd. Les tables associées à ces deux règles de redirection sont ~~<whitelist>~~ et ~~<blacklist>~~.

```
table <whitelist> persist file "etc/pf-tables/whitelist.txt"
table <blacklist> persist file "/etc/pf-tables/blacklist.txt"
```

Les trois autres règles sont fournies par la documentation de Spamd. Elles utilisent deux tables ~~<spamd>~~ et ~~<spamd-white>~~. La première table contient les IP des hôtes en cours d'approbation (grey listing) ou blacklistés (bit tarpitting). Ces hôtes sont envoyés vers Spamd (3). La seconde (4) contient les hôtes autorisés à parler directement au MTA réel (white list). L'obsolescence des enregistrements de cette liste est gérée par Spamlogd. La dernière règle (5) est un catch all pour récupérer le trafic SMTP non connu et l'envoyer à Spamd.

```
table <spamd> persist
table <spamd-white> persist
```

À présent, le hook pour PF est prêt, nous pouvons passer à la configuration proprement dite de Spamd

2.2 Spamd

La configuration pour les listes whites/blacks se passe au niveau de ~~/etc/mail/spamd.conf~~. On y définit les sources des différentes listes. Examinons notre ~~spamd.conf~~.

```
all:\
    :uatraps:mywhite:nixspam:mywhite:china:mywhite:korea:mywhite:
# Scripts in /var/spamd-scripts
# Take hardcoded domain names in script
# Take whitelist from cvs puremail
mywhite:\
    :white:\
    :method=file:\
    :file=/var/spamd/spamd-spf.txt
# University of Alberta greytrap hits.
# Addresses stay in it for 24 hours from time they misbehave.
uatraps:\
:black:\
        :msg=»Your address %A has sent mail to a ualberta.ca spamtrap\n\
        within the last 24 hours»:\
        :method=http:\
        :file=www.openbsd.org/spamd/traplist.gz
# Nixspam recent sources list.
# Mirrored from http://www.heise.de/ix/nixspam
nixspam:\
:black:\
        :msg=»Your address %A is in the nixspam list\n\
        See http://www.heise.de/ix/nixspam/dnsbl_en/ for details»:\
        :method=http:\
        :file=www.openbsd.org/spamd/nixspam.gz
# Mirrored from http://www.okean.com/chinacidr.txt
china:\
        :black:\
        :msg=»SPAM. Your address %A appears to be from China\n\
        See http://www.okean.com/asianspamblocks.html for more details»:\
        :method=http:\
        :file=www.openbsd.org/spamd/chinacidr.txt.gz:
# Mirrored from http://www.okean.com/koreacidr.txt
korea:\
:black:\
```

```
:black:\
        :msg=»SPAM. Your address %A appears to be from Korea\n\
        See http://www.okean.com/asianspamblocks.html for more details»:\
        :method=http:\
        :file=www.openbsd.org/spamd/koreacidr.txt.gz:
```

Deux choses intéressantes :

- La section ~~all~~ donne l'ordre d'application des règles. Cet ordre est un peu tordu. Par exemple, si on a la séquence ~~black1:white:black2~~ et qu'une IP est présente dans les trois listes, alors la transaction va se retrouver dans une blacklist (~~black2~~). Si on avait voulu donner la primeur aux whitelists (comme dans notre configuration), alors il faudrait enregistrer le quadruplé ~~black1:white:black2:white~~.
- La constitution des listes peut se faire à partir de fichiers (~~:method=file:~~ et ~~:file=chemin:~~) ou de ressources réseau (~~:method=http:~~ et ~~:file=URL:~~).

La commande ~~spamd-setup~~ va lire le fichier, télécharger les ressources réseau, mettre à jour la base de données de Spamd et les tables PF associées. Hop, ~~spamd-setup~~ dans un petit cron et la vie est belle (~~/var/cron/tabs/root~~) :

```
0    *    *    *    *    /usr/libexec/spamd-setup
```

Pendant que l'on est dans les automatisations de tâches, il faut spécifier la rotation du fichier de log de Spamd, sinon il va faire exploser notre système de fichier. Ajoutons dans ~~/etc/newsyslog.conf~~ :

```
 /var/log/spamd    _spamd:wheel   640  7   250  *   Z
```

Ce qui veut dire que ~~/var/log/spamd~~ va être « rotaté » sept fois avant d'être détruit.
La whitelist utilisée dans notre ~~spamd.conf~~ en exemple est une composition de différentes sources :

```
# cat /var/spamd/generate-whitelist.sh
#!/bin/sh
FILE=/var/spamd/spamd-spf.txt

rm -f $FILE
touch $FILE

for domain in \
aol.com \
apple.com \
amazon.com \
gmx.net \
_spf.google.com \
spf-a.hotmail.com \
spf-b.hotmail.com \
spf-c.hotmail.com \
spf-d.hotmail.com \
_spf-a.microsoft.com \
_spf-b.microsoft.com \
_spf-c.microsoft.com
do
echo \#$domain >> $FILE;
dig $domain TXT +short | tr "\ " "\n" | grep ^ip4: | cut -d: -f2 >> $FILE;
done

echo '# http://cvs.puremagic.com/viewcvs/*checkout*/greylisting/schema/whitelist_ip.txt' >> $FILE;
ftp -o - http://cvs.puremagic.com/viewcvs/*checkout*/greylisting/schema/whitelist_ip.txt \
| awk '/^[^#]/ {print $0}' >> $FILE;
```

On vérifie que nos listes blanches sont bien synchrones :

```
# spamdb | grep WHITE | wc -l
    3403
# pfctl -t spamd-white -T show | wc -l
    3403
```

C'est un composé de domaines proposant des enregistrements SPF (~~$domain~~) et de la whitelist issue de ~~puremagic~~ [2]. On pourra également dumper la table ~~<spamd-white>~~ de temps en temps pour pouvoir le réinjecter dans Spamd en cas de catastrophe (par exemple, pour un établissement d'environ 200 personnes, on fait des pointes à près de 700 enregistrements en whitelist) :

```
pfctl -t spamd-white -T show | sed 's/^[ \t]*//' > /var/spamd/spamd-white_`date +%Y-%m-%d`
```

La commande spamdb manipule la base de données de ~~Spamd~~. Pour consulter les mails en cours d'examen :

```
# spamdb | grep GREY
[...]
```

Pour récupérer les transactions en cours pour une adresse mail :

```
# spamdb | grep nicolas.greneche@example.org
GREY|62.209.218.70|usgs.gov|<jraber@valkyrie.net>|<nicolas.greneche@example.org>|1194181594|1194195994|119419599
```

Cet enregistrement dispose de 10 champs :

- le type d'enregistrement (WHITE ou GREY) ;
- l'IP de l'émetteur du message ;
- le message HELO envoyé par l'émetteur ;
- adresse mail source ;

- adresse mail source ;
- adresse mail destination ;
- date de la première entrée dans la spamdb (première tentative de connexion) ;
- date à laquelle l'enregistrement sera promu sur liste blanche ;
- date d'expiration de l'enregistrement dans la spamdb ;
- nombre de fois où une telle connexion a reçu une « temporary failure » de Spamd ;
- nombre de fois où une telle connexion est passée sur liste blanche.

Ajouter un hôte en whitelist (à utiliser avec le dump précédemment mentionné pour remonter sa liste blanche Spamd) :

```
# spamdb -a <IP_à_whitelister>
```

2.3 Lancement & paramètres

Sans surprises, il faut ajouter des choses dans le ~~/etc/rc.conf~~.

```
spamd_flags=»-v -l 127.0.0.1 -S 10 -n Postfix -h realmta.example.org -G 7:4:864»
spamd_grey=YES
spamlogd_flags=»-l pflog0»
```

Pour lancer Spamd :

```
/usr/libexec/spamd -v -l 127.0.0.1 -S 10 -n Postfix -h realmta.example.org -G 7:4:864
```

Option	Explication
v	mode verbeux
l	interface sur laquelle se binder
S	temps de *stuterring* pour les connexions sur liste grise
n	bannière du MTA présenté à la connexion
h	FQDN du MTA pour la bannière
G	triplet de temps : ▶ temps au-delà duquel une tentative de reconnexion en liste grise fait passer la connexion en liste blanche (en minutes) ; ▶ temps au-delà duquel une connexion sur liste grise est supprimée de la base de données de Spamd (en heures) ; ▶ temps d'inactivité maximum d'un hôte sur liste blanche (spamd-white) avant sa suppression.

our lancer Spamlogd :
/usr/libexec/spamlogd -l pflog0

3. Conclusion

3.1 Retours

On notera que nos paramètres sont plutôt sympas. Le man de Spamd propose une configuration plus dure du triplet temps. En le configurant par défaut, nous avons perdu des mails licites (Wanadoo et Gmail). La raison pour Wanadoo est un trop long délai de réémission (au-delà du second élément du triplet temps). Pour Gmail, c'est plus vicieux : le MTA de Gmail change à chaque tentative. Un coup, c'est le A qui va envoyer, puis le B va réessayer avant de retenter avec A, etc. Ainsi, le message reste indéfiniment en greylist jusqu'à ce que Gmail se lasse. Pour éliminer le spam résiduel, on peut ajouter une filtre supplémentaire au niveau du MTA (Amavisd-new + SpamAssassin serait un bon complément).

3.2 Synchronisation

À l'instar de pfsync pour PF, Spamd dispose d'un système de synchronisation de la spamdb. Cette synchronisation recopie les enregistrements d'une spamdb en cours d'utilisation vers un ou plusieurs hôtes de failover. Ces mises à jour peuvent se faire en unicast (dans le cas où l'on a une seule machine de backup) ou en multicast (plusieurs machines de backup). La sécurité des transmissions entre les différentes instances de Spamd est assurée par un algorithme HMAC. Pour mettre cette solution en œuvre, il faut commencer par générer un secret sur la machine « maître » et le répliquer sur la machine esclave :

```
# dd if=/dev/arandom of=/etc/mail/spamd.key bs=2048 count=1
```

Supposons que nos deux machines aient une interface em0 dédiée à la synchronisation, il nous faudra renseigner Spamd sur le fait qu'il doit se synchroniser. Sur l'esclave, il faut accepter les messages de synchronisation, sur em0, il faut ajouter ~~-y em0~~ dans les options de Spamd (~~spamd_flags~~ du ~~/etc/rc.conf~~) et sur le maître on ajoutera ~~-Y <fqdn_esclave>~~ pour spécifier l'esclave en cible de la synchronisation.

Liens:

- [1] http://bsdly.blogspot.com/search/label/spamtrap
- [2] http://projects.puremagic.com/greylisting/index.html